Department of Computer Science

Quaid-e-Azam University, Islamabad

Assignment 4

Student Course Registration System

Prepared by:

HURR MEHDI
M. SEHAL BILAL (Leader)
M. ASAD
ZOHAIB AHMED

Submitted to:

DR. ONAIZA MAQBOOL

24 March 2025

DATA DICTIONARY

STUDENT COURSE REGISTRATION SYSTEM

Created by	Group 8
Date Created	24-03-2025

Field Name	***Description***	***Type***	***Length***	***Conditions/Constraints***
Student_ID	Unique identifier for students	String	10	Must be unique for each student
Student_Name	Full name of the student	String	50	Must contain only alphabets
Email	Student email address	String	50	Must be a valid email format
Coordinator_ID	Unique identifier for a course or timetable coordinator	Integer	10	Must be unique.
Coordinator_Name	Full name of the coordinator	String	50	Required.
Coordinator_Email	Email address of the coordinator	String	50	Must be a valid email format.
Course_ID	Unique identifier for a course	String	10	Must be unique for each course
Course_Name	Name of the course	String	20	Cannot be null
Course_Credits	Number of credit hours for the course	Integer	2	Must be between 1 and 6.
Prerequisites	List of prerequisite courses.	List<String>	-	Valid course IDs only.
Semester_ID	Unique identifier for a semester.	String	6	Format: YYMM (e.g., 2501 for Jan 2025).
Start_Date	Semester Start date	Date	-	MM/DD/YYYY format.
End_Date	Semester End date	Date	-	MM/DD/YYYY format.
Report_ID	Unique identifier for an academic report	Integer	10	Must be unique.

Pass_Fail_Status	Pass or fail status of the student's performance	String	10	Values: "Pass" or "Fail."
GPA	Grade Point Average	Decimal	4,2	Must be within 0.00–4.00 (or your institutional range).
Conflict_ID	Unique identifier for a timetable conflict	Integer	10	Must be unique.
Resolution_Details	Details on how a timetable conflict is resolved	String	255	Optional/May be empty if not resolved.
Student_ID (FK)	Student receiving the grade	String	10	Must exist in Student table
Scheme_ID	Unique identifier for a study scheme	Integer	10	Must be unique.
Batch	Represents the batch/year group for the study scheme	String	10	Example: "Fall2025," "2025-2029."

Data Structures

1. **Student**
   Student = Student_ID + Student_Name + Email

2. **Course Coordinator**
   Coordinator_ID + Coordinator_Name + Coordinator_Email

3. **Timetable Coordinator**
   Coordinator_ID + Coordinator_Name + Coordinator_Email

4. **Course**
   Course_ID + Course_Name + Course_Credits + Prerequisites

5. **Academic Report**
   Report_ID + Pass_Fail_Status + GPA

6. **Semester**
   Semester_ID + Start_Date + End_Date

7. **Timetable Conflict**
   Conflict_ID + Resolution_Details

8. **Study Scheme**
   Scheme_ID + Batch

USE CASE DESCRIPTION & SYSTEM SEQUENCE DIAGRAM	**STUDENT COURSE REGISTRATION SYSTEM**
Created by	Hurr Mehdi (04072312048)
Date Created	24-03-2025

Use Case: View Academic Progress

Primary Actor(s):

- *Student*

Use Case Description:

This use case allows a student to view their academic progress, including courses they have passed, failed, and pending requirements.

Stakeholders and Interests:

- *Student: Wants to track their academic performance and plan future registrations.*
- *University Administration: Ensures accurate record-keeping and compliance with academic policies.*
- *Course Coordinator: Uses progress data to advise students and update curriculum requirements.*

Preconditions:

- *The student must be logged into the system.*
- *The student's academic records (course registrations, grades) must exist in the system.*

Postconditions:

- *The student's academic progress is displayed.*
- *No changes are made to the database (read-only action).*

Inputs:

- *StudentID*

Outputs:

- *AcademicProgressDetails (passed courses, failed courses, pending prerequisites, GPA).*

Main Success Scenario:

1. *The student navigates to the Academic Progress section.*
2. *The system retrieves the student’s academic records (registrations, grades, prerequisites).*
3. *The system generates a summary of:*
    - *Passed Courses: Courses marked as "Pass."*
    - *Failed Courses: Courses marked as "Fail."*
    - *Pending Requirements: Prerequisites not yet completed.*
4. *The system displays the academic progress summary to the student in a tabular or visual format.*
5. *The student reviews the information and exits the section.*

Alternative Scenarios:

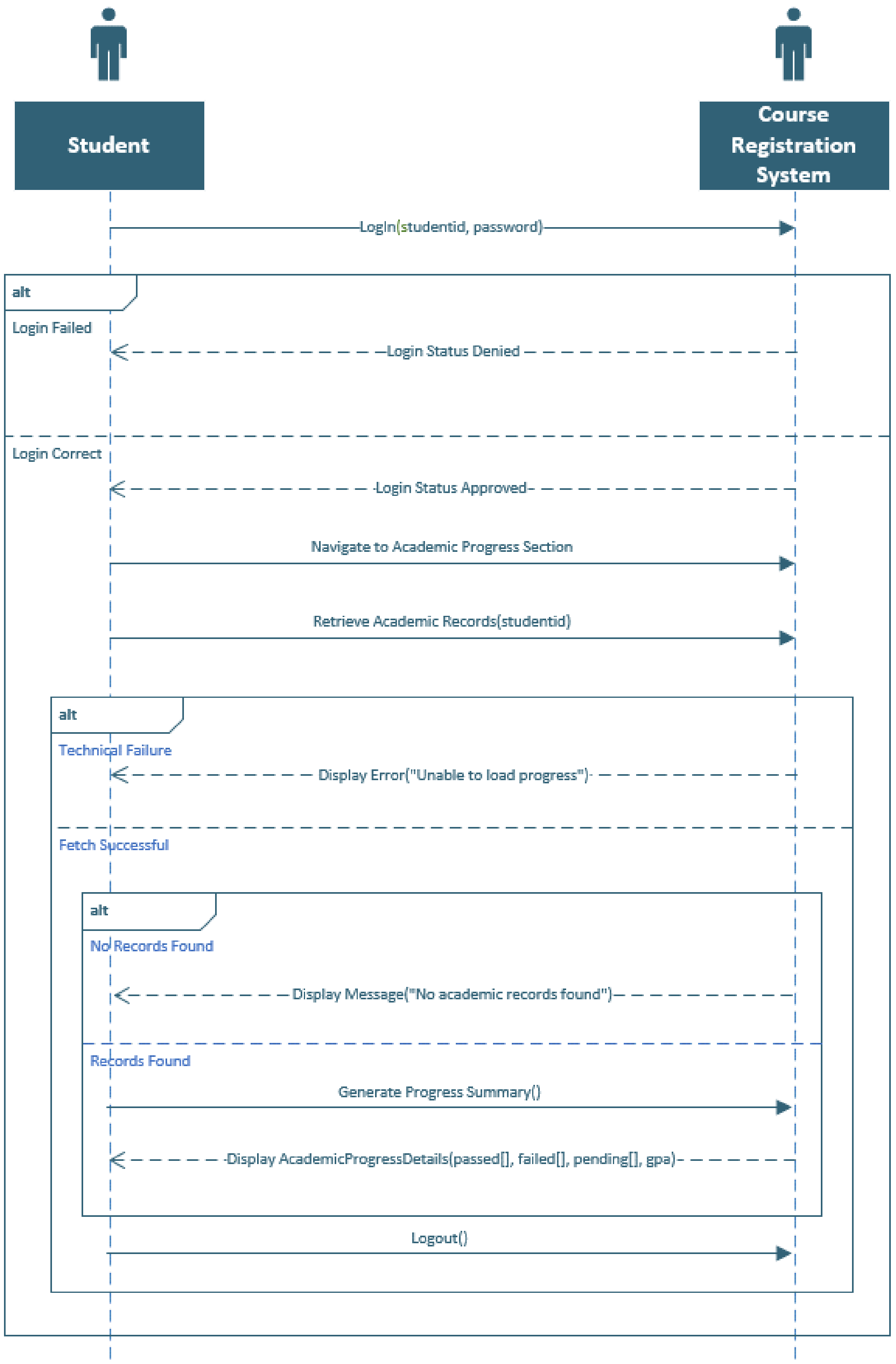
- *No Academic Records Found:*

The system displays a message: “No academic records available. Please register for courses first.”

- *Technical Failure:*

If the system fails to retrieve data, an error message is displayed: “Unable to load academic progress. Please try again later.”

Sequence Diagram: View Academic Progress


Student
Course Registration System
LogIn(studentid, password)
alt
Login Failed
Login Status Denied
Login Correct
Login Status Approved
Navigate to Academic Progress Section
Retrieve Academic Records(studentid)
alt
Technical Failure
Display Error("Unable to load progress")
Fetch Successful
alt
No Records Found
Display Message("No academic records found")
Records Found
Generate Progress Summary()
Display AcademicProgressDetails(passed[], failed[], pending[], gpa)
Logout()

2. Actors and Use Cases

Use Case: Login

Primary Actor(s):

- *Student*
- *Course Coordinator*
- *TimeTable Coordinator*

Use Case Description:

This use case allows users (students, course coordinators, and timetable coordinators) to log into the system to access their respective functionalities.

Stakeholders and Interests:

- *Student: Requires access to academic features.*
- *Course Coordinator: Needs system access for course-related tasks.*
- *TimeTable Coordinator: Manages class schedules and prerequisites.*

Preconditions:

- *The user must have valid login credentials.*

Postconditions:

- *The user is authenticated and granted access.*

Inputs:

- *Username*
- *Password*

Outputs:

- *LoginStatus (Success/Failure)*

Main Success Scenario:

1. *The user navigates to the login page.*
2. *The user enters a valid username and password.*
3. *The system verifies the credentials.*
4. *The system grants access and redirects the user to their dashboard.*

Alternative Scenarios:

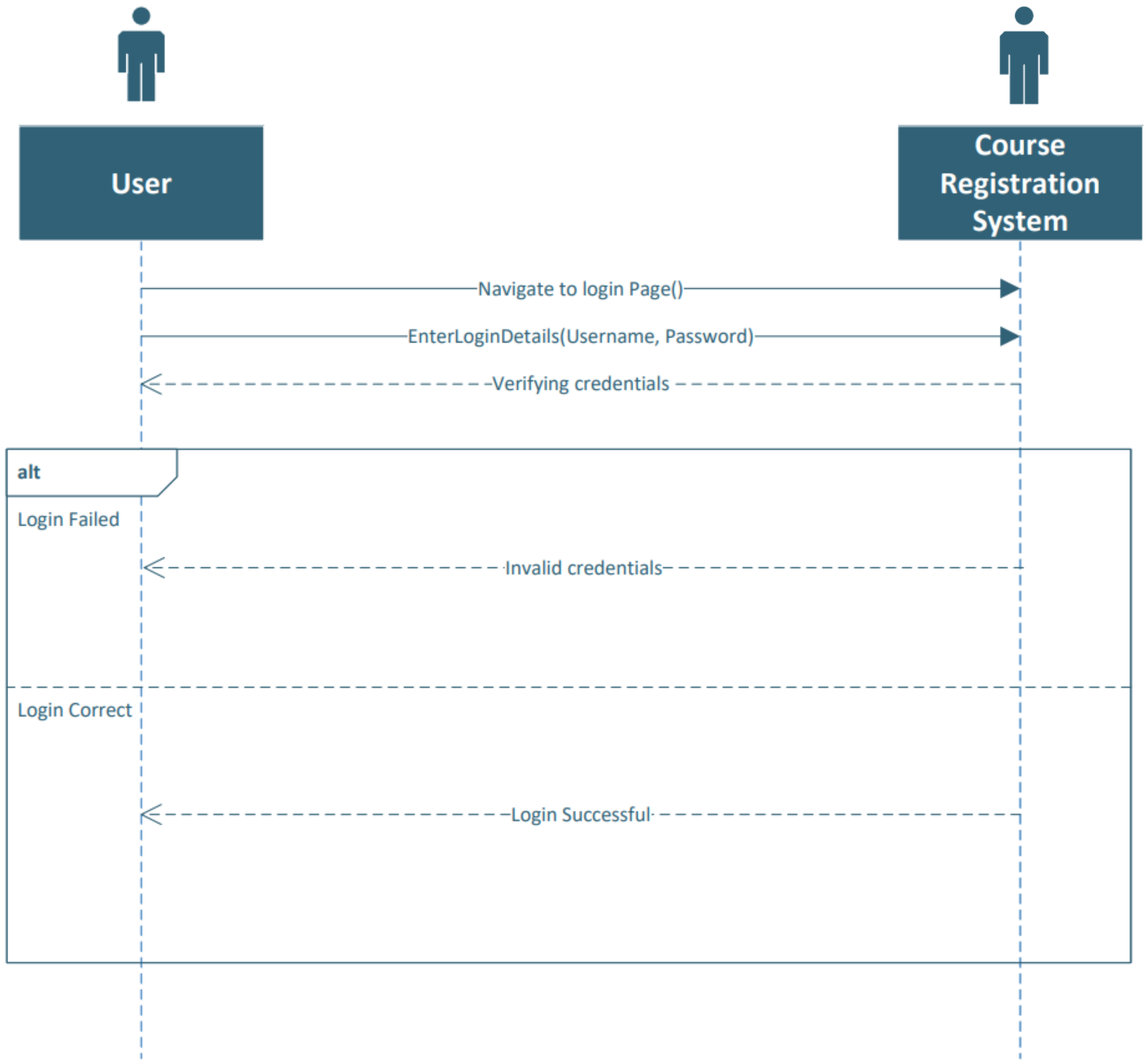
- *Invalid Credentials:*

  If the entered credentials are incorrect, the system displays an error message and the user is prompted to re-enter valid credentials.

- *System Error:*

  If an unexpected issue occurs, the system notifies the user and suggests retrying later.

Sequence Diagram: Login



Use Case: View Courses

Primary Actor(s):

- *Student*

Use Case Description:

This use case allows students to view available courses for registration and academic tracking.

Stakeholders and Interests:

- *Student: Requires course information for registration and planning.*
- *Course Coordinator: Ensures course details are correctly displayed.*

Preconditions:

- *The student must be logged into the system.*

Postconditions:

- *The student can view the list of available courses.*

Inputs:

- *StudentID*

Outputs:

- *CourseList*

Main Success Scenario:

1. *The student navigates to the "View Courses" section.*
2. *The system retrieves available courses for the student.*
3. *The system displays the course list.*

Alternative Scenarios:

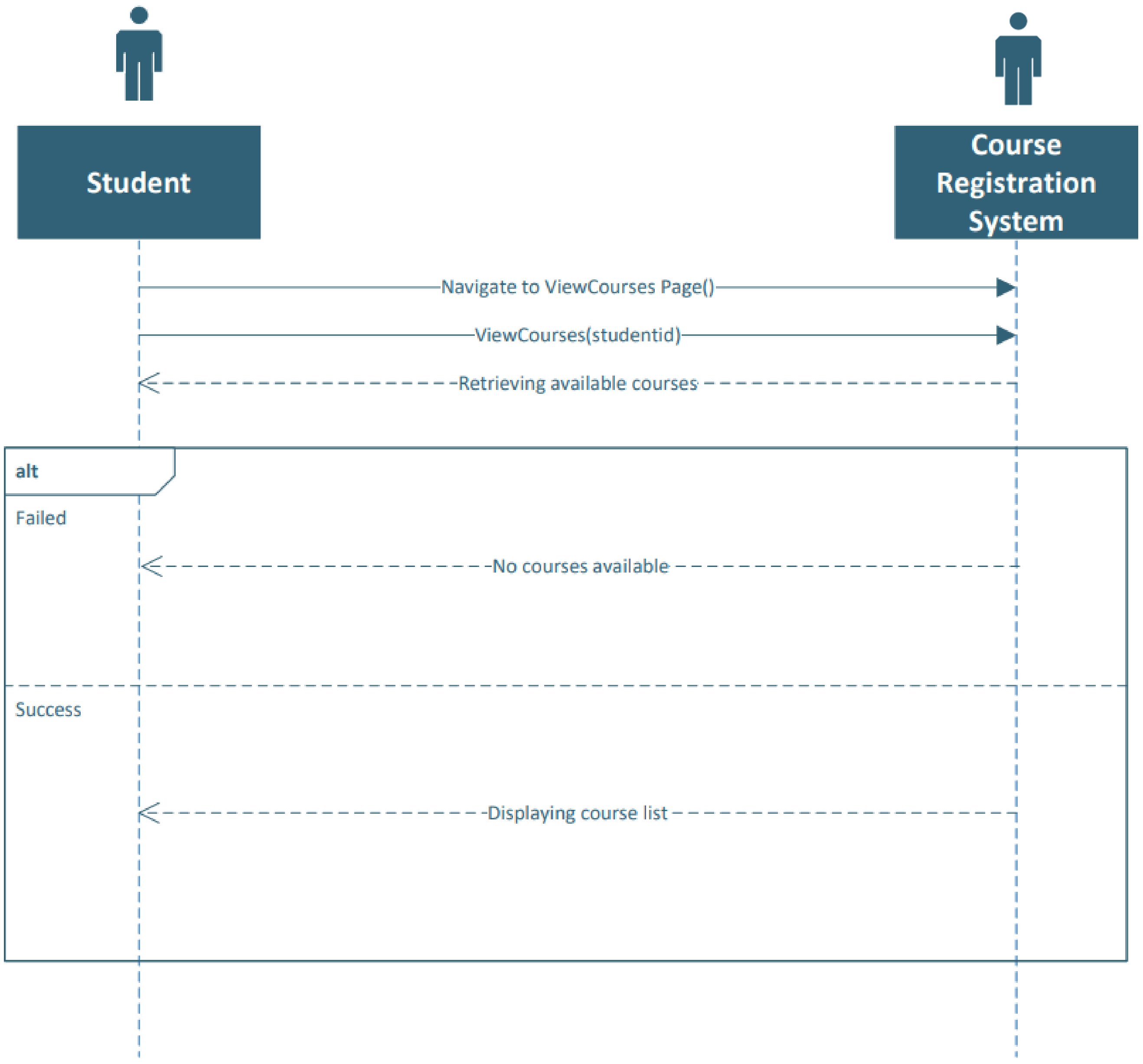
- *No Courses Available:*

  If no courses are found, the system informs the student that no courses are currently available.

- *System Error:*

If a technical issue occurs, the system notifies the student to try again later.

Sequence Diagram: View Courses



USE CASE DESCRIPTION & SYSTEM SEQUENCE DIAGRAM	**STUDENT COURSE REGISTRATION SYSTEM**
Created by	M. Sehal Bilal (04072312015)
Date Created	24-03-2025

Use Case: Register Courses

Primary Actor(s):

- *Student*

Use Case Description:

This use case allows a student to register for courses during the course registration period. The system ensures that the student meets prerequisites before enrolling them in a course.

Stakeholders and Interests:

- *Student: Wants to successfully register for courses without conflicts.*
- *University Administration: Ensures students enroll in the correct courses.*
- *Course Coordinator: Monitors course enrollments and prerequisite compliance.*
- *Timetable Coordinator: Manages scheduling conflicts and class capacities.*

Preconditions:

- *The student must be logged into the system.*
- *The course registration period must be active.*

Postconditions:

- *The registered course is stored in the database.*
- *The system updates the student's registered courses.*

Inputs:

- *StudentID*
- *CourseID*

Outputs:

- *RegistrationStatus (Confirmed or Denied)*
- *UpdatedCourseList*

Main Success Scenario:

1. *The student navigates to the course registration section.*
2. *The student enters the course ID for the desired course.*
3. *The system checks if the student meets the course prerequisites.*
4. *If prerequisites are met, the system registers the student.*
5. *The system updates the student's registered courses.*
6. *The system confirms the registration to the student by displaying a ✓(checkmark) or 'True' status next to the registered course ID.*

Alternative Scenarios:

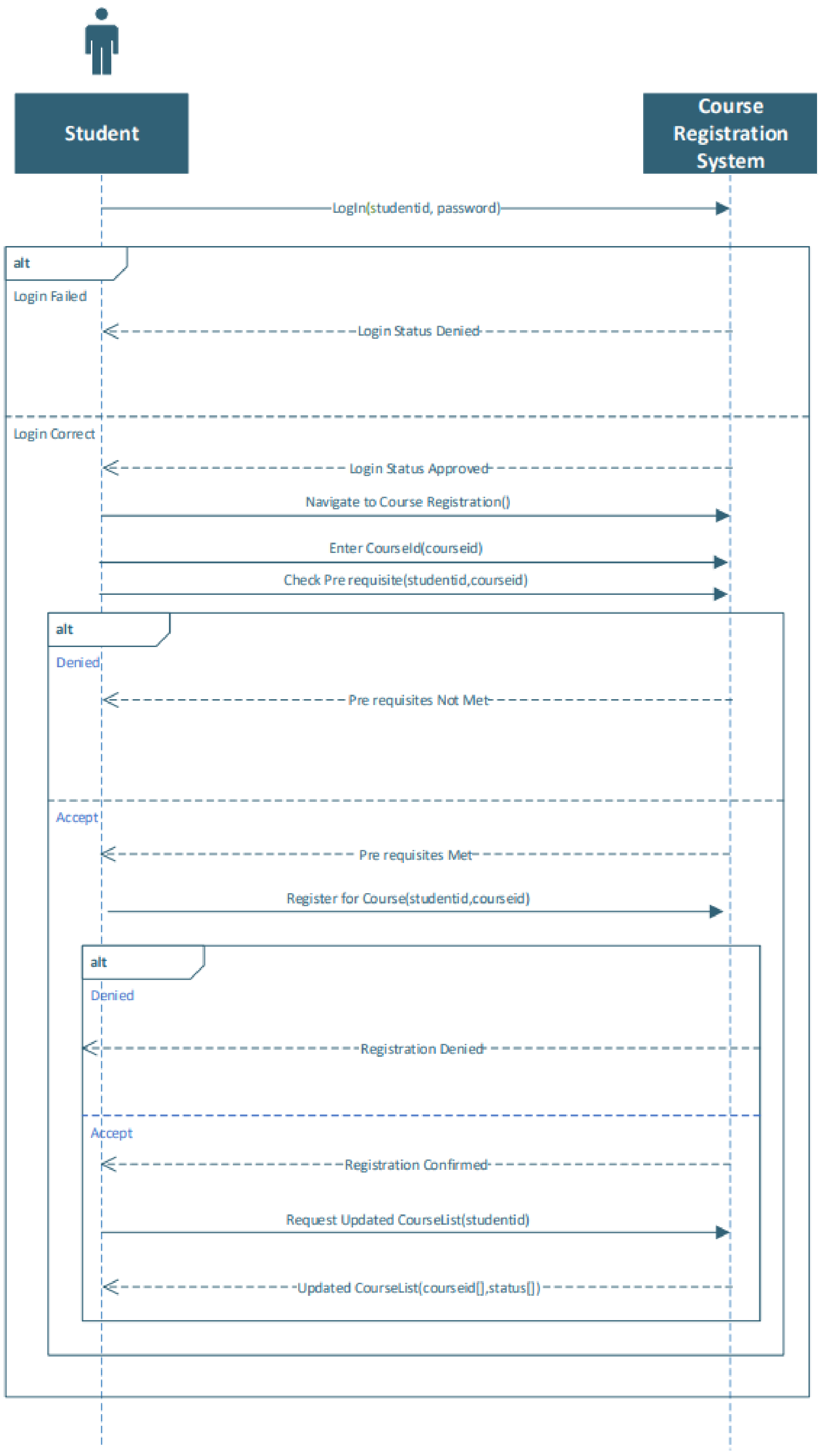
- *Prerequisites Not Met*

  The system notifies the student that they do not meet the prerequisites.The student is unable to register for the course.

- *Technical Failure*

  If the system crashes or encounters an error, the student cannot complete registration.The system logs the failure, and the student is advised to retry later.

Sequence Diagram: Register Courses


Student
Course Registration System
LogIn(studentid, password)
alt
Login Failed
Login Status Denied
Login Correct
Login Status Approved
Navigate to Course Registration()
Enter CourseId(courseid)
Check Pre requisite(studentid,courseid)
alt
Denied
Pre requisites Not Met
Accept
Pre requisites Met
Register for Course(studentid,courseid)
alt
Denied
Registration Denied
Accept
Registration Confirmed
Request Updated CourseList(studentid)
Updated CourseList(courseid[],status[])

Use Case: Generate Student Report

Primary Actor(s):

- *Course Coordinator*

Use Case Description:

This use case allows the Course Coordinator to generate a Student Academic Report for a specific student and semester.

Stakeholders and Interests:

- *Course Coordinator: Needs student reports for academic monitoring.*
- *University Administration: Uses reports for record-keeping and analysis.*
- *Students: May request reports for personal academic tracking.*

Preconditions:

- *The Course Coordinator must be logged into the system.*

Postconditions:

- *The system generates a Student Academic Report.*
- *The report is available for viewing.*

Inputs:

- *StudentID*
- *Semester*

Outputs:

- *GeneratedStudentReport*

Main Success Scenario:

1. *The Course Coordinator navigates to the "Generate Student Report" section.*
2. *The Course Coordinator enters the StudentID and Semester for which the report is required.*
3. *The system retrieves the academic data of the specified student for the given semester.*
4. *The system generates the Student Academic Report.*
5. *The system confirms report generation and provides a viewing option.*

Alternative Scenarios:

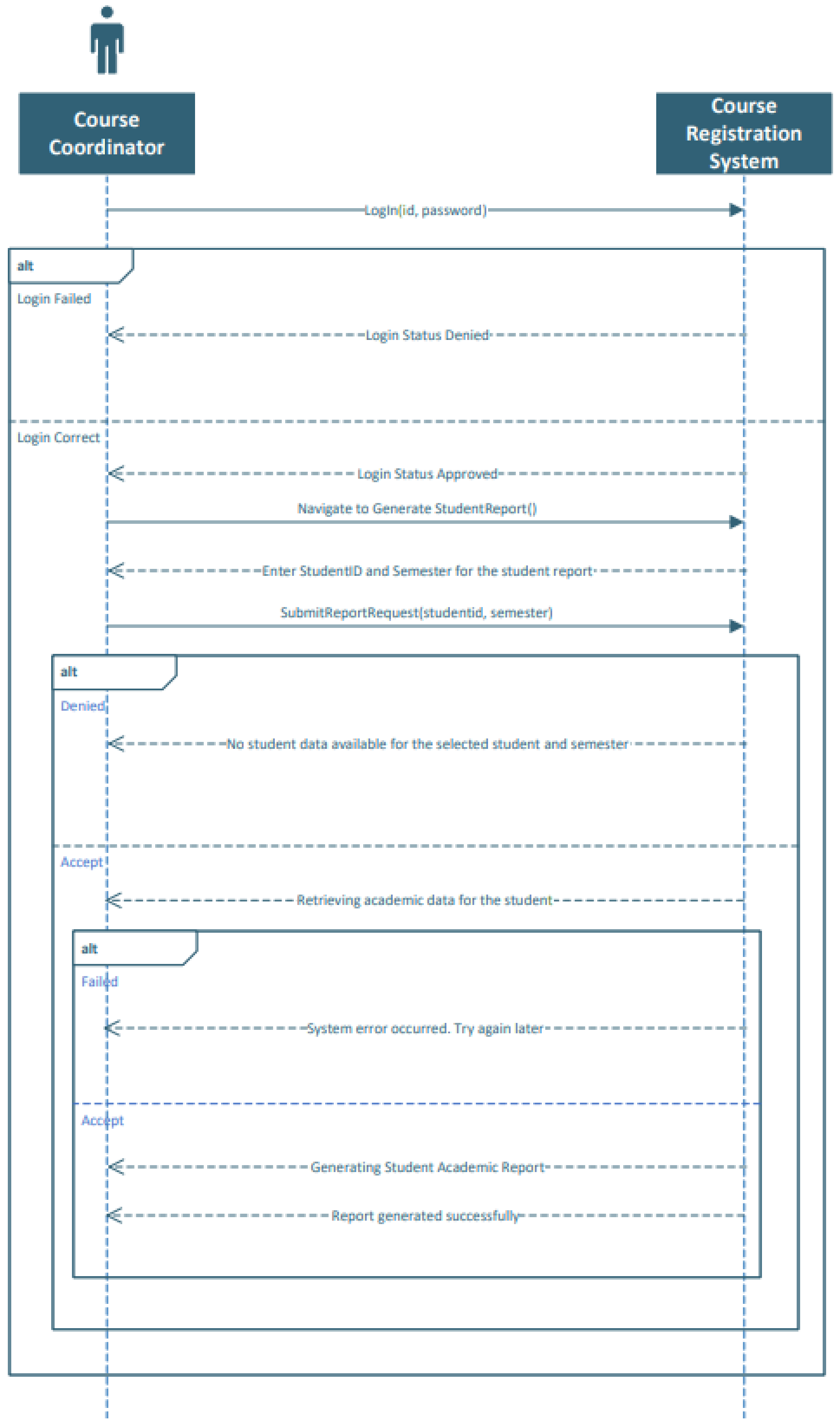
- *No Student Data Available:*

  No student data is available for the selected student and semester.The system informs the Course Coordinator that no records exist for the specified student and semester.

- *Technical Failure:*

  A system error occurs while generating the report. The system informs the Course Coordinator to try again later due to the error.

Sequence Diagram: Generate Student Report


Course Coordinator
Course Registration System
LogIn(id, password)
alt
Login Failed
Login Status Denied
Login Correct
Login Status Approved
Navigate to Generate StudentReport()
Enter StudentID and Semester for the student report
SubmitReportRequest(studentid, semester)
alt
Denied
No student data available for the selected student and semester
Accept
Retrieving academic data for the student
alt
Failed
System error occurred. Try again later
Accept
Generating Student Academic Report
Report generated successfully

Use Case: Upload Scheme of Study

Primary Actor(s):

- *Course Coordinator*

Use Case Description:

This use case allows the Course Coordinator to upload a Scheme of Study, defining course weightage and prerequisites for different student batches.

Stakeholders and Interests:

- *Course Coordinator: Ensures course structure is correctly defined.*
- *Students: Depend on an accurate scheme of study for academic planning.*
- *University Administration: Requires updated curriculum information.*

Preconditions:

- *The Course Coordinator must be logged into the system.*

Postconditions:

- *The Scheme of Study is recorded in the system.*
- *Students can view updated course prerequisites and weightage.*

Inputs:

- *SchemeOfStudyFile*

Outputs:

- *UploadStatus (Successful / Failed)*

Main Success Scenario:

1. *The Course Coordinator navigates to the "Upload Scheme of Study" section.*
2. *The Course Coordinator selects and uploads the Scheme of Study file.*
3. *The system validates the file format and content.*
4. *If the file is valid, the system records it.*
5. *The system updates course prerequisites and weightage.*
6. *The system confirms the successful upload.*

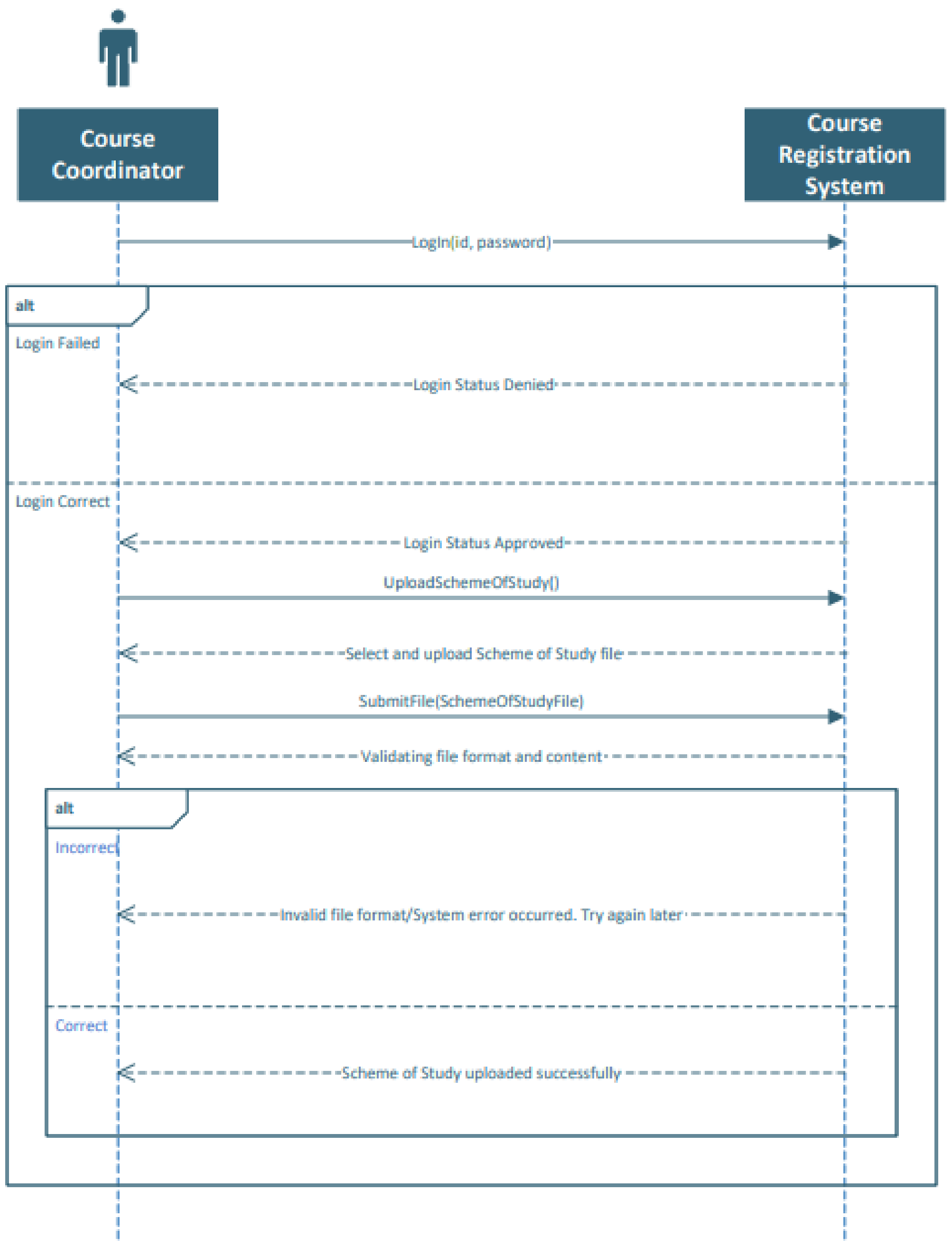
Alternative Scenarios:

- *File Validation Failed:*

  The uploaded file format is invalid.The system informs the Course Coordinator to upload a valid Scheme of Study.
- *Technical Failure:*

  A system error occurs while processing the upload.The system informs the Course Coordinator to try again later.

Sequence Diagram: Upload Scheme of Study


Course Coordinator
Course Registration System
LogIn(id, password)
alt
Login Failed
Login Status Denied
Login Correct
Login Status Approved
UploadSchemeOfStudy()
Select and upload Scheme of Study file
SubmitFile(SchemeOfStudyFile)
Validating file format and content
alt
Incorrect
Invalid file format/System error occurred. Try again later
Correct
Scheme of Study uploaded successfully

Use Case: View Student Progress

Primary Actor(s):

- *Course Coordinator*

Use Case Description:

This use case allows the Course Coordinator to monitor individual student performance, including passed and failed courses, pending prerequisites, and GPA.

Stakeholders and Interests:

- *Course Coordinator: Monitors student progress to provide academic guidance.*
- *Student: Receives academic feedback and course planning support.*
- *University Administration: Ensures accurate record-keeping and compliance with academic policies.*

Preconditions:

- *The Course Coordinator must be logged into the system.*
- *The student must be enrolled in at least one semester.*

Postconditions:

- *The student's academic progress is displayed.*
- *No modifications are made to student records (view-only action).*

Inputs:

- *StudentID*

Outputs:

- *AcademicProgressDetails (Passed Courses, Failed Courses, Pending Prerequisites, GPA)*

Main Success Scenario:

1. *The Course Coordinator navigates to the "View Student Progress" section.*
2. *The System retrieves the student's academic records (registrations, grades, prerequisites).*
3. *The System generates a summary of:*
   - *Passed Courses*
   - *Failed Courses*
   - *Pending Prerequisites*
   - *GPA*
4. *The System displays the academic progress summary to the Course Coordinator.*
5. *The Course Coordinator reviews the information for advising purposes.*

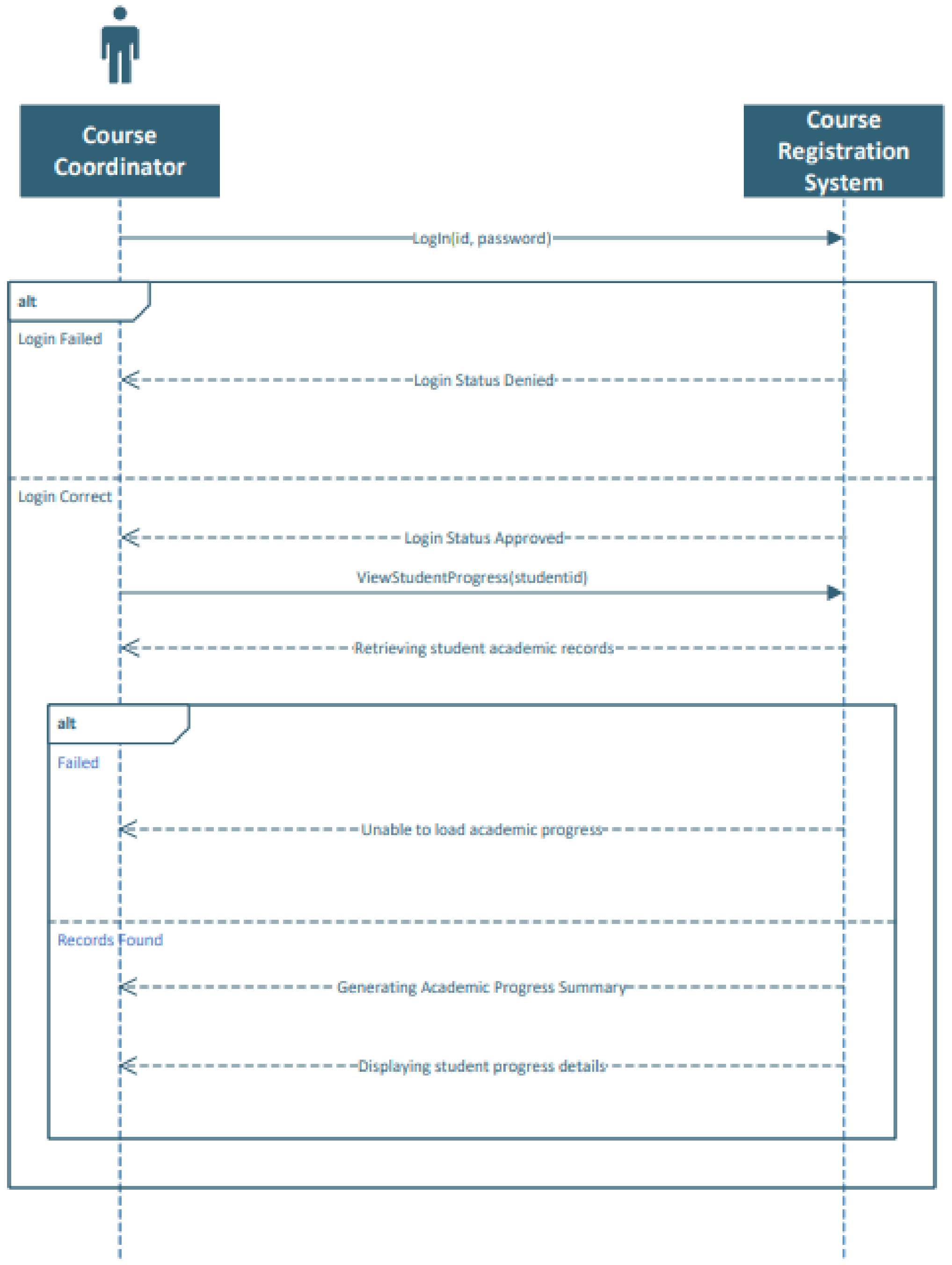
Alternative Scenarios:

- *No Academic Records Found:*

  If no academic records are available for the student, the system informs the Course Coordinator that no academic records are available for this student.
- *Technical Failure:*

  If a system error occurs while retrieving academic records, the system informs the Course Coordinator that it is unable to load academic progress and to try again later

Sequence Diagram: View Student Progress


Course Coordinator
Course Registration System
LogIn(id, password)
alt
Login Failed
Login Status Denied
Login Correct
Login Status Approved
ViewStudentProgress(studentid)
Retrieving student academic records
alt
Failed
Unable to load academic progress
Records Found
Generating Academic Progress Summary
Displaying student progress details

USE CASE DESCRIPTION & SYSTEM SEQUENCE DIAGRAM	**STUDENT COURSE REGISTRATION SYSTEM**
Created by	Zohaib Ahmed (04072213001)
Date Created	24-03-2025

Use Case: View Course Registrations

Primary Actor(s):

- *Timetable Coordinator*

Use Case Description:

This use case allows the timetable coordinator to view student course registrations to assist in class scheduling and avoid conflicts. The system retrieves and displays enrollment data for different courses.

Stakeholders and Interests:

- *Timetable Coordinator: Needs registration data to plan class schedules efficiently.*
- *Students: Benefit from properly managed class schedules with no conflicts.*
- *Course Coordinator: May need to adjust course offerings based on registration trends.*

Preconditions:

- *The timetable coordinator must be logged into the system.*
- *Course registration data must be available.*

Postconditions:

- *The system displays up-to-date student enrollment data for different courses.*

Inputs:

- *Semester*
- *Course Name*

Outputs:

- *List of students registered for the course.*

Main Success Scenario:

1. *The timetable coordinator navigates to the "View Course Registrations" section.*
2. *The coordinator enters search criteria (e.g., semester or specific courses).*
3. *The system retrieves and displays a list of students registered in each course.*
4. *The timetable coordinator reviews the data for scheduling adjustments.*

Alternative Scenarios:

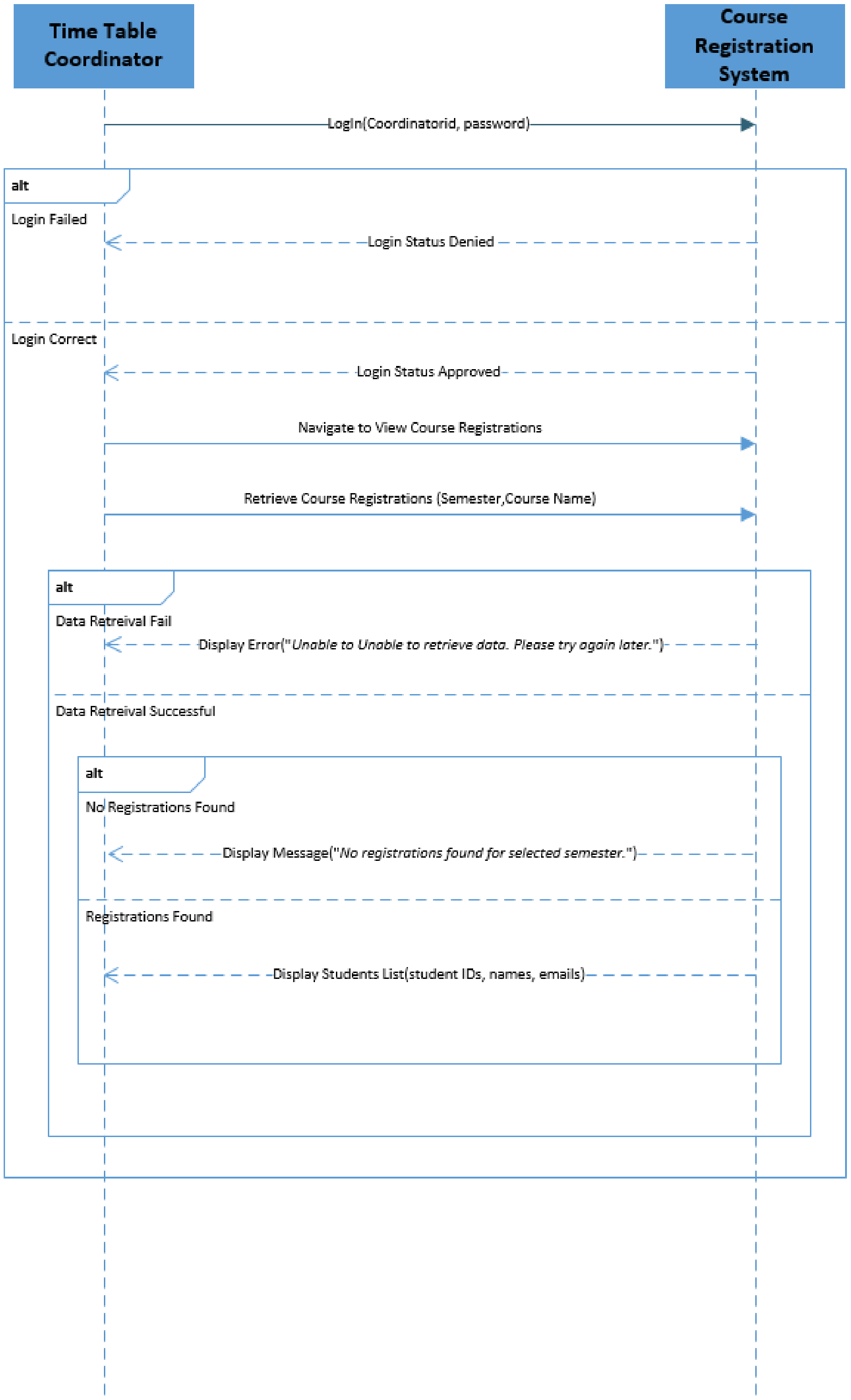
- *No Registrations Found*

The system notifies the coordinator that no registrations exist for the selected criteria.

- *Data Retrieval Error*

The system displays an error message, and the coordinator is advised to try again later.

Sequence Diagram: View Course Registrations


Time Table Coordinator
Course Registration System
LogIn(Coordinatorid, password)
alt
Login Failed
Login Status Denied
Login Correct
Login Status Approved
Navigate to View Course Registrations
Retrieve Course Registrations (Semester,Course Name)
alt
Data Retreival Fail
Display Error("Unable to Unable to retrieve data. Please try again later.")
Data Retreival Successful
alt
No Registrations Found
Display Message("No registrations found for selected semester.")
Registrations Found
Display Students List(student IDs, names, emails)

Use Case: Change Class Schedule

Primary Actor(s):

- *Timetable Coordinator*

Use Case Description:

This use case allows the Timetable Coordinator to modify the class schedule, such as changing time slots or instructors. The system ensures updates do not conflict with existing schedules.

Stakeholders and Interests:

- *Timetable Coordinator: Responsible for scheduling courses efficiently.*
- *Students: Need a clear schedule to plan their studies.*
- *Faculty: Require confirmed schedules to conduct classes.*
- *University Administration: Ensures proper resource allocation for lectures.*

Preconditions:

- *The Timetable Coordinator must be logged into the system.*
- *Course offerings for the semester must be finalized.*

Postconditions:

- *A new timetable is added to the system.*
- *The timetable is accessible to students and faculty.*

Inputs:

- *Course ID*
- *New Schedule Details (Time Slot, Instructor, Classroom)*

Outputs:

- *Schedule Update Status (Success or Conflict)*

Main Success Scenario:

1. *The Timetable Coordinator navigates to the "Change Class Schedule" section.*
2. *The Coordinator selects a course and enters the new schedule details.*
3. *The system checks for conflicts (e.g., overlapping classes, instructor availability).*
4. *If no conflicts, the system updates the schedule in the database.*
5. *The system confirms the schedule has been successfully changed.*

Alternative Scenarios:

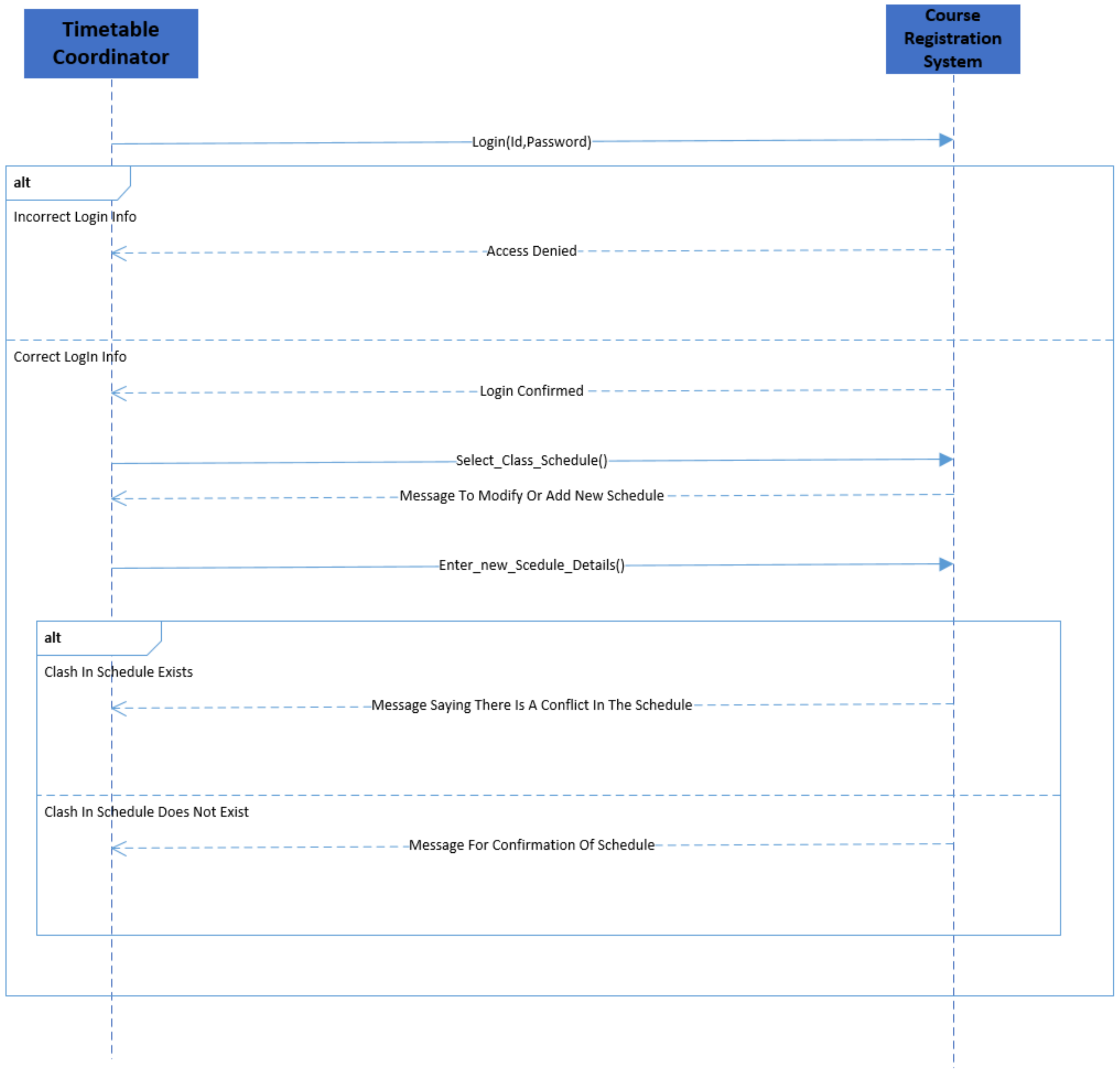
- *Room Unavailability:*

  The system notifies the Coordinator that a room is unavailable and suggests alternatives. The Coordinator selects a different room or time slot.

- *Instructor Scheduling Conflict:*

  The system detects a scheduling conflict for an instructor. The Coordinator revises the schedule to resolve the conflict.

Sequence Diagram: Change Class Schedule


Timetable Coordinator
Course Registration System
Login(Id,Password)
alt
Incorrect Login Info
Access Denied
Correct Login Info
Login Confirmed
Select_Class_Schedule()
Message To Modify Or Add New Schedule
Enter_new_Scedule_Details()
alt
Clash In Schedule Exists
Message Saying There Is A Conflict In The Schedule
Clash In Schedule Does Not Exist
Message For Confirmation Of Schedule

Use Case: Check Timetable Conflicts

Primary Actor(s):

- *Timetable Coordinator*

Use Case Description:

This use case allows the Timetable Coordinator to check for conflicts in the timetable, such as overlapping courses, room double-bookings, and instructor scheduling conflicts.

Stakeholders and Interests:

- *Timetable Coordinator: Ensures smooth scheduling without conflicts.*
- *Students: Avoids overlapping class schedules.*
- *Faculty: Ensures they do not have overlapping teaching assignments.*
- *University Administration: Maintains efficient resource allocation.*

Preconditions:

- *A timetable must exist in the system.*
- *Course and instructor details must be entered into the system.*

Postconditions:

- *Identified conflicts are flagged for resolution.*
- *The timetable is validated for conflicts before finalization.*

Inputs:

- *Semester timetable*
- *Instructor schedules*
- *Room availability*

Outputs:

- *List of conflicts (if any)*
- *Suggestions for resolving conflicts*

Main Success Scenario:

1. *The Timetable Coordinator navigates to the "Check Timetable Conflicts" section.*
2. *The system scans the timetable for: • Overlapping courses in the same room. • Instructors assigned to multiple courses at the same time. • Students enrolled in overlapping courses.*
3. *The system generates a report listing detected conflicts.*
4. *The Coordinator makes adjustments to resolve conflicts.*
5. *The system confirms a conflict-free timetable.*

Alternative Scenarios:

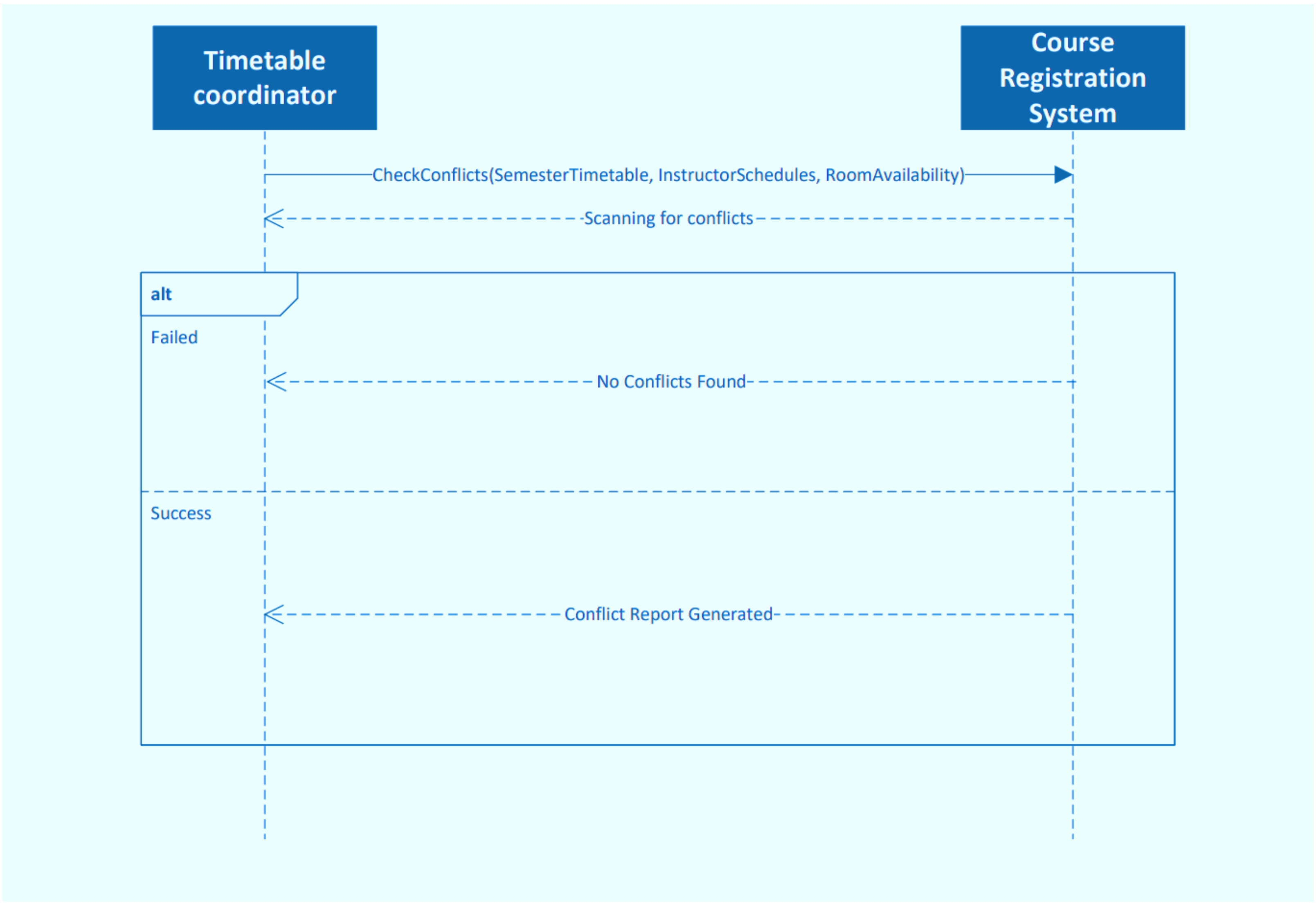
- *Multiple Conflicts Found:*

  The system displays multiple conflict warnings.The Coordinator prioritizes and resolves conflicts one by one.

- *Technical Error:*

  If the system fails to process the timetable check, the Coordinator retries after system maintenance.

Sequence Diagram: Check Timetable Conflicts


Timetable coordinator
Course Registration System
CheckConflicts(SemesterTimetable, InstructorSchedules, RoomAvailability)
Scanning for conflicts
alt
Failed
No Conflicts Found
Success
Conflict Report Generated

USE CASE DESCRIPTION & SYSTEM SEQUENCE DIAGRAM	**STUDENT COURSE REGISTRATION SYSTEM**
Created by	Muhammad Asad (04072312023)
Date Created	24-03-2025

Use Case: Edit Courses

Primary Actor(s):

- *Course Coordinator*

Use Case Description:

This use case allows the Course Coordinator to edit the details of a course, such as course name, prerequisites, or other relevant information. The system ensures that the changes are validated and updated in the database.

Stakeholders and Interests:

- *Course Coordinator: Wants to update course details accurately and efficiently.*
- *University Administration: Ensures that course information is up-to-date and accurate.*
- *Students: Rely on accurate course information for registration and academic planning.*
- *Timetable Coordinator: Needs updated course details to manage scheduling conflicts.*

Preconditions:

- *The Course Coordinator must be logged into the system.*
- *The course to be edited must exist in the system.*

Postconditions:

- *The course details are updated in the database.*
- *The system reflects the updated course information for all users.*

Inputs:

- *CourseID*
- *Updated course details (e.g., course name, prerequisites, etc.)*

Outputs:

- *EditStatus (Confirmed or Denied)*
- *UpdatedCourseDetails.*

Main Success Scenario:

1. *The Course Coordinator navigates to the course management section.*
2. *The Course Coordinator selects the course to be edited by entering the CourseID.*
3. *The system retrieves the current course details.*
4. *The Course Coordinator updates the course details (e.g., course name, prerequisites, etc.).*
5. *The system validates the updated course details.*
6. *If the validation is successful, the system updates the course details in the database.*

Alternative Scenarios:

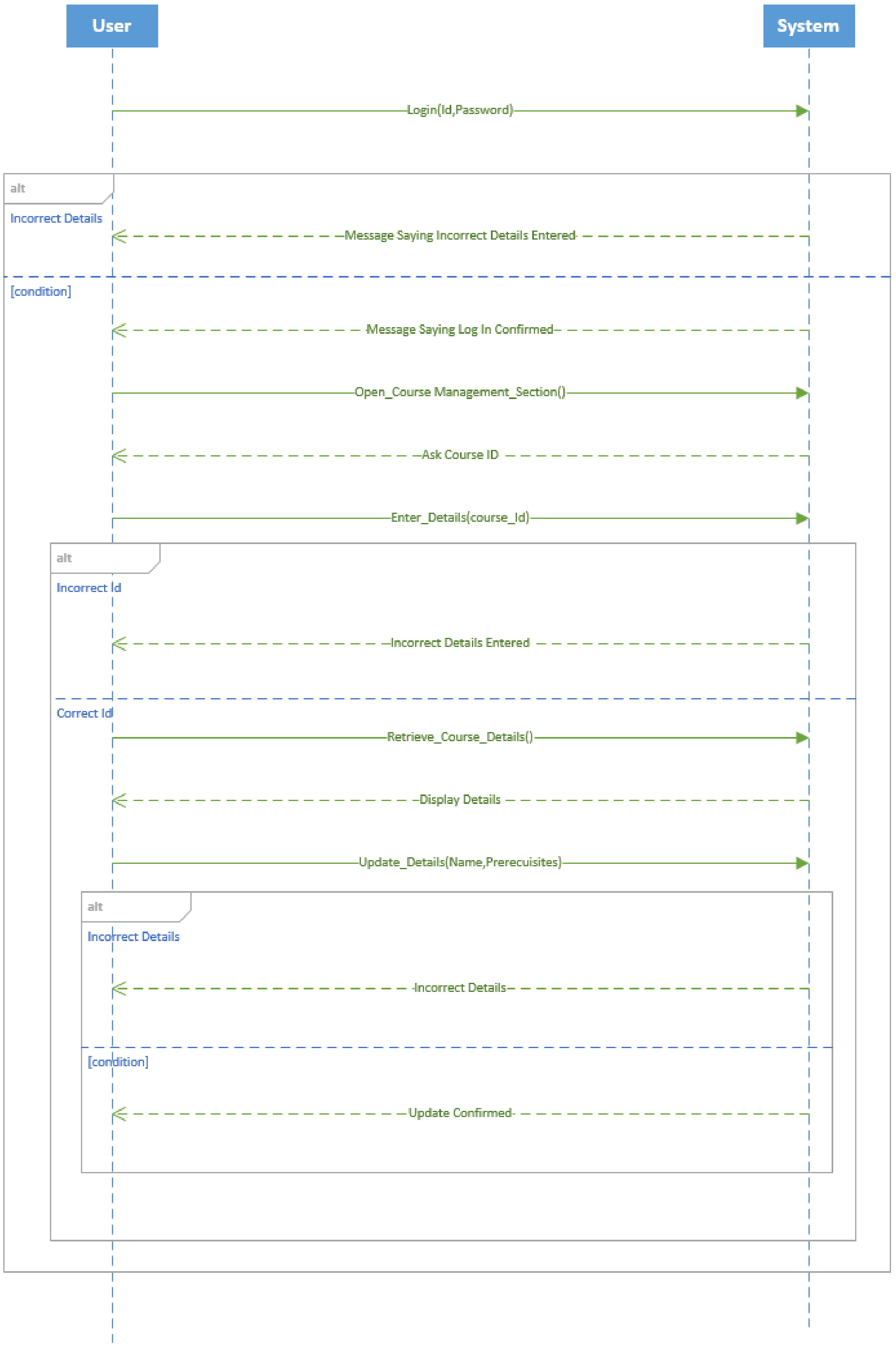
- *Validation Failure:*

  If the updated course details fail validation (e.g., invalid prerequisites or missing information), the system notifies the Course Coordinator. The Course Coordinator is prompted to correct the details and resubmit.

- *Technical Failure:*

  If the system crashes or encounters an error during the update process, the Course Coordinator cannot complete the edit. The system logs the failure, and the Course Coordinator is advised to retry later.

Sequence Diagram: Edit Courses



Use Case: Offer Course

Primary Actor(s):

- *Course Coordinator*

Use Case Description:

This use case allows the Course Coordinator to offer new courses for an academic semester. The system ensures that course details are validated and stored in the database.

Stakeholders and Interests:

- *Course Coordinator: Wants to offer new courses for students.*
- *Students: Need courses to be available for registration.*
- *University Administration: Ensures proper academic offerings.*

Preconditions:

- *The Course Coordinator must be logged into the system.*
- *The new course details (name, code, credits, prerequisites) must be available.*

Postconditions:

- • *The new course is added.*
- • *The course becomes available for student registration.*

Inputs:

- *Course Name*
- *Course Code*
- *Credit Hours*
- *Prerequisites (if any)*

Outputs:

Course Offering Status (Success or Failure)

Main Success Scenario:

1. *The Course Coordinator navigates to the "Offer Course" section.*
2. *The Course Coordinator enters the course details.*
3. *The system validates the details (e.g., unique course code, credit hour limits).*
4. *If valid, the system stores the course.*
5. *The system confirms the course has been successfully added.*

Alternative Scenarios:

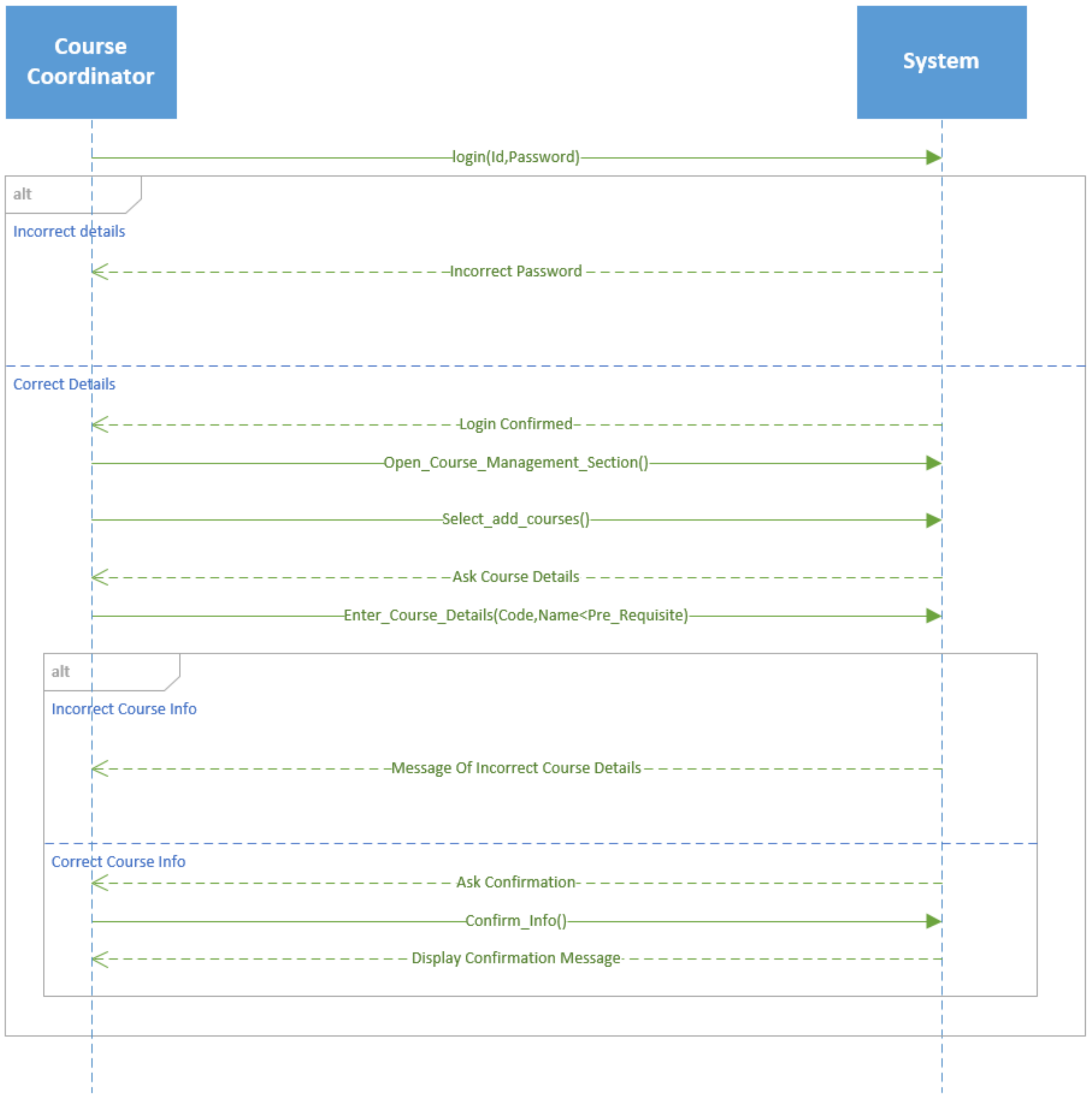
- *Validation Failure:*

  If course details are incorrect, the system prompts for corrections.

- *Technical Failure:*

  If the system crashes, the course cannot be added, and the Coordinator is advised to try later.

Sequence Diagram: Offer Course



Course Coordinator
System
login(Id,Password)
alt
Incorrect details
Incorrect Password
Correct Details
Login Confirmed
Open_Course_Management_Section()
Select_add_courses()
Ask Course Details
Enter_Course_Details(Code,Name<Pre_Requisite)
alt
Incorrect Course Info
Message Of Incorrect Course Details
Correct Course Info
Ask Confirmation
Confirm_Info()
Display Confirmation Message