College Enrollment System

Software Requirements Specification

Revision History

Date	Revision	Description	Author
02/23/2025	1.0	Initial Version.	Aakkash Muthukumar
02/24/2025	1.1	Finished Purpose Section.	Aakkash Muthukumar
02/25/2025	1.2	Filled in references page in purpose section.	Felipe Gutierrez
02/28/2025	1.2.1	Removed all the wrong references.	Aakkash Muthukumar
02/28/2025	1.3	Completed the requirements specification document.	Aakkash Muthukumar
03/02/2025	1.3.1	Started the Use Case Specification Document.	Aakkash Muthukumar
3/02/2025	1.4	Added to overview for client-side operation overview.	Christian Ramos
3/02/2025	1.5	Added to functional requirements and student module requirements.	Christian Ramos
03/03/2025	1.6	Fixed the document formatting and font styles. Also, I added some requirements.	Svein Quintos

Table Of Contents

1. Purpose 4
    1.1. Scope 4
    The College Enrollment System is a Java-based desktop application that facilitates student enrollment and course management across multiple universities. It allows administrators to create and manage course schedules, while students can enroll, drop, and manage their course selections. The system also supports prerequisite enforcement, waitlists, and reporting. 4
    1.2. Definitions, Acronyms, Abbreviations 4
    1.3. References 4
    1.4. Overview 4
2. Overall Description 5
    2.1. Product Perspective 5
    2.2. Product Architecture 5
    2.3. Product Functionality/Features 5
    2.4. Constraints 5
    2.5. Assumptions and Dependencies 6
3. Specific Requirements 7
    3.1. Functional Requirements 7
        3.1.1. Common Requirements: 7
        3.1.2. Administrator Module Requirements: 7
        3.1.3. Student Module Requirements: 7
        3.1.4. Reporting Module Requirements: 7
    3.2. External Interface Requirements 7
    3.3. Internal Interface Requirements 7
4. Non-Functional Requirements 8
    4.1. Security and Privacy Requirements 8
    4.2. Environmental Requirements 8
    4.3. Performance Requirements 8
5. Use Case Specification Document 9
    5.1. Student Registration 9
    5.2. Course Enrollment 10
    5.3. Manage Student Profile 11
    5.4. Payment Processing 12
    5.5. Generate Enrollment Reports 13
6. UML Use Case Diagrams Document
Use Case 1: Student Registration

Use Case 2: Course Enrollment

Use Case 3: Manage Student Profile

Use Case 4: Payment Processing

Use Case 5: Generate Enrollement Reports..... 15
7. Class Diagram:..... 18
8. Sequence Diagrams..... 19

1. Purpose

1.1. Scope

The College Enrollment System is a Java-based desktop application that facilitates student enrollment and course management across multiple universities. It allows administrators to create and manage course schedules, while students can enroll, drop, and manage their course selections. The system also supports prerequisite enforcement, waitlists, and reporting.

1.2. Definitions, Acronyms, Abbreviations

1.2.1. GUI: Graphical User Interface

1.2.2. TCP/IP: Transmission Control Protocol/Internet Protocol

1.2.3. Admin: University personnel responsible for managing course offerings

1.2.4. Student: University member enrolling in courses

1.2.5. SRS: Software Requirements Specification

1.3. References

1.4. Overview

1.4.1. This document outlines the functional and non-functional requirements of the College Enrollment System.

2. Overall Description

2.1. Product Perspective

2.1.1. The system operates a client-server application with a GUI-based Java application. The server handles requests related to course enrollment, course scheduling, and reporting. No database or external frameworks will be used. Client is operated by the student or an administrator to enroll, drop, or change courses based on prerequisites for courses being met. Admins can add new courses to catalog and put holds on student accounts.

2.2. Product Architecture

2.2.1. Client Application: Allows students and admins to interact with the system

2.2.2. Server Application: This handles course management, enrollment, and reporting over TCP/IP

2.3. Product Functionality/Features

2.3.1. Admins can create, update, delete, and view courses

2.3.2. Students can browse available courses and enroll if space allows

2.3.3. Prerequisites are enforced before enrollment

2.3.4. If a course is full, students can join the waitlist

2.3.5. Students can add a course within a specified deadline

2.3.6. Students can drop a course within a specified deadline

2.3.7. Students can view their enrolled courses with schedule details

2.3.8. Admins can generate reports on course enrollment, waitlisted students, and available seats

2.3.9. The system supports multiple universities

2.3.10. Students have a course unit cap

2.3.11. Unique IDs will be assigned to users to differentiate those with similar names

2.3.12. Students can be put on hold if a balance is owed on the account

2.4. Constraints

2.4.1. No web or HTML components

2.4.2. No databases or JSON file formats

2.4.3. Operates strictly over TCP/IP

2.4.4. Written in Java with a GUI-based interface

2.5. Assumptions and Dependencies

2.5.1. Users have network access to connect to the server

2.5.2. The system assumes admins maintain accurate course prerequisite data

2.5.3. Each university has unique course offerings and policies

2.5.4. Students only belong to a single university

3. Specific Requirements

3.1. Functional Requirements

3.1.1. Common Requirements:

3.1.1.1. Unique user IDs for students and admins

3.1.1.2. Secure authentication and role-based access

3.1.2. Administrator Module Requirements:

3.1.2.1. Create, update, delete, and view courses

3.1.2.2. Set course prerequisites and enrollment limits

3.1.2.3. Generate reports on student enrollment and waitlists

3.1.2.4. Manage multiple universities within the system

3.1.3. Student Module Requirements:

3.1.3.1. Browse available courses

3.1.3.2. Enroll in courses if prerequisites are met and space is available

3.1.3.3. Drop courses within the allowed deadline

3.1.3.4. Join a waitlist for full courses

3.1.3.5. View current course enrollment and waitlist status

3.1.3.6. View holds (ie. balance owed)

3.1.4. Reporting Module Requirements:

3.1.4.1. Generate reports on:

3.1.4.1.1. Course enrollments

3.1.4.1.2. Students on waitlists

3.1.4.1.3. Available seats in each course

3.2. External Interface Requirements

3.2.1. The GUI should be intuitive and user-friendly

3.2.2. The system should provide clear error messages for failed enrollments

3.3. Internal Interface Requirements

3.3.1. Client-server communication via TCP/IP

3.3.2. Efficient handling of multiple user requests concurrently

4. Non-Functional Requirements

4.1. Security and Privacy Requirements

4.1.1. Secure login authentication for students and admins

4.1.2. Role-based access control to prevent unauthorized actions

4.2. Environmental Requirements

4.2.1. Must run on standard Windows/macOS/Linux machines with Java installed.

4.2.2. Requires stable network connectivity.

4.3. Performance Requirements

4.3.1. Must handle multiple concurrent enrollments

4.3.2. System responses should not exceed 2 seconds under normal load conditions

4.3.3. Enrollment and waitlist updates must be reflected in real time

5. Use Case Specification Document

5.1. Student Registration

Use Case ID: UC01

Relevant Requirements:

- Requirements Document ID: RD001

Primary Actor:

- Student

Pre-conditions:

- The student has access to the enrollment portal.
- The system is operational.

Post-conditions:

- The student is registered in the system.
- The student receives a confirmation of registration.

Basic Flow or Main Scenario:

1. The student accesses the enrollment system.
2. The system prompts the student to enter personal details.
3. The student fills out and submits the registration form.
4. The system validates the provided information.
5. Upon successful validation, the system registers the student.
6. The system sends a confirmation message to the student. Extensions or Alternate Flows:

- If validation fails, an error message is displayed, and the student must correct the errors.

Exceptions:

- If the system encounters an error, the registration process is halted, and an error alert is generated. Related Use Cases:

- UC02: Course Enrollment

- UC03: Manage Student Profile

5.2. Course Enrollment

Use Case ID: UC02

Relevant Requirements:

- Requirements Document ID: RD002

Primary Actor:

- Student

Pre-conditions:

- The student is registered in the system.
- Course registration is open.

Post-conditions:

- The student is enrolled in selected courses.
- Enrollment confirmation is sent to the student.

Basic Flow or Main Scenario:

1. The student logs into the enrollment system.
2. The system displays available courses.
3. The student selects desired courses.
4. The system checks course availability and prerequisites.
5. If valid, the system enrolls the student.
6. The system sends confirmation to the student. **Extensions or Alternate Flows:**

- If the course is full, the student is placed on a waitlist. **Exceptions:**
- If the system encounters an error, enrollment fails, and an error alert is generated.

Related Use Cases:

- UC01: Student Registration

- UC04: Payment Processing

5.3. Manage Student Profile

Use Case ID: UC03

Relevant Requirements:

- Requirements Document ID: RD003

Primary Actor:

- Student

Pre-conditions:

- The student is logged into the system.

Post-conditions:

- Student profile is updated in the system.

Basic Flow or Main Scenario:

1. The student logs into the system.
2. The system displays the student profile.
3. The student edits personal details.
4. The system validates and updates the profile.
5. The system sends a confirmation of changes. **Extensions or Alternate Flows:**

- If validation fails, an error message is displayed. **Exceptions:**
- If the system encounters an error, profile updates are halted. **Related Use Cases:**
- UC01: Student Registration

5.4. Payment Processing

Use Case ID: UC04

Relevant Requirements:

- Requirements Document ID: RD004

Primary Actor:

- Student
- Payment System

Pre-conditions:

- The student has enrolled in courses.
- The payment system is operational. **Post-conditions:**
- Payment is successfully processed.
- A payment receipt is generated.

Basic Flow or Main Scenario:

1. The student selects the payment option.
2. The system redirects to the payment portal.
3. The student enters payment details.
4. The system processes the payment.
5. Upon success, the system generates a receipt. **Extensions or Alternate Flows:**

- If the payment fails, the student is prompted to retry or use another method. **Exceptions:**
- If the payment gateway is down, an error alert is generated. **Related Use Cases:**
- UC02: Course Enrollment

5.5. Generate Enrollment Reports

Use Case ID: UC05

Relevant Requirements:

- Requirements Document ID: RD005

Primary Actor:

- Administrator

Pre-conditions:

- The administrator has valid credentials.
- Enrollment data is available.

Post-conditions:

- A report is generated and displayed.

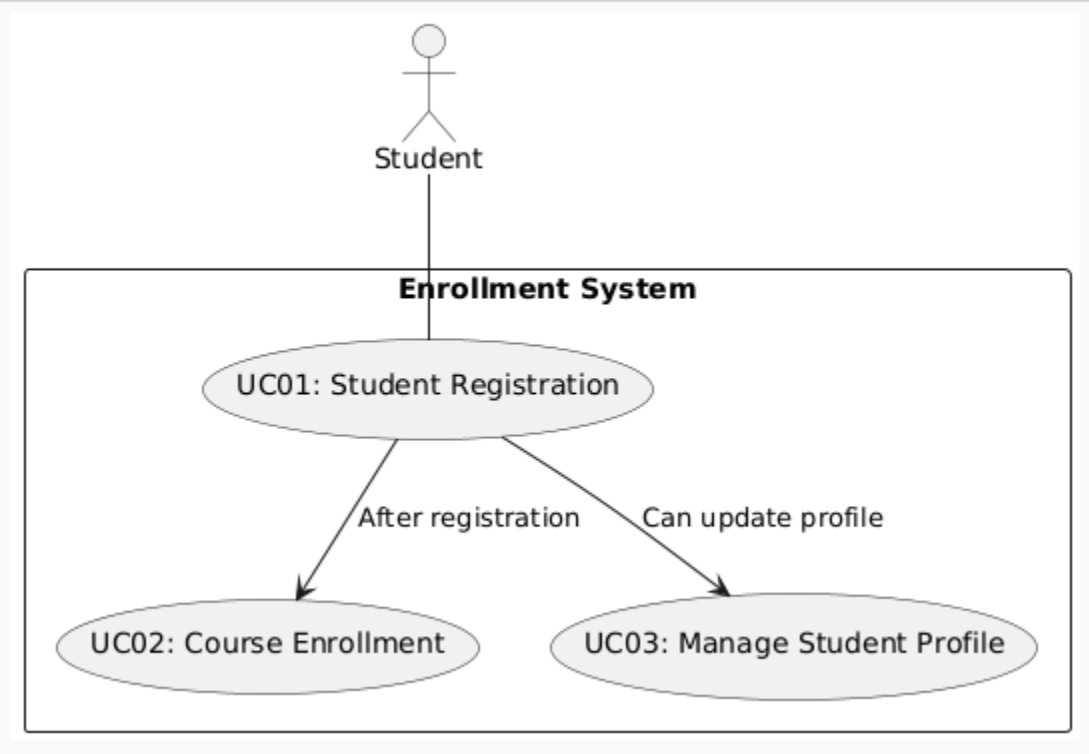
Basic Flow or Main Scenario:

1. The administrator logs into the system.
2. The system prompts the administrator to select report criteria.
3. The administrator enters the criteria and submits the request.
4. The system retrieves and processes the relevant data.
5. The system generates the report.
6. The administrator can download or print the report. **Extensions or Alternate Flows:**

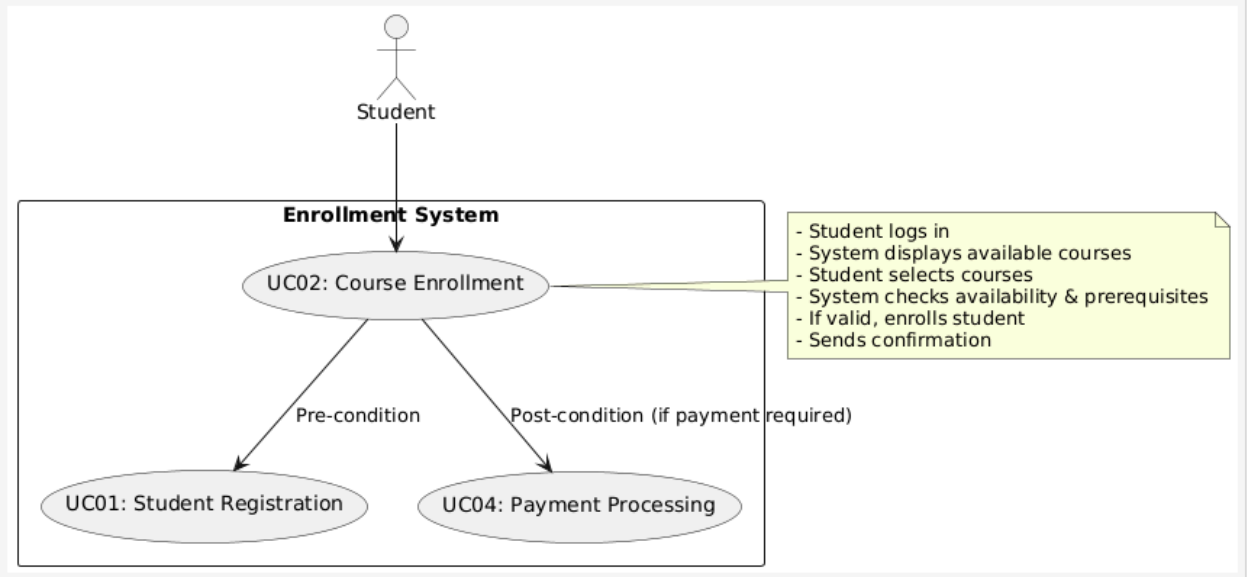
- If no data matches the criteria, an empty report is generated. **Exceptions:**
- If the system encounters an error, the report generation fails. **Related Use Cases:**
- UC01: Student Registration
- UC02: Course Enrollment

6. UML Use Case Diagrams Document

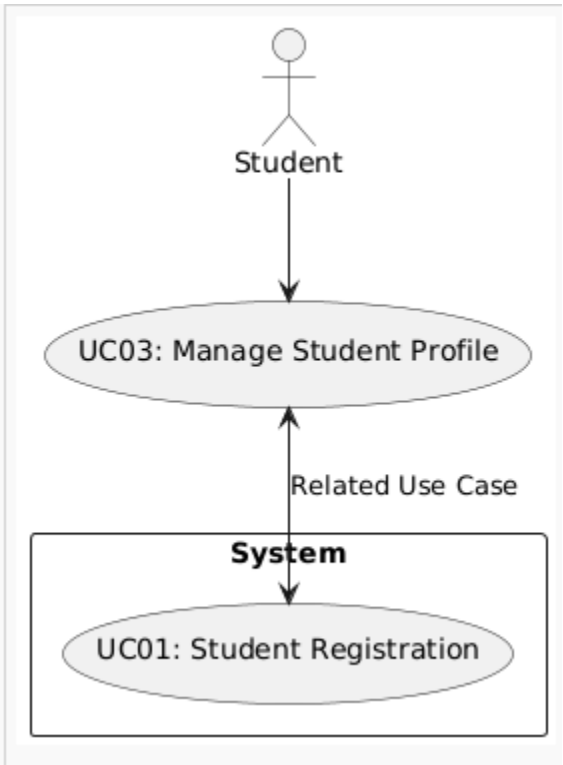
Use Case 1: Student Registration



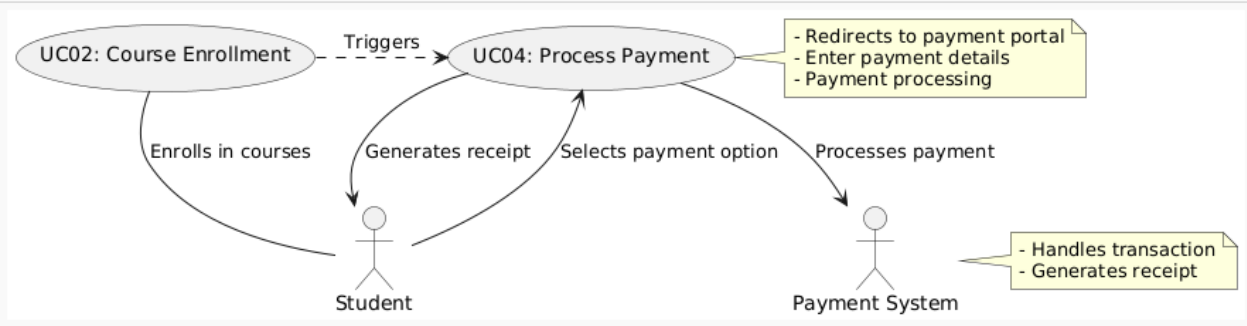
Use Case 2: Course Enrollment



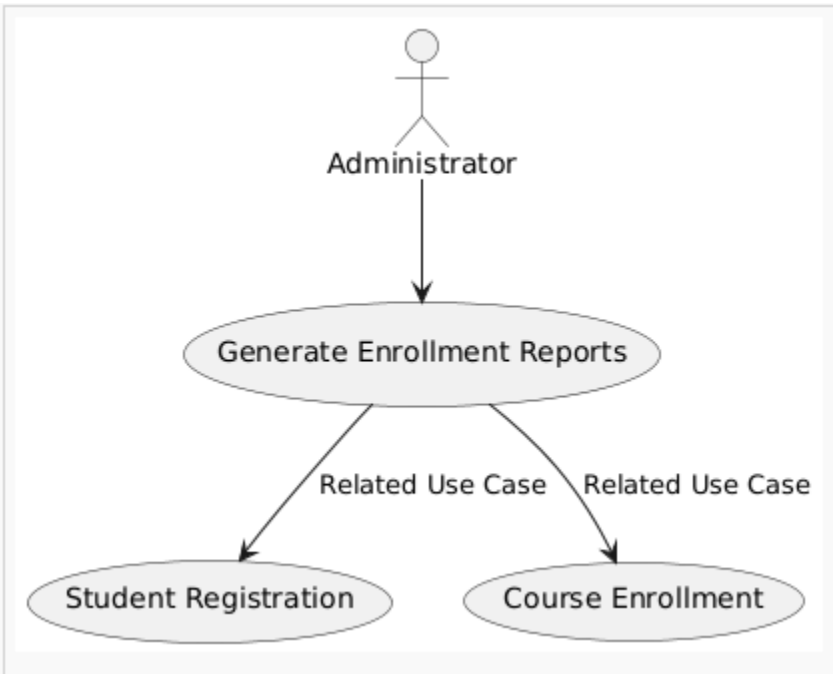
Use Case 3: Manage Student Profile



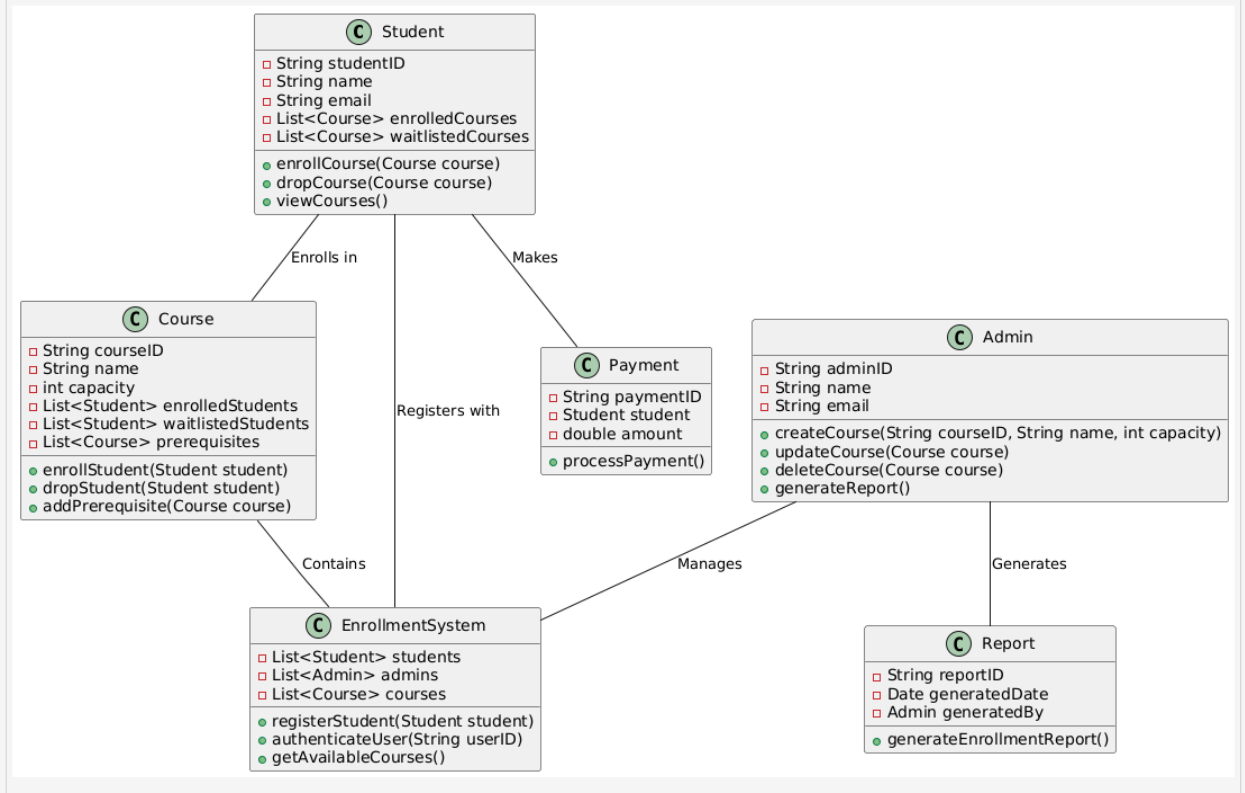
Use Case 4: Payment Processing



Use Case 5: Generate Enrollement Reports

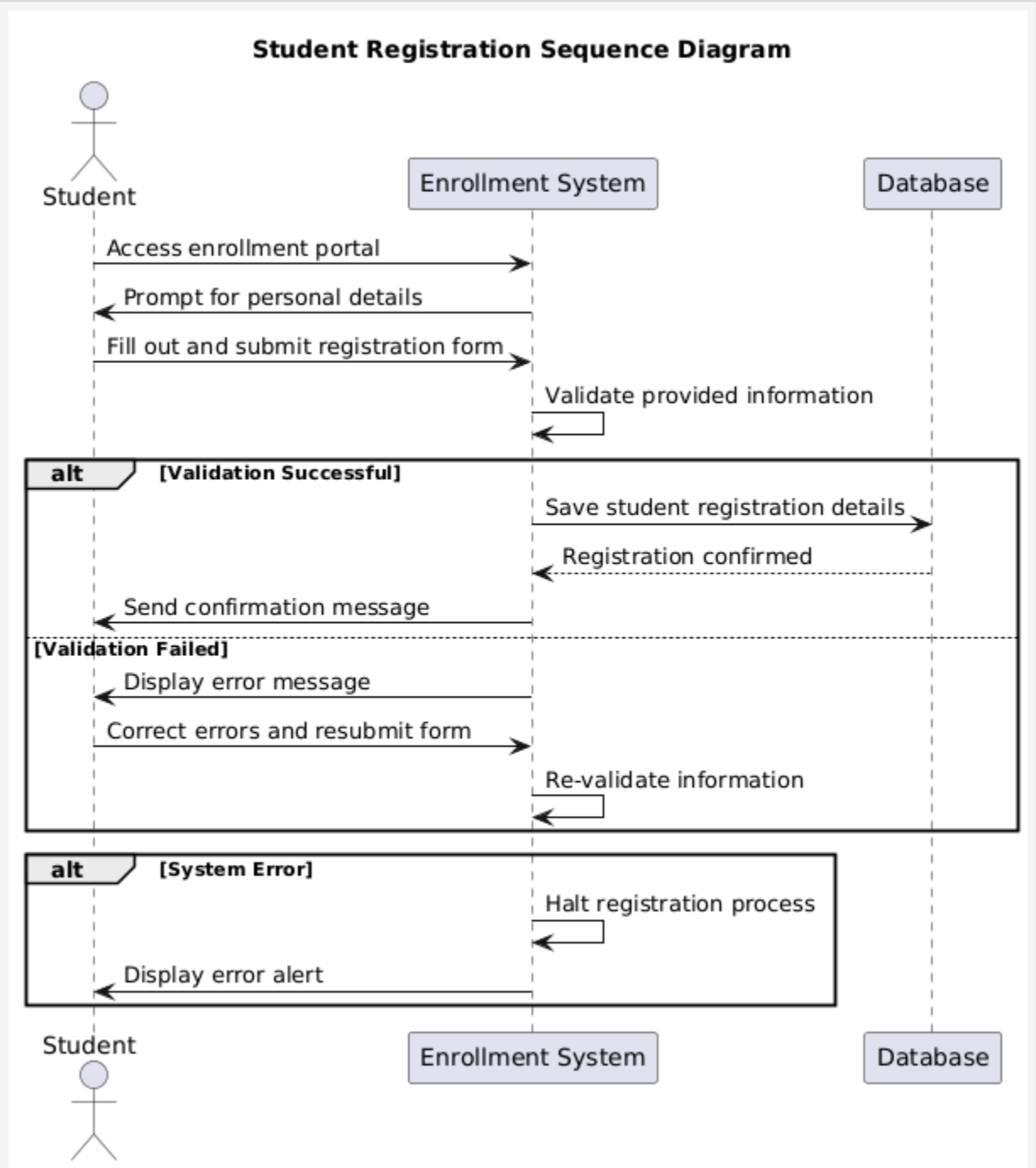


7. Class Diagram:

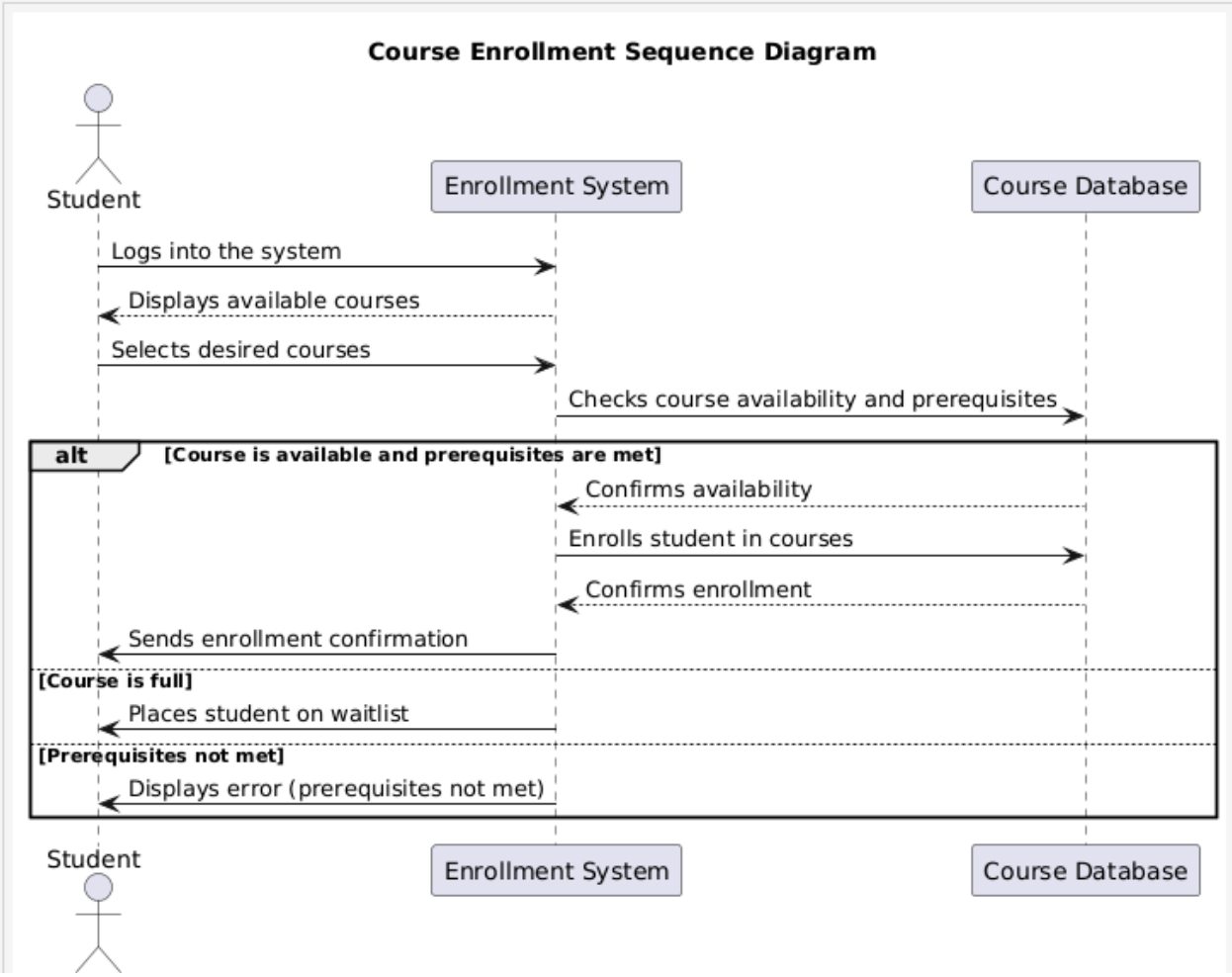


8. Sequence Diagrams

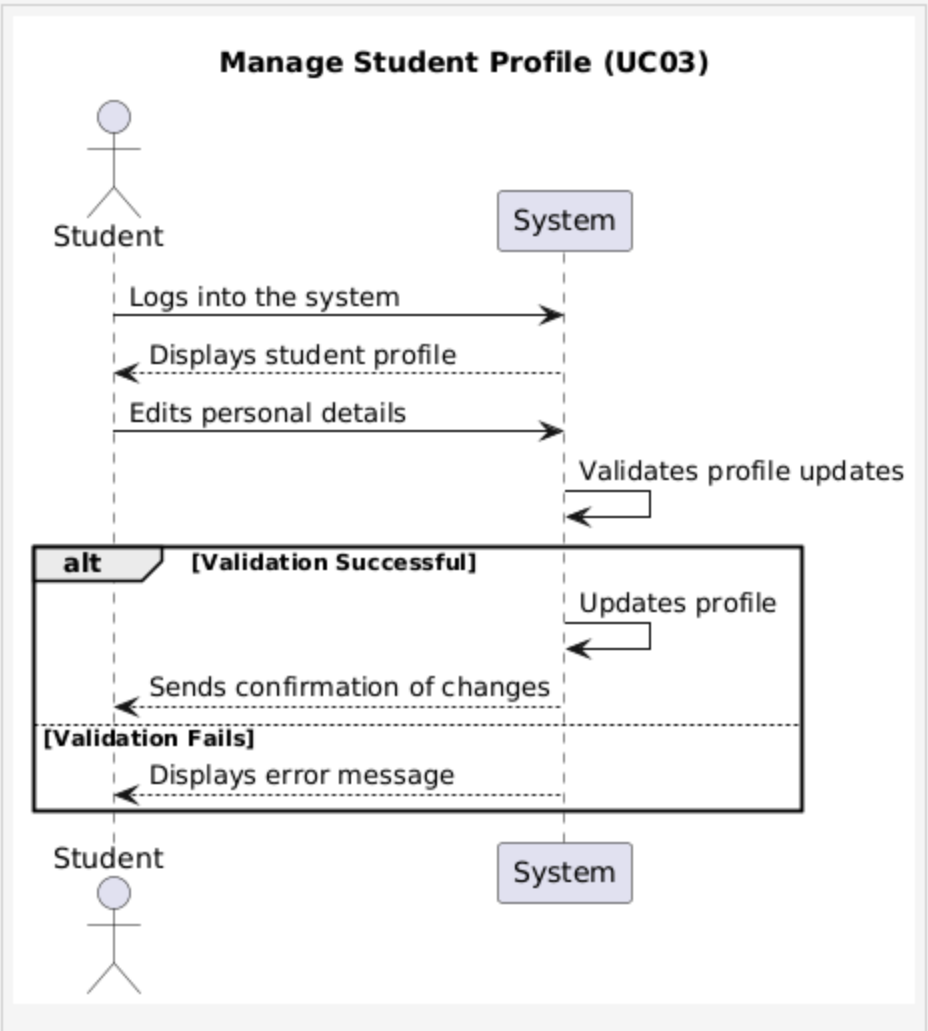
Sequence Diagram 1: Student Registration



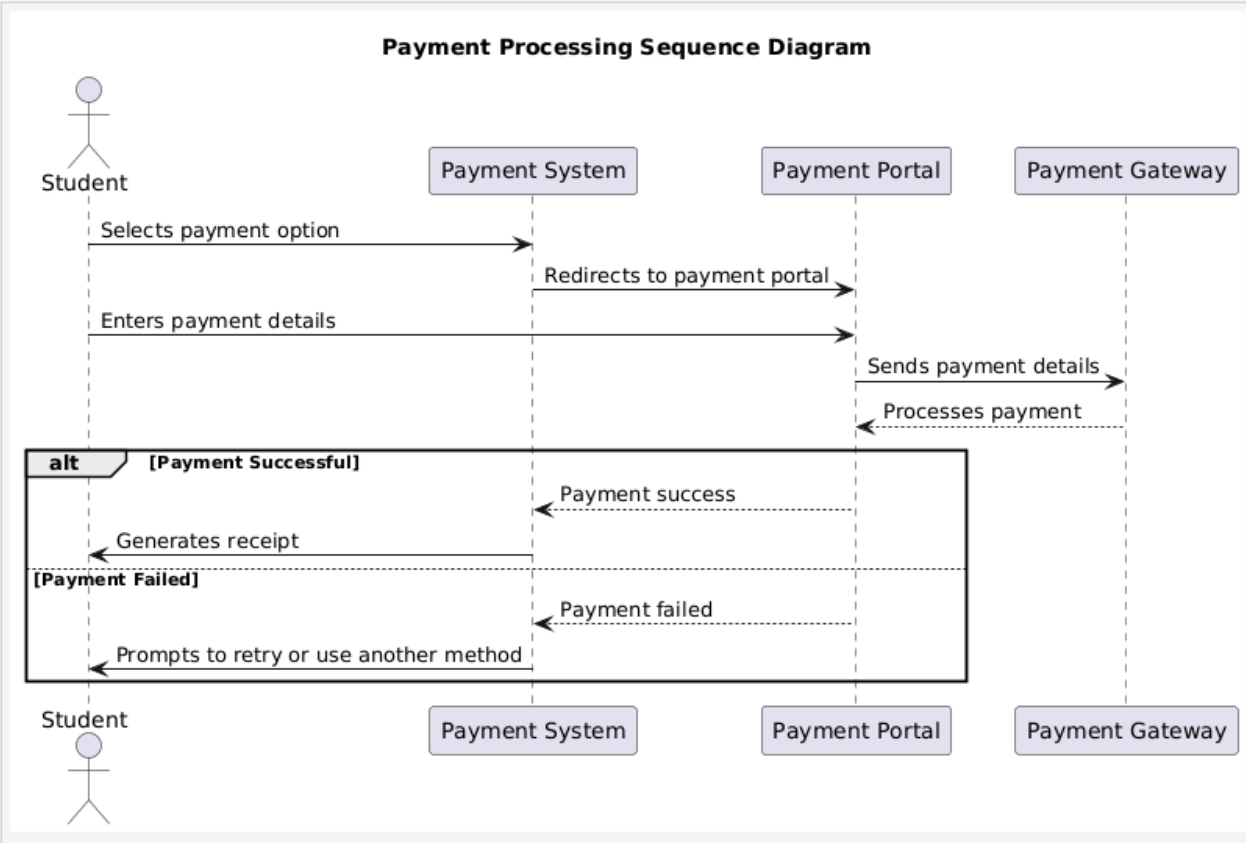
Sequence 2: Course Enrollment



Use Case 3: Manage Student Profile


Manage Student Profile (UC03)
Student
System
Logs into the system
Displays student profile
Edits personal details
Validates profile updates
alt
[Validation Successful]
Updates profile
Sends confirmation of changes
[Validation Fails]
Displays error message
Student
System

Sequence Diagram 4: Payment Processing



Sequence Diagram 5: Generate Enrollement Reports