Modern E-Learning Platform using Strapi and Nextjs

{Solely making it in PHP as discussed with the supervisor}

**Design Document**

Version 1.0



**Group Id:** F24PROJECTD409E

**Supervisor Name :** Amjad Iqbal Khan

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date (dd/mm/yyyy)** | **Version** | **Description** | **Author** |
| 26/02/2025 | 1.0 | ***"Learning Management System in PHP"*** is a web-based platform designed to facilitate online education by connecting students, instructors, and administrators. The system provides secure authentication for all users, allowing students to browse and enroll in courses, take quizzes, and submit reviews. Instructors can create and manage courses, add assessments, and track enrolled students. The admin oversees the entire system, managing course approvals, enrollments, user accounts, and financial transactions. With an integrated payment gateway, seamless enrollment, and a structured review system, this LMS ensures a comprehensive and interactive learning experience. | BC190203051 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Table of Contents**

1. Introduction of Design Document
2. Entity Relationship Diagram (ERD)
3. Sequence Diagrams
4. Architecture Design Diagram
5. Class Diagram
6. Database Design
7. Interface Design
8. Test Cases

**Design Document**

***1. Introduction to Design Document:***

After completing the Software Requirement Specification (SRS) document, the next phase in the software development process is the **Software Design Document**. This document provides a detailed overview of how the Learning Management System (LMS) in PHP will be developed by software developers or development teams. It ensures that design specifications are clear, understandable, and align with the project requirements.

A well-structured **Software Design Document (SDD)** helps all stakeholders understand what is possible, what is not, and which systems need to be developed. It ensures that the final product meets agreed-upon requirements and maintains consistency throughout development. This document also plays a crucial role in system architecture planning, future system expansion, and maintainability.

This **Design Document** is a detailed guide for building the **Modern E-Learning Platform**. It explains how different parts of the system work together and provides diagrams to show the system’s structure and flow.

The platform is developed using **PHP** and will have important features like **managing content, user logins, secure payments, and course organization**. This document includes key design elements such as **Entity Relationship Diagrams (ERD), Sequence Diagrams, System Architecture, Class Diagrams, Database Design, User Interface Design, and Test Cases** to ensure a smooth development process.

By following this document, the development team can create an **easy-to-use, efficient, and secure e-learning system** that supports online education and provides a great user experience.

**Design Document Includes:**

**1. Entity Relationship Diagram (ERD)**

The **ERD** for the LMS represents the logical structure of the database by defining entities, attributes, and relationships. It helps visualize how different entities, such as **students, instructors, courses, payments, enrollments, and assessments**, interact within the system. ERDs use:

* **Rectangles** to represent entities
* **Ellipses** to define attributes
* **Diamonds** to represent relationships

**2. Sequence Diagrams**

The **Sequence Diagrams** illustrate how various entities interact within the LMS for different functionalities. These diagrams define interactions **step-by-step**, ensuring clarity on system behavior. The LMS will have separate sequence diagrams for **students, instructors, and administrators**, covering essential functionalities such as **authentication, course enrollment, assessments, reviews, and payment processing**.

**3. Architecture Design Diagram**

An **Architecture Design Diagram** presents a high-level view of the LMS system, showing the relationships, constraints, and boundaries between different software components. It helps in understanding the **system’s layers, such as the frontend, backend, database, and external APIs (e.g., payment gateway)**.

**4. Class Diagram**

The **Class Diagram** describes the characteristics and behaviors of different classes within the LMS. It includes:

* **Classes** such as Student, Instructor, Admin, Course, Payment, Enrollment, and Assessment
* **Attributes and methods** of each class
* **Relationships between classes**, such as inheritance and associations

**5. Database Design**

The **Database Design** outlines the structure of tables, fields, relationships, and constraints. A well-designed database ensures **data consistency, integrity, and efficiency**. The LMS database includes tables such as:

* **Users** (Students, Instructors, Admins)
* **Courses** (Details, Categories, Status)
* **Enrollments** (Student-Course Relationship)
* **Assessments** (Quizzes, Submissions, Scores)
* **Payments** (Transactions, Status, Method)

**6. Interface Design**

The **User Interface (UI) Design** defines how users will interact with the LMS. It ensures a smooth user experience (UX) by providing:

* **A responsive and user-friendly dashboard** for students, instructors, and admins
* **Course browsing, enrollment, and progress tracking features**
* **An intuitive quiz and assessment system**
* **A secure payment processing interface**

**7. Test Cases**

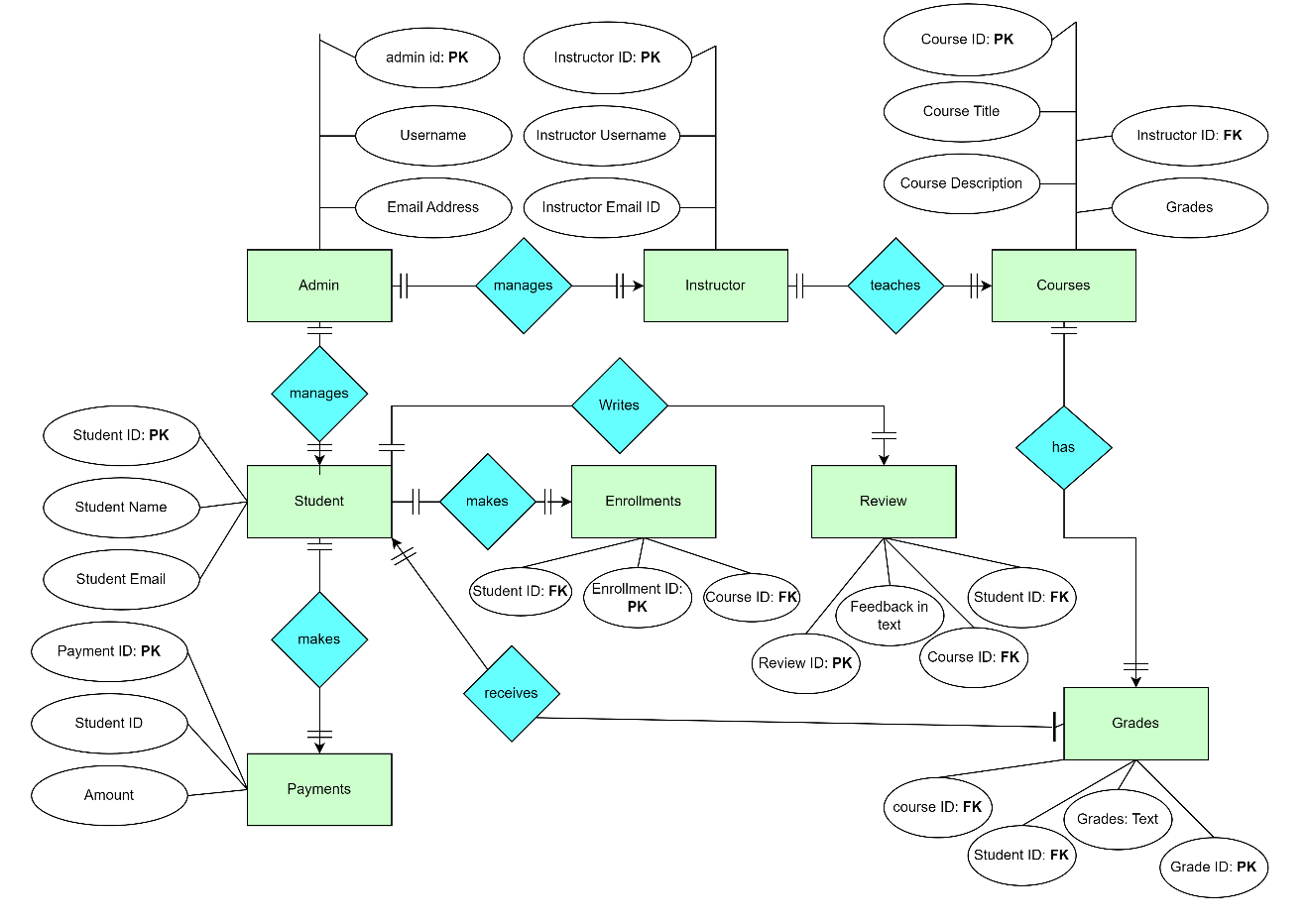
**Test Cases** document the testing strategy used to ensure the system meets the defined requirements. Each test case includes:

* **Test Data**
* **Preconditions**
* **Expected Results**
* **Post Condition/ Actions**
* **Priority**

Test cases cover critical areas such as **user authentication, course management, enrollment flow, assessment submission, and payment processing**.

***2. Entity Relationship Diagram (ERD)***

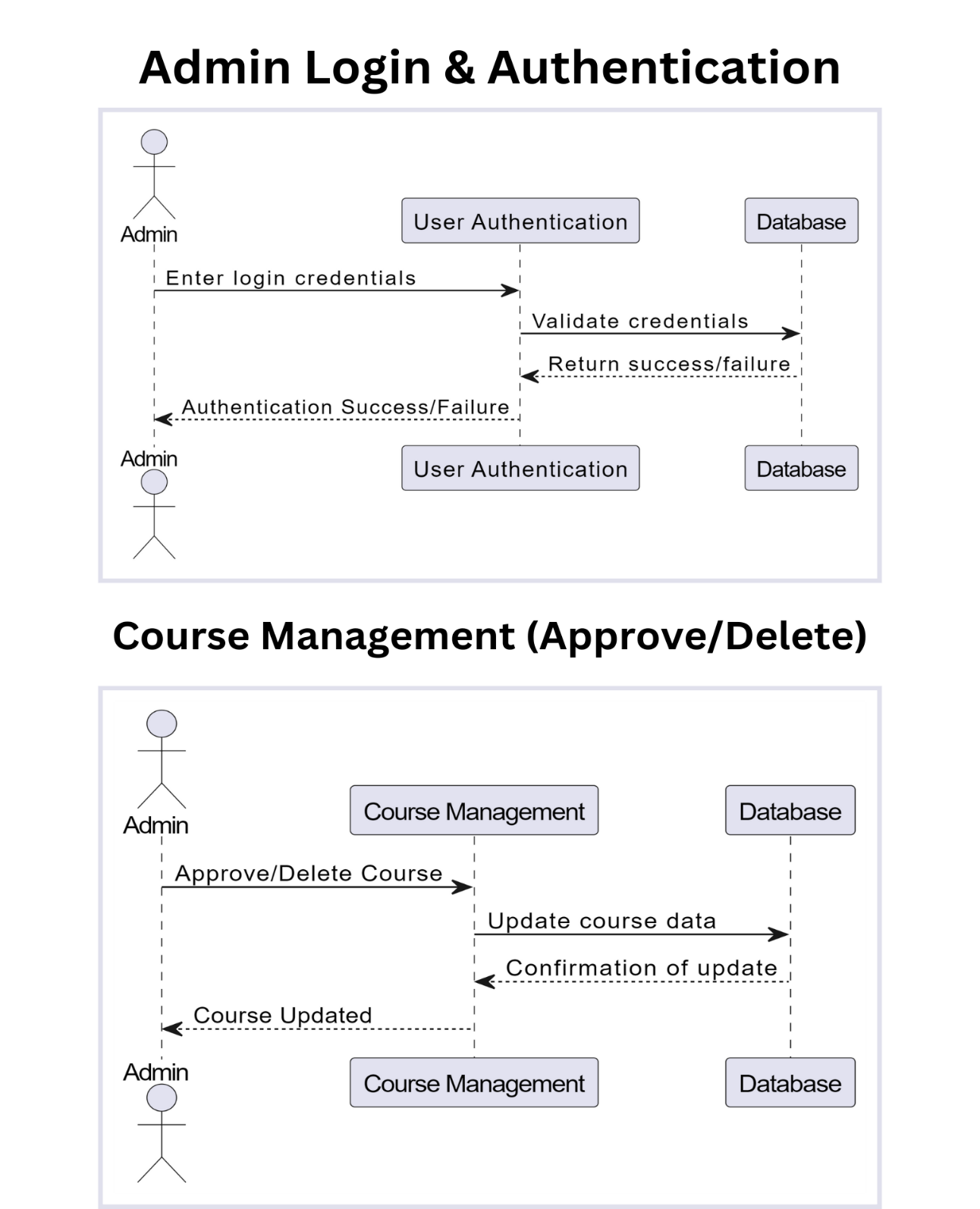
{ER diagrams made with online diagram making tool **Draw.io}**

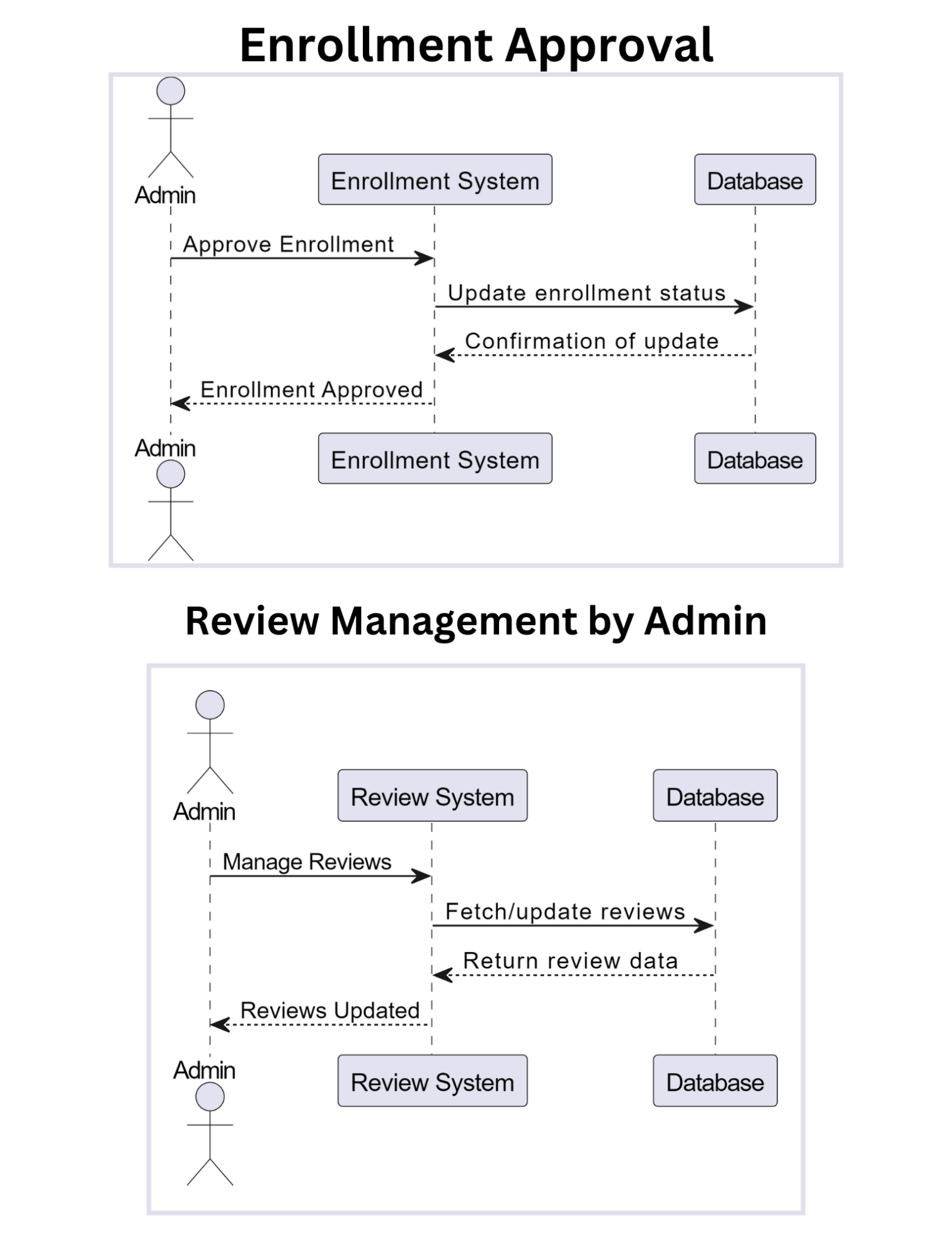
**“Please zoom in Document to View Clear ERD Clearly”**

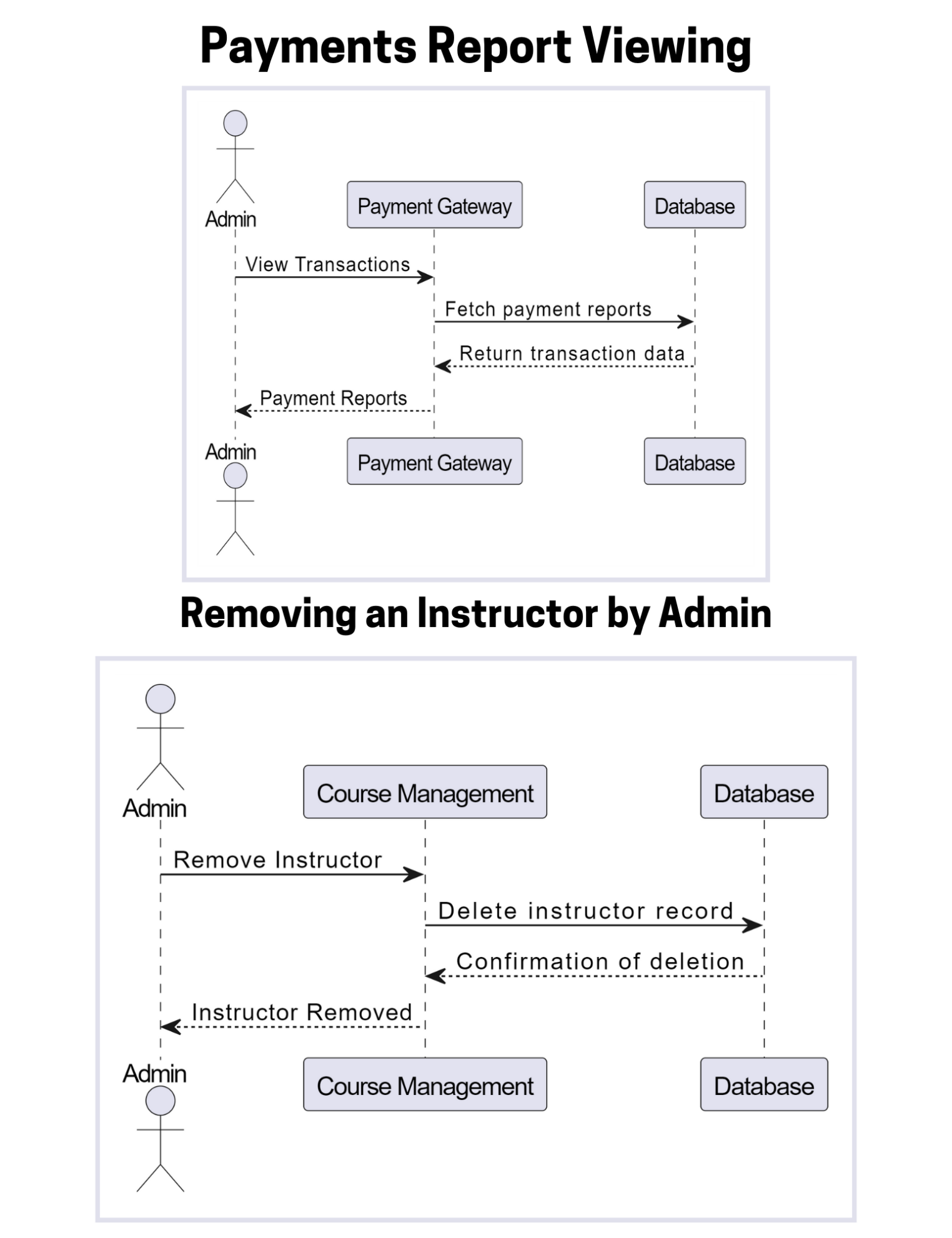
***3. Sequence Diagram***

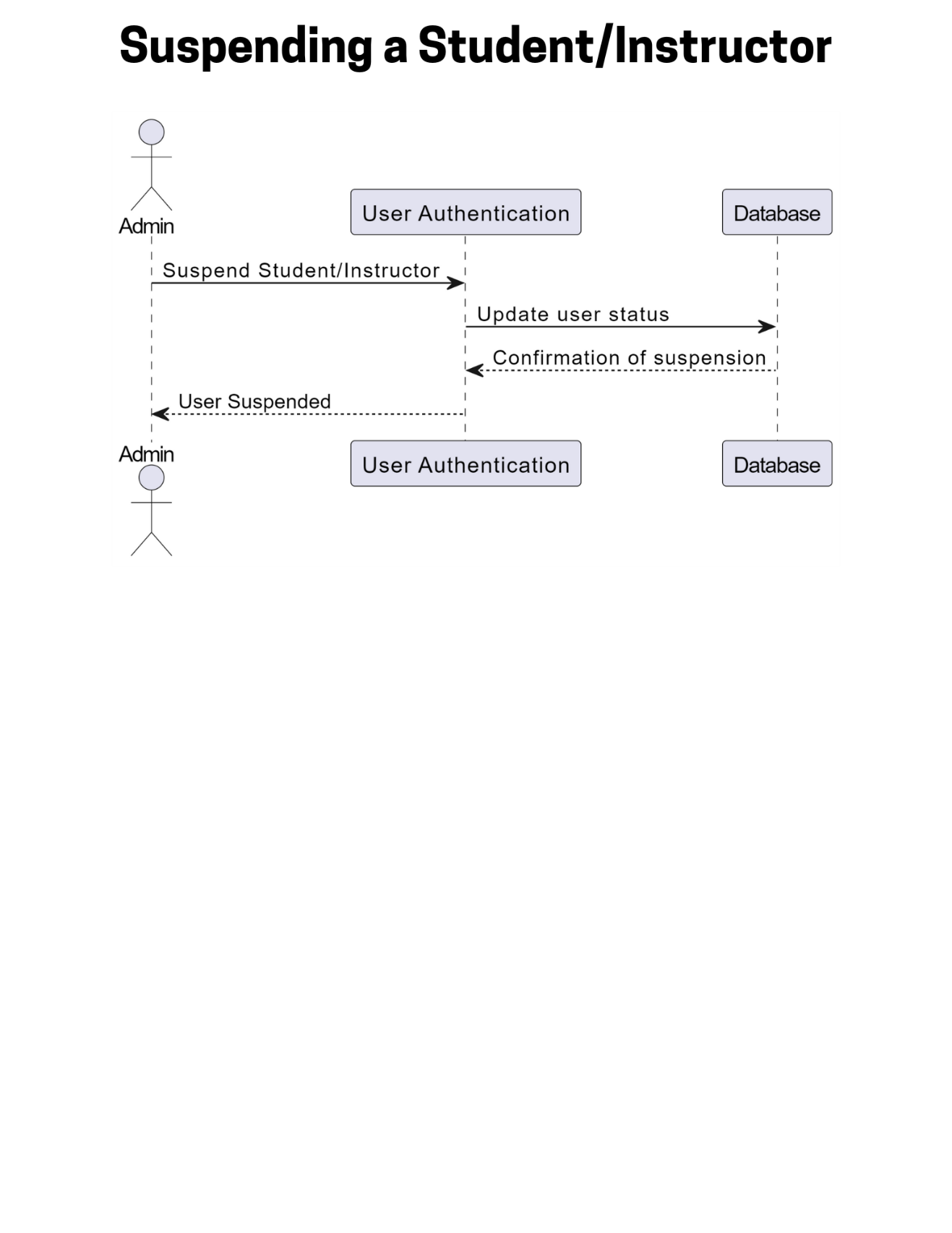
{Sequence diagram made with online diagram making tool **Draw.io** & Used **Canva** to Merge 2 diagrams at one place**}**

**“Please zoom in Document to View Sequence Diagrams Clearly”**

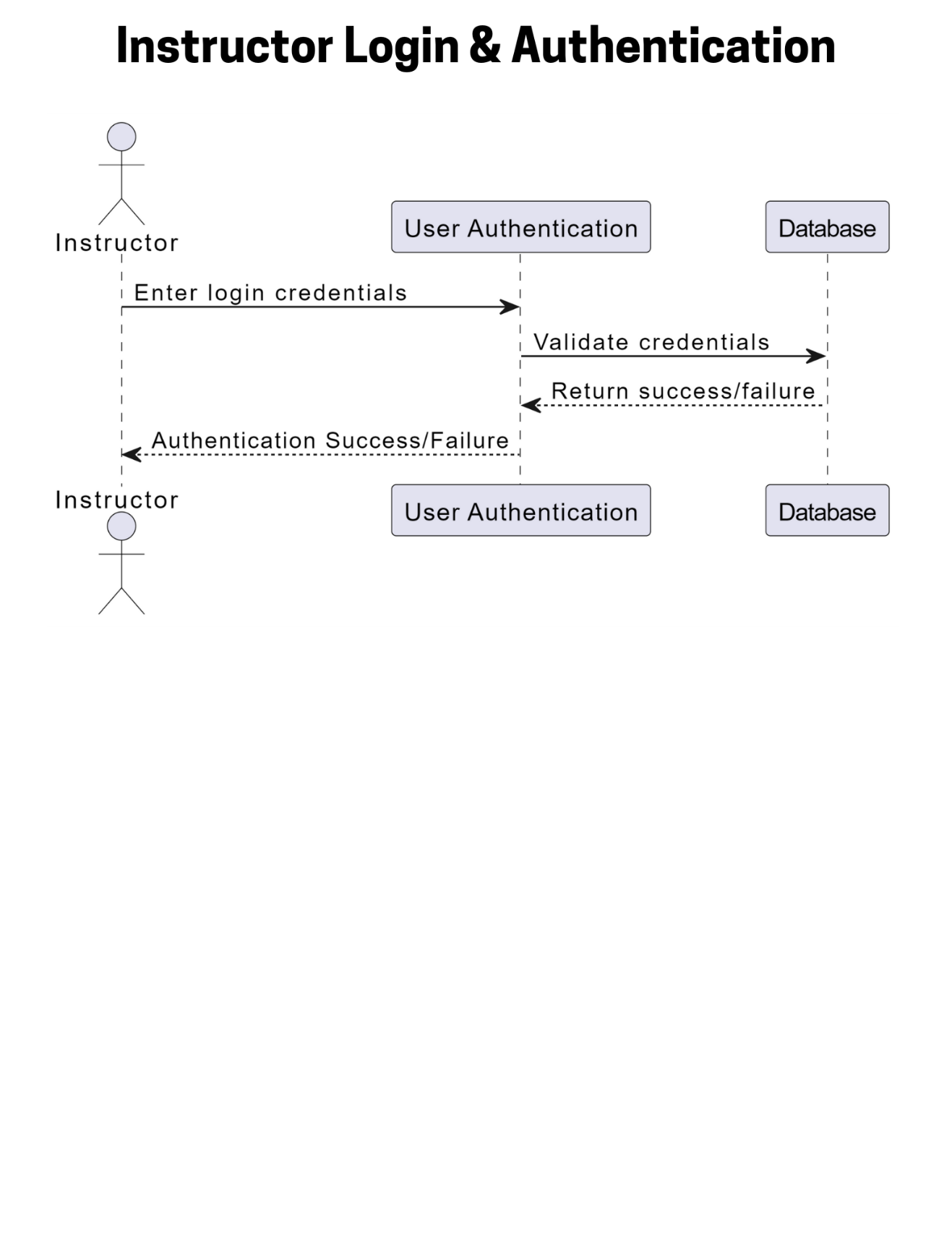
**1. Admin Role Sequence Diagrams:**

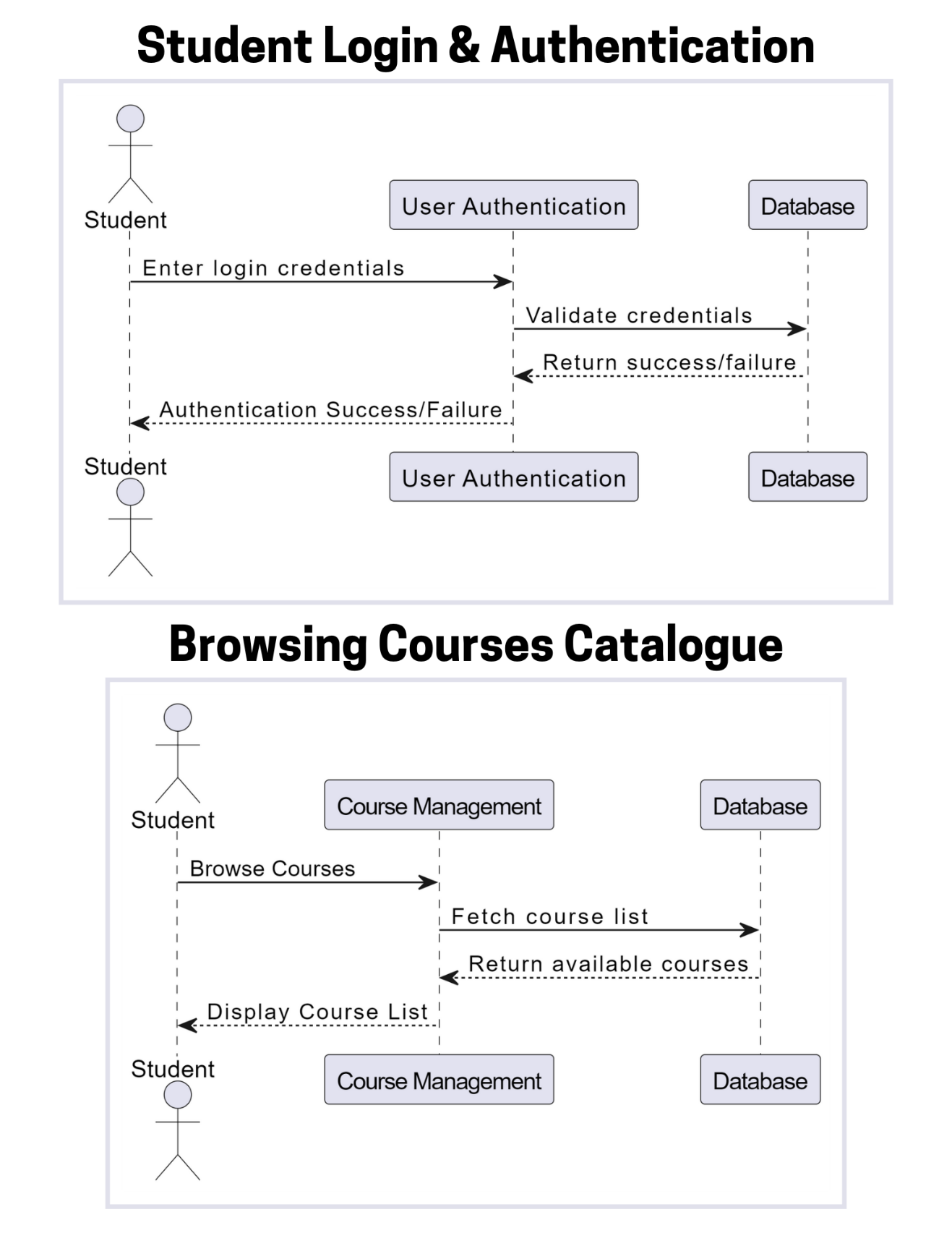
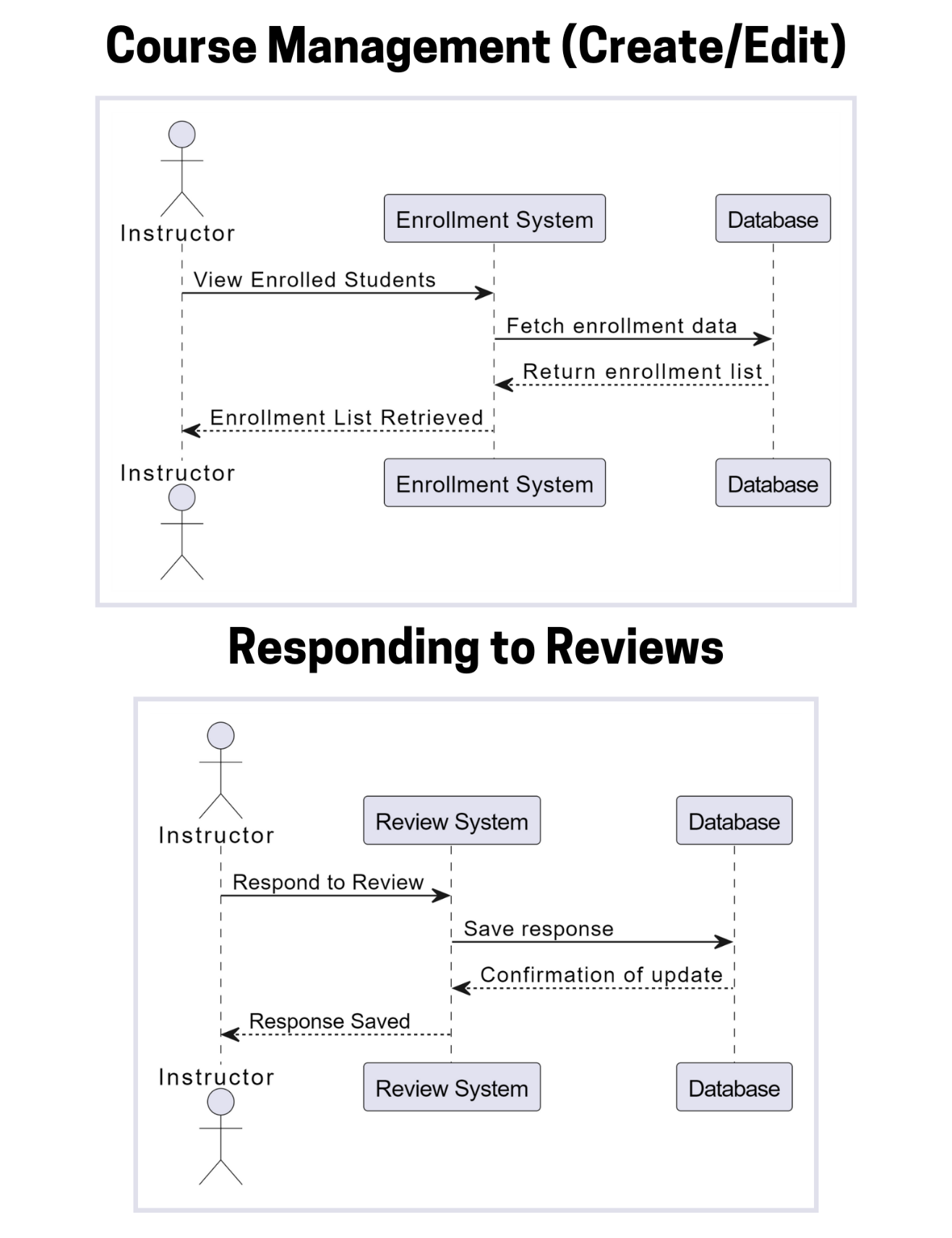
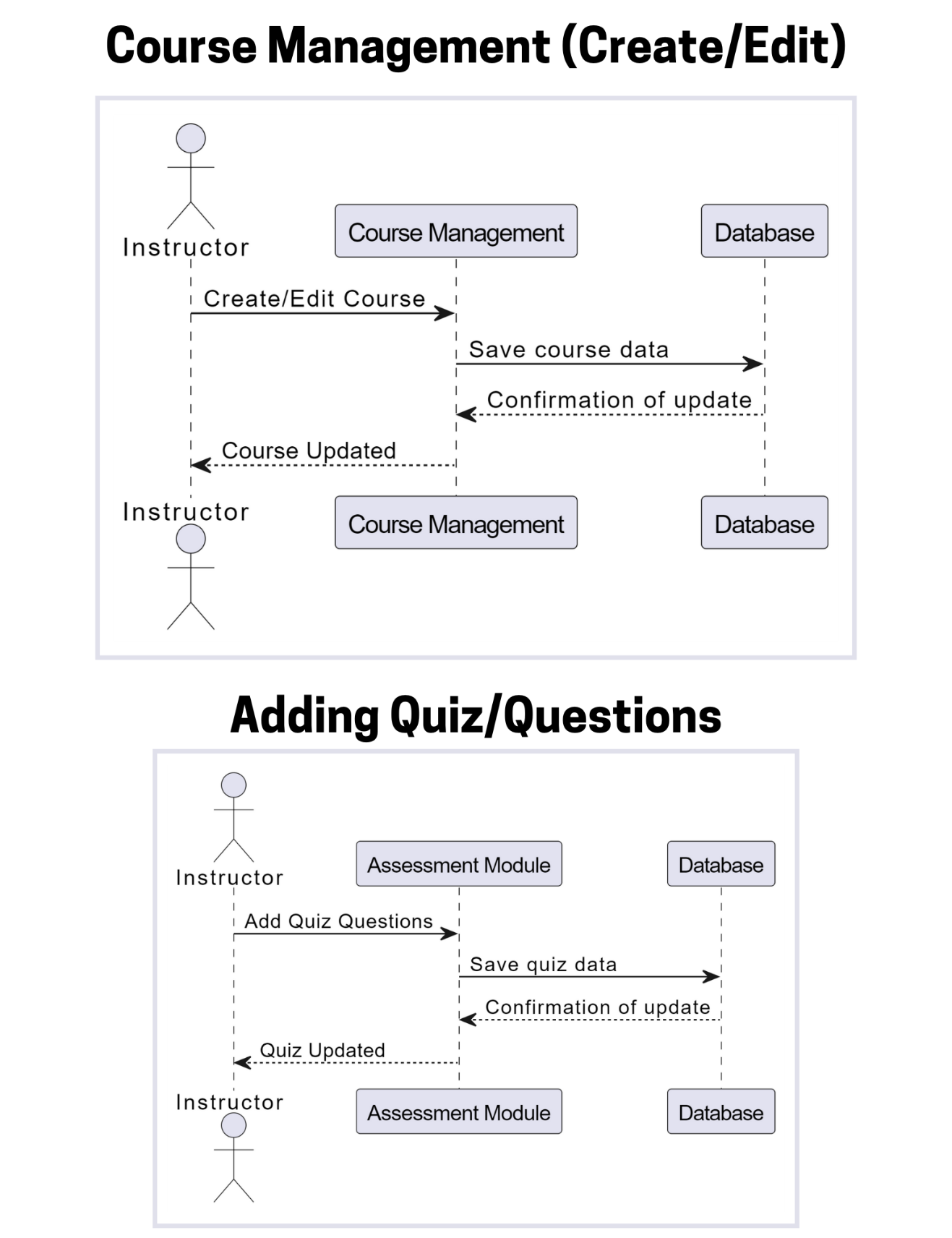
****

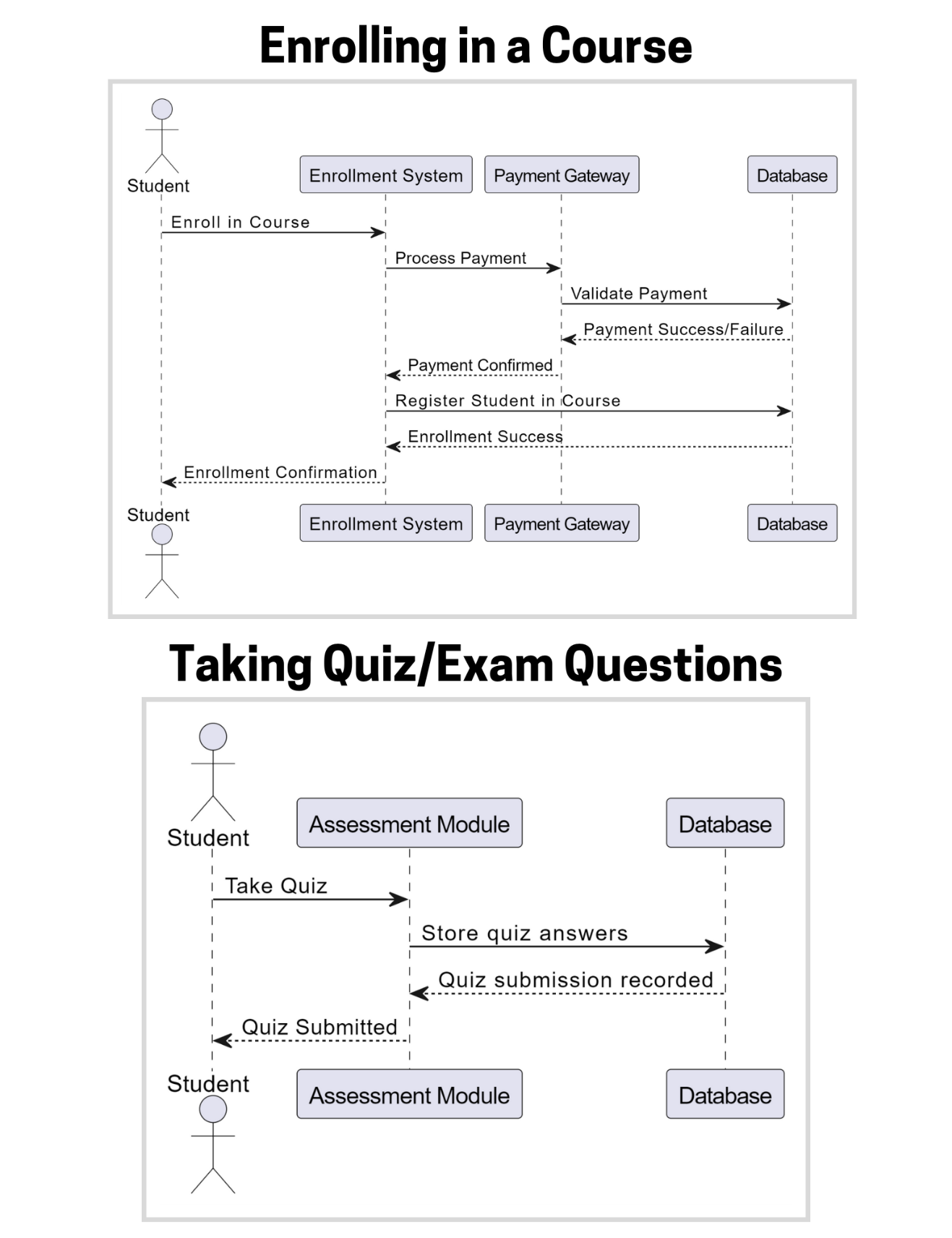
****

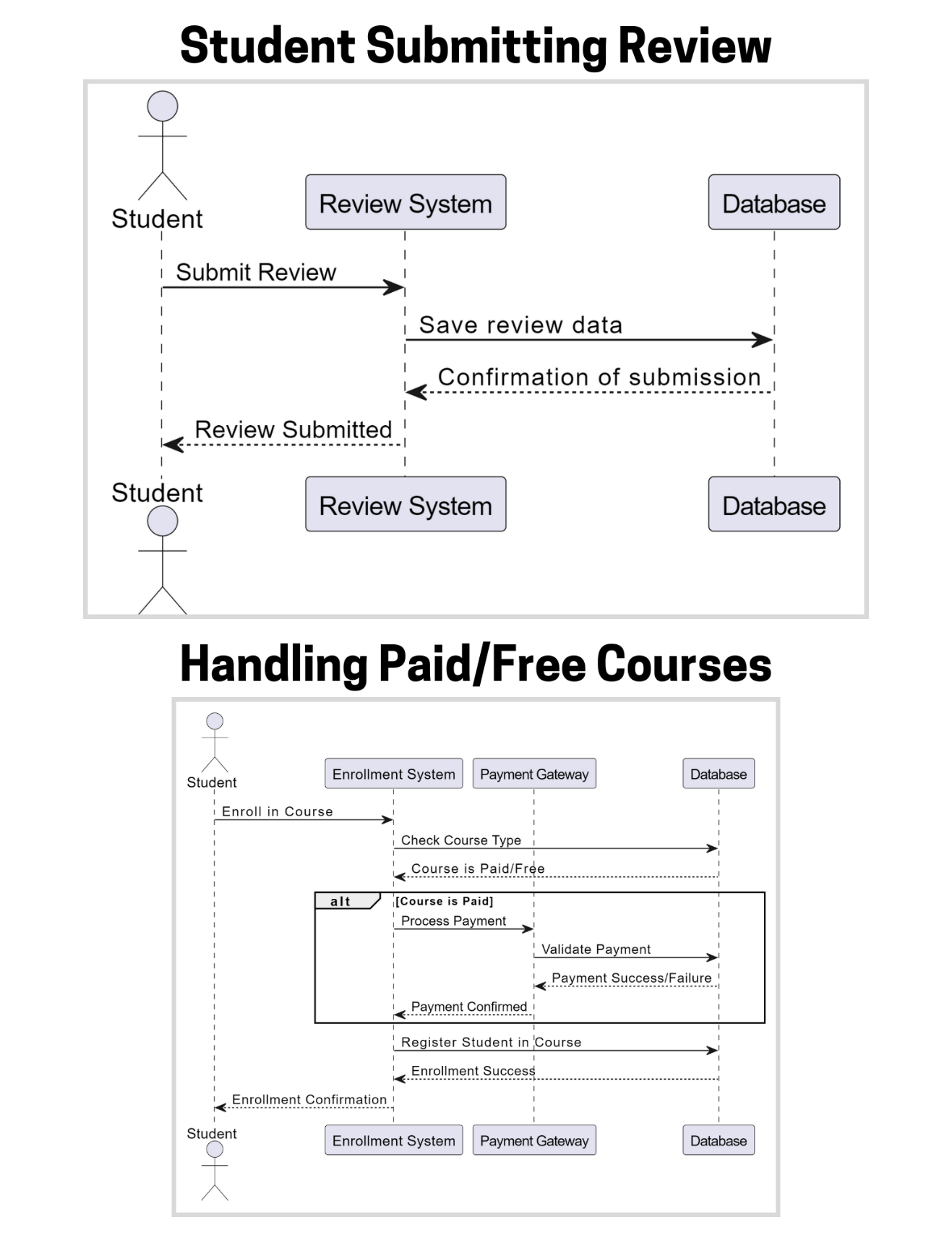
****

**2. Instructor Role Sequence Diagram:**

****

**3. Student Role Sequence Diagram:**

****

****

***4. Architecture Design Diagram***

Users (Students, Instructors, Admins)

Interface / Forms (Web & Mobile)

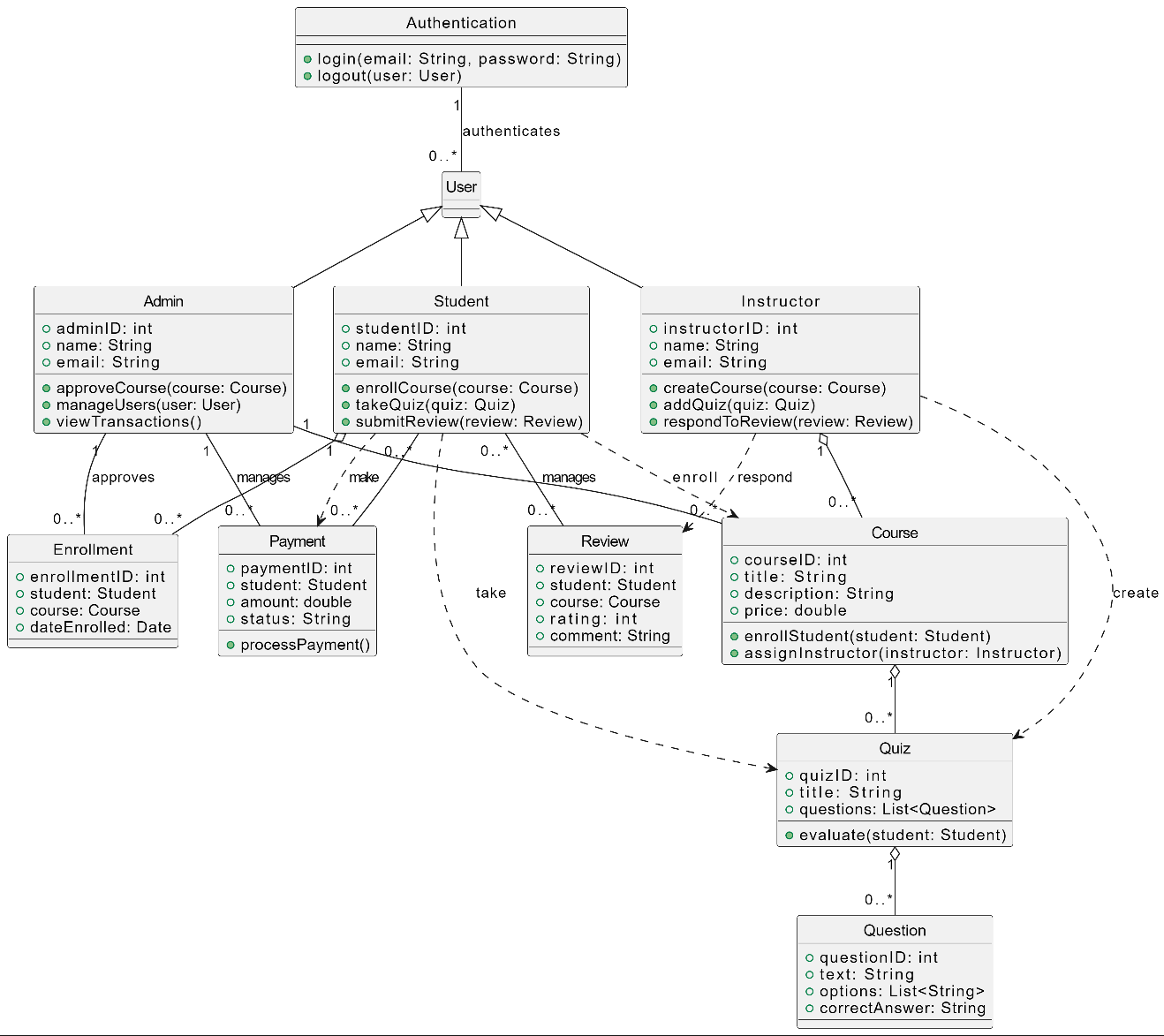
Business Logic (Course Management, Enrollments, Payments, Reviews)

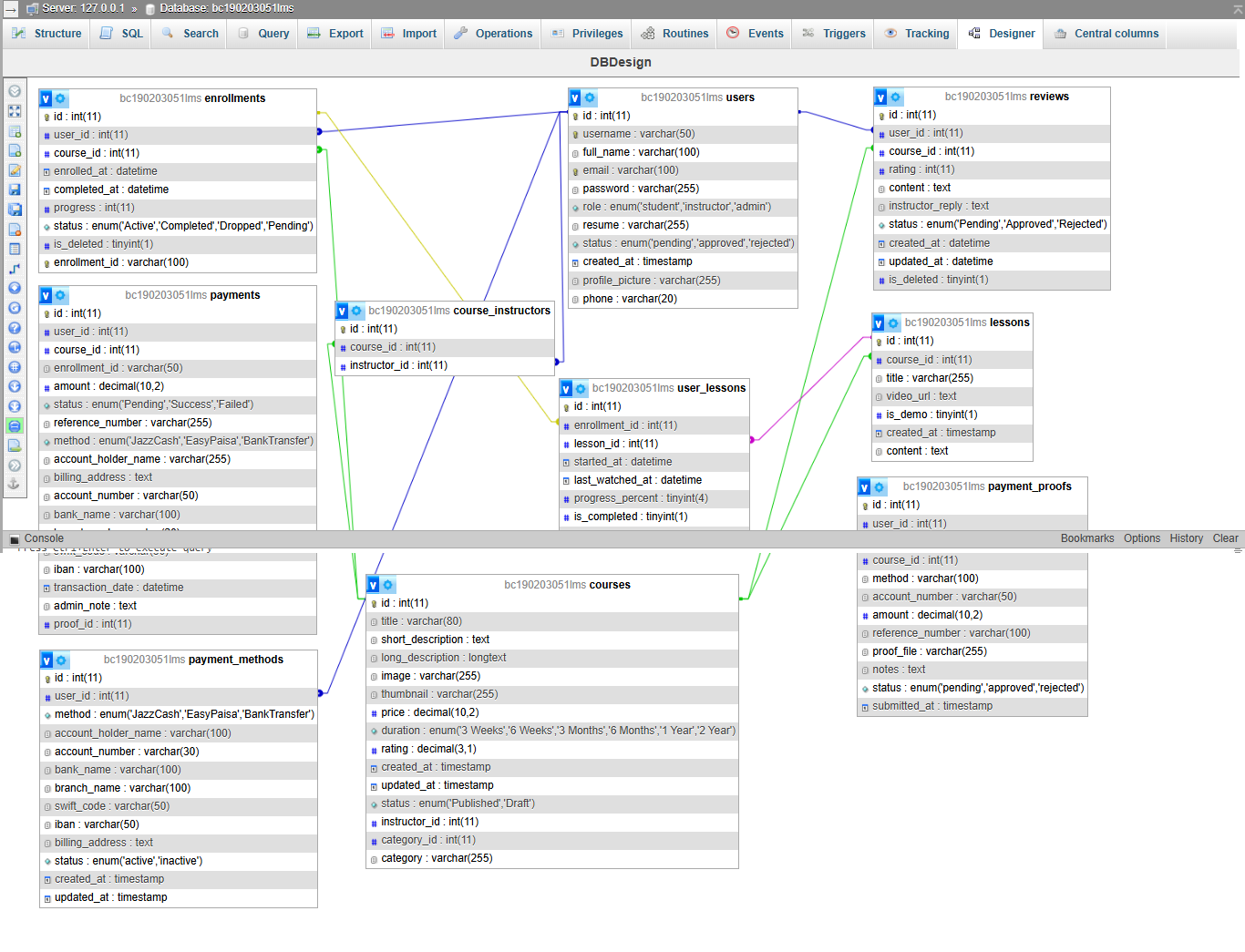
Data Access Component (API & Database Communication)

SQL Server Database

***5. Class Diagram***

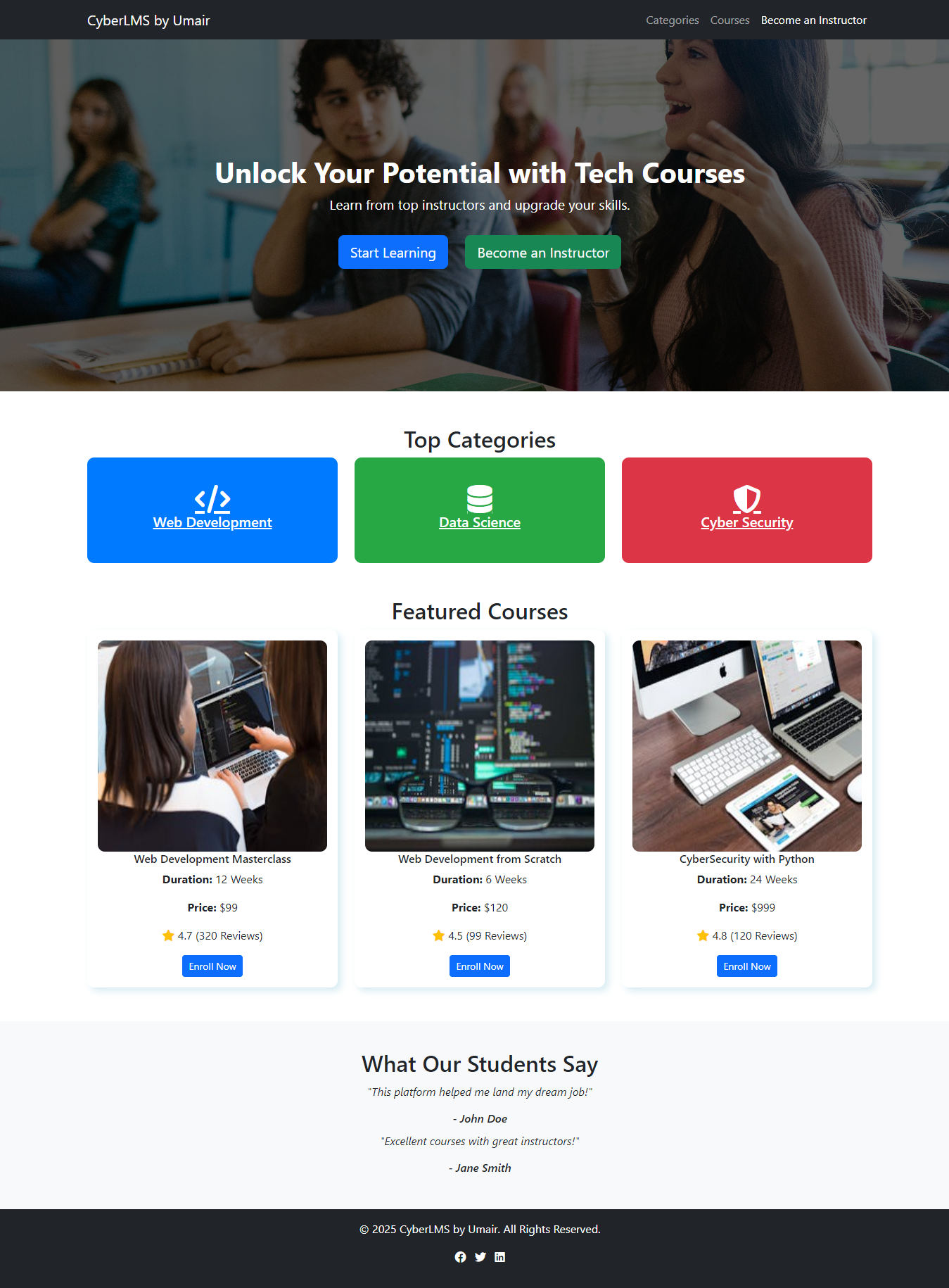
{Class diagram made with online diagram making tool **Draw.io}**

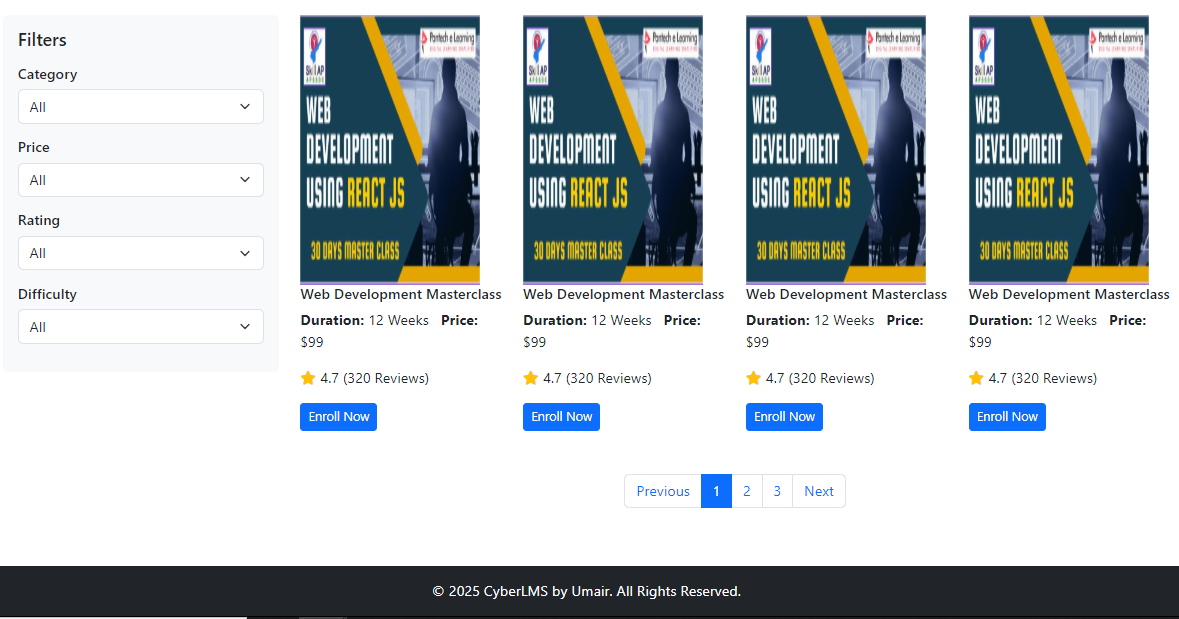
**“Please zoom in Document to View Class Diagram Clearly”**

***6. Database Design***

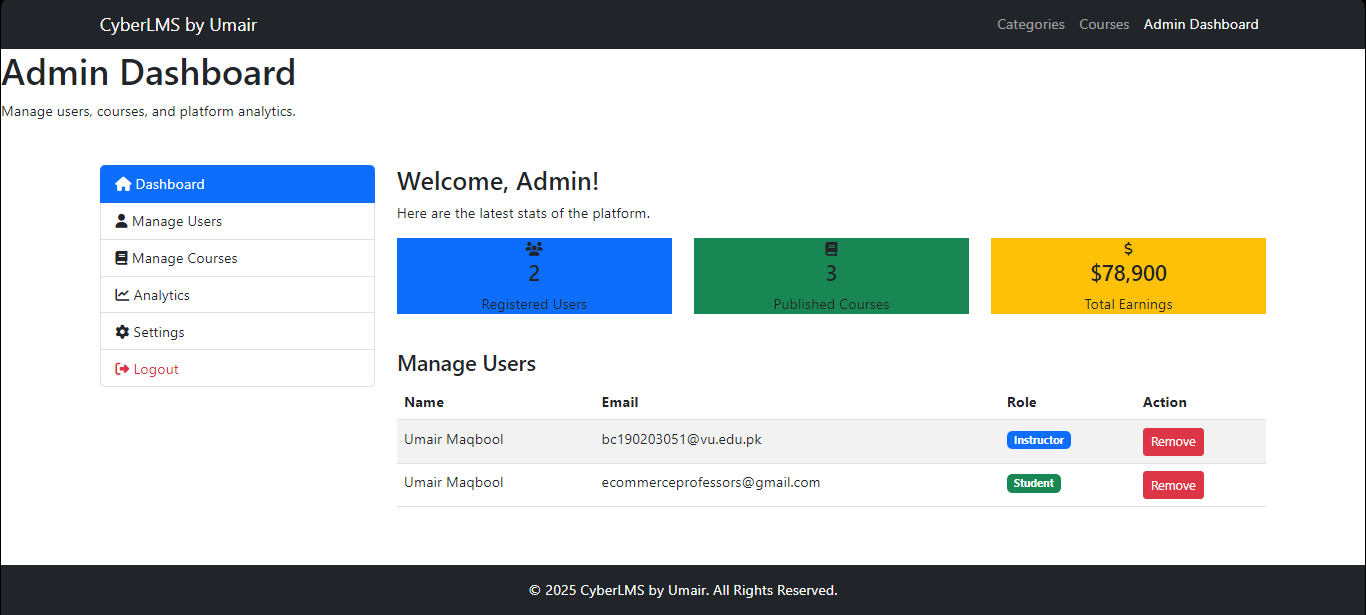
***7.Interface Design***

1. HomePage:

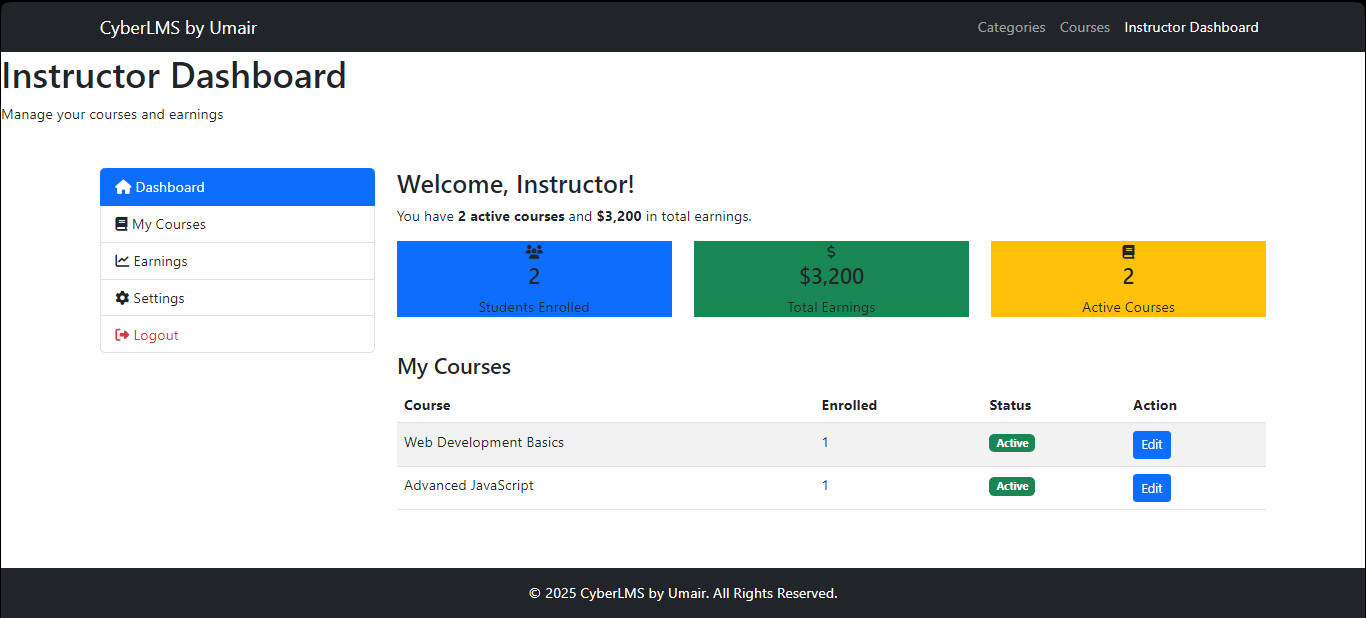


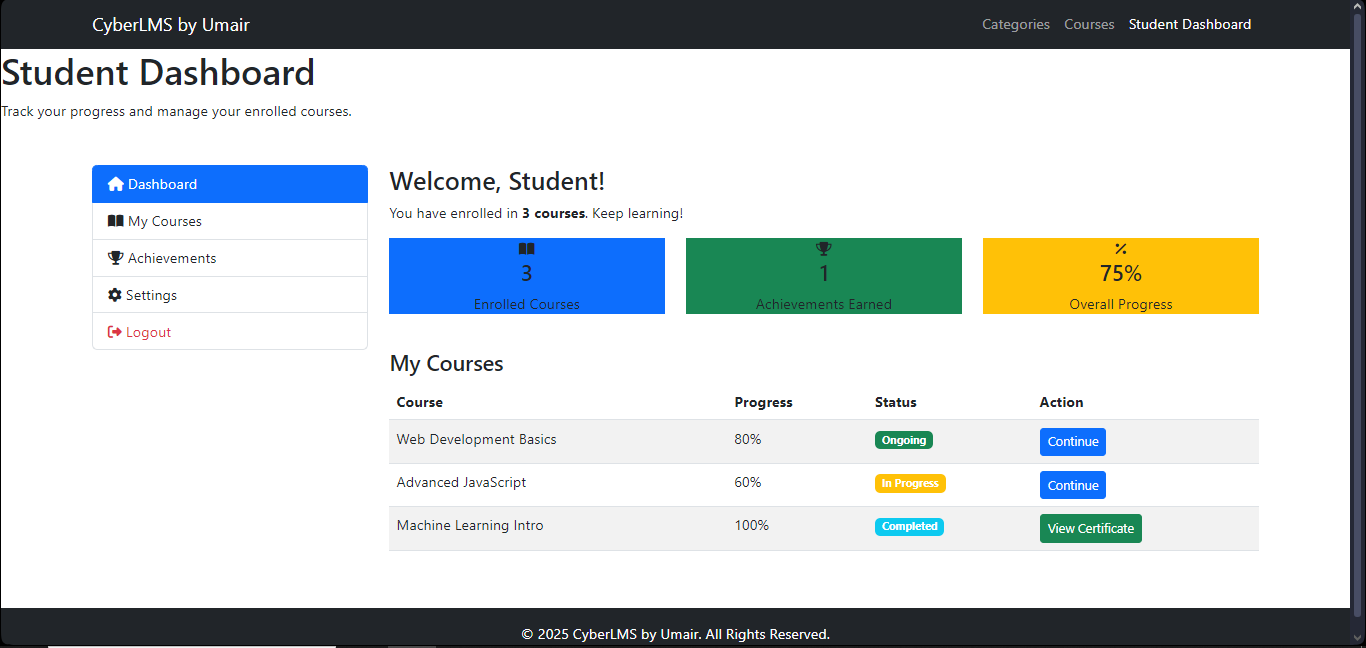
2. Courses Page:

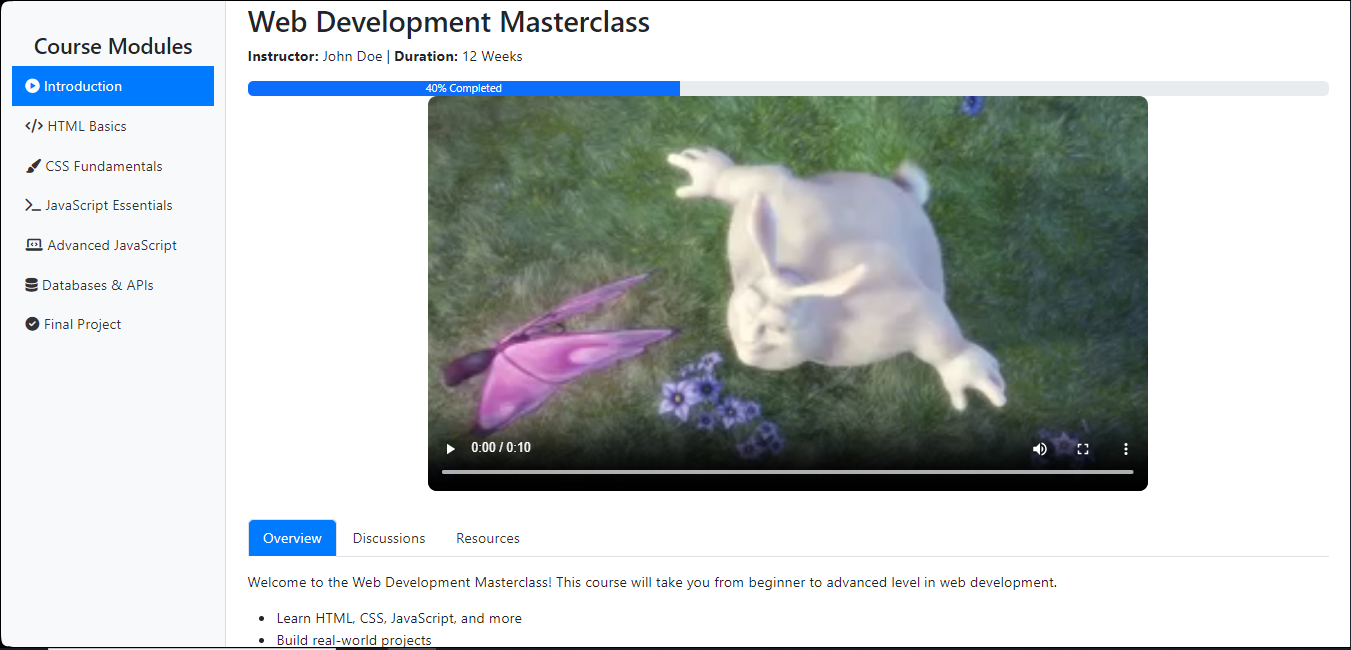
3. Categories Page:

4. Admin Dashboard:

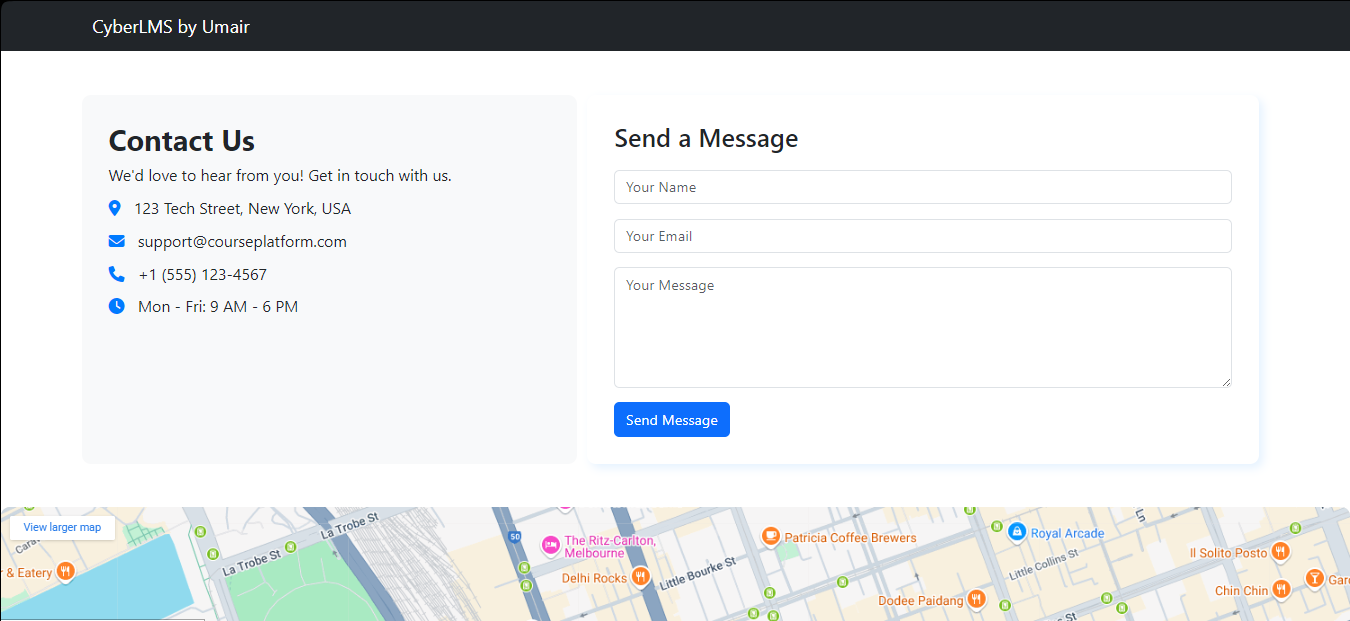
5. Instructor Dashboard:



6. Student Dashboard:

7. Course View Dashboard Page:

8. Contact Us Page Interface:



***8.Test Case’s***

**Test Case No. 1:**

|  |  |
| --- | --- |
| **Test Case** | **UC-OELP001-TC01** |
| **Test Case Name** | Verify course creation |
| **Pre-Condition** | Admin is logged in with appropriate privileges |
| **Actions** | 1. Navigate to course management 2. Click "Add Course" 3. Enter course title, description, and content 4. Click "Save" |
| **Expected Results** | Course should be created successfully and appear in the catalog |
| **Priority** | High |
| **Tested by** | Mr. Umair Maqbool |

**Test Case No. 2:**

|  |  |
| --- | --- |
| **Test Case** | **UC-OELP001-TC02** |
| **Test Case Name** | Verify course deletion |
| **Pre-Condition** | Course exists in the system |
| **Actions** | 1. Navigate to course management 2. Select a course 3. Click **"Delete"** |
| **Expected Results** | Course should be removed from the catalog |
| **Priority** | High |
| **Tested by** | Mr. Umair Maqbool |

**Test Case No. 3:**

|  |  |
| --- | --- |
| **Test Case** | **UC-OELP001-TC03** |
| **Test Case Name** | Verify error handling for failed course creation |
| **Pre-Condition** | Admin is logged in |
| **Actions** | 1. Try adding a course with missing required fields 2. Click **"Save"** |
| **Expected Results** | System should display an error message |
| **Priority** | Medium |
| **Tested by** | Mr. Umair Maqbool |

**Test Case No. 4:**

|  |  |
| --- | --- |
| **Test Case** | **UC-OELP002-TC04** |
| **Test Case Name** | Verify adding a new user |
| **Pre-Condition** | Admin is logged in |
| **Actions** | 1. Navigate to user management 2. Click **"Add User"** 3. Enter user details and select role 4. Click **"Save"** |
| **Expected Results** | User should be added successfully. |
| **Priority** | High |
| **Tested by** | Mr. Umair Maqbool |

**Test Case No. 5:**

|  |  |
| --- | --- |
| **Test Case** | **UC-OELP002-TC05** |
| **Test Case Name** | Verify modifying user roles |
| **Pre-Condition** | User exists in the system |
| **Actions** | 1. Select a user 2. Change role from student to instructor 3. Click **"Save"** |
| **Expected Results** | User role should be updated |
| **Priority** | Medium |
| **Tested by** | Mr. Umair Maqbool |

**Test Case No. 6:**

|  |  |
| --- | --- |
| **Test Case** | **UC-OELP002-TC06** |
| **Test Case Name** | Verify deleting a user |
| **Pre-Condition** | User exists in the system |
| **Actions** | 1. Select a user 2. Click **"Delete"** |
| **Expected Results** | User should be removed from the system |
| **Priority** | High |
| **Tested by** | Mr. Umair Maqbool |

**Test Case No. 7:**

|  |  |
| --- | --- |
| **Test Case** | **UC-OELP003-TC07** |
| **Test Case Name** | Verify course creation by instructor |
| **Pre-Condition** | Instructor is logged in |
| **Actions** | 1. Navigate to course creation **dashboard** 2. Enter course details 3. Click "**Save**" |
| **Expected Results** | Course should be created successfully |
| **Priority** | High |
| **Tested by** | Mr. Umair Maqbool |

**Test Case No. 8:**

|  |  |
| --- | --- |
| **Test Case** | **UC-OELP003-TC08** |
| **Test Case Name** | Verify error handling for missing course details. |
| **Pre-Condition** | Instructor is logged in |
| **Actions** | 1. Leave required fields blank 2. Click "**Save**" |
| **Expected Results** | System should show an error message |
| **Priority** | Medium |
| **Tested by** | Mr. Umair Maqbool |

**Test Case No. 9:**

|  |  |
| --- | --- |
| **Test Case** | **UC-OELP003-TC09** |
| **Test Case Name** | Verify media upload during course creation |
| **Pre-Condition** | Instructor is logged in |
| **Actions** | 1. Upload course materials (videos, PDFs) 2. Click "**Save**" |
| **Expected Results** | Files should be uploaded successfully. |
| **Priority** | Medium |
| **Tested by** | Mr. Umair Maqbool |

**Test Case No. 10:**

|  |  |
| --- | --- |
| **Test Case** | **UC-OELP004-TC10** |
| **Test Case Name** | Verify student enrollment |
| **Pre-Condition** | Student is logged in |
| **Actions** | 1. Browse available courses 2. Select a course 3. Click "**Enroll**" |
| **Expected Results** | Student should be enrolled successfully |
| **Priority** | Medium |
| **Tested by** | Mr. Umair Maqbool |

**Test Case No. 11:**

|  |  |
| --- | --- |
| **Test Case** | **UC-OELP005-TC11** |
| **Test Case Name** | Verify feedback submission & Spam Prevention |
| **Pre-Condition** | 1. Student has completed a course  2. Student is Logged in. |
| **Actions** | 1. Navigate to feedback section 2. Submit rating and comment  3. Submitting Multiple Reviews for the Same Course (Spam) |
| **Expected Results** | Feedback should be posted Successfully 1st time only.  Multiple Feeback submission should be rejected/Blocked. **(Spam Prevention)** |
| **Priority** | Medium |
| **Tested by** | Mr. Umair Maqbool |

**Test Case No. 12:**

|  |  |
| --- | --- |
| **Test Case** | **UC-OELP006-TC12** |
| **Test Case Name** | Verify progress tracking updates |
| **Pre-Condition** | Student is enrolled in a course |
| **Actions** | 1. Complete a lesson 2. Check progress bar |
| **Expected Results** | Progress should update dynamically |
| **Priority** | Medium |
| **Tested by** | Mr. Umair Maqbool |

**Test Case No. 13:**

|  |  |
| --- | --- |
| **Test Case** | **UC-OELP007-TC13** |
| **Test Case Name** | Verify SQL Injection Prevention |
| **Pre-Condition** | User is on login form |
| **Actions** | 1. Enter ' OR **'1'='1** in the email/password field 2. Click "**Login**" |
| **Expected Results** | System should reject the input and prevent unauthorized access |
| **Priority** | High |
| **Tested by** | Mr. Umair Maqbool |

**Test Case No. 14:**

|  |  |
| --- | --- |
| **Test Case** | **UC-OELP008-TC14** |
| **Test Case Name** | Verify system response under heavy load |
| **Pre-Condition** | Simulate 1000+ concurrent users |
| **Actions** | 1. Use a load testing tool (e.g., JMeter) 2. Simulate multiple users logging in, enrolling, and accessing courses |
| **Expected Results** | System should handle load without crashing, and response time should remain within acceptable limits. |
| **Priority** | High |
| **Tested by** | Mr. Umair Maqbool |