INSTITUTE MANAGEMENT SYSTEM

Diploma in Software Engineering

Final Project Documentation

National Institute of Business Management

Kandy Regional Center

No: 2, Asgiri Vihara Mawatha,

Kandy

2023

INSTITUTE MANAGEMENT SYSTEM

K.D.D. Dissanayake KADSE222F-004

Y.S. Panannala KADSE222F-031

S.A.S. Dhanayake KADSE222F-016

K.V.M. Sulakshana KADSE222F-039

Diploma in Software Engineering

Supervisor:

Mr. Sandaruwan Herath

Management Information System Division,

National Institute of Business Management

“The project is submitted in partial fulfilment of the requirement of the Diploma in Software Engineering of the National Institute of Business Management.”

December 2023

## Declaration

“I certify that this project does not incorporate without acknowledgement, any material previously submitted for a Diploma in any institute and to the best of my knowledge and belief. It does not contain any material previously published or written by another person or by myself except where due reference is made in the text. I also hereby give consent for my project report, if accepted, to be made available for photocopying and interlibrary loans, and for the title and summary to be made available outside organizations.”

|  |  |  |
| --- | --- | --- |
| Student Name | Index No | Signature |
| K.D.D. Dissanayake | KADSE222F-004 |  |
| Y.S. Panannala | KADSE222F-031 |  |
| S.A.S. Dhanayake | KADSE222F-016 |  |
| K.V.M. Sulakshana | KADSE222F-039 |  |

Certified by

Supervisor: Mr. Sandaruwan Herath

Signature: ………………

Date: …………………

Course Directors:

Mr. Sandaruwan Herath Ms. Inoka Abhayasinghe

Signature: ……………. Signature: …………….

Date: ………………… Date: …………………

## Summary

Mr. Dhammika Thisara Sanchiarrachi who is the manager and sole lecturer at his Institute was facing problems with managing his records. So, to give a solution to his problem we build a computerized system to better manage his record keeping tasks.

All the Standard protocols were followed to build the application. And The Application features an easy to navigate user interface which has built in validations, and helpful features to ensure the validity of the data.

Also, our project features an additional component which is a Web Site for students to use. They can use it to ask questions, view notes, view results, view their schedule, and view answers to their questions.

A Local Database was implemented to store and access these data.

Overall, we believe that we have achieved the goal of our project which was to help Mr. Dhammika to free him from his time-consuming daily record keeping tasks with an automated system.

## Acknowledgement

First, we would like to express our heartfelt gratitude to Mr. Dhammika Thisara Sanchiarrachi for giving us the opportunity to build this project.

Also, we would like to extend our thanks to Mr. Sandaruwan Herath, our Project Supervisor, and Ms. Inoka Abhayasinghe, and Mr. Sandaruwan Herath our Course Directors, for providing us with knowledge, skills, guidance, and support.

We would also like to thank all the instructors and staff at NIBM for their assistance in many ways.

Finally, we would like to express our gratitude to our families and friends, who have always supported us in every possible way.

## List of Figures

[Figure 1: Lifeway Education Services 3](file:///C:\Users\Maheesha\Desktop\Home%20Work\Year%2001\IMS\Project%20Report\PROJECT%20REPORT%20-%20Copy%20(2).docx#_Toc155180230)

[Figure 2: Software Development Method (Waterfall Method) 6](file:///C:\Users\Maheesha\Desktop\Home%20Work\Year%2001\IMS\Project%20Report\PROJECT%20REPORT%20-%20Copy%20(2).docx#_Toc155180231)

[Figure 3: Technologies used. 7](file:///C:\Users\Maheesha\Desktop\Home%20Work\Year%2001\IMS\Project%20Report\PROJECT%20REPORT%20-%20Copy%20(2).docx#_Toc155180232)

[Figure 4: Use Case Diagram for the current system 10](file:///C:\Users\Maheesha\Desktop\Home%20Work\Year%2001\IMS\Project%20Report\PROJECT%20REPORT%20-%20Copy%20(2).docx#_Toc155180233)

[Figure 5: Use Case Diagram for the proposed system. 11](file:///C:\Users\Maheesha\Desktop\Home%20Work\Year%2001\IMS\Project%20Report\PROJECT%20REPORT%20-%20Copy%20(2).docx#_Toc155180234)

[Figure 6: DFD Diagram (Context Level) 12](file:///C:\Users\Maheesha\Desktop\Home%20Work\Year%2001\IMS\Project%20Report\PROJECT%20REPORT%20-%20Copy%20(2).docx#_Toc155180235)

[Figure 7: DFD Diagram (Level 1) 13](file:///C:\Users\Maheesha\Desktop\Home%20Work\Year%2001\IMS\Project%20Report\PROJECT%20REPORT%20-%20Copy%20(2).docx#_Toc155180236)

[Figure 8: DFD Diagram (Level 1) 14](file:///C:\Users\Maheesha\Desktop\Home%20Work\Year%2001\IMS\Project%20Report\PROJECT%20REPORT%20-%20Copy%20(2).docx#_Toc155180237)

[Figure 9: DFD Diagram (Level 1) 15](file:///C:\Users\Maheesha\Desktop\Home%20Work\Year%2001\IMS\Project%20Report\PROJECT%20REPORT%20-%20Copy%20(2).docx#_Toc155180238)

[Figure 10: Class Diagram 16](file:///C:\Users\Maheesha\Desktop\Home%20Work\Year%2001\IMS\Project%20Report\PROJECT%20REPORT%20-%20Copy%20(2).docx#_Toc155180239)

[Figure 11: Class Diagram 17](file:///C:\Users\Maheesha\Desktop\Home%20Work\Year%2001\IMS\Project%20Report\PROJECT%20REPORT%20-%20Copy%20(2).docx#_Toc155180240)

[Figure 12: ER Diagram 18](file:///C:\Users\Maheesha\Desktop\Home%20Work\Year%2001\IMS\Project%20Report\PROJECT%20REPORT%20-%20Copy%20(2).docx#_Toc155180241)

[Figure 13: Sequence Diagram (Manage Course Details) 19](#_Toc155180242)

[Figure 14: Sequence Diagram (Manage Course Details) 20](#_Toc155180243)

[Figure 15: Sequence Diagram (Manage Student Details) 21](file:///C:\Users\Maheesha\Desktop\Home%20Work\Year%2001\IMS\Project%20Report\PROJECT%20REPORT%20-%20Copy%20(2).docx#_Toc155180244)

[Figure 16: Sequence Diagram (Store Notes) 23](file:///C:\Users\Maheesha\Desktop\Home%20Work\Year%2001\IMS\Project%20Report\PROJECT%20REPORT%20-%20Copy%20(2).docx#_Toc155180245)

[Figure 17: Sequence Diagram (Store Exam Marks) 24](file:///C:\Users\Maheesha\Desktop\Home%20Work\Year%2001\IMS\Project%20Report\PROJECT%20REPORT%20-%20Copy%20(2).docx#_Toc155180246)

[Figure 18: Sequence Diagram for Manage Schedule 25](#_Toc155180247)

[Figure 19: Sequence Diagram for Manage Schedule 26](#_Toc155180248)

[Figure 20: Sequence Diagram (Store payments) 27](file:///C:\Users\Maheesha\Desktop\Home%20Work\Year%2001\IMS\Project%20Report\PROJECT%20REPORT%20-%20Copy%20(2).docx#_Toc155180249)

[Figure 21: Sequence Diagram (View Attendance) 28](#_Toc155180250)

[Figure 22: Sequence Diagram (View Questions) 29](#_Toc155180251)

[Figure 23: Sequence Diagram (Answer Questions) 30](#_Toc155180252)

[Figure 24: Sequence Diagram for View Results 31](#_Toc155180253)

[Figure 25: Sequence Diagram for View Schedule 32](#_Toc155180254)

[Figure 26: Sequence Diagram for Ask Questions 33](file:///C:\Users\Maheesha\Desktop\Home%20Work\Year%2001\IMS\Project%20Report\PROJECT%20REPORT%20-%20Copy%20(2).docx#_Toc155180255)

[Figure 27: Sequence Diagram for View Results 34](#_Toc155180256)

[Figure 28: Sequence Diagram for View Lecture Notes 35](#_Toc155180257)

[Figure 29: Login Page 38](#_Toc155180258)

[Figure 30: Pseudo code for login 39](file:///C:\Users\Maheesha\Desktop\Home%20Work\Year%2001\IMS\Project%20Report\PROJECT%20REPORT%20-%20Copy%20(2).docx#_Toc155180259)

[Figure 31: Manage Course Details Screen 40](#_Toc155180260)

[Figure 32: Pseudo code for course management 41](file:///C:\Users\Maheesha\Desktop\Home%20Work\Year%2001\IMS\Project%20Report\PROJECT%20REPORT%20-%20Copy%20(2).docx#_Toc155180261)

[Figure 33: Manage Student Details Screen 42](#_Toc155180262)

[Figure 34: Pseudo code for student management 43](file:///C:\Users\Maheesha\Desktop\Home%20Work\Year%2001\IMS\Project%20Report\PROJECT%20REPORT%20-%20Copy%20(2).docx#_Toc155180263)

[Figure 35: Manage Lecturer Details Screen 44](#_Toc155180264)

[Figure 36: Pseudo code for lecturer management 45](file:///C:\Users\Maheesha\Desktop\Home%20Work\Year%2001\IMS\Project%20Report\PROJECT%20REPORT%20-%20Copy%20(2).docx#_Toc155180265)

[Figure 37: Record Payment Screen 46](#_Toc155180266)

[Figure 38: Pseudo code for manage payments 46](file:///C:\Users\Maheesha\Desktop\Home%20Work\Year%2001\IMS\Project%20Report\PROJECT%20REPORT%20-%20Copy%20(2).docx#_Toc155180267)

[Figure 39: Manage Schedule Screen 47](#_Toc155180268)

[Figure 40: Pseudo code for schedule management 48](file:///C:\Users\Maheesha\Desktop\Home%20Work\Year%2001\IMS\Project%20Report\PROJECT%20REPORT%20-%20Copy%20(2).docx#_Toc155180269)

[Figure 41: Store Lecture Notes Screen 49](#_Toc155180270)

[Figure 42: Pseudo code for store lecture notes 49](file:///C:\Users\Maheesha\Desktop\Home%20Work\Year%2001\IMS\Project%20Report\PROJECT%20REPORT%20-%20Copy%20(2).docx#_Toc155180271)

[Figure 43: Store Exam Marks Screen 50](file:///C:\Users\Maheesha\Desktop\Home%20Work\Year%2001\IMS\Project%20Report\PROJECT%20REPORT%20-%20Copy%20(2).docx#_Toc155180272)

[Figure 44: Pseudo code to store exam marks 50](file:///C:\Users\Maheesha\Desktop\Home%20Work\Year%2001\IMS\Project%20Report\PROJECT%20REPORT%20-%20Copy%20(2).docx#_Toc155180273)

[Figure 45: Ask Questions screen 51](#_Toc155180274)

[Figure 46: Pseudo code to ask questions 51](file:///C:\Users\Maheesha\Desktop\Home%20Work\Year%2001\IMS\Project%20Report\PROJECT%20REPORT%20-%20Copy%20(2).docx#_Toc155180275)

[Figure 47: View Results Screen 52](#_Toc155180276)

[Figure 48: Pseudo code for view exam results 52](file:///C:\Users\Maheesha\Desktop\Home%20Work\Year%2001\IMS\Project%20Report\PROJECT%20REPORT%20-%20Copy%20(2).docx#_Toc155180277)

[Figure 49: View Notes Screen 53](#_Toc155180278)

[Figure 50: Pseudo code for view notes 53](file:///C:\Users\Maheesha\Desktop\Home%20Work\Year%2001\IMS\Project%20Report\PROJECT%20REPORT%20-%20Copy%20(2).docx#_Toc155180279)

[Figure 51: View Questions Screen 54](#_Toc155180280)

[Figure 52: Pseudo code for View Questions 54](file:///C:\Users\Maheesha\Desktop\Home%20Work\Year%2001\IMS\Project%20Report\PROJECT%20REPORT%20-%20Copy%20(2).docx#_Toc155180281)

[Figure 53: View Student Attendance Screen 55](#_Toc155180282)

[Figure 54: Pseudo code for view student attendance 55](file:///C:\Users\Maheesha\Desktop\Home%20Work\Year%2001\IMS\Project%20Report\PROJECT%20REPORT%20-%20Copy%20(2).docx#_Toc155180283)

[Figure 55: View Schedule Screen 56](file:///C:\Users\Maheesha\Desktop\Home%20Work\Year%2001\IMS\Project%20Report\PROJECT%20REPORT%20-%20Copy%20(2).docx#_Toc155180284)

[Figure 56: Pseudo code to view schedule 56](file:///C:\Users\Maheesha\Desktop\Home%20Work\Year%2001\IMS\Project%20Report\PROJECT%20REPORT%20-%20Copy%20(2).docx#_Toc155180285)

[Figure 57: Home Screen 57](#_Toc155180286)

[Figure 58: Pseudo code for home screen 58](file:///C:\Users\Maheesha\Desktop\Home%20Work\Year%2001\IMS\Project%20Report\PROJECT%20REPORT%20-%20Copy%20(2).docx#_Toc155180287)

[Figure 59: Parent Details Screen 59](#_Toc155180288)

[Figure 60: Pseudo code to maintain Parent Details 59](file:///C:\Users\Maheesha\Desktop\Home%20Work\Year%2001\IMS\Project%20Report\PROJECT%20REPORT%20-%20Copy%20(2).docx#_Toc155180289)

[Figure 61: Database Schema 60](file:///C:\Users\Maheesha\Desktop\Home%20Work\Year%2001\IMS\Project%20Report\PROJECT%20REPORT%20-%20Copy%20(2).docx#_Toc155180290)

[Figure 62: Work Breakdown Sheet i](file:///C:\Users\Maheesha\Desktop\Home%20Work\Year%2001\IMS\Project%20Report\PROJECT%20REPORT%20-%20Copy%20(2).docx#_Toc155180291)

## List of Tables

[Table 1: Project Timeline 5](#_Toc154517563)

[Table 2: Table of Interfaces 37](#_Toc154517564)

**Table of Contents**

[Declaration i](#_Toc155179486)

[Summary ii](#_Toc155179487)

[Acknowledgement iii](#_Toc155179488)

[List of Figures iv](#_Toc155179489)

[List of Tables v](#_Toc155179490)

[Chapter 1 1](#_Toc155179491)

[Introduction 1](#_Toc155179492)

[1.1 Introduction to the Organization 1](#_Toc155179493)

[1.2 Organizational Structure 1](#_Toc155179494)

[1.3 Current Operation Method 1](#_Toc155179495)

[1.4 Users and Responsibilities of Organization 1](#_Toc155179496)

[1.5 Problem Definition 2](#_Toc155179497)

[1.6 Project Objectives 2](#_Toc155179498)

[1.7 Proposed Solution 2](#_Toc155179499)

[1.8 Chapter Summary 3](#_Toc155179500)

[Chapter 2 4](#_Toc155179501)

[Methodology 4](#_Toc155179502)

[2.1 Introduction 4](#_Toc155179503)

[2.2 Project Plan 4](#_Toc155179504)

[2.3 Data Collection Methods 5](#_Toc155179505)

[2.4 Software Development Method 6](#_Toc155179506)

[2.5 Software Development Tools 7](#_Toc155179507)

[2.6 Testing Methods 8](#_Toc155179508)

[2.7 Implementation plan 8](#_Toc155179509)

[2.8 Chapter Summary 8](#_Toc155179510)

[Chapter 3 9](#_Toc155179511)

[Analysis 9](#_Toc155179512)

[3.1 Introduction 9](#_Toc155179513)

[3.2 Diagrams 10](#_Toc155179514)

[Use case diagram for the current system 10](#_Toc155179515)

[Use Case Diagram for the proposed system. 11](#_Toc155179516)

[DFD Diagram (Context Level) 12](#_Toc155179517)

[DFD Diagram (Level 1) 13](#_Toc155179518)

[DFD Diagram (Level 1) 14](#_Toc155179519)

[DFD Diagram (Level 1) 15](#_Toc155179520)

[Class Diagram 16](#_Toc155179521)

[Class Diagram 17](#_Toc155179522)

[ER Diagram 18](#_Toc155179523)

[Sequence Diagram for Manage Course Details 19](#_Toc155179524)

[Sequence Diagram for Manage Student 21](#_Toc155179525)

[Sequence Diagram for Manage Lecturer 22](#_Toc155179526)

[Sequence Diagram for Manage Notes 23](#_Toc155179527)

[Sequence Diagram for Manage Exam Marks 24](#_Toc155179528)

[Sequence Diagram for Manage Schedule 24](#_Toc155179529)

[Sequence Diagram for Manage Payments 27](#_Toc155179530)

[Sequence Diagram for View Attendance 28](#_Toc155179531)

[Sequence Diagram for View Questions 29](#_Toc155179532)

[Sequence Diagram for Answer Questions 30](#_Toc155179533)

[Sequence Diagram for View Results 31](#_Toc155179534)

[Sequence Diagram for View Schedule 32](#_Toc155179535)

[Sequence Diagram for Ask Questions 33](#_Toc155179536)

[Sequence Diagram for View Answers 34](#_Toc155179537)

[Sequence Diagram for View Lecture Notes 35](#_Toc155179538)

[3.3 Chapter Summary 36](#_Toc155179539)

[Chapter 4 37](#_Toc155179540)

[Solution Design 37](#_Toc155179541)

[4.1 Introduction 37](#_Toc155179542)

[4.2 Interface Design 37](#_Toc155179543)

[4.3 Database Schema 60](#_Toc155179544)

[4.4 System Requirements 61](#_Toc155179545)

[Chapter 5 62](#_Toc155179546)

[Conclusion 62](#_Toc155179547)

[Project Conclusion 62](#_Toc155179548)

[References 63](#_Toc155179549)

[Appendices i](#_Toc155179550)

[Work Breakdown Chart i](#_Toc155179551)

[Future Enhancements ii](#_Toc155179552)

[Program Code iii](#_Toc155179553)

[Desktop Application Code iii](#_Toc155179554)

[Student Entity Class iii](#_Toc155179555)

[Student Database Class v](#_Toc155179556)

[Manage Student Details Form viii](#_Toc155179557)

[DbConnection Class xvi](#_Toc155179558)

[DbOperations Interface xvii](#_Toc155179559)

[Home Window xvii](#_Toc155179560)

[Student Website Application Code xx](#_Toc155179561)

[Ask Questions Web Page xx](#_Toc155179562)

[View Results Web Page xxii](#_Toc155179563)

[View Notes Web Page xxiv](#_Toc155179564)

[Mark Attendance Page xxvi](#_Toc155179565)

# Chapter 1

# Introduction

## Introduction to the Organization

Lifeway Education Services, located in Nuwara Eliya, has been providing ICT education for children for over a decade. They cover everything from basic computer systems to advanced programming, helping students to navigate and learn everything that they need to know about ICT.

Dhammika Thisara Sanchiarrachi, who manages the institution all by himself, also is the lecturer for his students. Managing both the institution and the lessons has been more difficult than ever, as more and more students enroll in his classes each month. With over 50 students currently enrolled in his lessons, and new groups enrolling each month, his workload has increased to a point where he can’t manage it all by himself.

The one thing that takes more time out of his day except teaching, is record-keeping tasks. Manually entering, updating, removing, and searching for records from a long list of students has been more difficult than ever before.

So, to give a solution for this problem we decided to build an application to better manage his day-to-day record-keeping tasks.

## Organizational Structure

In the current organizational structure, Dhammika Thisara Sanchiarrachi manages the entire operation of the Institute all by himself. Including recording data, maintaining them, and everything else.

## Current Operation Method

Currently, all data and information regarding every part of the organization are recorded and handled manually. Such as registering students, marking attendance, viewing student details, recording payments, etc.

## Users and Responsibilities of Organization

Currently, the only users of the organization are the client and the students who come to learn in the institute.

Responsibilities of client

* Maintain all the records of the organization.
* Perform day-to-day checking of the records.

Responsibilities of students

* There are no identified responsibilities of students

## 1.5 Problem Definition

**Time Consuming manual data manipulation**

The main problem that the client faces is to maintain all the records by hand.

Which takes more time than he likes to spend on it, resulting in reduced time that he can use to teach kids.

**Error prone to manual mistakes**

Very prone to have errors when referring and recording details from one book to another.

**Daunting manual search operations**

Searching for a record/records has become a difficult task, as he has to look at a long list of records to find results, and also he has to note down the details somewhere else to look at later.

## Project Objectives

The main objective is to replace almost all the manual record-keeping tasks with an automated computerized system.

## Proposed Solution

The proposed solution is to make a User-Friendly GUI application that the client can use to better manage his record-keeping jobs.

To achieve that, we will provide functionality to the application to automatically store, update, delete, and search for records more quickly, making it easier for him to do his work.

## Chapter Summary

In this chapter we got to know about the purpose for the project, who were building the project for, the structure of the organization, its daily tasks, how they are performed, the people who is involved with it, their responsibilities, and the problems that they face.

Also, we discussed how we can provide a better solution for the problems that they’re facing by providing a tailor-made application to better manage their day-to-day tasks.

**A wall with a logo

Description automatically generated**

Figure 1: Lifeway Education Services

# Chapter 2

# Methodology

## 2.1 Introduction

In this chapter you will get to know about the project plan, data collection methods, software development method, software development tools, testing strategies, and the implementation plan that will be used for the application.

## 2.2 Project Plan

First, an interview with the client was conducted to understand the needs for the system. After that, a field study was conducted to fine tune his requirements. Next, past records of the institute were reviewed to collect what type of data he wants to be stored. After that, an SRS document was created for review. After it was accepted by the client, we began to build the system.

According to the requirements, designs were created to help us build the system. The components of the system were identified by understanding the organization and its operations. By using various diagrams, we were able to map out the organization clearly.

Then after that, the implementation phase began according to the designs. Various bugs and errors were identified along the way, but we resolved them and moved forward.

Finally, after passing through a thorough testing phase and making sure the application meets all the requirements, the system was successfully implemented in the client’s environment.

The below plan was followed to complete the project.

|  |  |
| --- | --- |
| Gathered the user requirements | 1 week |
| Conducted a field study and reviewed past records | 1 week |
| Prepared the SRS | 1 week |
| Designed the System | 2 weeks |
| Implemented the System | 2 weeks |
| Conducted tests of the System | ½ week |
| Prepared the documentation | 2 ½ weeks |
| Finalized the system | 2 weeks |

Table : Project Timeline

## 2.3 Data Collection Methods

We gathered all the required information by using,

* Interviews with the client
* Observations of the Organization
* Reviewing Past Records of the organization

## 2.4 Software Development Method

To make the application as best as we can,we chose to follow the Waterfall model to guide us through the Software Development Life Cycle. We chose this model because it helps us to go through all the SDLC phases, phase by phase without compromising the quality of each stage.

Based on that first, an interview and a field study of the organization was conducted to better understand the requirements for the system and to understand what can be converted into an automated system and what cannot.

After that, requirements were confirmed with the client to make sure that the proposed system meets all his needs.

After that, the System was designed using Flow Charts, DFDs, Use Case Diagrams, Sequence Diagrams, ER Diagrams and more.

Next, the implementation began. While having a strict testing method in place to help us achieve the expected quality of the system.

After the testing was completed, and making sure that everything works according to the client’s needs and expectations. Working System was handed over to the Client with all the documentation.

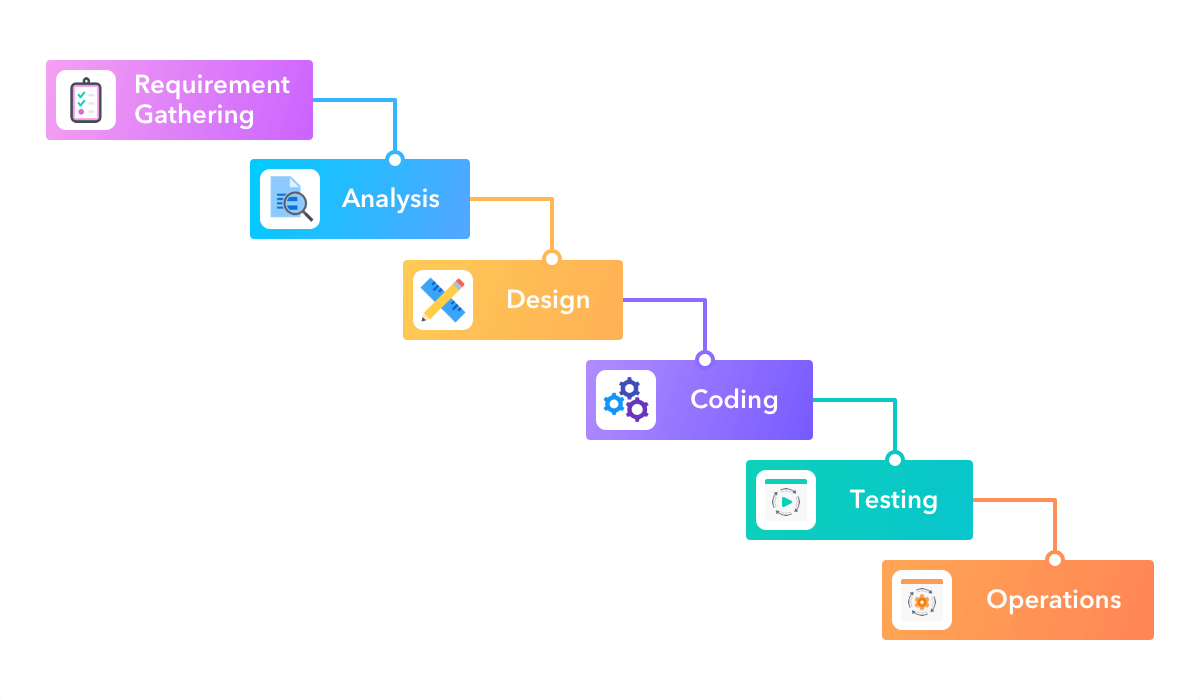
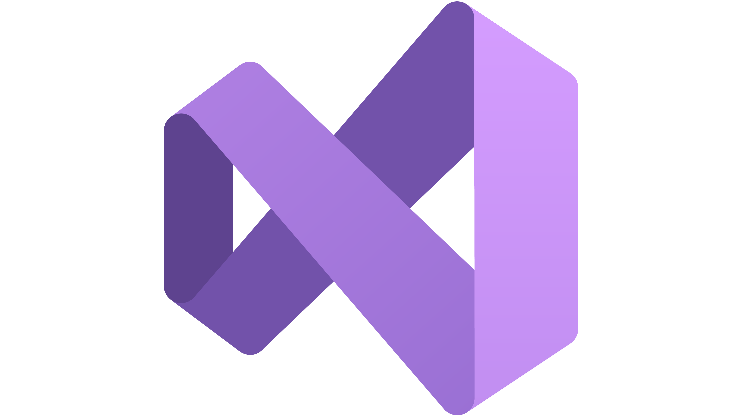
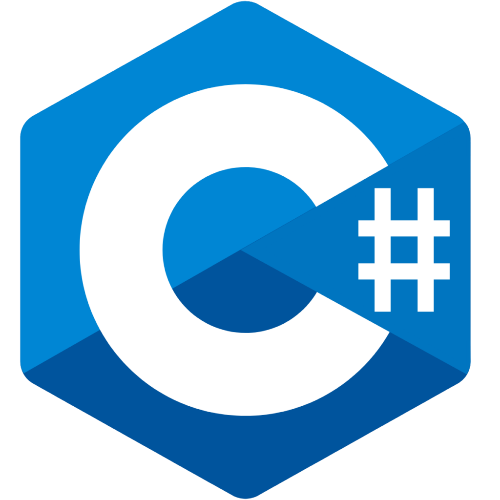


Figure 2: Software Development Method (Waterfall Method)

## 2.5 Software Development Tools

For our client’s application we decided to build it using C# and we used Visual Studio IDE as the development environment.

And for the students’ website, we decided to build it using HTML, CSS, and PHP, also we used Visual Studio Code as development environment.



**A dolphin and text on a white background

Description automatically generated**A logo of a website

Description automatically generated

A blue and white logo

Description automatically generated

A blue and black logo

Description automatically generated

Figure : Technologies used.

## 2.6 Testing Methods

Throughout the implementation process we tested every module, class, and integration to ensure our system performs as expected.

We conducted,

* White Box Testing
* Black Box Testing

for every part of the code.

## 2.7 Implementation plan

We plan to implement the system in the client’s environment by following the parallel method which will allow us to use the current system and the new system at the same time for a planned amount of time which allows us to make sure our application performs exactly as planned. Also, we will use this time to identify any new improvements for the application before we completely implement it in the client’s environment.

The advantage of using this method is that, if we encounter a problem in this time, we can quickly refer to the old system. Without having to stop operations immediately.

## 2.8 Chapter Summary

In this chapter we discussed the project plan, data collection methods, software development method, software development tools, testing strategies, and the implementation plan for the application.

# Chapter 3

# Analysis

## 3.1 Introduction

In this chapter includes diagrams which shows the current operational methods of the organization, the proposed operations of the organization, and the blueprint for the new system and for its components.

## 3.2 Diagrams

Here we will see the diagrams that explain the current system and the proposed system for the client.

### Use case diagram for the current system

A diagram of a person

Description automatically generated

*This diagram shows the current system in place and it’s uses*

Figure : Use Case Diagram for the current system

### Use Case Diagram for the proposed system.

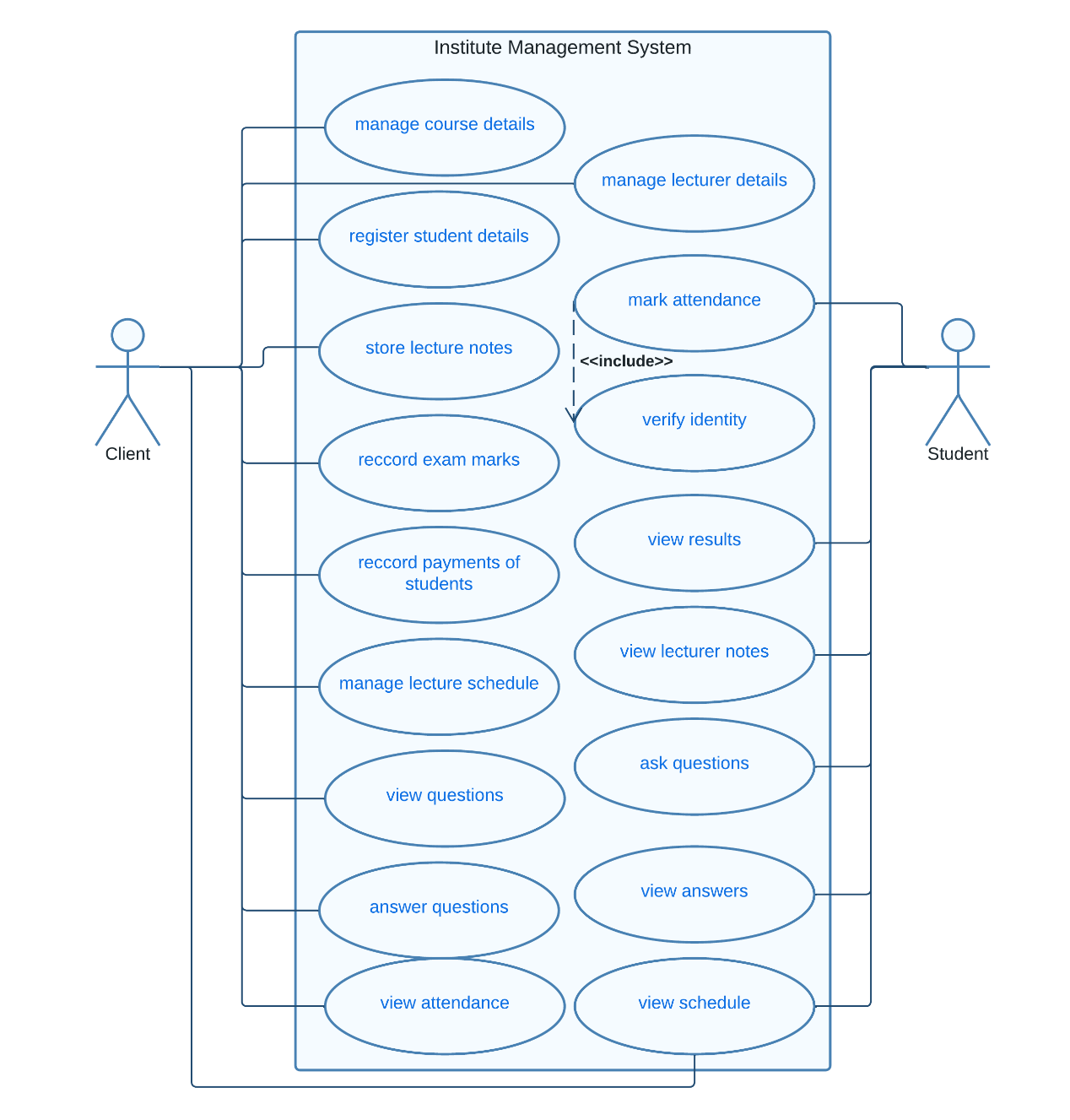
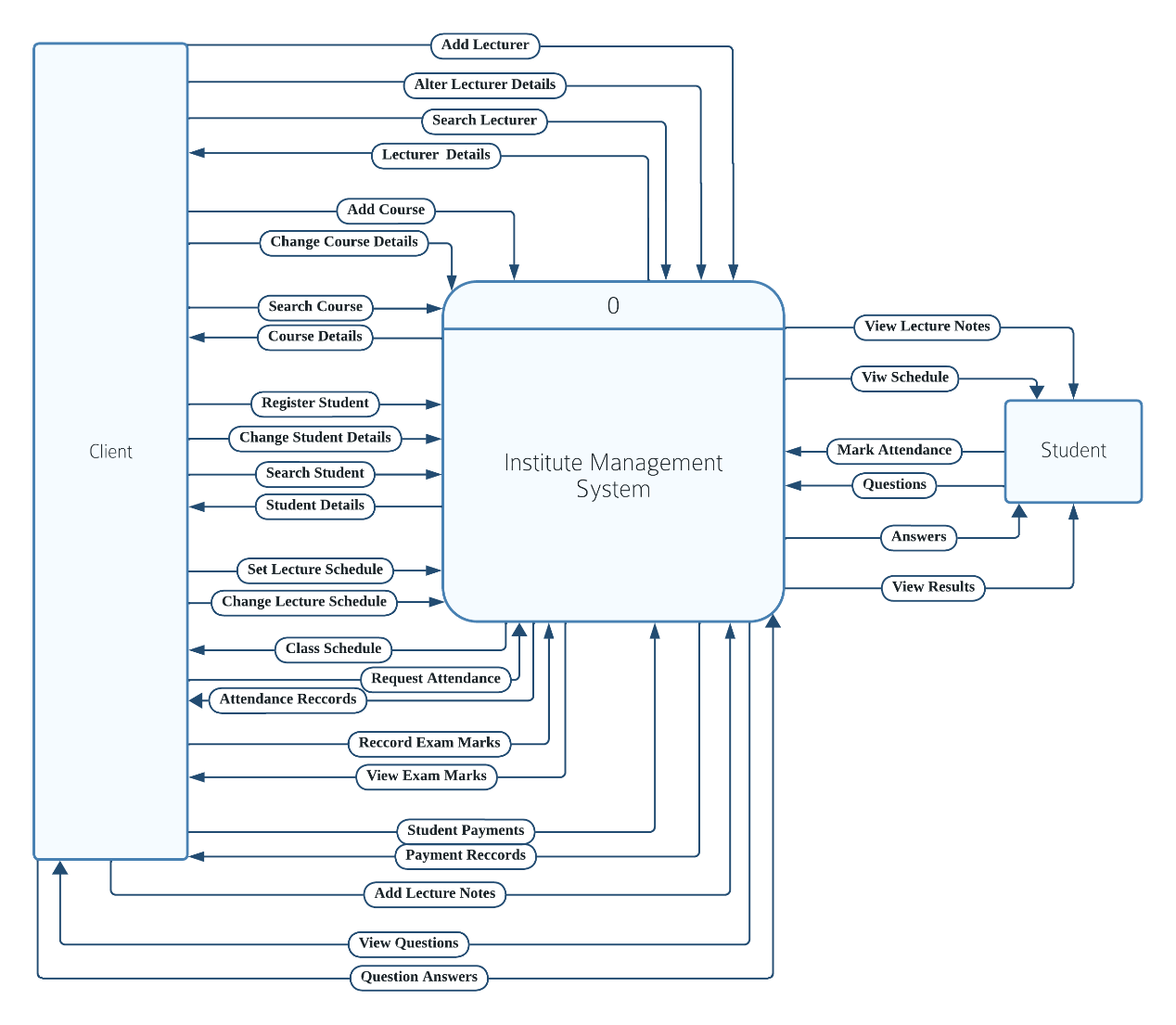


Figure : Use Case Diagram for the proposed system.

*This diagram shows the proposed system and it’s uses*

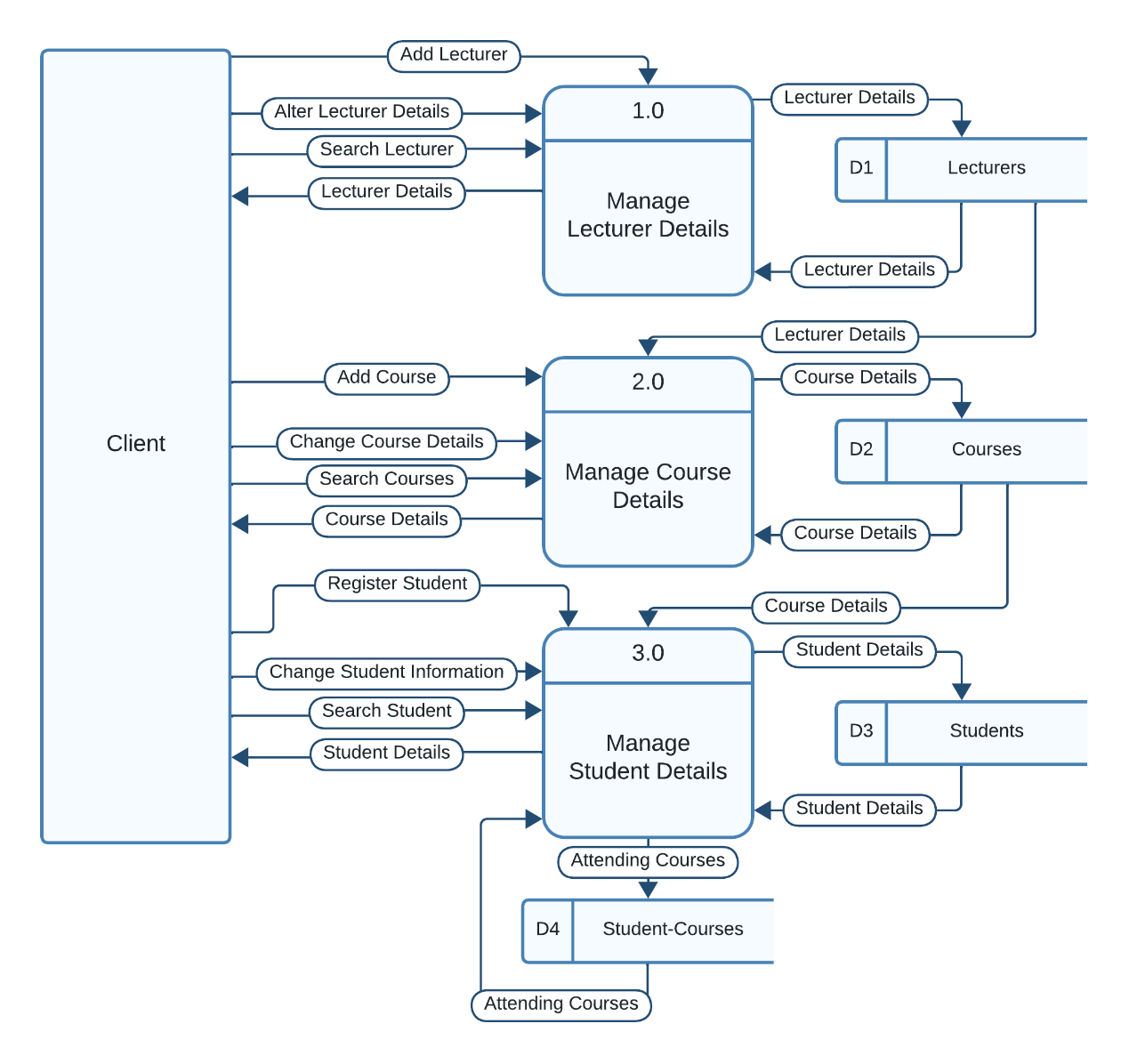
### DFD Diagram (Context Level)



*This diagram shows the overall operations of the system and their related data flows*

Figure : DFD Diagram (Context Level)

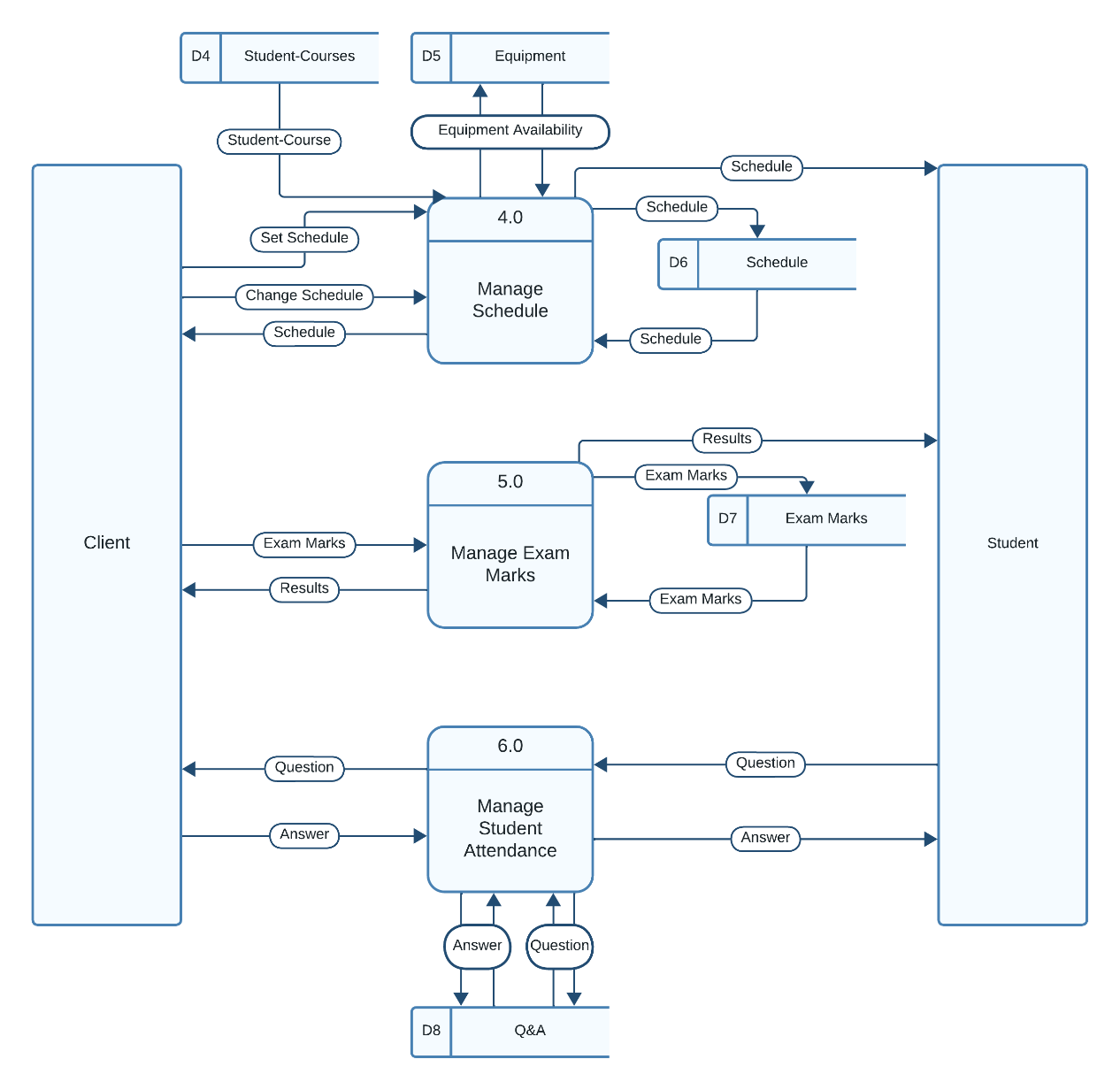
### DFD Diagram (Level 1)



*This diagram shows the flow of data when managing lecturer details, course details, student details. In operations like, inserting, updating, deleting, or searching for a record.*

Figure : DFD Diagram (Level 1)

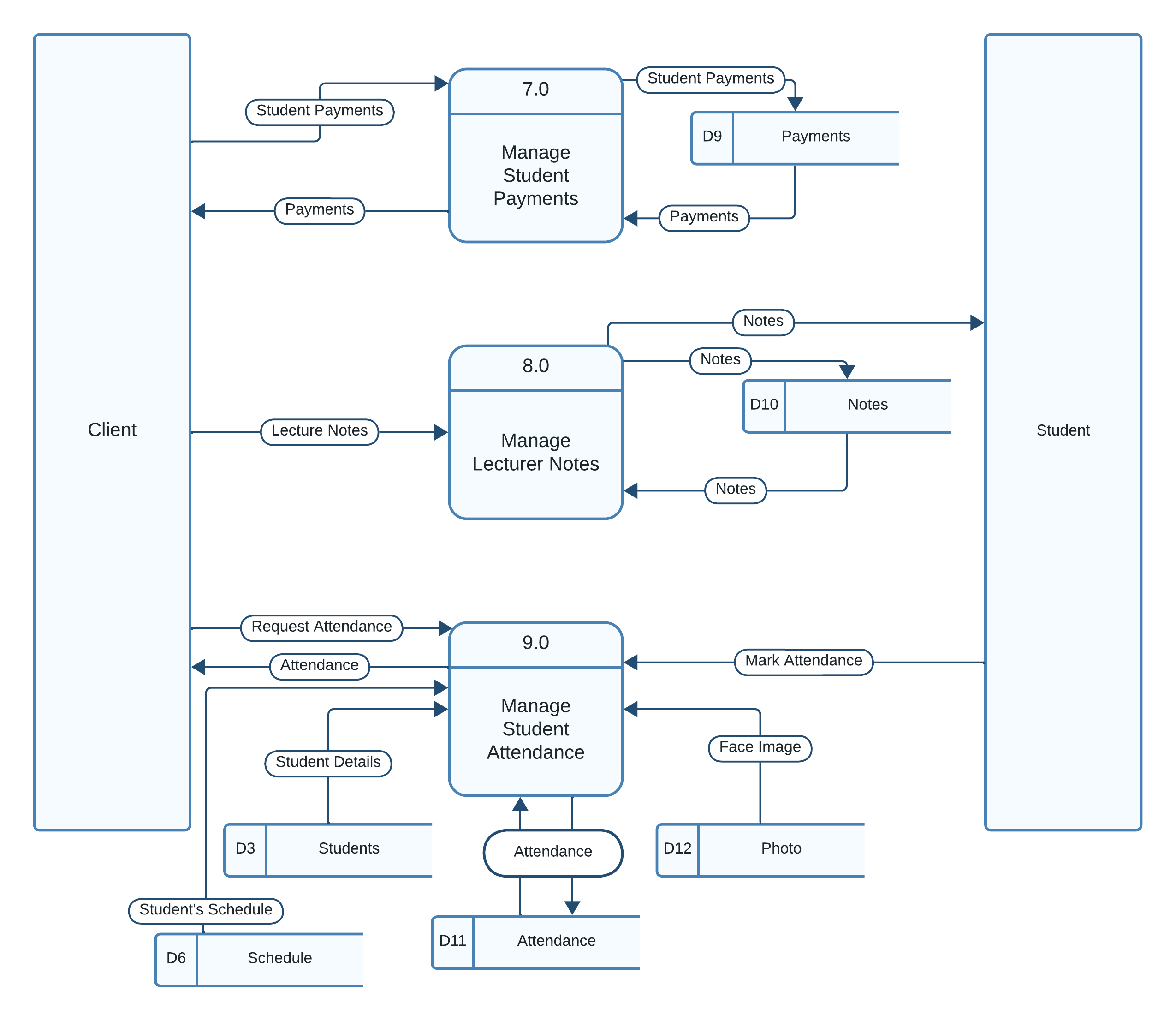
### DFD Diagram (Level 1)



*This diagram shows the flow of data when managing schedule details, exam details, student questions. In operations like, inserting, updating, deleting, or searching for a record.*

Figure : DFD Diagram (Level 1)

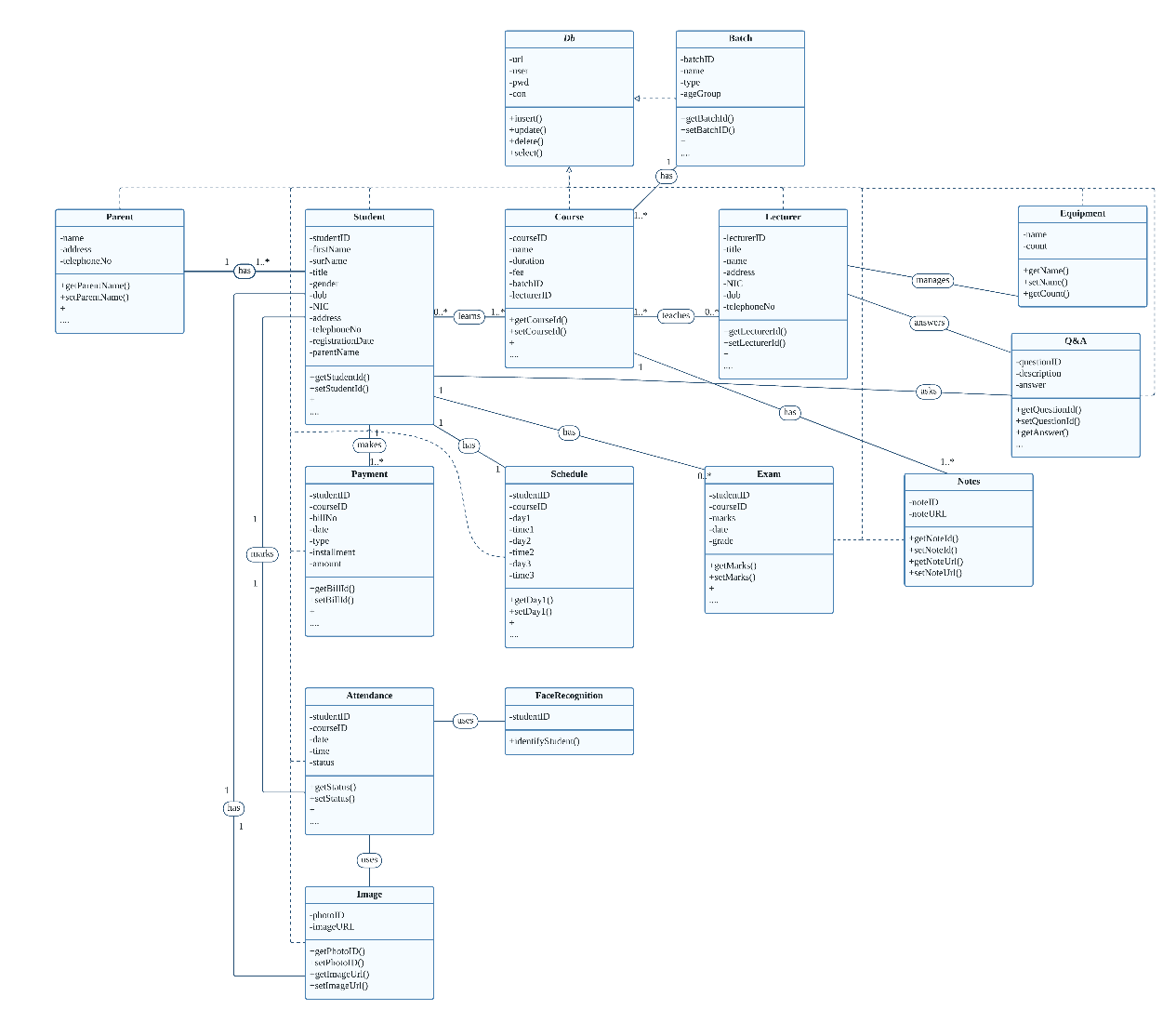
### DFD Diagram (Level 1)



*This diagram shows the flow of data when managing payment details, lecture notes, student attendance. In operations like, inserting, updating, deleting, or searching for a record.*

Figure : DFD Diagram (Level 1)

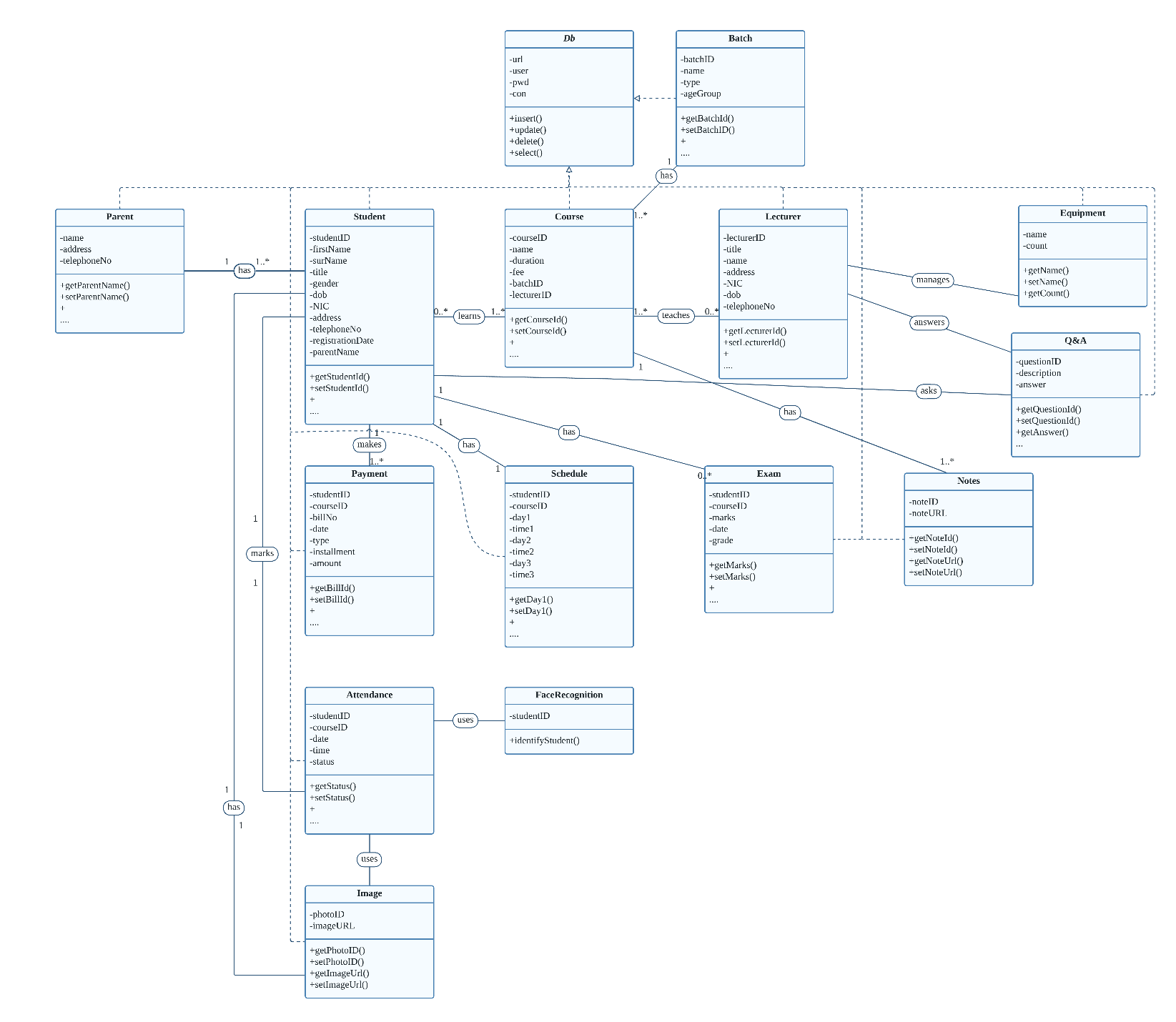
### Class Diagram



*This diagram shows all identified classes that is needed to implement the system. By using these classes we can achieve the expected functionality of the program.*

Figure : Class Diagram

### Class Diagram



*This diagram shows all the classes that is implemented in the system to achieve the functionality of the application. By using these classes we can achieve the expected functionality of the program.*

Figure : Class Diagram

### ER Diagram

A diagram of a flowchart

Description automatically generated

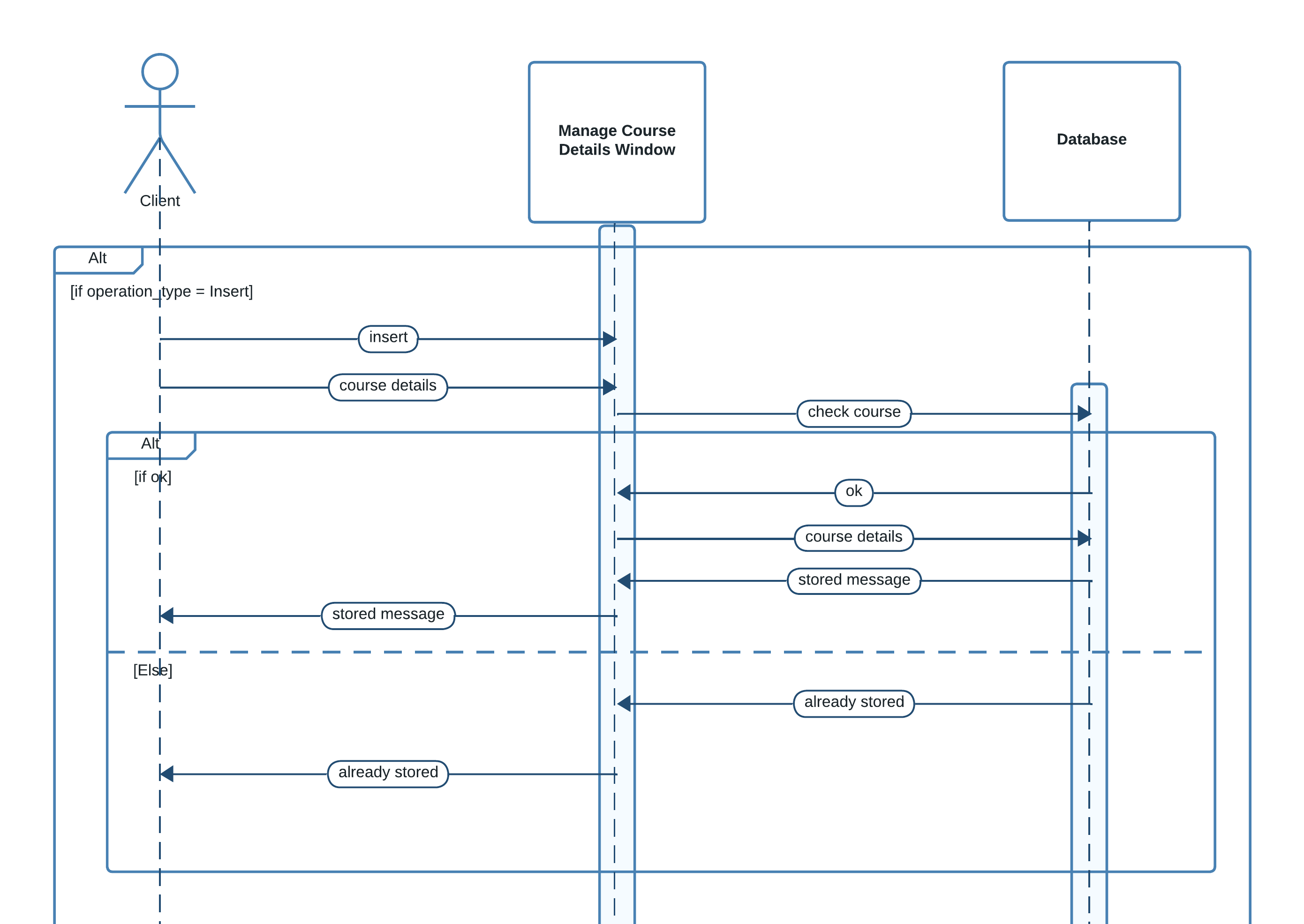
*This diagram shows all the entities present and their relationships.*

Figure : ER Diagram

exam



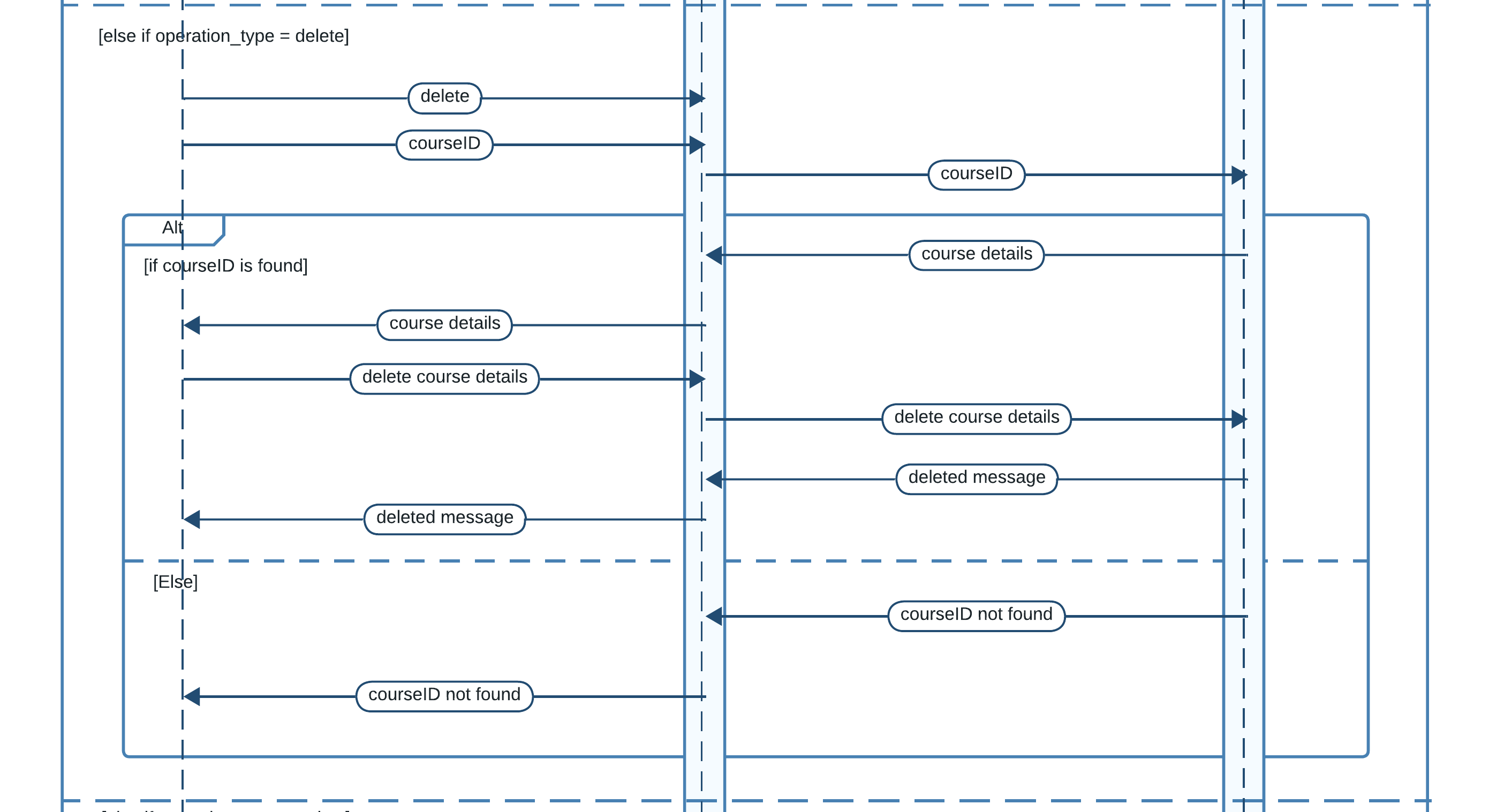
### Sequence Diagram for Manage Course Details

A diagram of course details

Description automatically generated

Figure : Sequence Diagram (Manage Course Details)

*This diagram shows the order of processes when a lecturer is about to insert, update, delete, or search for course details.*

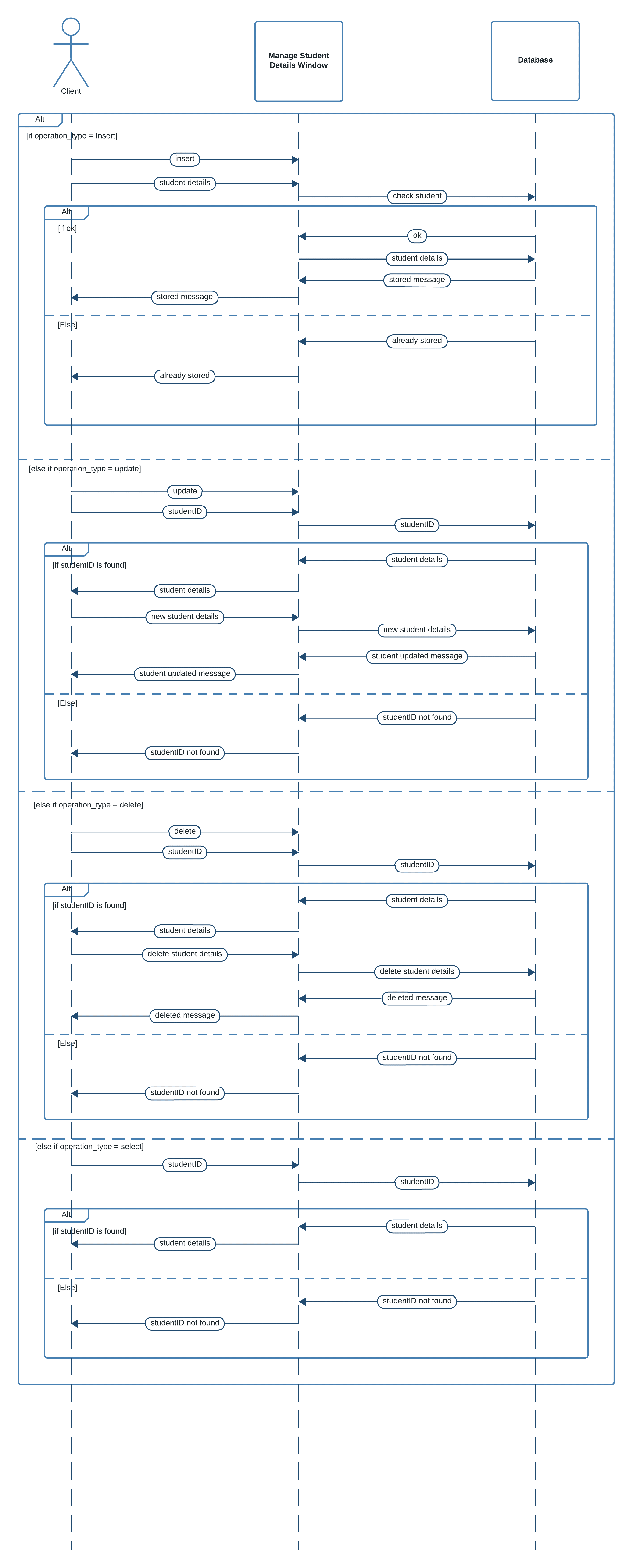
A diagram of a course

Description automatically generated

Figure : Sequence Diagram (Manage Course Details)

*This diagram shows the order of processes when a lecturer is about to insert, update, delete, or search for course details.*

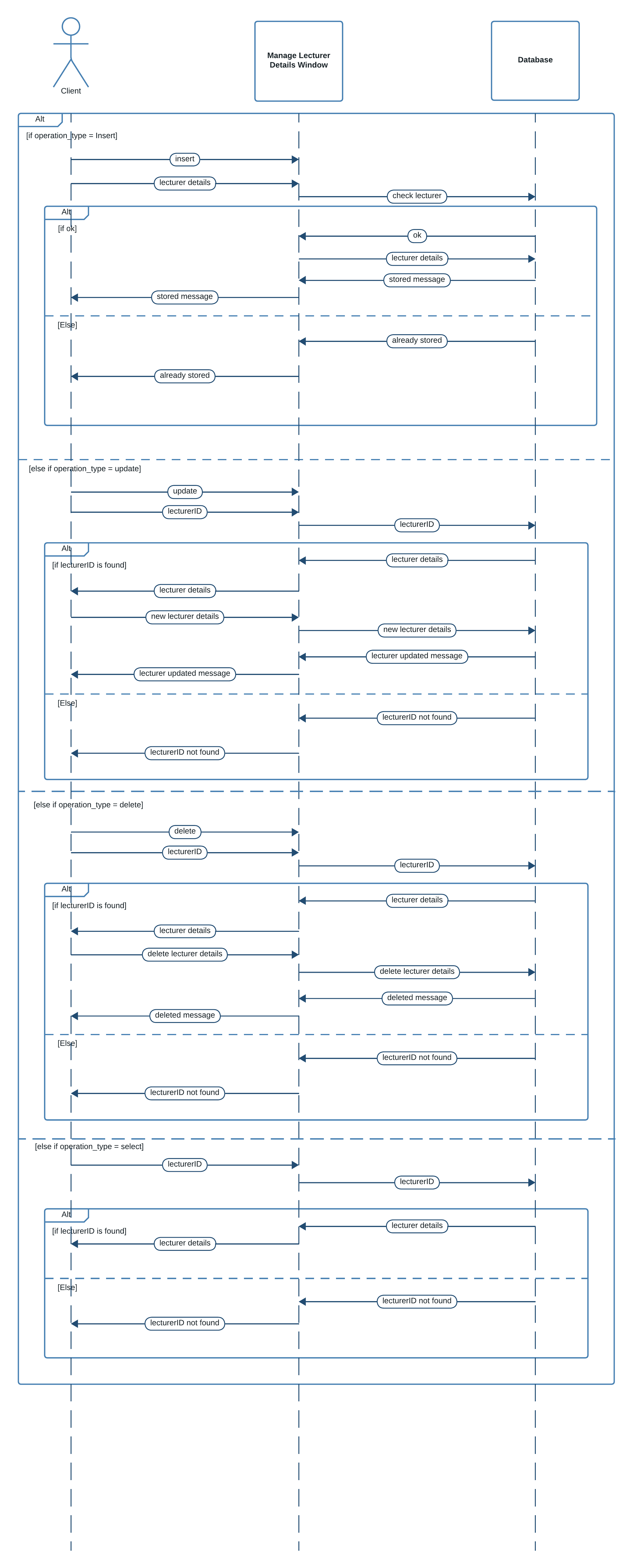
### Sequence Diagram for Manage Student



*This diagram shows the order of processes when a lecturer is about to insert, update, delete, or search for student details.*

Figure : Sequence Diagram (Manage Student Details)

### Sequence Diagram for Manage Lecturer



*This diagram shows the order of processes when a lecturer is about to insert, update, delete, or search for lecturer details.*

Figure 16: Sequence Diagram (Manage Lecturer Details)

### Sequence Diagram for Manage Notes

*This diagram shows the order of processes when the lecturer is about to store lecture notes.*

Figure : Sequence Diagram (Store Notes)

### Sequence Diagram for Manage Exam Marks

### Sequence Diagram for Manage Schedule

*This diagram shows the order of processes when the lecturer is about to store student exam marks*

Figure : Sequence Diagram (Store Exam Marks)

A diagram of a software project

Description automatically generatedA diagram of a schedule

Description automatically generated

Figure : Sequence Diagram for Manage Schedule

*This diagram shows the order of processes when a lecturer is about to insert, update, delete, or search for schedule details.*

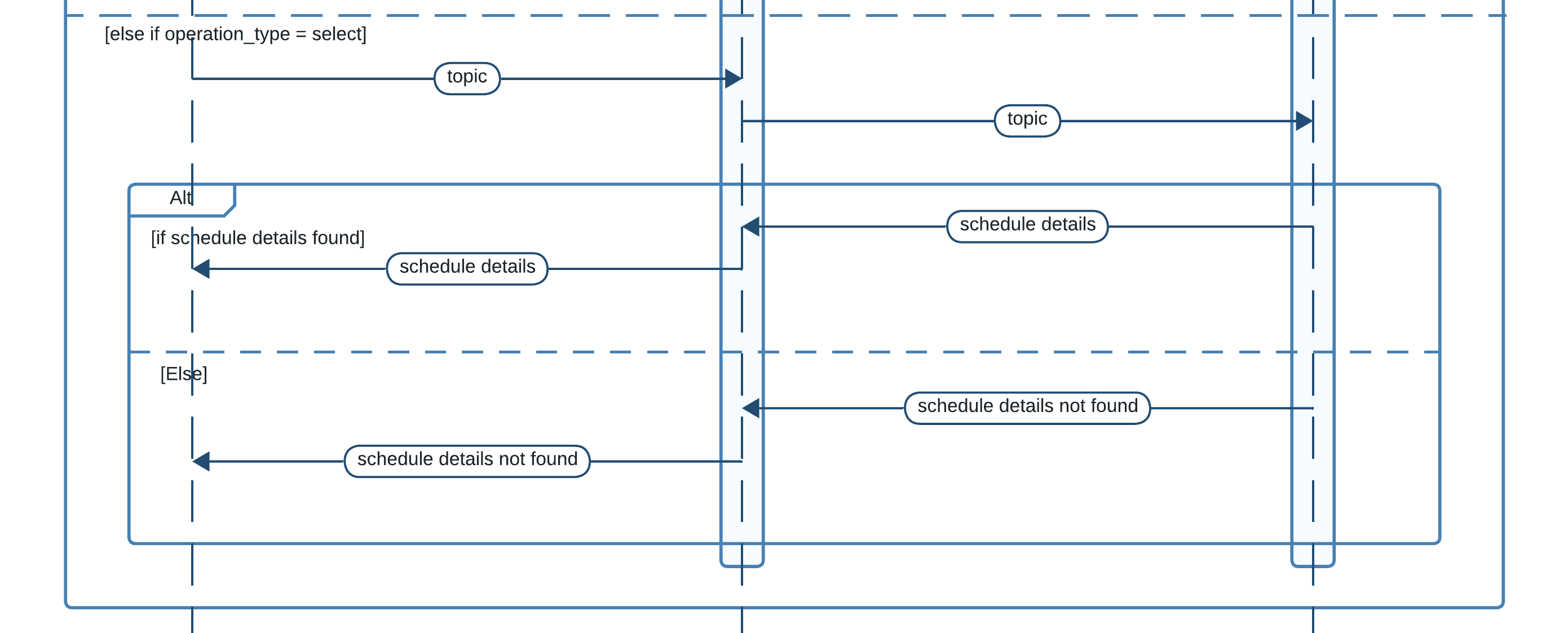
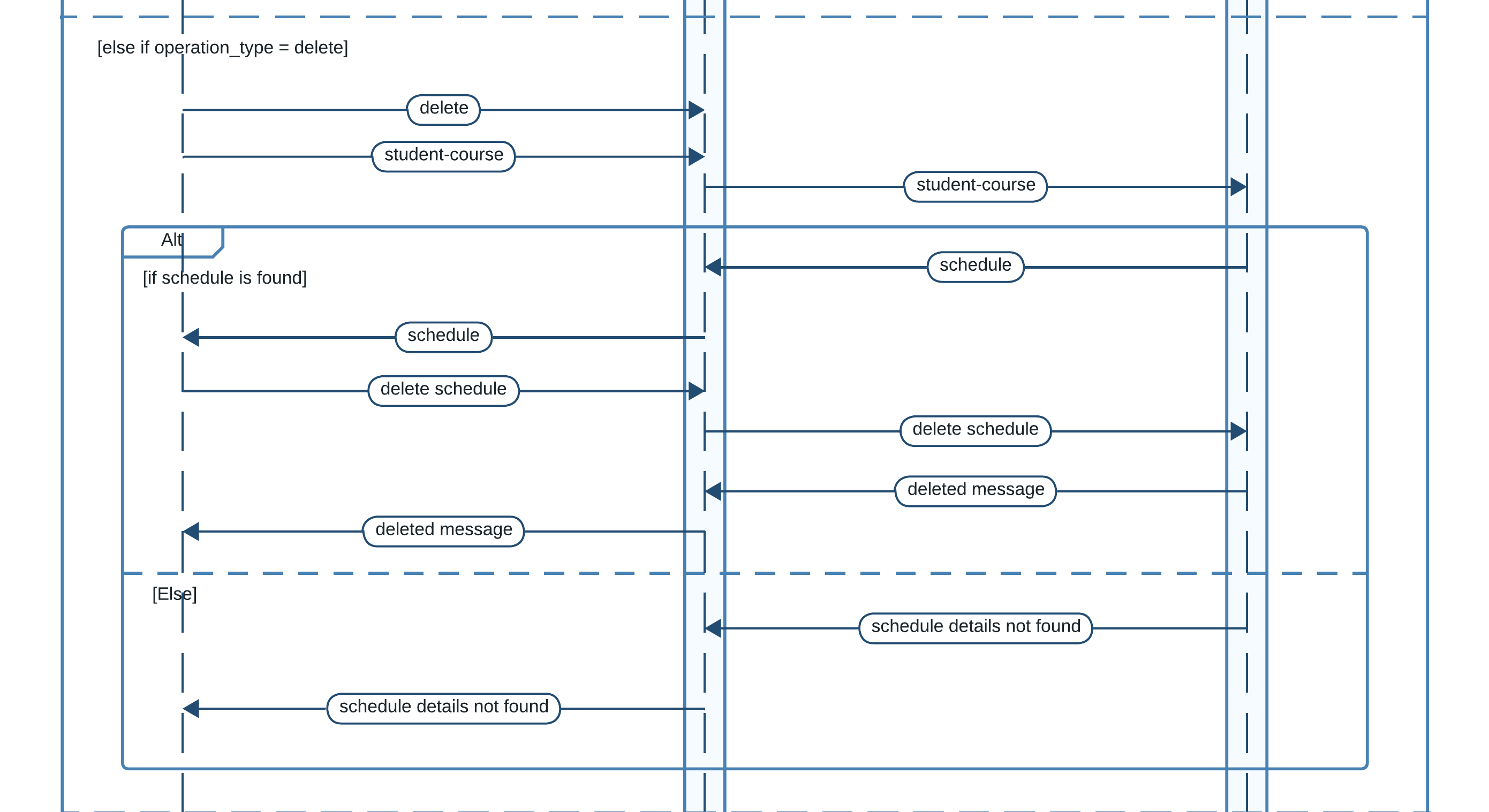


Figure : Sequence Diagram for Manage Schedule

*This diagram shows the order of processes when a lecturer is about to insert, update, delete, or search for schedule details.*

### Sequence Diagram for Manage Payments

*This diagram shows the order of processes when the lecturer is storing payment details*

Figure : Sequence Diagram (Store payments)

### Sequence Diagram for View Attendance

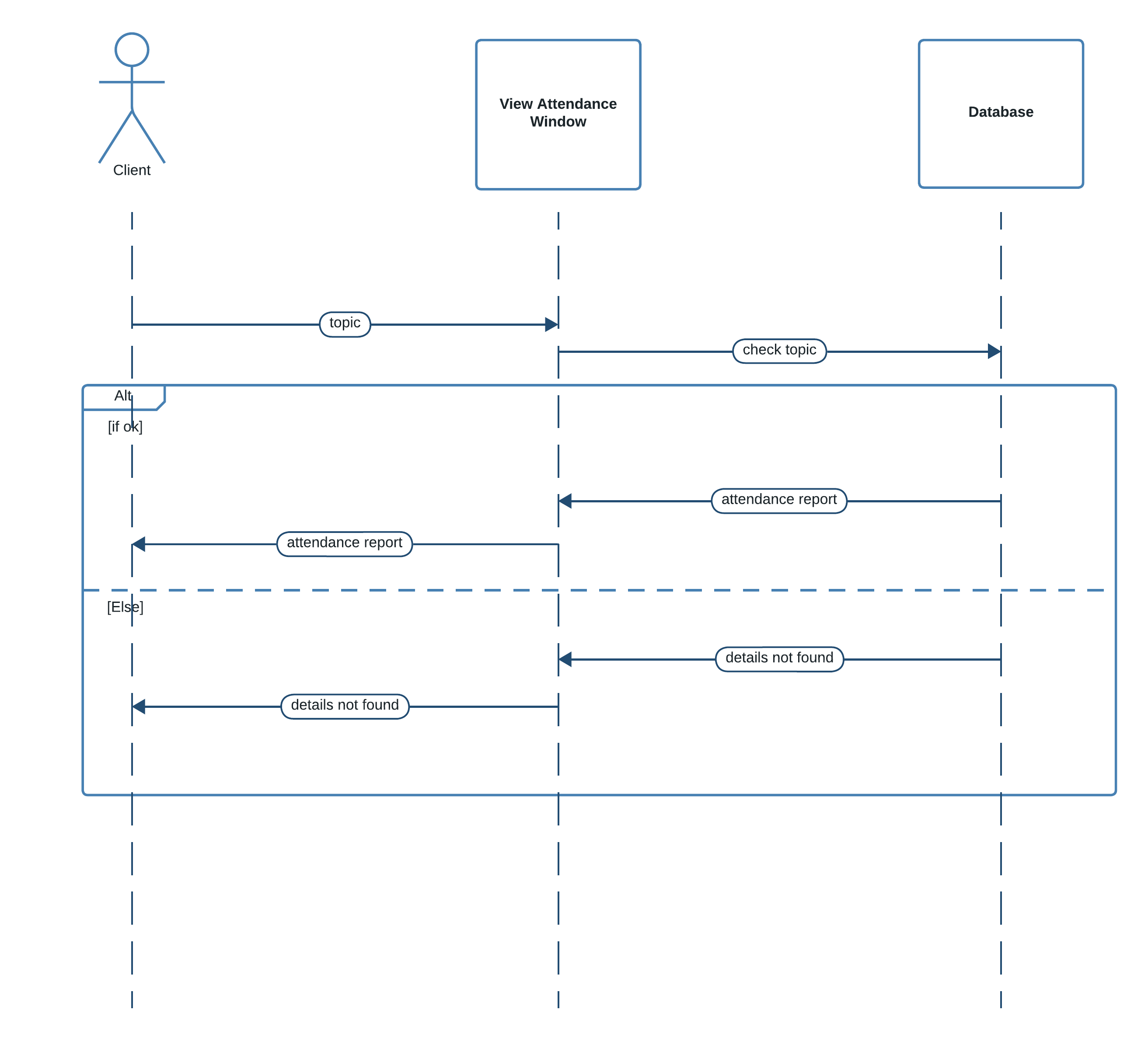


Figure : Sequence Diagram (View Attendance)

*This diagram shows the sequences when the lecturer is viewing student attendance*

### Sequence Diagram for View Questions

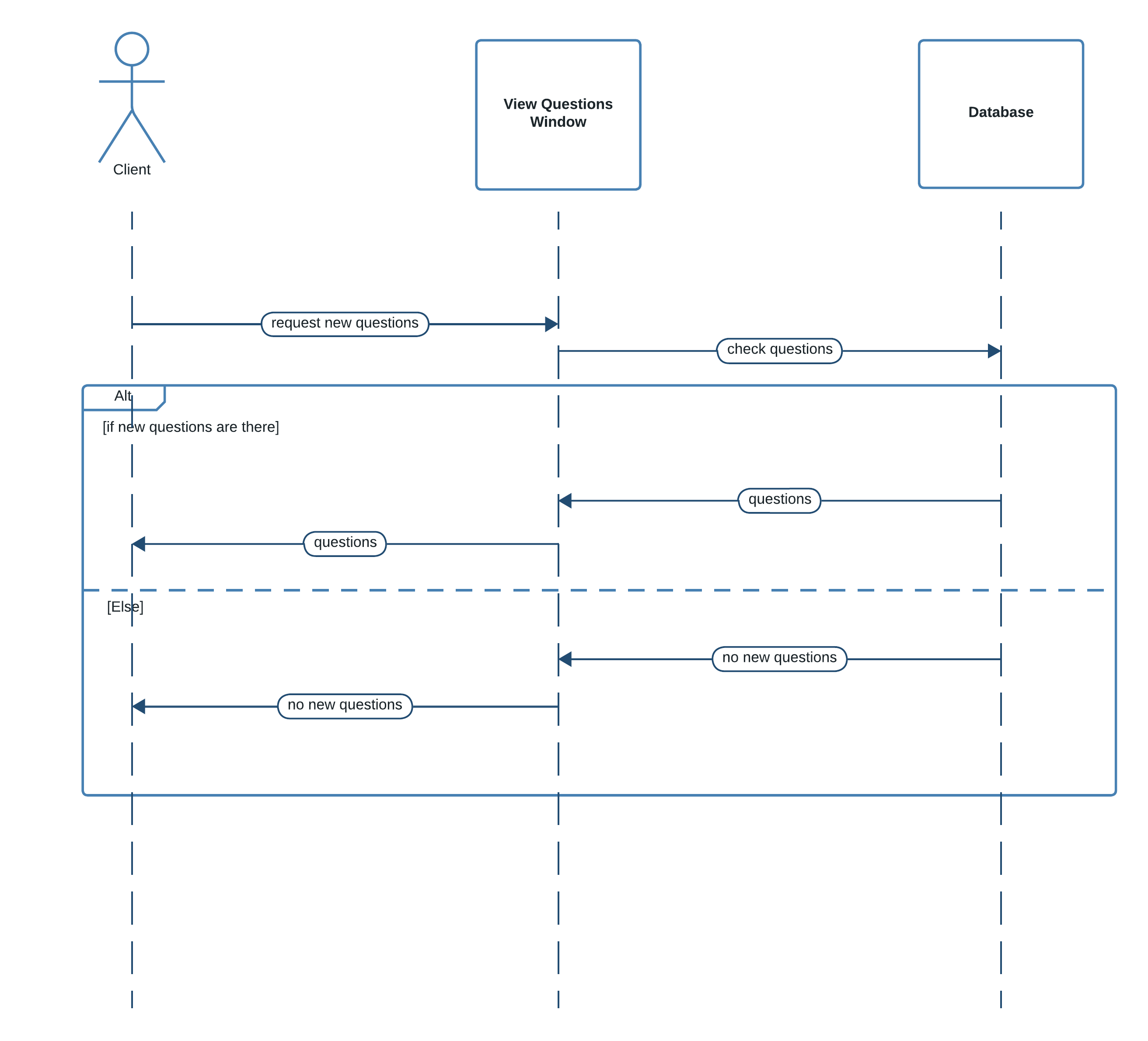


Figure : Sequence Diagram (View Questions)

*This diagram shows the order of processes when the lecturer is viewing student questions*

### Sequence Diagram for Answer Questions

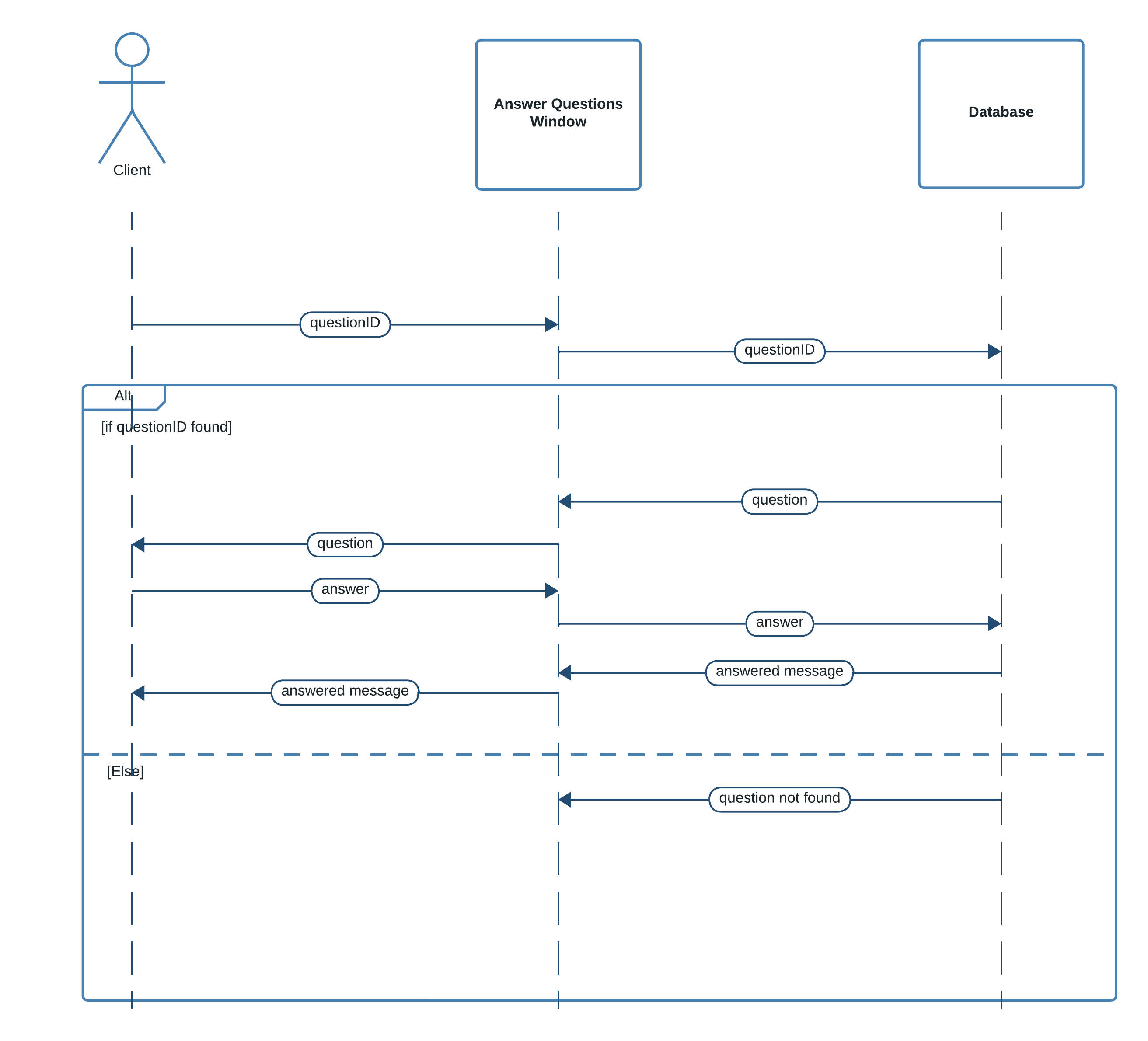


Figure : Sequence Diagram (Answer Questions)

*This diagram shows the order of processes when the lecturer answers student questions.*

### Sequence Diagram for View Results

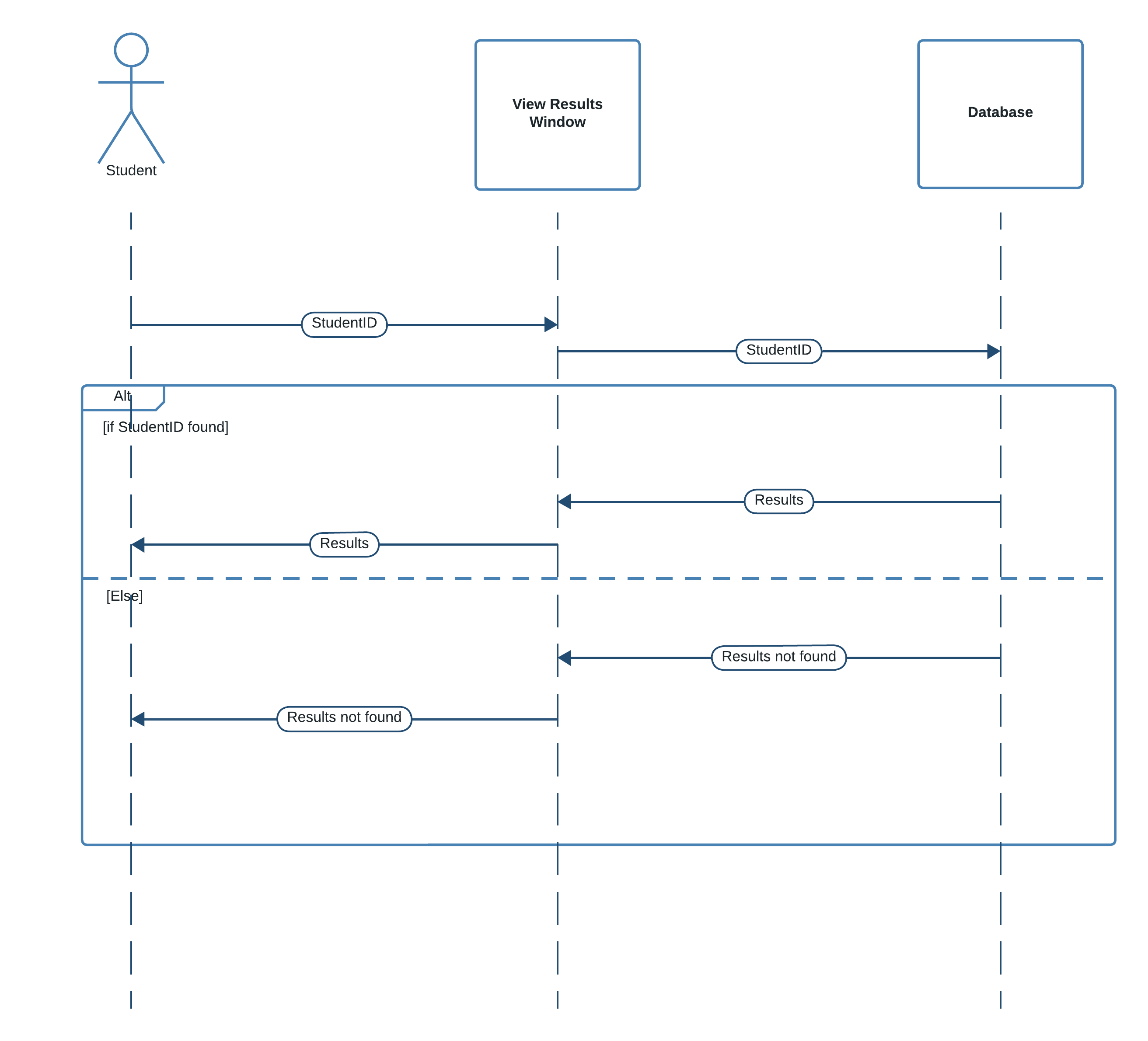


Figure : Sequence Diagram for View Results

*This diagram shows the order of processes when students request to view exam results.*

### Sequence Diagram for View Schedule

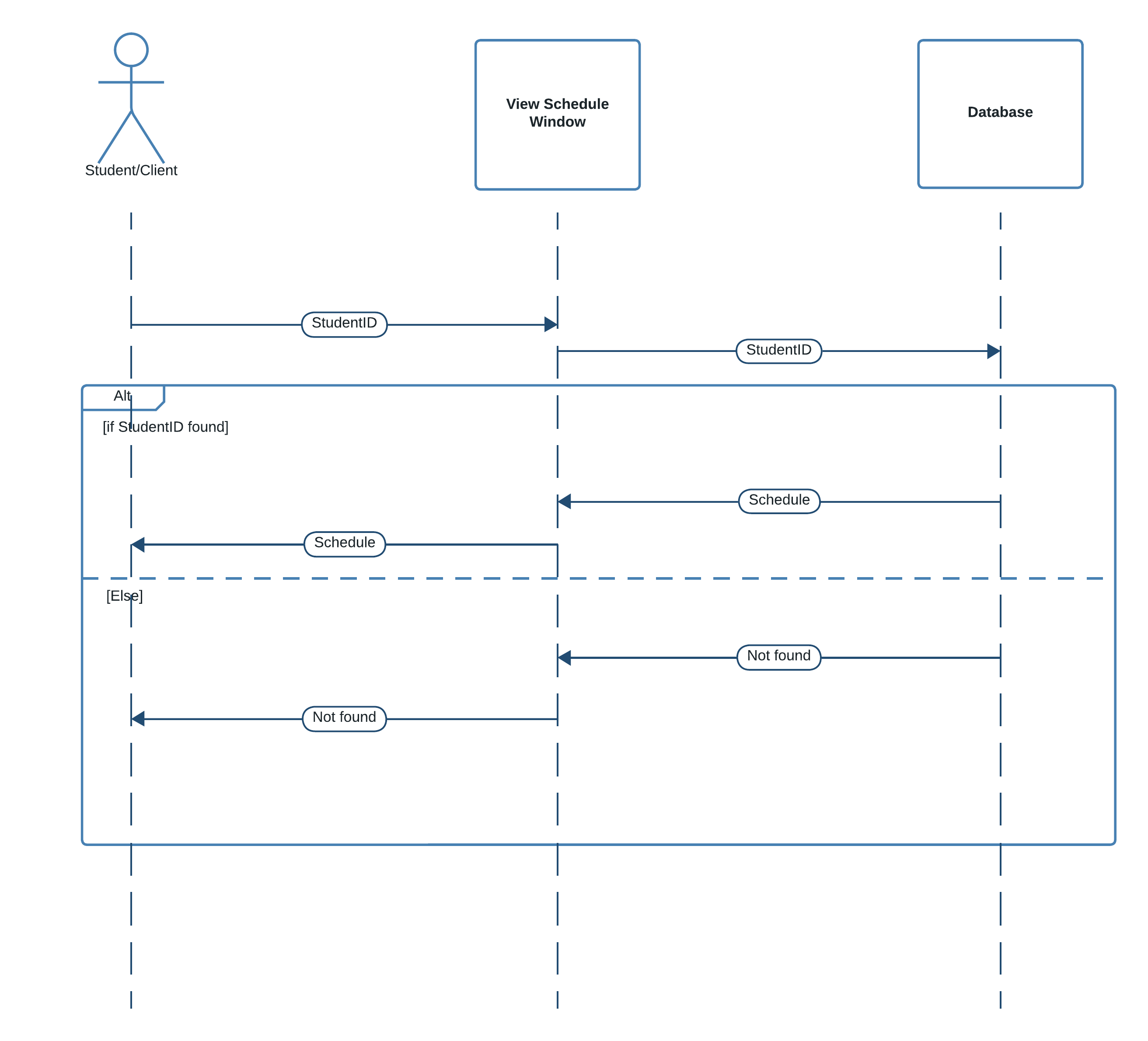


Figure : Sequence Diagram for View Schedule

*This diagram shows the order of processes when students request to view the lecturer schedule*

### Sequence Diagram for Ask Questions



Figure : Sequence Diagram for Ask Questions

*This diagram shows the order of processes when students are asking questions*

### Sequence Diagram for View Answers

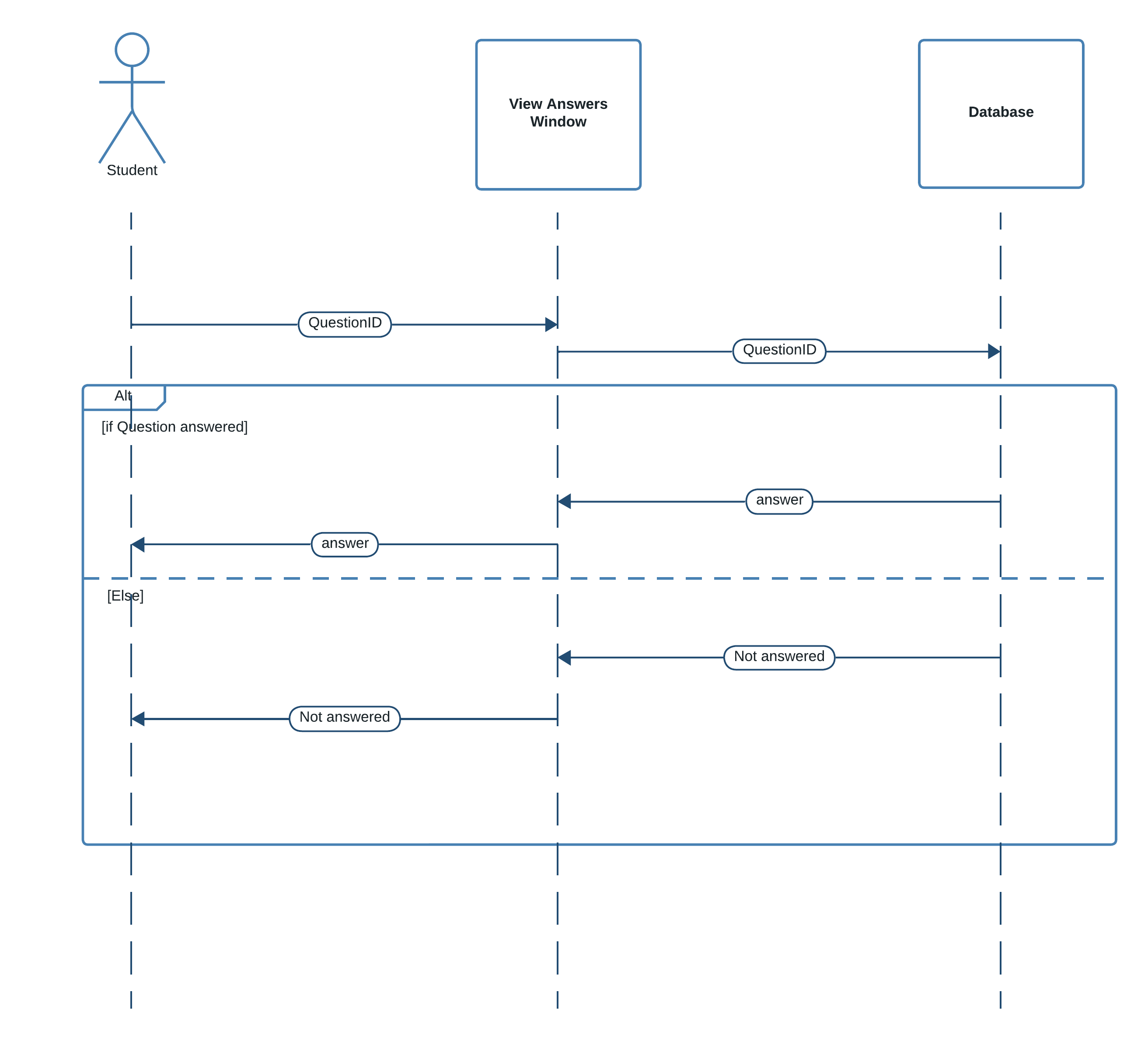


Figure : Sequence Diagram for View Results

*This diagram shows the order of processes when students requesting to view answers for questions.*

### Sequence Diagram for View Lecture Notes

A diagram of a person

Description automatically generated

Figure : Sequence Diagram for View Lecture Notes

*This diagram shows the order of processes when students are requesting to view lecturer notes.*

## 3.3 Chapter Summary

In this chapter we looked at the designs of the system which helped us to create the application. The diagrams made it easy to implement the program because we knew what to implement and how to implement it exactly.

# Chapter 4

# Solution Design

## 4.1 Introduction

In this section we will be looking at the Interface Designs, and Database Schema Design of the System

## 4.2 Interface Design

|  |  |
| --- | --- |
| **Interface number** | **Interface name** |
| 1 | Login Screen |
| 2 | Course Details |
| 3 | Student Details |
| 4 | Lecturer Details |
| 5 | Payment Details |
| 6 | Set Schedule |
| 7 | Store Lecture Note |
| 8 | Store Exam Marks |
| 9 | Ask Questions |
| 10 | View Results |
| 11 | View Notes |
| 12 | View Questions |
| 13 | View Attendance |
| 14 | View Schedule |
| 15 | Home Screen |
| 16 | Parent Details |

Table : Table of Interfaces

Interface No: 01

Interface Name: Login Screen

Description: Used to login to the system

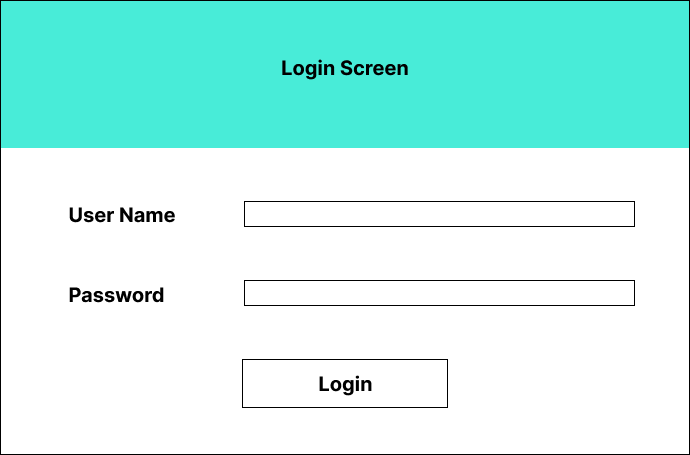


Figure : Login Page

Pseudo Code

Figure : Pseudo code for login

START

//VALIDATIONS

INPUT USERNAME

INPUT PASSWORD

IF USERNAME == “DHAMMIKA” && PASSWORD == “1234”

THEN

OUTPUT LOGIN SUCCESSFUL

SHOW HOME PAGE

ELSE

OUTPUT USERNAME OR PASSWORD IS INVALID

END IF

END

Interface No: 02

Interface Name: Course Details

Description: Used to handle all the Course Details related operations

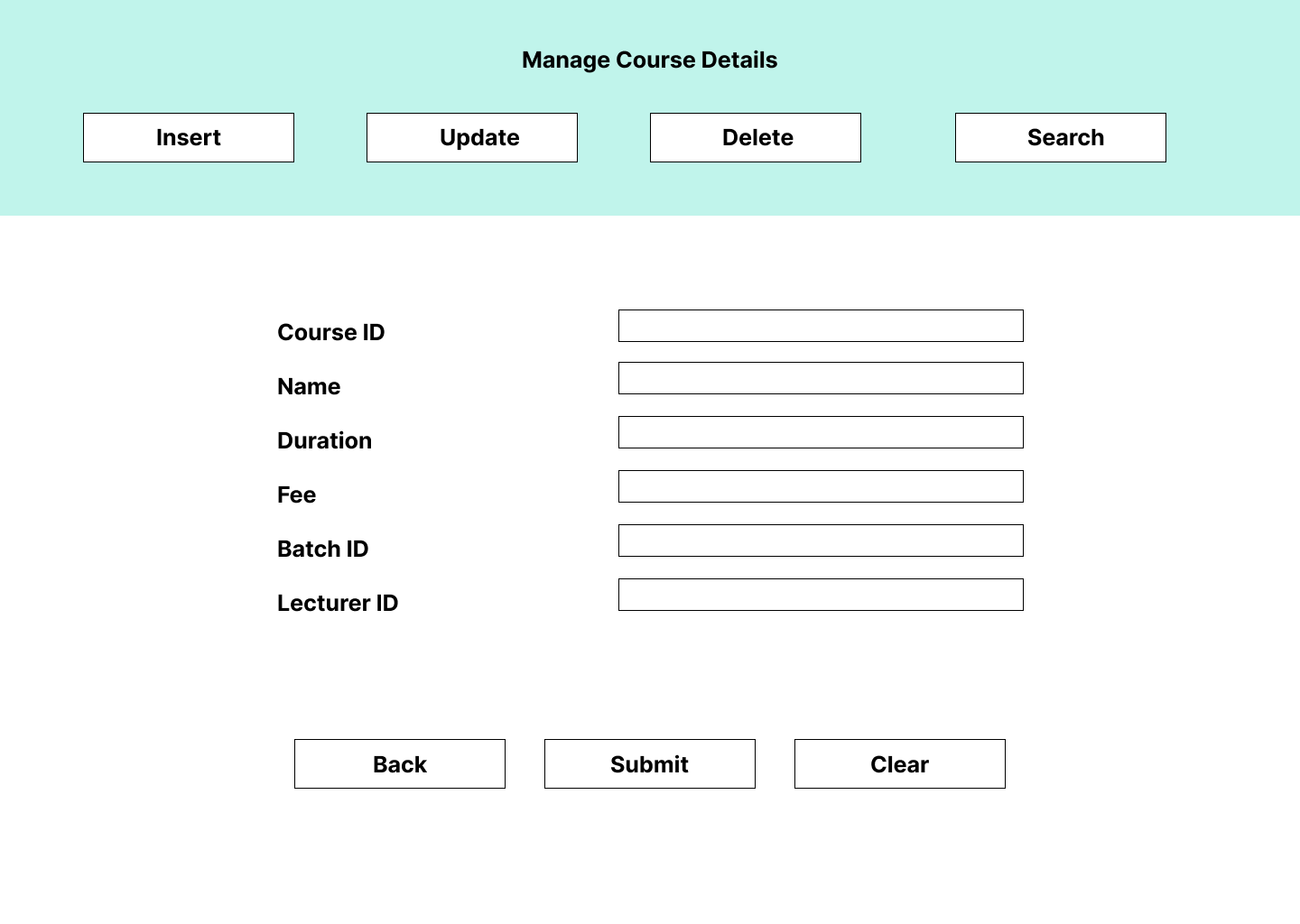


Figure : Manage Course Details Screen

Pseudo Code

Figure : Pseudo code for course management

START

//VALIDATIONS

OPEN DBCON

IF INSERT\_CLICKED

THEN

executeQuery(INSERT INTO COURSE (“COURSEID”,…))

ELSE

IF UPDATE\_CLICKED

THEN

executeQuery(UPDATE COURSE SET FEE = “2000.00” WHERE COURSEID =”C0001”)

ELSE

IF DELETE\_CLICKED

THEN

executeQuery(DELETE FROM COURSE WHERE COURSEID=”C0001”)

ELSE

IF SELECT\_CLICKED

THEN

Result = executeQuery(SELECT \* FROM COURSE)

OUTPUT Result

END IF

CLOSE DBCON

END

Interface No: 03

Interface Name: Student Details

Description: Used to handle all the Student Details related operations

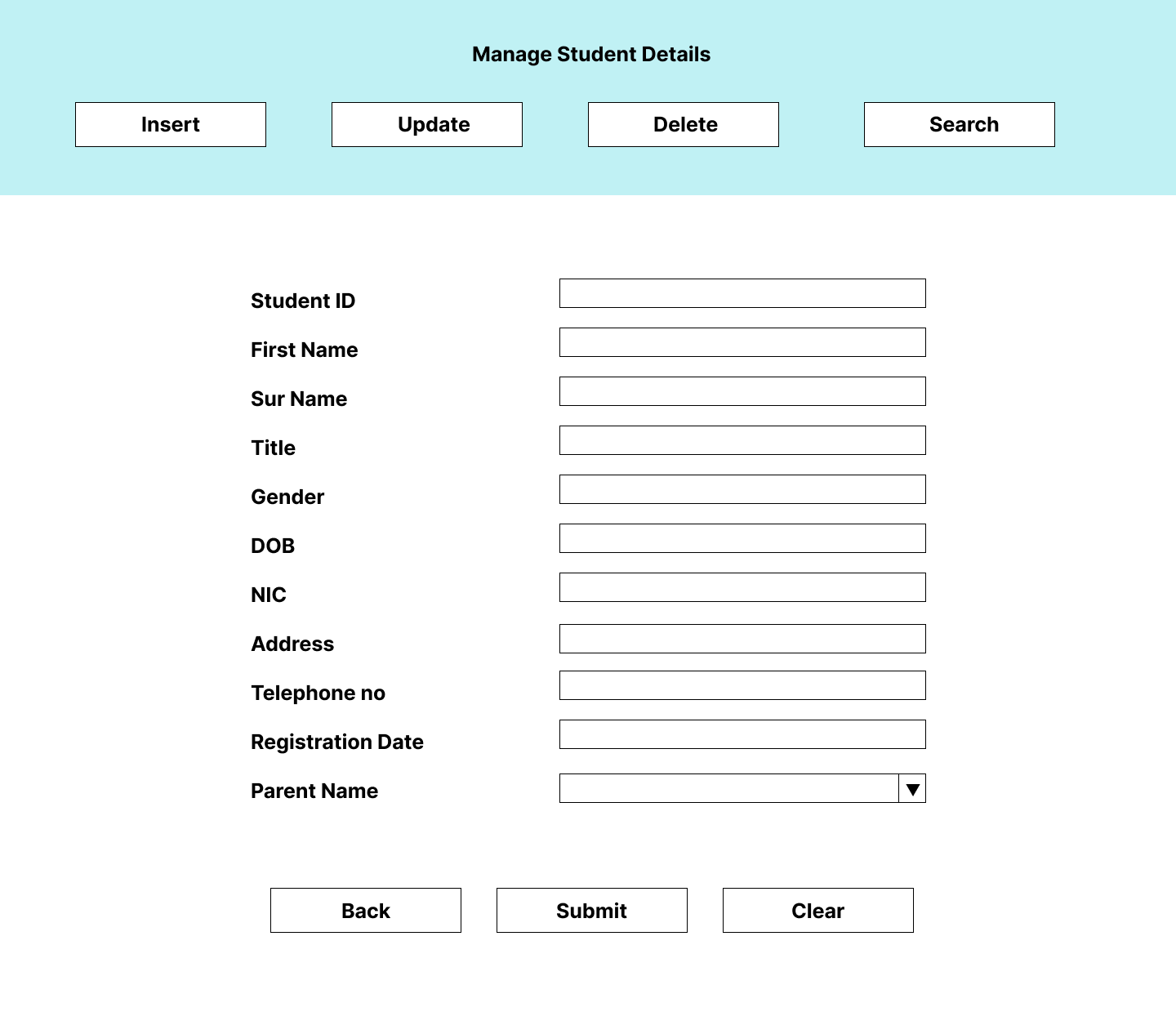


Figure : Manage Student Details Screen

Pseudo Code

Figure : Pseudo code for student management

START

//VALIDATIONS

INCLUDE DbCon

OPEN DBCON

IF INSERT\_CLICKED

THEN

executeQuery(INSERT INTO STUDENTS (“STUDENTID”,…))

ELSE

IF UPDATE\_CLICKED

THEN

executeQuery(UPDATE STUDENTS SET ADDRESS=”” WHERE STUDENTID = “S0001”)

ELSE

IF DELETE\_CLICKED

THEN

executeQuery(DELETE FROM STUDENTS WHERE STUDENTID = “S0001”)

ELSE

IF SELECT\_CLICKED

THEN

Result = executeQuery(SELECT \* FROM STUDENTS)

OUTPUT Result

END IF

CLOSE DBCON

END

Interface No: 04

Interface Name: Lecturer Details

Description: Used to handle all the Lecturer Details related operations



Figure : Manage Lecturer Details Screen

Pseudo Code

Figure : Pseudo code for lecturer management

START

INCLUDE DbCon

//VALIDATIONS

OPEN DBCON

IF INSERT\_CLICKED

THEN

executeQuery(INSERT INTO LECTURERS (“LECTURERID”, …))

ELSE

IF UPDATE\_CLICKED

THEN

executeQuery(UPDATE LECTURER SET ADDRESS=”” WHERE LECTURERID = “L0001”)

ELSE

IF DELETE\_CLICKED

THEN

executeQuery(DELETE FROM LECTURER WHERE LECTURERID = “L0001”)

ELSE

IF SELECT\_CLICKED

THEN

Result = executeQuery(SELECT \* FROM LECTURER)

OUTPUT Result

END IF

CLOSE DBCON

END

Interface No: 05

Interface Name: Payment Details

Description: Used to handle all the Payment Details related operations



Figure 37: Record Payment Screen

Pseudo Code

Figure : Pseudo code for manage payments

START

INCLUDE DbCon

//VALIDATIONS

OPEN DBCON

IF INSERT\_CLICKED

THEN

executeQuery(INSERT INTO PAYMENTS (“STUDENTID”, “COURSEID”, “TYPE”, “AMOUNT”, “INSTALLMENT”, ‘BILLNO’);

END IF

CLOSE DBCON

END

Interface No: 06

Interface Name: Set Schedule

Description: Used to set schedule of the institute

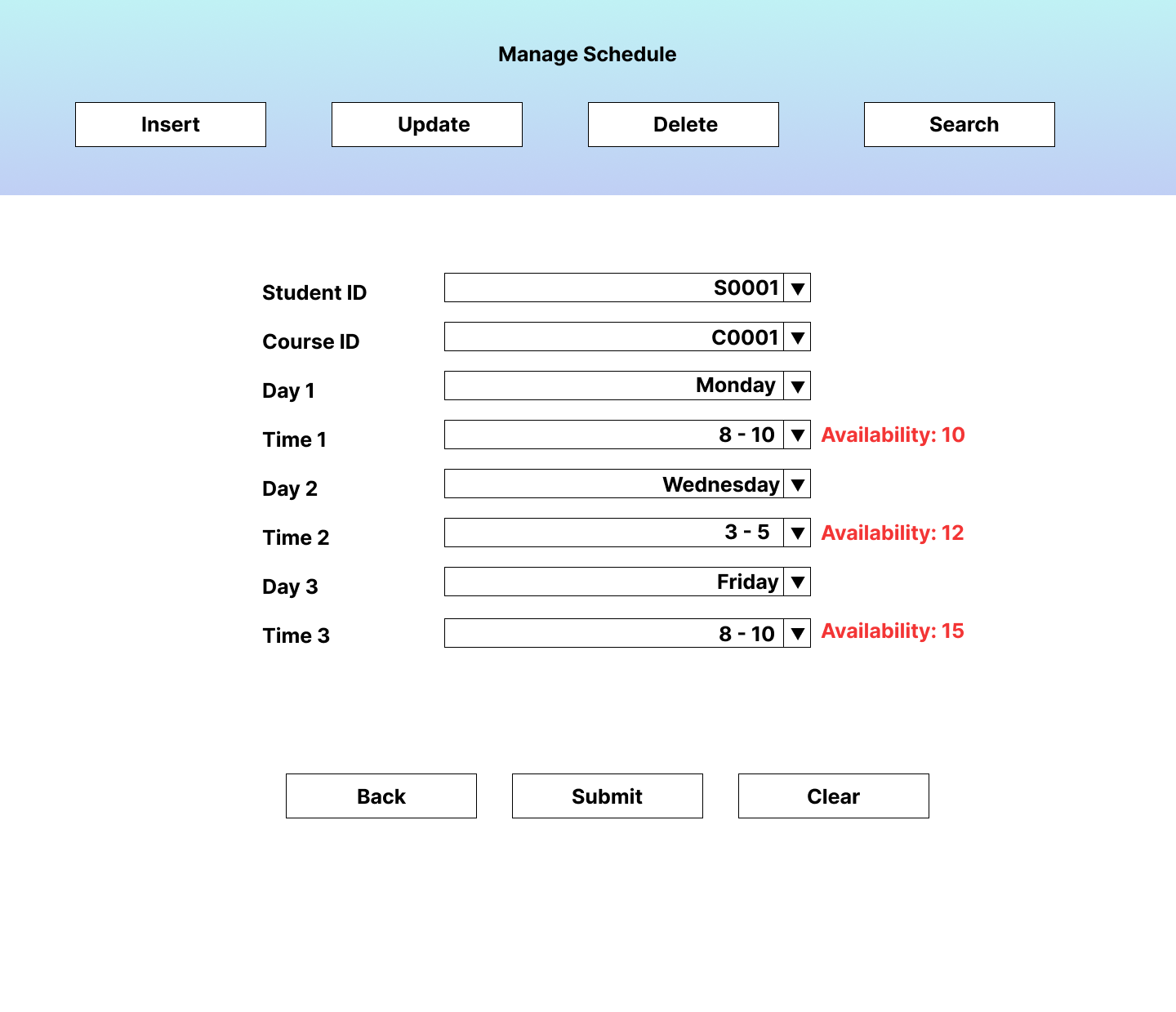


Figure : Manage Schedule Screen

Pseudo Code

Figure : Pseudo code for schedule management

START

INCLUDE DbCon

//VALIDATIONS

IF INSERT\_CLICKED

THEN

executeQuery(INSERT INTO SCHEDULE (“STUDENTID”, ..))

ELSE

IF UPDATE\_CLICKED

THEN

executeQuery(UPDATE SCHEDULE SET TIME=”10 - 12” WHERE BILLNO = “S0001”)

ELSE

IF DELETE\_CLICKED

THEN

executeQuery(DELETE FROM SCHEDULE WHERE STUDENTID = “S0001”)

ELSE

IF SELECT\_CLICKED

THEN

Result = executeQuery(SELECT \* FROM SCHEDULE)

OUTPUT Result

END IF

END

Interface No: 07

Interface Name: Store Lecture Note

Description: Used to store lecture notes to the system

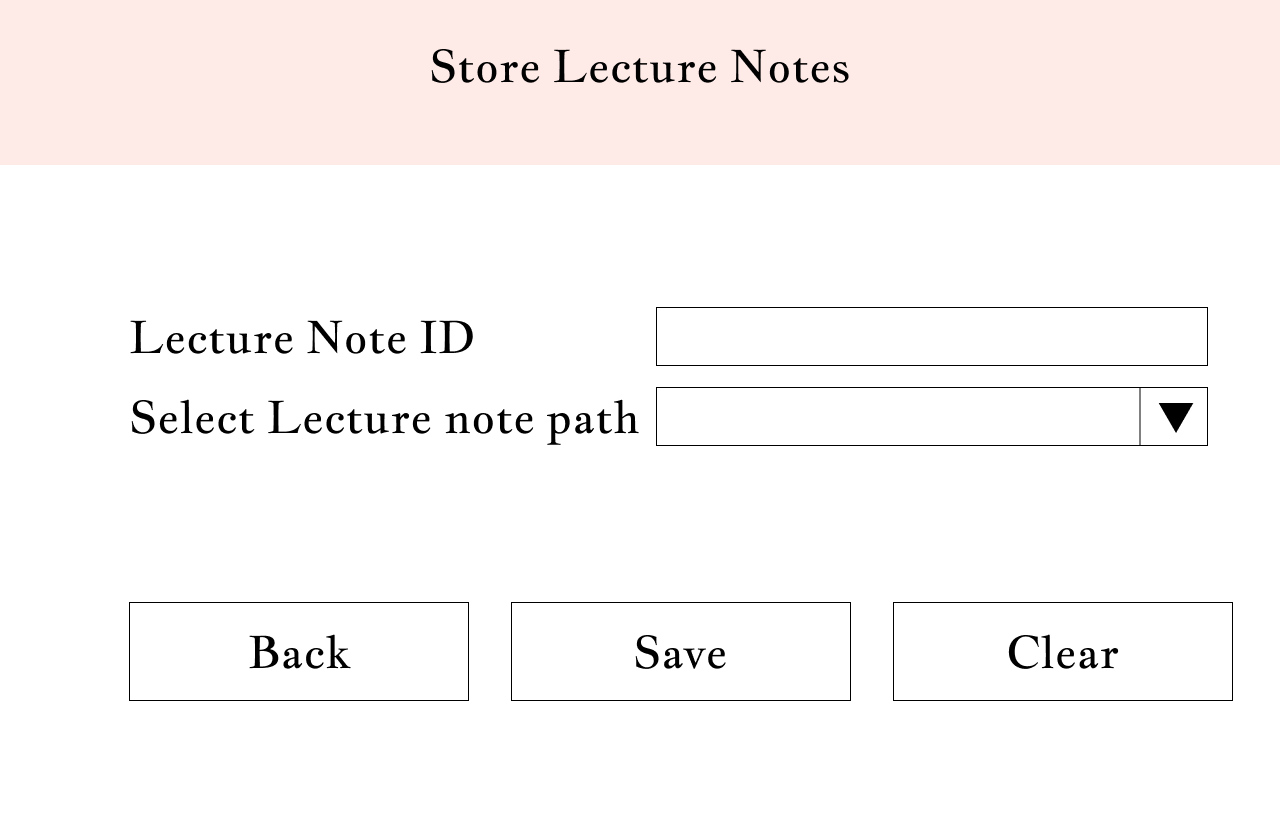


Figure : Store Lecture Notes Screen

Pseudo Code

Figure : Pseudo code for store lecture notes

START

INCLUDE DbCon

//VALIDATIONS

OPEN DBCON

IF INSERT\_CLICKED

THEN

executeQuery(INSERT INTO NOTES (“NOTEID”, “NOTEPATH”))

END IF

CLOSE DBCON

END

Interface No: 08

Interface Name: Store Exam Marks

Description: Used to store Exam Marks

A close-up of a form

Description automatically generated

Figure : Store Exam Marks Screen

Pseudo Code

Figure : Pseudo code to store exam marks

START

INCLUDE DbCon

//VALIDATIONS

IF INSERT\_CLICKED

THEN

executeQuery(INSERT INTO MARKS (“STUDENTID”, “COURSEID”, “EXAM”,

“MARKS”))

END IF

END

Interface No: 09

Interface Name: Ask Questions

Description: Used to ask questions from the lecturer

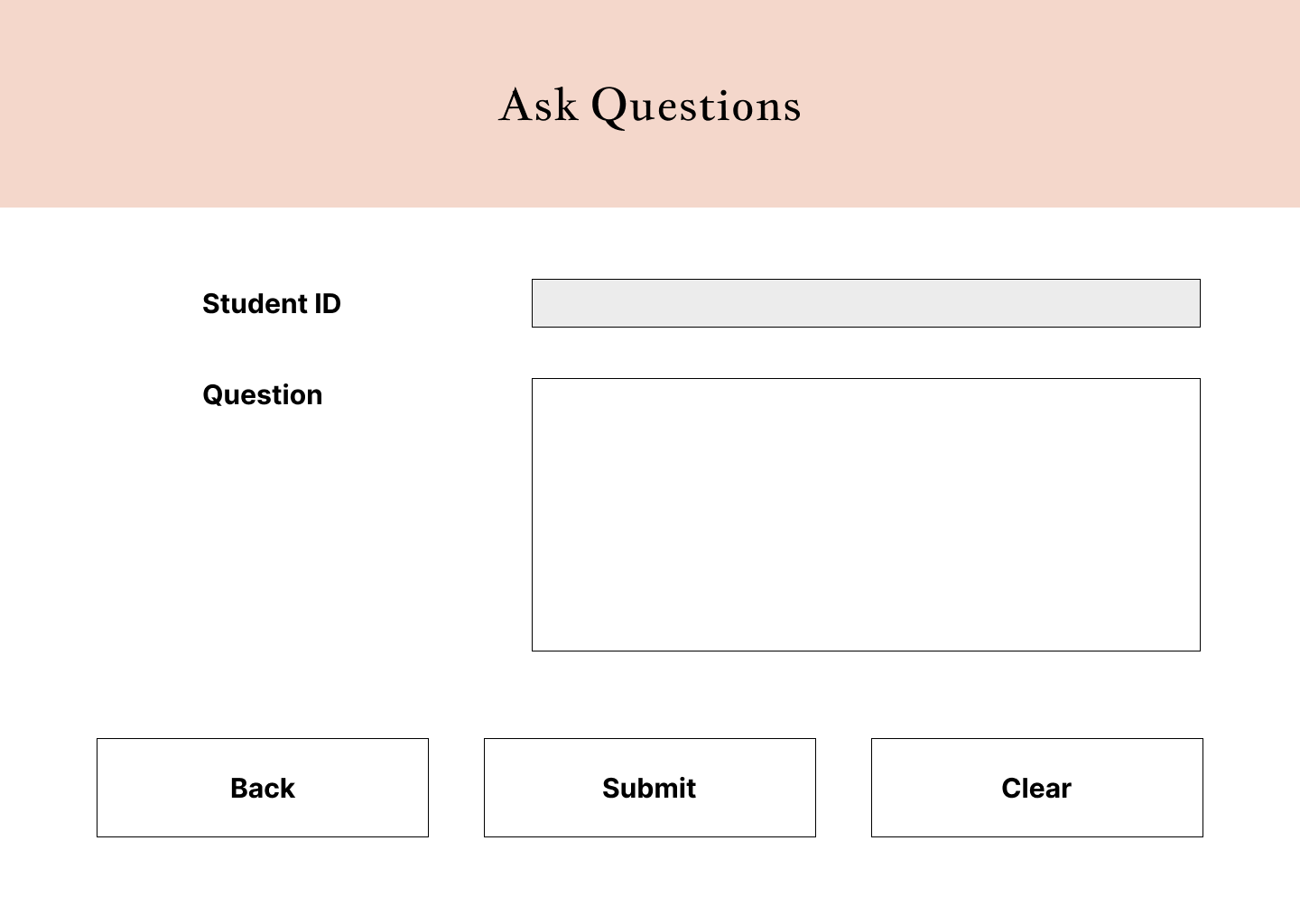


Figure : Ask Questions screen

Pseudo Code

Figure : Pseudo code to ask questions

START

INCLUDE DbCon

INPUT STUDENTID

INPUT QUESTION

OPEN DBCON

IF INSERT\_CLICKED

THEN

executeQuery(INSERT INTO QUESTIONS (STUDENTID, QUESTION))

END IF

CLOSE DBCON

END

Interface No: 10

Interface Name: View Results

Description: Used to view results of exams

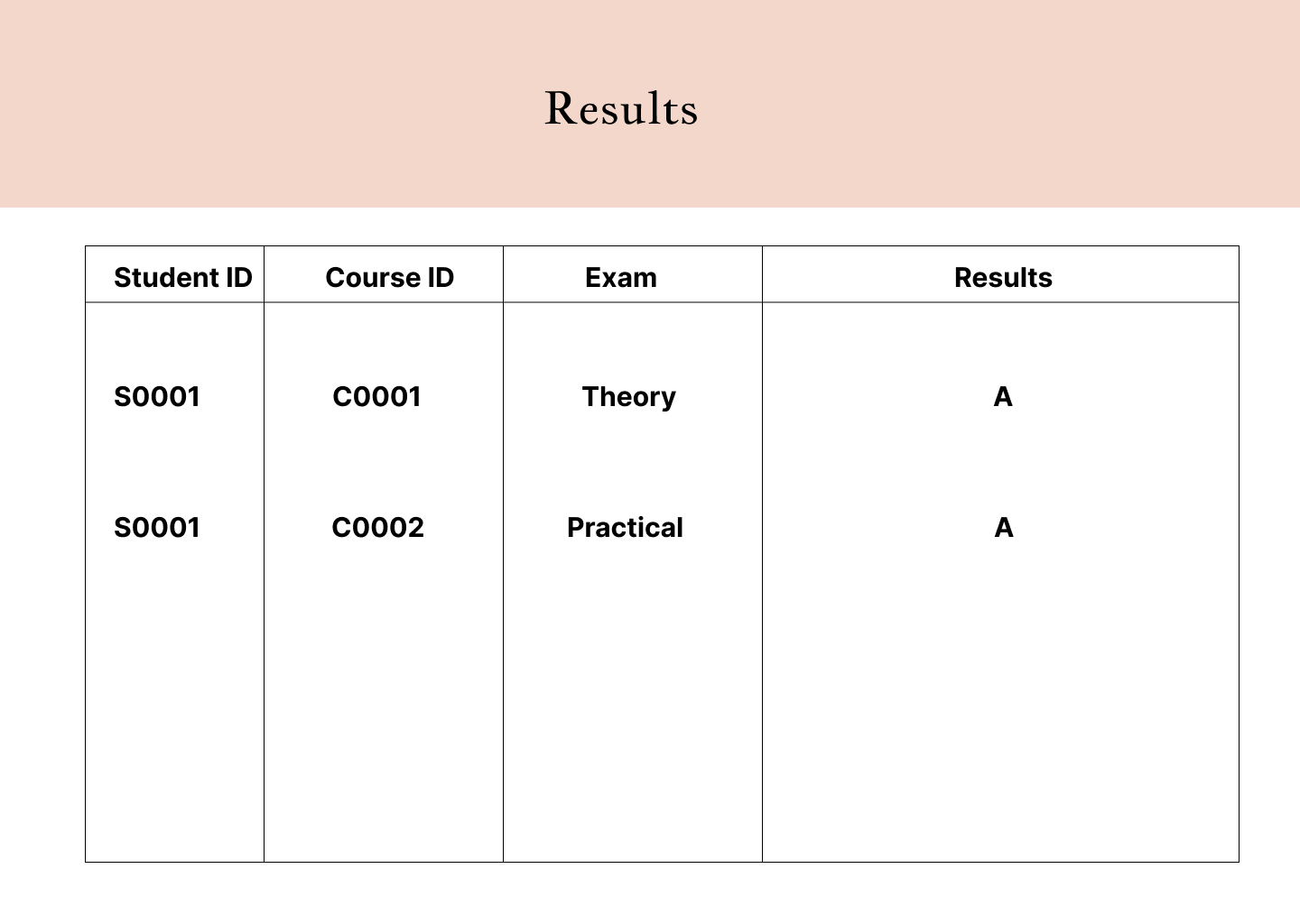


Figure : View Results Screen

Pseudo Code

Figure : Pseudo code for view exam results

START

INCLUDE DbCon

OPEN DBCON

ON LOAD

Result = executeQuery(SELECT \* FROM RESULTS)

OUTPUT Result

END

CLOSE DBCON

END

Interface No: 11

Interface Name: View Notes

Description: Used to view and select lecturer notes

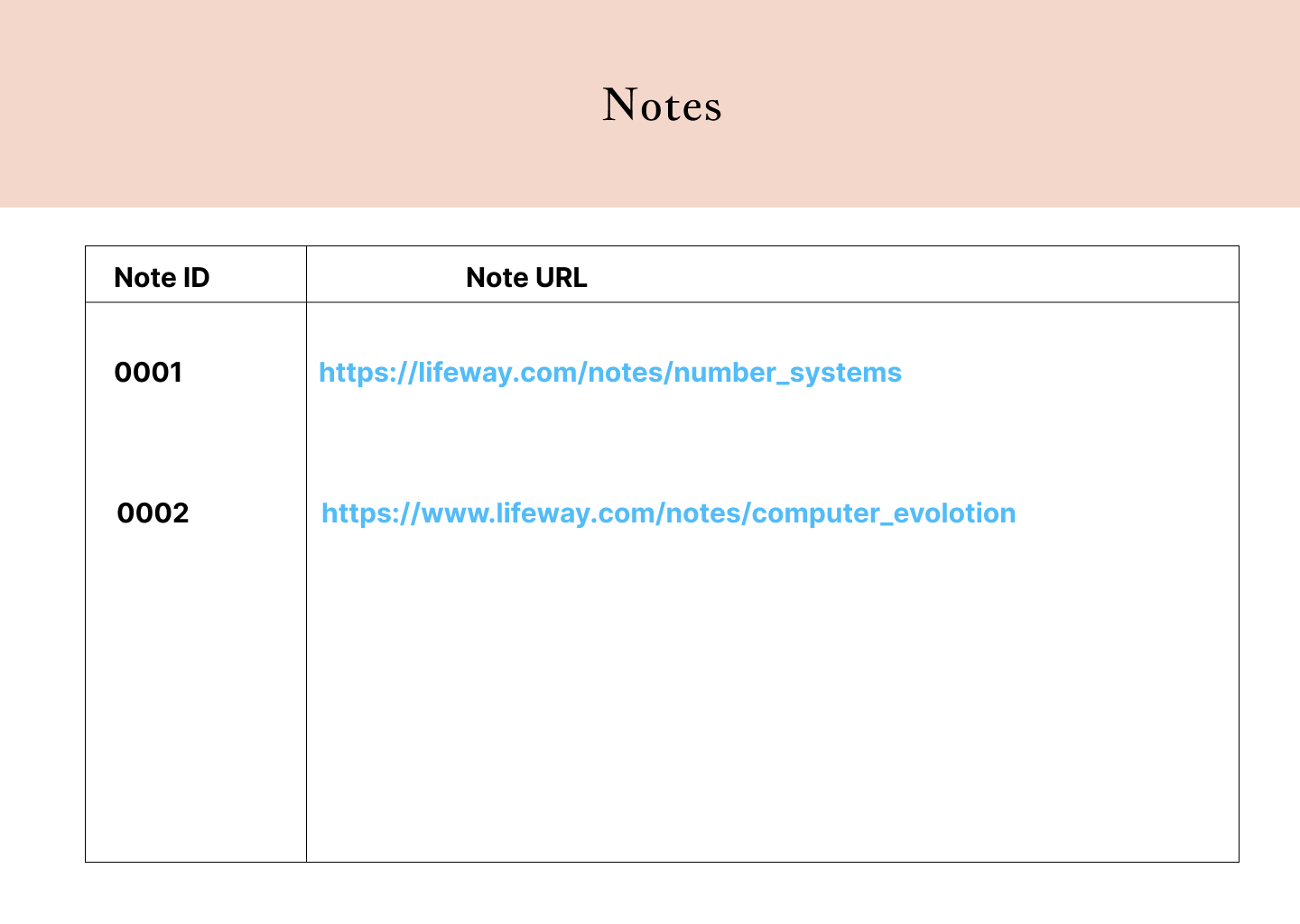


Figure : View Notes Screen

Pseudo Code

Figure : Pseudo code for view notes

START

INCLUDE DbCon

OPEN DBCON

ON LOAD

Result = executeQuery(SELECT \* FROM NOTES)

OUTPUT Result

END

END

Interface No: 12

Interface Name: View Questions

Description: Used to view Questions that the students have sent and select and answer them

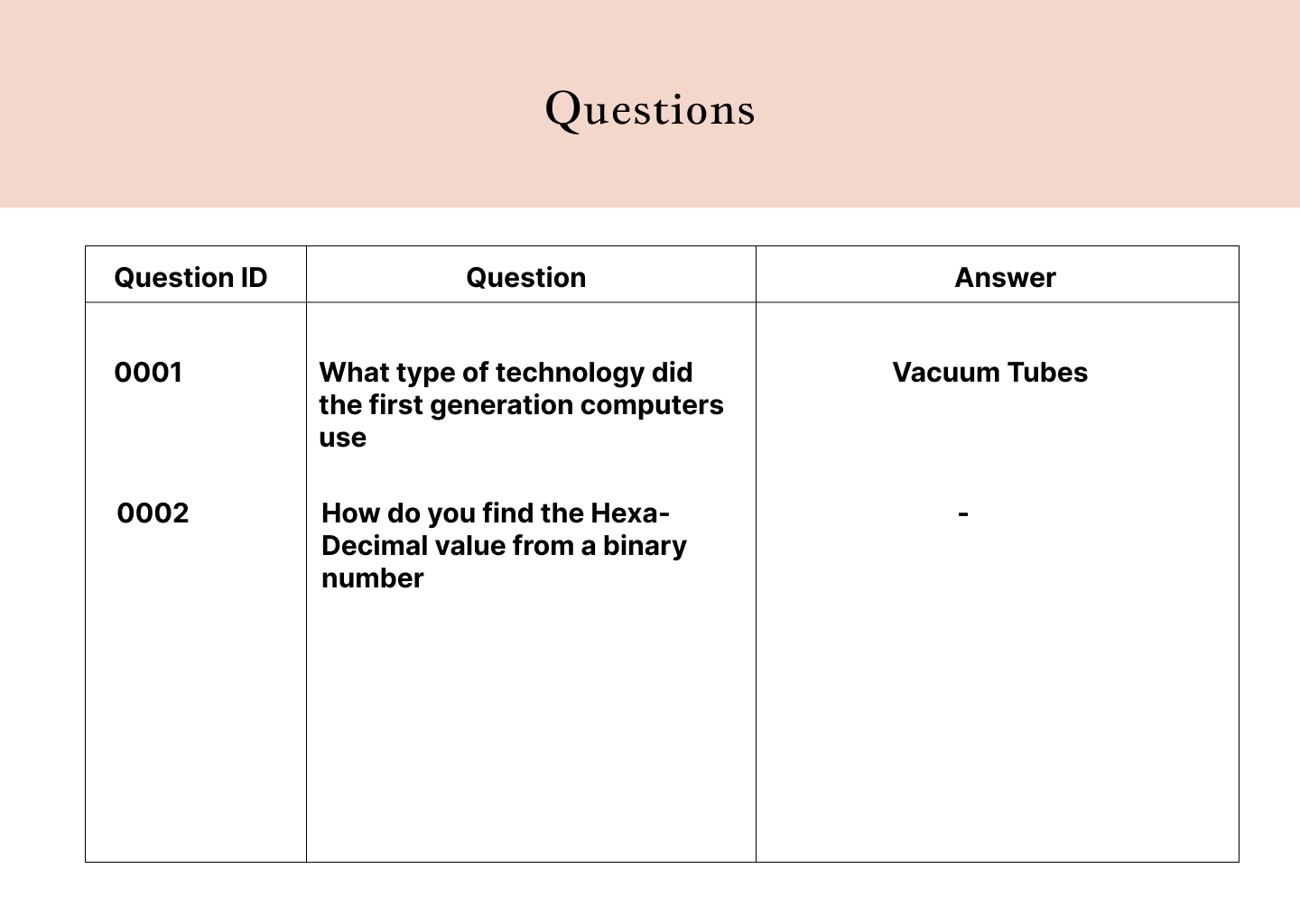


Figure : View Questions Screen

Pseudo Code

Figure : Pseudo code for View Questions

START

INCLUDE DbCon

ON LOAD

Result = executeQuery(SELECT \* FROM QUESTIONS)

OUTPUT Result

END

END

Interface No: 13

Interface Name: View Attendance

Description: Used to view the attendance of students

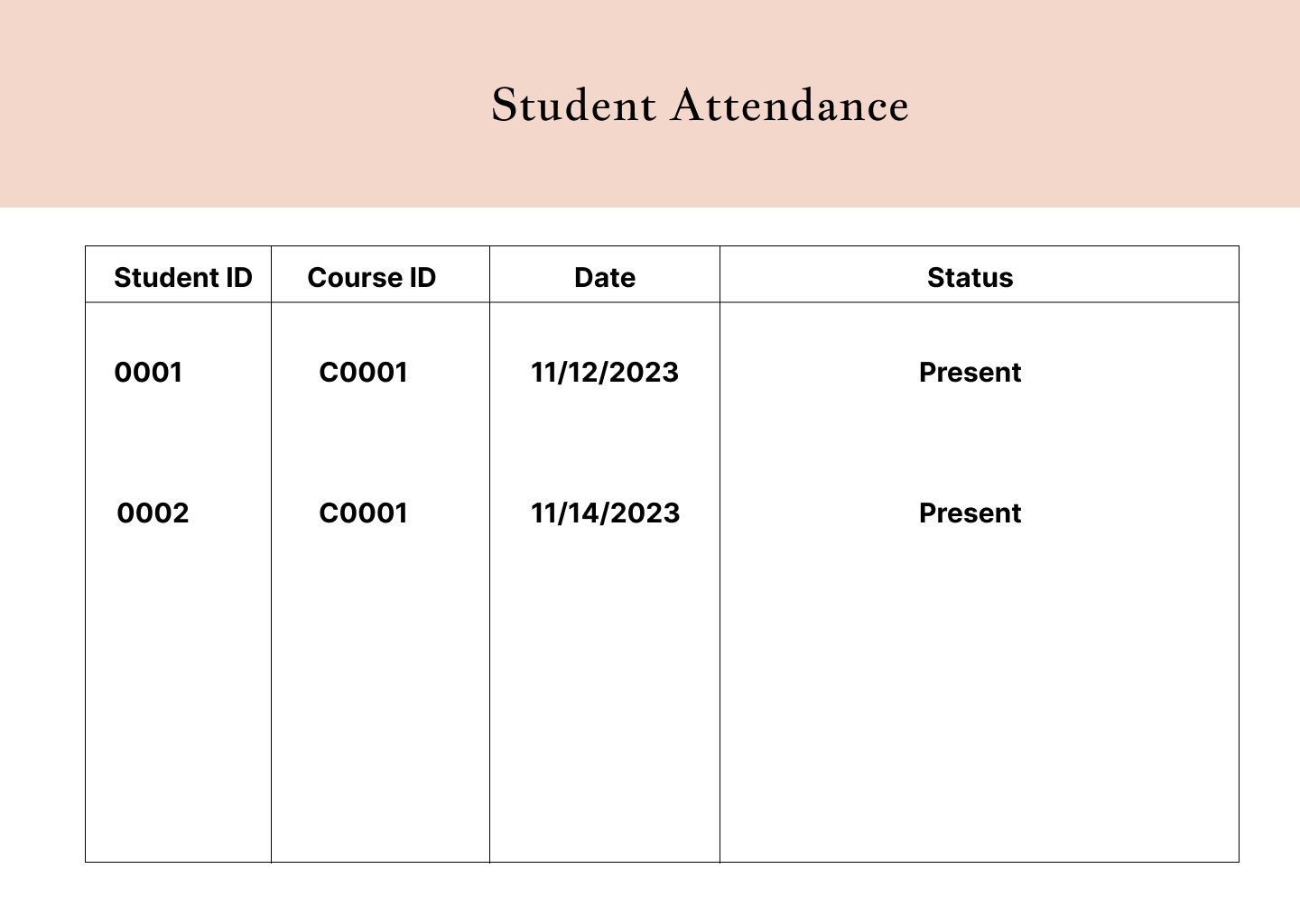


Figure : View Student Attendance Screen

Pseudo Code

Figure : Pseudo code for view student attendance

START

INCLUDE DbCon

OPEN DBCON

ON LOAD

Result = executeQuery(SELECT \* FROM ATTENDANCE)

OUTPUT Result

END

CLOSE DBCON

END

Interface No: 14

Interface Name: View Schedule

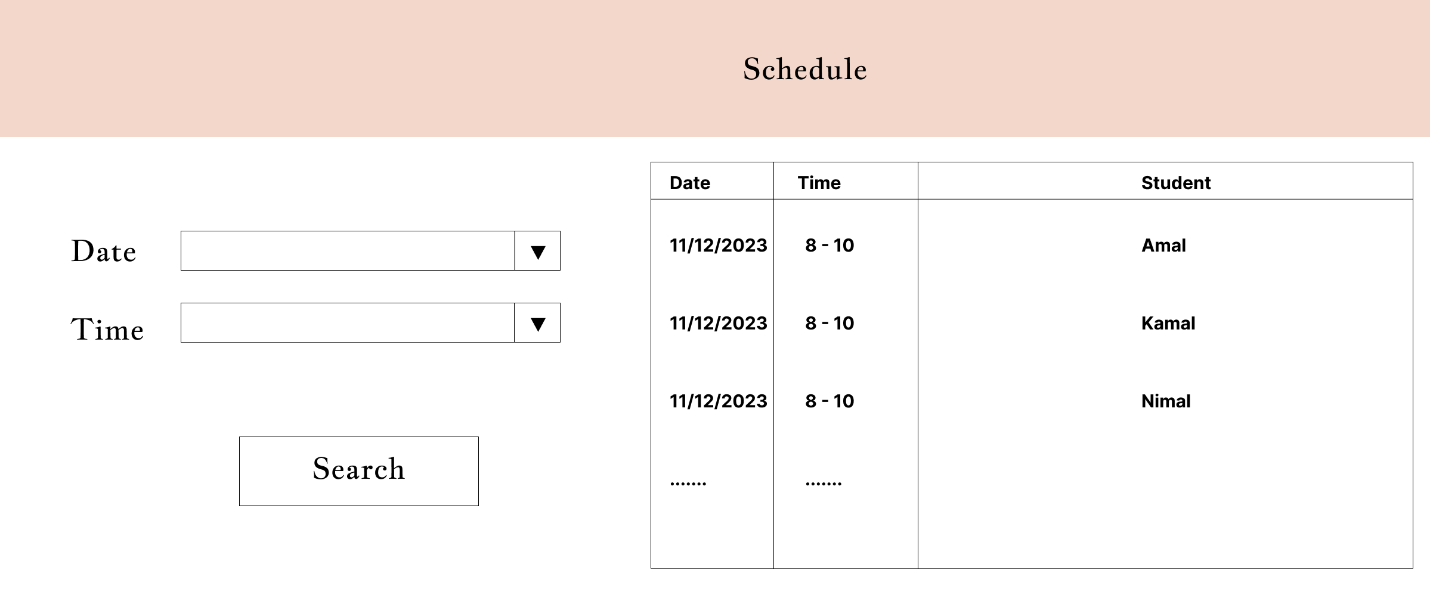
Description: Used to view the schedule

Figure : View Schedule Screen

Pseudo Code

Figure : Pseudo code to view schedule

START

INCLUDE DbCon

INPUT DATE

INPUT TIME

OPEN DBCON

IF SEARCH\_CLICKED

THEN

Result = executeQuery(SELECT \* FROM QUESTIONS WHERE DATE=“11/12/2023” AND TIME=”8-10”)

OUTPUT Result

END

CLOSE DBCON

END

Interface No: 15

Interface Name: Home Screen

Description: Used to navigate through the system by clicking buttons

A group of blue rectangles with black text

Description automatically generated

Figure : Home Screen

Pseudo Code

Figure : Pseudo code for home screen

START

IF COURSE\_CLICKED

THEN

SHOW COURSE DETAILS WINDOW

ELSE IF STUDENTS\_CLICKED

THEN

SHOW STUDENTS DETAILS WINDOW

ELSE IF LECTURER\_CLCKED

THEN

SHOW LECTURER DETAILS WINDOW

ELSE IF SCHEDULE\_CLICKED

THEN

SHOW SCHEDULE WINDOW

ELSE IF LECTURE NOTES\_CLICKED

THEN

SHOW NOTES WINDOW

IF LOGOUT\_CLICKED

THEN

LOGOUT SESSION

CLEAR DATA

TERMINATE APPLICATION

END IF

END

Interface No: 16

Interface Name: Parent Details

Description: Used to store parent details of a student

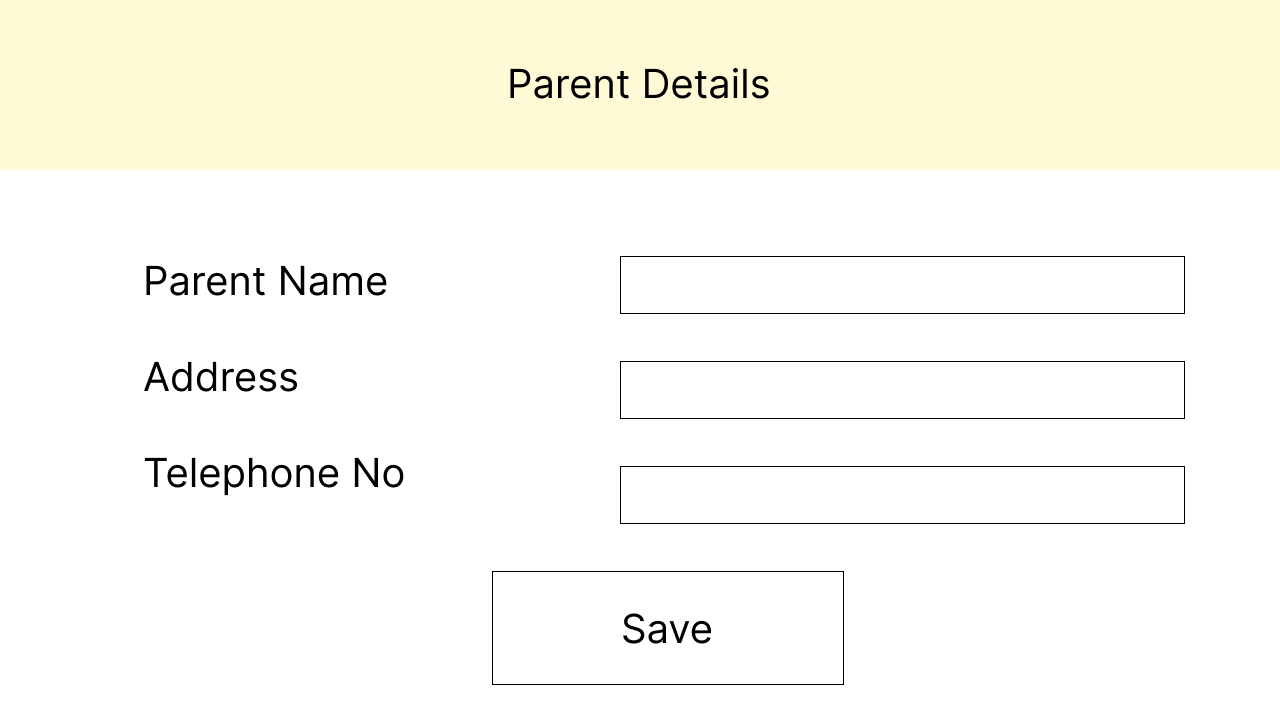


Figure : Parent Details Screen

Pseudo Code

Figure : Pseudo code to maintain Parent Details

START

INCLUDE DbCon

INPUT PARENT\_NAME

INPUT ADDRESS

INPUT TELEPHONENO

IF SAVE\_CLICKED

THEN

ExecuteQuery(INSERT INTO PARENTS (PARENT\_NAME, ADDRESS,

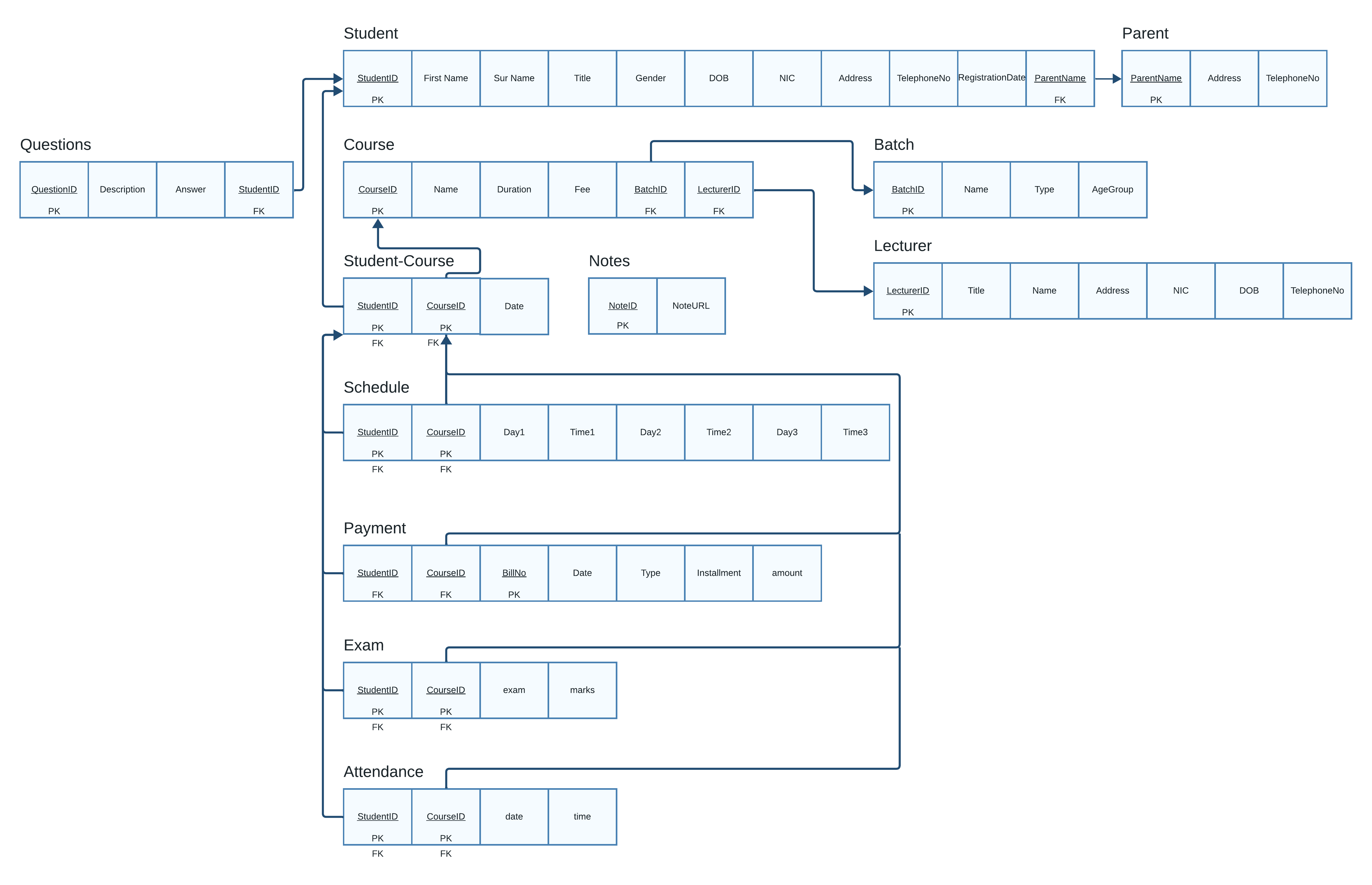
TELEPHONENO))

OUTPIT SAVED

END IF

END

## 4.3 Database Schema



*This diagram shows the database schema of the implemented database of the application.*

Figure : Database Schema

## 4.4 System Requirements

**Hardware**

* Core i3 Processor
* 2GB RAM
* 1GB Hard Disc Space
* Monitor
* Webcam
* Keyboard and Mouse

**Software**

* Windows 7
* Windows 8
* Windows 10

(With .Net framework 3.5 above)

# Chapter 5

# Conclusion

## Project Conclusion

Time, we all try to spend our valuable time as effectively as possible. For Mr. Dhammika this became a particularly hard thing to do. As he was playing both the manager and the sole lecturer at his institute. Because he was one the bet teachers in his area, more and more students enrolled in his classes. Although he loved teaching, he didn’t like spending much time in record keeping. As his student count group grew day-by-day it came to a point where he couldn’t manage the record-keeping work manually anymore. As one of our group members was one of his students, he knew exactly what Mr. Dhammika was going through.

So, we came together to build an application for him, to take him out of this record-keeping task. To do that first we went to his institute, we observed, and gathered all the needed information needed. Next, we discussed with Mr. Dhammika to get a clear idea of what he wants. And after confirming what his expectations are we started to build the application.

We followed all the standard practices that are involved in creating a good application. And we think that our application satisfies our client’s expectations.

We worked hard and created a feature rich application for him to use which has all the functionality that he needed. Which were managing all the records related to his institute more easily, searching for records more easily, and overall, not being worried about human mistakes.

Also, we went beyond what he needed and created an application for his students to use as well. Where the students can ask questions, review notes, view exam results, view their schedule and much more.

Having an application that does the things that you were manually doing before saves you much more time and effort.

So, for the Achievements of the project

* Implemented a new record keeping system that is more efficient.
* Which saves time and effort needed to maintain data within the organization.
* Gives the ability to quickly search for something easily.
* Allows students to be involved in their studies than ever before.
* Allows students record their attendance more easily.

And, for the Weaknesses of the project

* Students will mark their attendance manually.

But overall, we think that we have achieved what this project meant to our client.

And it wouldn’t be possible if we didn’t have the support and guidance from our Supervisor, Course Directors, Lecturers, our family, and our friends. So, for that we would like to convey our heartfelt gratitude to every who helped us.

# References

(2023) ‘Learn C# Programming’, Available at: <https://www.programiz.com/csharp-programming> (Accessed: 20 December 2023)

(2023) ‘C# Tutorial (C Sharp)’, Available at: <https://www.w3schools.com/cs/index.php>. (Accessed: 20 December 2023)

(2023) ‘C# documentation’, Available at: <https://learn.microsoft.com/en-us/dotnet/csharp/> (Accessed: 20 December 2023)

# Appendices

## Work Breakdown Chart

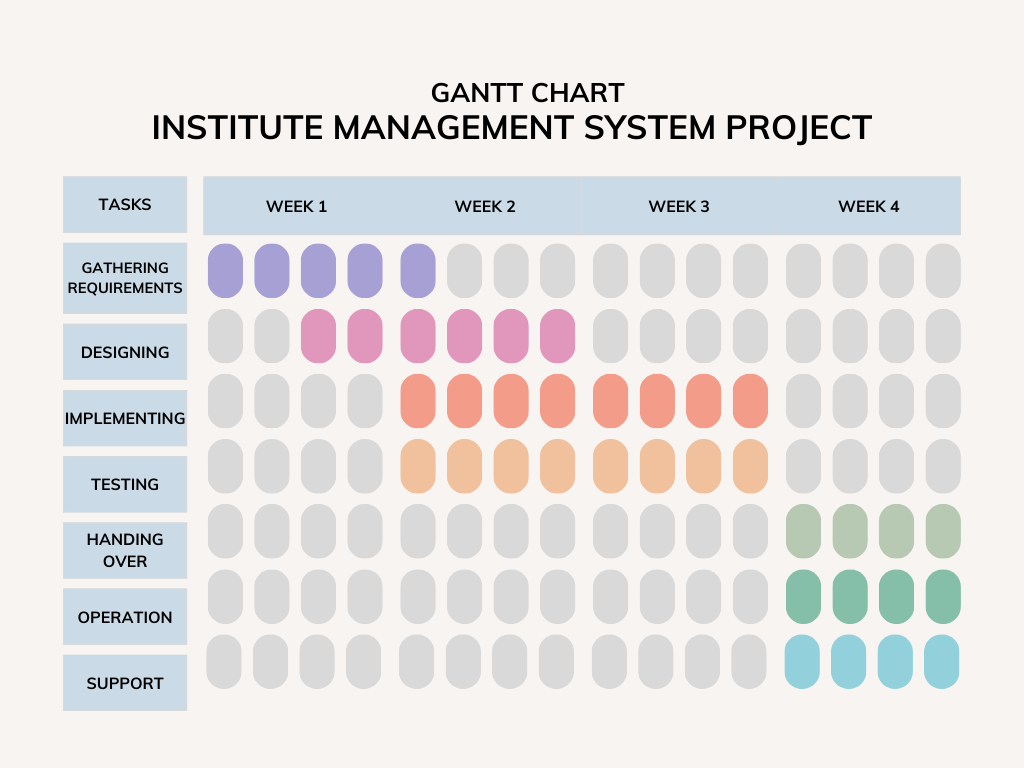
The below chart shows how we managed the given time to complete our project.

Figure : Work Breakdown Sheet

## Future Enhancements

As for the future enhancements, we are looking forward to improving the student attendance marking system by introducing a mobile-based application.

And to provide more functionality to the students.

Also, we will help Mr. Dhammika with any additional features that he might want in the future.

## Program Code

## Desktop Application Code

### Student Entity Class

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Lifeway\_Institute\_Management\_System

{

public class Student

{

private String studentID;

private String fname;

private String surname;

private String title;

private String gender;

private String dob;

private int nic;

private String address;

private int tel;

private String registrationDate;

private String parent;

public Student()

{

studentID = "";

fname = "";

surname = "";

title = "";

gender = "";

dob = "";

nic = 0;

address = "";

tel = 0;

registrationDate = "";

parent = "";

}

public String StudentID

{

get { return studentID; }

set { studentID = value; }

}

public String Fname

{

get { return fname; }

set { fname = value; }

}

public String Surname

{

get { return surname; }

set { surname = value; }

}

public String Title

{

get { return title; }

set { title = value; }

}

public String Gender

{

get { return gender; }

set { gender = value; }

}

public String DOB

{

get { return dob; }

set { dob = value; }

}

public int NIC

{

get { return nic; }

set { nic = value; }

}

public String Address

{

get { return address; }

set { address = value; }

}

public int Tel

{

get { return tel; }

set { tel = value; }

}

public String Registrationdate

{

get { return registrationDate; }

set { registrationDate = value; }

}

public String Parent

{

get { return parent; }

set { parent = value; }

}

### Student Database Class

using MySql.Data.MySqlClient;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Lifeway\_Institute\_Management\_System

{

internal class StudentDb : DbConnection, DbOperations

{

Student student = new Student();

MySqlConnection con;

MySqlCommand com;

public StudentDb()

{

}

public StudentDb(Student student)

{

this.student = student;

}

public void insert()

{

con = getConnection();

String query = "Insert into Students(Fname, Surname, Title, Gender, DOB, NIC, Address, Tel, RegistrationDate, Parent) " +

"values( '"+student.Fname+"'," +

"'"+student.Surname+"'," +

"'"+student.Title+"'," +

"'"+student.Gender+"'," +

"'"+student.DOB+"'," +

""+student.NIC+"," +

"'"+student.Address+"'," +

""+student.Tel+"," +

"'"+student.Registrationdate+"'," +

"'"+student.Parent+"')";

com = new MySqlCommand(query, con);

con.Open();

com.ExecuteNonQuery();

con.Close();

}

public void update()

{

con = getConnection();

String query = "Update Students set Address = '" + student.Address + "', Tel = " + student.Tel + " " +

"where StudentID = "+student.StudentID+"";

com = new MySqlCommand(query, con);

con.Open();

com.ExecuteNonQuery();

con.Close();

}

public void delete()

{

con = getConnection();

String query = "Delete from students where StudentID = '" + student.StudentID + "'";

com = new MySqlCommand(query, con);

con.Open();

com.ExecuteNonQuery();

con.Close();

}

//gets the most recent studentID

public int getID()

{

try

{

int ID = 0;

con = getConnection();

String query = "Select max(StudentID) from Students";

com = new MySqlCommand(query, con);

con.Open();

MySqlDataReader dr = com.ExecuteReader();

while (dr.Read())

{

ID = dr.GetInt32(0);

}

con.Close();

return ID;

}

catch (Exception ex)

{

showErrorMessage(ex, this);

con.Close();

return 0;

}

}

//gets a list of stored IDs

public List<int> getIDs()

{

con = getConnection();

String query = "Select StudentID from Students";

com = new MySqlCommand(query, con);

con.Open();

List<int> IDs = new List<int>();

MySqlDataReader dr = com.ExecuteReader();

while (dr.Read())

{

IDs.Add(dr.GetInt32(0));

}

con.Close();

return IDs;

}

//gets a list of stored First Names

public List<String> getFNames()

{

con = getConnection();

String query = "Select Fname from Students";

com = new MySqlCommand(query, con);

con.Open();

MySqlDataReader dr = com.ExecuteReader();

List<String> names = new List<String>();

while (dr.Read())

{

names.Add(dr.GetString(0));

}

con.Close();

return names;

}

//gets a list of stored Last Names

public List<String> getSNames()

{

con = getConnection();

String query = "Select Surname from Students";

com = new MySqlCommand(query, con);

con.Open();

MySqlDataReader dr = com.ExecuteReader();

List<String> names = new List<String>();

while (dr.Read())

{

names.Add(dr.GetString(0));

}

con.Close();

return names;

}

//gets details of student whose ID is passed

public Student getStudent(int ID)

{

Student student = new Student();

con = getConnection();

String query = "Select \* from Students where StudentID = " + ID + "";

com = new MySqlCommand(query, con);

con.Open();

MySqlDataReader dr = com.ExecuteReader();

dr.Read();

student.StudentID = dr.GetString(0);

student.Fname = dr.GetString(1);

student.Surname = dr.GetString(2);

student.Title = dr.GetString(3);

student.Gender = dr.GetString(4);

student.DOB = dr.GetString(5);

student.NIC = dr.GetInt32(6);

student.Address = dr.GetString(7);

student.Tel = dr.GetInt32(8);

student.Registrationdate = dr.GetString(9);

student.Parent = dr.GetString(10);

con.Close();

return student;

}

//gets detials of student whose first name and last name is passed

public Student getStudent(String fname, String sname)

{

Student student = new Student();

con = getConnection();

String query = "Select \* from Students where Fname = '"+fname+"' and Surname = '"+sname+"'";

com = new MySqlCommand(query, con);

con.Open();

MySqlDataReader dr = com.ExecuteReader();

dr.Read();

student.StudentID = dr.GetString(0);

student.Fname = dr.GetString(1);

student.Surname = dr.GetString(2);

student.Title = dr.GetString(3);

student.Gender = dr.GetString(4);

student.DOB = dr.GetString(5);

student.NIC = dr.GetInt32(6);

student.Address = dr.GetString(7);

student.Tel = dr.GetInt32(8);

student.Registrationdate = dr.GetString(9);

student.Parent = dr.GetString(10);

con.Close();

return student;

}

}

}

### Manage Student Details Form

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Lifeway\_Institute\_Management\_System

{

public partial class frmStudent : Form

{

String type = "";

Student student = new Student();

StudentDb studentdb = new StudentDb();

frmHome frmHome = new frmHome();

public frmStudent()

{

InitializeComponent();

}

public frmStudent(frmHome frmHome)

{

InitializeComponent();

this.frmHome = frmHome;

}

//clears the text fields

public void clear()

{

cboID.Text = "";

cbo\_firstname.Text = "";

cbo\_firstname.Items.Clear();

cbo\_surname.Text = "";

cbo\_surname.Items.Clear();

txttitle.Clear();

cbogender.SelectedIndex = -1;

txtdob.Clear();

txt\_NIC.Clear();

txt\_address.Clear();

txt\_tel.Clear();

txtRegistrationDate.Clear();

cbo\_Parent.SelectedIndex = -1;

cbo\_Parent.Text = "";

if (type == "insert")

{

setInsertEnvironment();

}

else if (type == "update")

{

setUpdateEnvironment();

}

else if (type == "delete")

{

setDeleteEnvironment();

}

else if (type == "select")

{

setSelectEnvironment();

}

}

public void populateIDs()

{

cboID.Items.Clear();

List<int> IDs = studentdb.getIDs();

foreach (int ID in IDs)

{

cboID.Items.Add(ID);

}

}

public void populateNames()

{

cbo\_firstname.Items.Clear();

cbo\_surname.Items.Clear();

List<String> fnames = new List<String>();

List<String> snames = new List<String>();

fnames = studentdb.getFNames();

snames = studentdb.getSNames();

foreach (String name in fnames)

{

cbo\_firstname.Items.Add(name);

}

foreach (String name in snames)

{

cbo\_surname.Items.Add(name);

}

}

public void populateParentNames()

{

cbo\_Parent.Items.Clear();

ParentDb parentdb = new ParentDb();

List<String> names = parentdb.getNames();

foreach (String name in names)

{

cbo\_Parent.Items.Add(name);

}

cbo\_Parent.Items.Add("Add New...");

}

public void setInsertEnvironment()

{

lblID.Visible = false;

cboID.Visible = false;

cbo\_firstname.Enabled = true;

cbo\_surname.Enabled = true;

txttitle.Enabled = true;

cbogender.Enabled = true;

txtdob.Enabled = true;

txt\_address.Enabled = true;

txt\_NIC.Enabled = true;

txt\_tel.Enabled = true;

txtRegistrationDate.Enabled = true;

cbo\_Parent.Enabled = true;

btnGetDetails.Visible = false;

btnSubmit.Visible = true;

cbo\_firstname.Items.Clear();

cbo\_surname.Items.Clear();

txtRegistrationDate.Text = System.DateTime.Now.Date.ToShortDateString();

populateParentNames();

}

public void setUpdateEnvironment()

{

lblID.Visible = true;

cboID.Visible = true;

cboID.Enabled = true;

cbo\_firstname.Enabled = false;

cbo\_surname.Enabled = false;

txttitle.Enabled = false;

cbogender.Enabled = false;

txtdob.Enabled = false;

txt\_address.Enabled = false;

txt\_NIC.Enabled = false;

txt\_tel.Enabled = false;

txtRegistrationDate.Enabled = false;

cbo\_Parent.Enabled = false;

cbo\_firstname.Items.Clear();

cbo\_surname.Items.Clear();

btnGetDetails.Visible = false;

btnSubmit.Visible = false;

populateIDs();

}

public void setDeleteEnvironment()

{

lblID.Visible = false;

cboID.Visible = false;

cbo\_firstname.Enabled = true;

cbo\_surname.Enabled = true;

txttitle.Enabled = false;

cbogender.Enabled = false;

txtdob.Enabled = false;

txt\_address.Enabled = false;

txt\_NIC.Enabled = false;

txt\_tel.Enabled = false;

txtRegistrationDate.Enabled = false;

cbo\_Parent.Enabled = false;

btnSubmit.Visible = false;

btnGetDetails.Visible = false;

populateNames();

}

public void setSelectEnvironment()

{

lblID.Visible = false;

cboID.Visible = false;

cbo\_firstname.Enabled = true;

cbo\_surname.Enabled = true;

txttitle.Enabled = false;

cbogender.Enabled = false;

txtdob.Enabled = false;

txt\_address.Enabled = false;

txt\_NIC.Enabled = false;

txt\_tel.Enabled = false;

txtRegistrationDate.Enabled = false;

cbo\_Parent.Enabled = false;

btnSubmit.Visible = false;

btnGetDetails.Visible = false;

populateNames();

}

private void btnInsert\_Click(object sender, EventArgs e)

{

type = "insert";

clear();

}

private void btnSubmit\_Click(object sender, EventArgs e)

{

if (type == "insert")

{

//validations

if (cbo\_firstname.Text == "")

{

MessageBox.Show("Must Enter Student First Name");

return;

}

if (cbo\_surname.Text == "")

{

MessageBox.Show("Must Enter Student Sur Name");

return;

}

if (cbogender.SelectedIndex == -1)

{

MessageBox.Show("Must select gender");

return;

}

if (txtdob.Text == "")

{

MessageBox.Show("Must select student Date of Birth");

return;

}

if (txt\_NIC.Text == "")

{

MessageBox.Show("Must enter student NIC");

return;

}

if (txt\_NIC.Text.Length != 10)

{

MessageBox.Show("Must enter valid NIC number");

return;

}

if (txt\_address.Text == "")

{

MessageBox.Show("Must enter address of student");

return;

}

if (txt\_tel.Text == "")

{

MessageBox.Show("Must enter student contact number");

return;

}

if (txt\_tel.Text.Length < 10)

{

MessageBox.Show("Must enter a valid Telephone No");

return;

}

if (cbo\_Parent.SelectedIndex == -1)

{

MessageBox.Show("Must select Parent Name");

return;

}

Student student = new Student();

student.Fname = cbo\_firstname.Text;

student.Surname = cbo\_surname.Text;

student.Title = txttitle.Text;

student.Gender = cbogender.SelectedItem.ToString();

student.DOB = txtdob.Text;

student.NIC = Convert.ToInt32(txt\_NIC.Text);

student.Address = txt\_address.Text;

student.Tel = Convert.ToInt32(txt\_tel.Text);

student.Registrationdate = txtRegistrationDate.Text;

student.Parent = cbo\_Parent.SelectedItem.ToString();

studentdb = new StudentDb(student);

studentdb.insert();

int studentID = studentdb.getID();

if (studentID != 0)

{

MessageBox.Show("Student successfully registered\nStudentID: " + studentID);

clear();

if (MessageBox.Show("Do you want to continue to register the student with a course?", "", MessageBoxButtons.YesNo) == DialogResult.Yes)

{

this.Close();

frmStudentCourse frmstudentcourse = new frmStudentCourse(frmHome, studentID);

frmstudentcourse.Show();

}

}

else if (type == "update")

{

student.Address = txt\_address.Text;

student.Tel = Convert.ToInt32(txt\_tel.Text);

if (MessageBox.Show("Do you really want to update the details of " + student.Fname + " " + student.Surname, "Update Confirmation Dialog", MessageBoxButtons.YesNo, MessageBoxIcon.Information) == DialogResult.Yes)

{

studentdb = new StudentDb(student);

studentdb.update();

MessageBox.Show("Reccord successfully updated", "Updated Message");

clear();

}

else

{

MessageBox.Show("Student Details not updated");

}

}

else if (type == "delete")

{

if (MessageBox.Show("Do you really want to delete the details of " + student.Fname + " " + student.Surname, "Delete Confirmation Dialog", MessageBoxButtons.YesNo, MessageBoxIcon.Information) == DialogResult.Yes)

{

studentdb = new StudentDb(student);

studentdb.delete();

MessageBox.Show("Reccord successfully deleted", "Deleted Message");

clear();

}

else

{

MessageBox.Show("Student Details not deleted");

}

}

}

private void cbogender\_SelectedIndexChanged(object sender, EventArgs e)

{

if (cbogender.SelectedIndex != -1)

{

if (cbogender.SelectedItem.ToString() == "Male")

{

txttitle.Text = "Mr.";

}

else

{

txttitle.Text = "Miss.";

}

}

}

private void btnClear\_Click(object sender, EventArgs e)

{

clear();

}

private void cboID\_SelectedIndexChanged(object sender, EventArgs e)

{

if (cboID.SelectedIndex != -1)

{

btnGetDetails.Visible = true;

}

}

private void btnGetDetails\_Click(object sender, EventArgs e)

{

if (type == "update")

{

int ID = Convert.ToInt32(cboID.SelectedItem.ToString());

student = studentdb.getStudent(ID);

cboID.Enabled = false;

txt\_address.Enabled = true;

txt\_tel.Enabled = true;

btnSubmit.Visible = true;

}

else if (type == "delete" || type == "select")

{

String fname = cbo\_firstname.SelectedItem.ToString();

String sname = cbo\_surname.SelectedItem.ToString();

student = studentdb.getStudent(fname, sname);

cbo\_firstname.Enabled = false;

cbo\_surname.Enabled = false;

}

if (type == "delete")

{

btnSubmit.Visible = true;

}

cbo\_firstname.Text = student.Fname;

cbo\_surname.Text = student.Surname;

txttitle.Text = student.Title;

cbogender.Text = student.Gender;

txtdob.Text = student.DOB;

txt\_NIC.Text = student.NIC.ToString();

txt\_address.Text = student.Address;

txt\_tel.Text = student.Tel.ToString();

txtRegistrationDate.Text = student.Registrationdate;

cbo\_Parent.Text = student.Parent;

btnGetDetails.Visible = false;

}

private void btnUpdate\_Click(object sender, EventArgs e)

{

type = "update";

clear();

}

private void btnDelete\_Click(object sender, EventArgs e)

{

type = "delete";

clear();

}

private void cbo\_firstname\_SelectedIndexChanged(object sender, EventArgs e)

{

if (type == "delete" || type == "select")

{

if (cbo\_firstname.SelectedIndex != -1 && cbo\_surname.SelectedIndex !=1)

{

btnGetDetails.Visible = true;

}

}

}

private void cbo\_surname\_SelectedIndexChanged(object sender, EventArgs e)

{

if (type == "delete" || type == "select")

{

if (cbo\_firstname.SelectedIndex != -1 && cbo\_surname.SelectedIndex !=1)

{

btnGetDetails.Visible = true;

}

}

}

private void btnSelect\_Click(object sender, EventArgs e)

{

type = "select";

clear();

}

private void btnBack\_Click(object sender, EventArgs e)

{

this.Close();

}

private void cbo\_Parent\_SelectedIndexChanged(object sender, EventArgs e)

{

if (cbo\_Parent.Text != "")

{

if (cbo\_Parent.SelectedItem.ToString() == "Add New...")

{

cbo\_Parent.SelectedIndex = -1;

this.Hide();

btnrefresh.Visible = true;

frmParent frmParent = new frmParent(this);

frmParent.Show();

}

}

}

private void btnrefresh\_Click(object sender, EventArgs e)

{

populateParentNames();

btnrefresh.Visible = false;

}

private void frmStudent\_FormClosed(object sender, FormClosedEventArgs e)

{

frmHome.Show();

}

}

}

### DbConnection Class

using System;

using System.Collections.Generic;

using System.Diagnostics;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Xml.Serialization;

using MySql.Data.MySqlClient;

namespace Lifeway\_Institute\_Management\_System

{

public class DbConnection

{

private MySqlConnection connection;

private String host = "localhost";

private String username = "root";

private String pwd = "";

private String database = "Lifeway";

private String str;

public DbConnection() {

//sets the connection string to connect with the database

this.str = "server=" + host + ";user=" + username + ";pwd=" + pwd + ";database=" + database + "";

//create the connection to the database

try

{

connection = new MySqlConnection(str);

}

catch (Exception ex)

{

showErrorMessage(ex, this);

}

}

public void showErrorMessage(Exception ex, Object from)

{

MessageBox.Show("Something went wrong please contact developer \nError Details\nError generated : " +from.GetType()+ "\nError code: " +ex.Message);

}

public MySqlConnection getConnection()

{

return this.connection;

}

}

}

### DbOperations Interface

using MySql.Data.MySqlClient;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Lifeway\_Institute\_Management\_System

{

public interface DbOperations

{

public void insert();

public void update();

public void delete();

}

}

### Home Window

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Net.Http.Headers;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Lifeway\_Institute\_Management\_System

{

public partial class frmHome : Form

{

frmLogin frmlogin;

public frmHome()

{

InitializeComponent();

}

public frmHome(frmLogin frmlogin)

{

InitializeComponent();

this.frmlogin = frmlogin;

}

private void btnManageStudents\_Click(object sender, EventArgs e)

{

this.Hide();

frmStudent frmStudent = new frmStudent(this);

frmStudent.Show();

}

private void btnBack\_Click(object sender, EventArgs e)

{

this.Close();

}

private void btnManageLecturer\_Click(object sender, EventArgs e)

{

this.Hide();

frmLecturer frmLecturer = new frmLecturer(this);

frmLecturer.Show();

}

private void frmHome\_FormClosed(object sender, FormClosedEventArgs e)

{

frmlogin.Show();

}

private void btnManageCourses\_Click(object sender, EventArgs e)

{

this.Hide();

frmCourse frmcourse = new frmCourse(this);

frmcourse.Show();

}

private void btnManageBatches\_Click(object sender, EventArgs e)

{

this.Hide();

frmBatch frmbatch = new frmBatch(this);

frmbatch.Show();

}

private void btnPayments\_Click(object sender, EventArgs e)

{

this.Hide();

frmPayment frmpayment = new frmPayment(this);

frmpayment.Show();

}

private void btnStudentCourse\_Click(object sender, EventArgs e)

{

this.Hide();

frmStudentCourse frmstudentcourse = new frmStudentCourse(this);

frmstudentcourse.Show();

}

private void btnNotes\_Click(object sender, EventArgs e)

{

this.Hide();

frmNotes frmnotes = new frmNotes(this);

frmnotes.Show();

}

private void btnManageSchedule\_Click(object sender, EventArgs e)

{

this.Hide();

frmSchedule frmSchedule = new frmSchedule(this);

frmSchedule.Show();

}

private void btnAnswerQuestions\_Click(object sender, EventArgs e)

{

this.Hide();

frmAnswers frmAnswers = new frmAnswers(this);

frmAnswers.Show();

}

private void btnExamMarks\_Click(object sender, EventArgs e)

{

this.Hide();

frmexamMarks frmexammarks = new frmexamMarks(this);

frmexammarks.Show();

}

private void btnViewSchedule\_Click(object sender, EventArgs e)

{

this.Hide();

frmViewSchedule frmViewSchedule = new frmViewSchedule(this);

frmViewSchedule.Show();

}

private void btnViewAttendance\_Click(object sender, EventArgs e)

{

this.Hide();

frmViewAttendance frmviewAttendance = new frmViewAttendance(this);

frmviewAttendance.Show();

}

}

}

## Student Website Application Code

### Ask Questions Web Page

<?php

    include\_once 'DbConnection.php';

    $query = "Select studentid from students";

    $res = mysqli\_query($con, $query);

    $no\_rows = mysqli\_num\_rows($res);

    $c = 1;

?>

<html>

    <head>

        <Title>Ask Questions</Title>

        <link rel="stylesheet" href = "stylings.css"/>

    </head>

    <body style = "display: none;">

        <h1 id = "heading">Feel free to ask any Question you have regarding you're studies</h1>

        <table cellspacing = "0px" cellpadding = "5px" align = "center">

            <tr>

                <td id = "Title">Ask Questions</td>

            </tr>

            <!--Question form-->

            <tr>

                <td>

                    <form action="save-questions.php" method = "POST">

                        <!--Label and ComboBox to select StudentID to submit question-->

                        <label for="studentID">Student ID</label>

                        <select required name = "studentID">

                            <option value="">Please Select StudentID</option>

                            <?php while ( $c <= $no\_rows ) {$row = mysqli\_fetch\_assoc($res);?>

                                <option value = '<?php echo $row['studentid'];?>'>

                                    <?php echo $row['studentid'];?>

                                </option>

                            <?php $c = $c + 1; } ?>

                        </select> <br> <br>

                        <!--Label and TextArea to note the question-->

                        <label for="question">Question</label> <br>

                        <textarea required rows = "10" cols = "40" name = "question"></textarea> <br><br>

                        <!--Button to submit question-->

                        <button type = "Submit" name = "submit" id = "submit">

                            Submit

                        </button>

                    </form>

                </td>

            </tr>

        </table>

        <script src = "JQuery.js"></script>

        <script>

            $(function(){

                $('body').fadeIn(1000);

            });

        </script>

    </body>

</html>

### View Results Web Page

<?php

    include\_once "DbConnection.php";

    $studentID = $\_POST['studentID'];

    $query = "Select \* from exam\_marks where studentID = ".$studentID."";

    $res = mysqli\_query($con, $query);

    $no\_of\_rows = mysqli\_num\_rows($res);

    $c = 0;

?>

<html>

    <head>

        <title>Student Results</title>

        <style>

            div{

                font-size: 40px;

                font-weight: bold;;

                width: 100%;

                height: 100px;

                background-color:cornsilk;

                text-align: center;

            }

            table{

                position: relative;

                top: 125px;

                font-size: 40px;

            }

        </style>

    </head>

    <body>

        <div>

            Student Results

        </div>

        <table border = "1" cellspacing = "0px" cellpadding = "5px" align = "center">

            <tr>

                <th>Student ID</th>

                <th>Course ID</th>

                <th>Exam Type</th>

                <th>Marks</th>

            </tr>

            <?php while ($c < $no\_of\_rows){

                $row = mysqli\_fetch\_assoc($res);

                $c = $c + 1; ?>

                <tr>

                    <td> <?php echo $row['StudentID'] ?> </td>

                    <td> <?php echo $row['CourseID'] ?> </td>

                    <td> <?php echo $row['Exam'] ?> </td>

                    <td> <?php echo $row['Marks'] ?> </td>

                </tr>

            <?php } ?>

        </table>

    </body>

</html>

### View Notes Web Page

<?php

    include\_once 'DbConnection.php';

    $query = "Select \* from Notes";

    $res = mysqli\_query($con, $query);

    $no\_of\_rows = mysqli\_num\_rows($res);

    $c = 1;

?>

<html>

    <head>

        <title>View Notes</title>

        <style>

            table{

                table-layout: fixed;

                position: relative;

                top: 10px;

            }

            td{

                word-wrap: break-word;

            }

        </style>

    </head>

    <body>

        <table border = "1" width = "80%" cellpadding = "5px" cellspacing = "0px" align = "center">

            <tr>

                <th width = "20%">Note</th>

                <th width = "80%">Resource</th>

            </tr>

            <?php while ($c <= $no\_of\_rows){

                $row = mysqli\_fetch\_assoc($res); $c = $c + 1;

            ?>

                <tr>

                    <td> <?php echo $row['name']; ?> </td>

                    <td> <a href = "<?php echo $row['destination'] ?>" target = "\_blank"> Click here to view note </a></td>

                </tr>

            <?php } ?>

        </table>

    </body>

</html>

### Mark Attendance Page

<?php

    session\_start();

    $uname = $\_SESSION['name'];

    include\_once "DbConnection.php";

    date\_default\_timezone\_set("Asia/Colombo");

    $studentID = $\_POST['studentID'];

    $courseID = $\_POST['courseID'];

    $date = date("m/d/Y");

    $time = date("h:i");

    $query = "Select studentID from attendance where studentID = ".$studentID." and courseID = ".$courseID." and date = '".$date."'";

    $res = mysqli\_query($con, $query);

    $rows = mysqli\_num\_rows($res);

    if ($rows == 0) {

        $query = "insert into attendance values(".$studentID.", ".$courseID.", '".$date."', '".$time."')";

        $res = mysqli\_query($con, $query);

    }

?>

<html>

    <head>

        <style>

            div{

                width: 75%;

                height: 60%;

                background-color: honeydew;

                border: 1px double black;

                position: relative;

                left: 150px;

                top: 100px;

                display: none;

            }

            table{

                position: relative;

                top: 150px;

                font-size: 40px;

                font-family: monospace;

                color: darkcyan;

            }

        </style>

    </head>

    <body>

        <?php if ($rows == 0){ ?>

            <div class = "div">

                <table align = "center">

                    <tr>

                        <th>

                            Attendance Marked Successfully

                        </th>

                    </tr>

                    <tr>

                        <td style = "text-align: center;">

                            <br>

                            <a href = "home.php">Click here to go to home</a>

                        </td>

                    </tr>

                </table>

            </div>

        <?php }

        else{ ?>

            <div class = "div">

                <table align = "center">

                    <tr>

                        <th>

                            Student has already marked attendance for today

                        </th>

                    </tr>

                    <tr>

                        <td style = "text-align: center;">

                            <br>

                            <a href = "home.php">Click here to go to home</a>

                        </td>

                    </tr>

                </table>

            </div>

        <?php } ?>

    <script src = "JQuery.js"></script>

    <script>

        $(function(){

            $('.div').fadeIn(1000);

        });

    </script>

    </body>

</html>