**Final Project Report**

**On**

Student Information System (SIS)

(A desktop application)

**Prepared By**

**Muhammad Shahid**

**BCS-15-03**

**Session: 2015-19**

**Morning**

**Supervised By**

Dr.Shafiq Hussain

**Computer Sciences Department**

****

**Bahauddin Zakariya University Multan,**

**Campus Sahiwal**

**DEDICATION**

Our Parents and Teachers all who’ve given us their support during the development of this project and for giving good ideas to prove ourselves as intellectuals in front of our Respected Teachers.

**ACKNOWLEDGEMENT**

Praise to Allah Almighty, Lord of the worlds, the Merciful and the Beneficent, who gave us strength, thoughts and co-operative people to enable us to accomplish this goal and fulfill the required functionalities.

This was all not possible without the guidance, continuous appreciation and moral support by “**Dr.Shafiq Hussain”**. He was always there whenever we need his help and ideas. We are really thankful to him who made our concepts clearer.

At last, we would like to acknowledge all of the assistance and contributions of Bahauddin Zakariya University for supporting us with all that is needed starting from the books, and ending with the full care that it is providing us with, to help us to be professionals in the field of Computer Science.

**DECLARATION**

We hereby declare that we have developed this application and accompanied report entirely on the basis of our personal efforts. Not any of the portions of the application work presented has been submitted of any application for any other qualification or degree of this or any other university or institute of learning.

Student Name & Signature

**Muhammad Shahid**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**CERTIFICATE OF APPROVAL**

It is to certify that the final year project of **BS(CS) “Student Information System** (A desktop application)**”** was developed by **Muhammad Shahid** under the supervision of “**Dr.Shafiq Hussain**” and that in his opinion, it is in scope, fully adequacy and quality of the degree of Bachelors of Science in Computer Sciences.

**Supervisor External Examiner**

Dr.Shafiq Hussain Dr. Muhammad Asif

Assistant Professor Assistant Professor

Department of Computer Science Department of Computer Science

BZU, Sahiwal UOL, Pakpattan

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Head of Department**

Dr. Shafiq Hussain

Department of Computer Science

BZU, Sahiwal

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Abstract**

Student information System is a model system to store information about students, teachers, students result, classes, subjects, fees, attendance. The system is designed to meet the purpose of dealing with student information system.

As project mainly concentrate on student information system so keeping the friendly user interface the system should provide all necessary student information facilities. It provide the Administrator the facility to save, update, delete and view records of students, teachers, attendance and result.

Currently the System stores information in databases and retrieve information on the interfaces with the help of data access.

Table of Contents

[Chapter 1 1](#_Toc6932679)

[Introduction 1](#_Toc6932680)

[1.1. Project Overview 1](#_Toc6932681)

[1.2. Background study 1](#_Toc6932682)

[1.2.1. Historical perspective 1](#_Toc6932683)

[1.3. Scope 1](#_Toc6932684)

[1.3.1. Modules 1](#_Toc6932685)

[1.4. Features of Proposed System 1](#_Toc6932686)

[1.5 Tools & Techniques 1](#_Toc6932687)

[Chapter 2 1](#_Toc6932688)

[Requirement Analysis 1](#_Toc6932689)

[2.1. Software Requirement Specification 1](#_Toc6932690)

[2.1.1. Introduction 1](#_Toc6932691)

[2.1.2. Purpose 1](#_Toc6932692)

[2.1.3. Proposed System 1](#_Toc6932693)

[2.2. Specification Requirements 1](#_Toc6932694)

[2.2.1. Functional Requirements 1](#_Toc6932695)

[2.2.2. Non-Functional Requirements 1](#_Toc6932696)

[2.2.3. Optional Requirements 1](#_Toc6932697)

[2.3. Project Objective 1](#_Toc6932698)

[Chapter 3 1](#_Toc6932699)

[METHODOLOGY & WORKPLAN 1](#_Toc6932700)

[3.1 What is Mythology and why we need it? 1](#_Toc6932701)

[3.2 Adopted Methodology 1](#_Toc6932702)

[3.3 Roles & Responsibilities 1](#_Toc6932703)

[Chapter 4 1](#_Toc6932704)

[Software Design 1](#_Toc6932705)

[4.1 Activity Diagram 1](#_Toc6932706)

[4.1.1 Login Activity Diagram for Admin 1](#_Toc6932707)

[4.1.2 Activity Diagram for Student Information System 1](#_Toc6932708)

[4.2 Sequence Diagram 1](#_Toc6932709)

[4.2.1 SIS Sequence Diagram 1](#_Toc6932710)

[4.2.2 User Login 1](#_Toc6932711)

[4.2.3 Add Department 1](#_Toc6932712)

[4.2.4 Add Class 1](#_Toc6932713)

[4.2.5 Add Course 1](#_Toc6932714)

[4.2.6 Add Student Information 1](#_Toc6932715)

[4.2.7Add Teacher Information 1](#_Toc6932716)

[4.2.8Add Student Result 1](#_Toc6932717)

[4.2.9Add Student Attendance 1](#_Toc6932718)

[4.2.10 Add Student Fee Details 1](#_Toc6932719)

[4.3 Use Case Diagram 1](#_Toc6932720)

[4.3.1 Use Case Description 1](#_Toc6932721)

[*Use Case UC1: Login User* 1](#_Toc6932722)

[Main Success scenario: 1](#_Toc6932723)

[Extension: 1](#_Toc6932724)

[*Use Case UC2: Manage Department* 1](#_Toc6932725)

[Main Success scenario: 1](#_Toc6932726)

[Extension: 1](#_Toc6932727)

[*Use Case UC2: Manage Class* 1](#_Toc6932728)

[Main Success scenario: 1](#_Toc6932729)

[Extension: 1](#_Toc6932730)

[*Use Case UC2: Add Course* 1](#_Toc6932731)

[Main Success scenario: 1](#_Toc6932732)

[Extension: 1](#_Toc6932733)

[*Use Case UC2: Add Student Information* 1](#_Toc6932734)

[Main Success scenario: 1](#_Toc6932735)

[Extension: 1](#_Toc6932736)

[*Use Case UC2: Add Student Result* 1](#_Toc6932737)

[Main Success scenario: 1](#_Toc6932738)

[Extension: 1](#_Toc6932739)

[*Use Case UC2: Add Teacher Information* 1](#_Toc6932740)

[Main Success scenario: 1](#_Toc6932741)

[Extension: 1](#_Toc6932742)

[*Use Case UC2: Add Student Attendance* 1](#_Toc6932743)

[Main Success scenario: 1](#_Toc6932744)

[Extension: 1](#_Toc6932745)

[Main Success scenario: 1](#_Toc6932746)

[Extension: 1](#_Toc6932747)

[*Use Case UC2: Logout* 1](#_Toc6932748)

[Main Success scenario: 1](#_Toc6932749)

[Extension: 1](#_Toc6932750)

[4.4 Class Diagram 1](#_Toc6932751)

[4.5 ERD Diagram 1](#_Toc6932752)

[Chapter 5 1](#_Toc6932753)

[Implementation 1](#_Toc6932754)

[5.1 Introduction 1](#_Toc6932755)

[5.2 Tools and Technology Used 1](#_Toc6932756)

[5.2.1 Tools 1](#_Toc6932757)

[5.2.2 Technology 1](#_Toc6932758)

[5.2.3 Planning & Scheduling 1](#_Toc6932759)

[Gantt chart 1](#_Toc6932760)

[Chapter 6 1](#_Toc6932761)

[Database Design 1](#_Toc6932762)

[6.1 Database Design 1](#_Toc6932763)

[6.2 Database 1](#_Toc6932764)

[6.3 Database tables 1](#_Toc6932765)

[Chapter 7 1](#_Toc6932766)

[User Interface 1](#_Toc6932767)

[7.1 Introduction 1](#_Toc6932768)

[7.2 User Interface Front End 1](#_Toc6932769)

[7.2.1 Login 1](#_Toc6932770)

[7.2.2 Dashboard Page 1](#_Toc6932771)

[7.2.3 Department Page 1](#_Toc6932772)

[7.2.4 Course Page 1](#_Toc6932773)

[7.2.5 Course Mapping Page 1](#_Toc6932774)

[7.2.6 Student Register Page 1](#_Toc6932775)

[7.2.7 Student View page 1](#_Toc6932776)

[7.2.8 Teacher Register Page 1](#_Toc6932777)

[7.2.9 Teacher View Page 1](#_Toc6932778)

[7.2.10Add Result 1](#_Toc6932779)

[7.2.11 View Result 1](#_Toc6932780)

[7.2.12 Add attendance 1](#_Toc6932781)

[7.2.13 View Attendance 1](#_Toc6932782)

[7.2.14 Fee Module 1](#_Toc6932783)

[Chapter 8 1](#_Toc6932784)

[SYSTEM TESTING 1](#_Toc6932785)

[8.1 Introduction 1](#_Toc6932786)

[8.2 Testing Plan 1](#_Toc6932787)

[8.2.1 Unit Testing 1](#_Toc6932788)

[8.2.2 System Testing 1](#_Toc6932789)

[8.2.3 Integration Testing 1](#_Toc6932790)

[8.2.4 User Acceptance Testing 1](#_Toc6932791)

[8.3 Test Cases 1](#_Toc6932792)

[8.4 Testing Results 1](#_Toc6932793)

[Chapter 9 1](#_Toc6932794)

[Conclusion & Future Work 1](#_Toc6932795)

[9.1 Conclusion 1](#_Toc6932796)

[9.2 Future Work 1](#_Toc6932797)

[REFERENCES: 1](#_Toc6932798)

**Table of Figures**

[Figure 3.1:Adopeted Methodology](#_Toc6694132) 13

[Figure 4.1.1: Login Activity Diagram](#_Toc6694133) 14

[Figure 4.1.2: Student Information System Activity Diagram](#_Toc6694134) 15

[Figure 4.2.1: Student Information System Sequece Diagram](#_Toc6694135) 16

[Figure 4.2.2: Login Sequece Diagram](#_Toc6694135) 17

[Figure 4.2.3: Add department Sequece Diagram](#_Toc6694135) 18

[Figure 4.2.4: Add class Sequece Diagram](#_Toc6694135) 19

[Figure 4.2.5: Add Course Sequece Diagram](#_Toc6694135) 20

[Figure 4.2.6: Add Student Sequece Diagram](#_Toc6694135) 21

[Figure 4.2.7: Add Teacher Sequece Diagram](#_Toc6694135) 22

[Figure 4.2.8: Add Student Result Sequece Diagram](#_Toc6694135) 23

[Figure 4.2.9: Add Student Attedance Sequece Diagram](#_Toc6694135) 24

[Figure 4.2.10: Add Student Fee Sequece Diagram](#_Toc6694135) 25

[Figure 4.3: Student Information System Use Case Diagram](#_Toc6694135) 26

[Figure 4.4: Student Information System Class Diagram](#_Toc6694135) 37

[Figure 4.5: Student Information System ERD Diagram](#_Toc6694135) 38

[Figure 6.3.1: Admin Database Table(screenshot)](#_Toc6694135) 67

[Figure 6.3.2: Department Database Table(screenshot)](#_Toc6694135) 67

[Figure 6.3.3: Class Database Table (screenshot)](#_Toc6694135) 68

[Figure 6.3.4: Coursse Database Table(screenshot)](#_Toc6694135) 68

[Figure 6.3.5: CourseMapping Database Table(screenshot)](#_Toc6694135) 69

[Figure 6.3.6: Student Database Table(screenshot)](#_Toc6694135) 69

[Figure 6.3.7: Teacher Database Table(screenshot)](#_Toc6694135) 70

[Figure 6.3.8: Student Attendance Database Table(screenshot)](#_Toc6694135) 70

[Figure 6.3.9: Student Result Database Table(screenshot)](#_Toc6694135) 71

[Figure 6.3.10: Student Fee Database Table(screenshot)](#_Toc6694135) 71

[Figure 7.2.1: Login Page(screenshot)](#_Toc6694135) 73

[Figure 7.2.2: Dassh Board Page(screenshot)](#_Toc6694135) 74

[Figure 7.2.3: Department Page(screenshot)](#_Toc6694135) 74

[Figure 7.2.4: Course Page(screenshot)](#_Toc6694135) 75

[Figure 7.2.5: Course Mapping Page(screenshot)](#_Toc6694135) 75

[Figure 7.2.6: Student Registeration Page(screenshot)](#_Toc6694135) 76

[Figure 7.2.7: Student View Page(screenshot)](#_Toc6694135) 76

[Figure 7.2.8: Teacher Registeration Page(screenshot)](#_Toc6694135) 77

[Figure 7.2.9: Teacher View Page(screenshot)](#_Toc6694135) 77

[Figure 7.2.10: Add Result Page(screenshot)](#_Toc6694135) 78

[Figure 7.2.11: View Result Page(screenshot)](#_Toc6694135) 78

[Figure 7.2.12: Student Attendance Page(screenshot)](#_Toc6694135) 79

[Figure 7.2.13: Student Attendance View Page(screenshot)](#_Toc6694135) 79

[Figure 7.2.14: Student Fee Record Page(screenshot)](#_Toc6694135) 80

# Chapter 1

# Introduction

## Project Overview

Student data are the main element for any educational institute. “Student Information Management System” provides us a simple interface for maintenance of student information. It can be used by educational institutes or colleges to maintain the records of students easily. Achieving this objective is difficult using a manual system as the information is scattered, can be redundant and collecting relevant information may be very time consuming. All these problems are solved using Student Information Management System project. Throughout the project the focus has been on presenting information in an easy and intelligible manner. The project is very useful for school, colleges, academies and universities. The project provides facilities like department management, class management and course management, registration and profile creation of student as well as managing the student academic record and result thus reducing paperwork and automating the record generation process in an educational institution.

The objective of Student information System is to allow the administrator to edit and find out the academic record as well as students personal information. It’ll also facilitate keeping all the records of students, such as their id, name, mailing address, phone number, DOB etc. So, all the information about a student will be available in a few seconds. Overall, it’ll make Student Information Management an easier job for the administrator to manage the students in institute .The main purpose of this SRS document is to illustrate the requirements of the project Student information System and is intended to help any organization to maintain and manage its student’s academic record as well as personal data.

## Background study

Technology evolution played an important role in the success of many businesses and helping business to become more competitive in the industry as well as the economy. Better quality of life is often achieved with the effect of the creation of IT. As technology continues to evolve, computer is largely dominating the work of a lot of people. We are now benefiting the fruit of its wonders, advancement of connectivity, communication, accessible online devices were invented to enable man to add, subtract and to record simple transactions.

Student Information Management System is used in recording a student’s information. A well-built one will reduce the load on the people that normally have to do all the work. Student Information Management System is a good step for a school. It is useful especially when the school retrieves the important information from the student. Through it the school can trace what are the standings of the students. Lack of enrollment system in a school can lead to chaos and troubles, the students will be confused on what they should do and how they will do to be able to enroll. It is extremely useful in the school in the way of making the processes of enrolling much easy.

### Historical perspective

The Student Information Management System is the SIS that will hold the business in its goal to a paperless office environment. From the acronym itself, the SIS is that management process that will protect information and data to ensure that all the records and the files in the system database of the business organization is protected and easily accessed by users in the framework. In the shortest definition of the automated records management system, it is simply that process that is employed by the organization in the maintenance of records and files from the very moment that they arrive to the business or were created by the enterprise up to the period when they will be deemed ready for eradication and deletion from the archived or active databases. The records handled by the Student Information Management System may come in tangible forms that will be translated to digital formats with the use of the latest technologies in the market.

According to Muhenda & Lwanga (2000) noted that Student record used for storing large database or knowledge base. It can be used for knowing the current status of any aspect of the business due to its on-line real time processing capability. As governance issues take center stage in the management of Educational Institutions, the management of students’ records becomes imperative in the improvement of services offered in Higher Educational Institutions (HEI’s) in Uganda. Proper records management underpins policy formulation, decision making, protects interests of organization, and protects rights of employers and students in addition to helping Institutions conduct business and deliver services in a consistent and equitable manner. There are five Areas for Student Information Management System which includes:

* The Student Information Management System will affect in its inception to the business organization. Such as the system will work in the active records section. This mean that it can give you access and help you manage the most current documents and data that the operations are handling.
* The second sector would be the data protection and jumping, which is the area that will ensure all the files are properly filed and archived for record keeping purposes. In time, the next area and sector would be the determination of destruction of the documents in a secured manner that will ensure that all of the data from the files will still be kept secret and confidential.
* The other sections would be the records management area and the document management and imaging.

The student records management system will give the business organization streamlined operations that will reduce paper based transactions to the bare minimum. Simply, the automated records management system is a method that will do all the filing and management of documents for the organization.

## Scope

Without Student information Management System, managing and maintaining the details of the student is a tedious job for any Institute. Student Information system will store all the details of the students including their academic result course offered background information, educational qualifications, personal. Student Information System also manage departments, classes and course offered in an institute.

### Modules

1. Login module
2. Student registration Module
3. Student Information Module
4. Department Management Module
5. Class Management Module
6. Subject/Course Management Module
7. Student Result Management Module
8. Student Attendance Module
9. Student Fee Module
10. Teacher Registration Module
11. Teacher Profile View Module

#### Login Module

Login module will help in authentication of user account. User who has valid login id and password can only login into their respective accounts.

#### Student Registration Module

Student Registration module allow the administrator to add the student personal detail like name, father name, data of birth, CNIC, cell number, email and student address, department, class, session and registration date. The administrator can add, delete, and update the student information with ease.

#### Student View Module

Student view module allows the administrator to view the records of students registered in different departments. Administration can also view the single student details by using student id.

#### Department Management Module

Department management module will help the administrator to manage the departments in the institute or organization. The administrator can add new department, delete, and update and view the department detail.

#### Class Management Module

Class management module will help the administrator to manage the Class in the institute. The administrator can adds new Class, delete, and update and view the Class detail with ease.

#### Subject/Course Management Module

Course management module allows the user to add course detail like course name, course code, course credit hour. Course management module will help the administrator to manage and maintain all the courses. The administrator can adds new Course, delete, and update and view the course detail with ease.

#### Student Result Management Module

Student result management module allows the user to mark the result of student in each subject/course.

#### Student Attendance Module

Student attendance module allows the admin to enter, update and view the attendance of a student in each subject/course.

#### Student Fee Module

Student fee module allows the admin to keep record of students’ fee.

#### Teacher Registration Module

Teacher Registration module allow the administrator to add the teacher personal detail like name, father name, data of birth, CNIC, cell number, email and student address, department, and registration hiring date. The administrator can add, delete, update and view the teacher information with ease.

#### Teacher View Module

Teacher view module allows the administrator to view the records of teachers registered. Administration can also view the single teacher details by using student id.

## Features of Proposed System

Below is the list of main features of proposed system.

1. User login
2. Maintain student information (save, delete, update student information and also search the specific record).
3. Maintain student result (save, update student result record and also search the specific record).
4. Maintain department detail (save, delete, update department information and also search the specific record).
5. Maintain class detail (save, delete, update class information and also search the specific record).
6. Maintain course detail (save, delete, update course information and also search the specific record).
7. Maintain student attendance record(save, update and view attendance of a specific student or whole class)
8. Maintain student fee record(save, update and search the fee details of a specific student or whole class)
9. Maintain Teacher Information(save, delete, update and search the specific teacher details or can view the all teachers’ record)

## 1.5 Tools & Techniques

Student Information System is a desktop application consists of software and hardware tools.

**1.5.1 Hardware Specifications:**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Processor |  | Core 2 Quad (or above) |
|  | RAM |  | 4 GB (or above) |
|  | Hard disk |  | 80 GB(or above) |
| **1.5.2 Software Specifications:** | | | |
|  | Operating System | Window XP, Window 7 or above | |
|  | IDE | Eclipse Neon | |
|  | UI Design Tool | ScenceBuilder | |
|  | Database | | MySQL |

# Chapter 2

# Requirement Analysis

## Software Requirement Specification

### Introduction

Student data are the main element for any educational institute. “Student Information Management System” provides us a simple interface for maintenance of student information. It can be used by educational institutes or colleges to maintain the records of students easily. Achieving this objective is difficult using a manual system as the information is scattered, can be redundant and collecting relevant information may be very time consuming. All these problems are solved using Student Information Management System project. Throughout the project the focus has been on presenting information in an easy and intelligible manner. The project is very useful for school, colleges, academies and universities. The project provides facilities like department management, class management and course management, registration and profile creation of student as well as managing the student academic record and result thus reducing paperwork and automating the record generation process in an educational institution.

### Purpose

The current problem in the standard school is did not have systematic data arrangement in the student management. When the staff of administrator wants to record the data of the student (student academic record as well as personal information), they need to fill out by use the manual system, In this case data might be lost when several problem is occur. Other problem that can be happen is hard to search and update the student information and class arrangement. The systematic requirement is required so that all data is stored into the database for future reference and enhancement. Below is the specific problem statement that occurs in standard school via using manual system.

* Lack of data arrangement that is record by using manual system and using a lot of paper to record the student information, student result and performance.
* The manual system is hard to search and update about the student information, result and performance
* The manual system is not providing the security of the academic information that might be lost.

### Proposed System

The proposed SIS system provides detail general information about the students along with Educational, SIS is used for adding, viewing information and updating students’ details. SIS Emphasizes the system's functionality, database design and functional modules. Some advantages of proposed system are mentioned below:-

* It is very fast and clever.
* Not easy for data loss.
* Short time to knowledge and learning to use operate the system.
* Need short time to find any student information.
* No need for more time.
* Easy to update any data.

#### Modules of Proposed System

1. Login module
2. Student registration Module
3. Student Information Module
4. Department Management Module
5. Class Management Module
6. Subject/Course Management Module
7. Student Result Management Module
8. Student Attendance Module
9. Student Fee Module
10. Teacher Registration Module
11. Teacher Profile View Module

#### Login Module

Login module will help in authentication of user account. User who has valid login id and password can only login into their respective accounts.

#### Student Registration Module

Student Registration module allow the administrator to add the student personal detail like name, father name, data of birth, CNIC, cell number, email and student address, department, class, session and registration date. The administrator can add, delete, and update the student information with ease.

#### Student View Module

Student view module allows the administrator to view the records of students registered in different departments. Administration can also view the single student details by using student id.

#### Department Management Module

Department management module will help the administrator to manage the departments in the institute or organization. The administrator can add new department, delete, and update and view the department detail.

#### Class Management Module

Class management module will help the administrator to manage the Class in the institute. The administrator can adds new Class, delete, and update and view the Class detail with ease.

#### Subject/Course Management Module

Course management module allows the user to add course detail like course name, course code, course credit hours. Course management module will help the administrator to manage and maintain all the courses. The administrator can adds new course, delete, and update and view the course detail with ease.

#### Student Result Management Module

Student result management module allows the user to mark the result of student in each subject/course.

#### Student Attendance Module

Student attendance module allows the admin to enter the attendance of a student in each subject/course.

#### Student Fee Module

Student fee module allows the admin to keep record of students’ fee.

#### Teacher Registration Module

Teacher Registration module allow the administrator to add the teacher personal detail like name, father name, data of birth, CNIC, cell number, email and teacher address, department, class, session and registration date. The administrator can add, delete, update and view the teacher information with ease.

#### Teacher View Module

Teacher view module allows the administrator to view the records of teacher registered. Administration can also view the single student details by using teacher id.

## Specification Requirements

### Functional Requirements

1. Administrator shall have rights to maintain student information (save, delete, update student information and also search the specific record).
2. Administrator shall have rights to maintain student result (save, delete, update student result record and also search the specific record).
3. Administrator shall have rights to maintain department detail (save, delete, update department information and also search the specific record).
4. Administrator shall have rights to maintain class detail (save, delete, update class information and also search the specific record).
5. Administrator shall have rights to maintain course detail (save, delete, update course information and also search the specific record).
6. Administrator shall have rights to maintain student attendance detail (save, delete, update course attendance information and also search the specific record).
7. Administrator shall have rights to maintain students’ fee detail (save, delete, update fee information and also search the specific record).
8. Administrator shall have rights to maintain teacher detail (save, delete, update teacher information and also search the specific record).

### Non-Functional Requirements

#### Performance Requirements

The proposed system that we are going to develop will be used as the chief performance system for providing help to the organization in managing the whole database of the student studying in the institute. Therefore, it is expected that the database would perform functionally all the requirements that are specified.

#### Safety Requirements

The database may get crashed at any certain time due to virus or operating system failure. Therefore, it is required to take the database backup.

#### Security Requirements:

We are going to develop such system that will secure only administrator access the system after authentication and maintain and manage the student record and perform some other activity.

### Optional Requirements

1. Performance requirement
2. User interface consideration
3. Global access and functionality
4. Application and database management
5. User and system documentation and training

## Project Objective

The current problem in the standard school is did not have systematic data arrangement in the student management. When the staff of administrator wants to record the data of the student (student academic record as well as personal information), they need to fill out by use the manual system, In this case data might be lost when several problem is occur. Other problem that can be happen is hard to search and update the student information and class arrangement. The systematic requirement is required so that all data is stored into the database for future reference and enhancement.

# Chapter 3

# METHODOLOGY & WORKPLAN

## 3.1 What is Mythology and why we need it?

Project development team is consisting of two members. In order to accomplish a goal, documentation and development is equally distributed among them and each member work on parallel to avoid wastage of time.

Whenever a small or large project has started to develop, first thing all of programmers required is methodology. Methodology is a way of developing a project, in which all of the programmers gather the user’s requirements, design the project, implement it, and after all this testing and maintenance of the project, in a satisfaction of user and according to the project requirements.

## 3.2 Adopted Methodology

Incremental model is used to develop this project, in which we divided our work in multiple modules. All these modules are further divided into more easily managed modules which made up the actual implementation of the requirements.

Reason behind using this model is:

* It is easy to test and debug the product during iterations.
* Software released in increments over time is more likely to satisfy changing user requirements than if it were planned as a single overall release at the end of the same period.
* Generates working software quickly and early during the software life cycle.
* This model is more flexible – less costly to change scope and requirements.
* It is easier to test and debug during a smaller iteration.
* In this model customer can respond to each built.
* Lowers initial delivery cost.
* Easier to manage risk because risky pieces are identified and handled during it’d iteration.

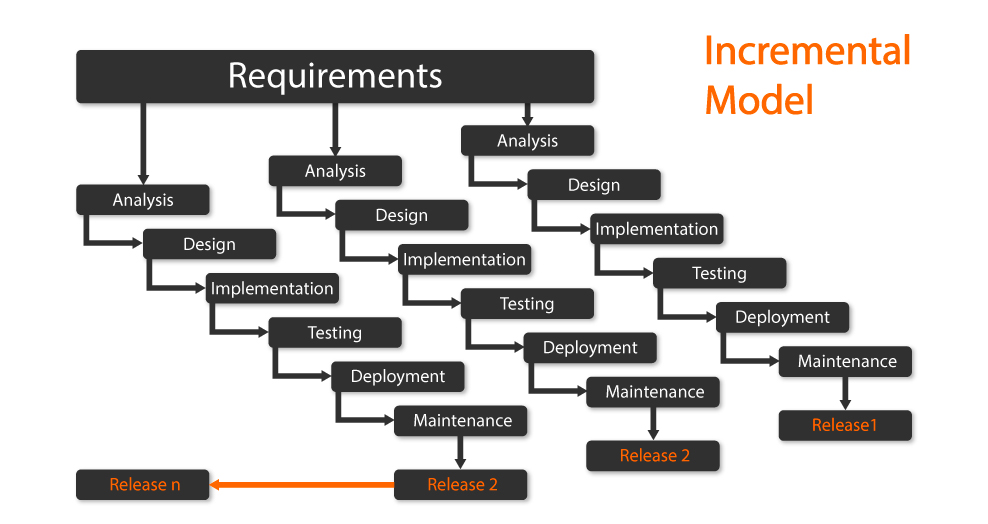
****

Figure 3.1: Adopted Methodology

|  |  |
| --- | --- |
| **Requirement Analysis** | * Requirement and specification of the software are collected |
| **Design** | * Some high-end function are designed during this stage |
| **Code** | * Coding of software is done during this stage |
| **Test** | * Once the system is deployed, it goes through the testing phase |

## 3.3 Roles & Responsibilities

Project development team is consisting of two members. In order to accomplish a goal, documentation and development is equally distributed among them and each member work on parallel to avoid wastage of time.

# Chapter 4

# Software Design

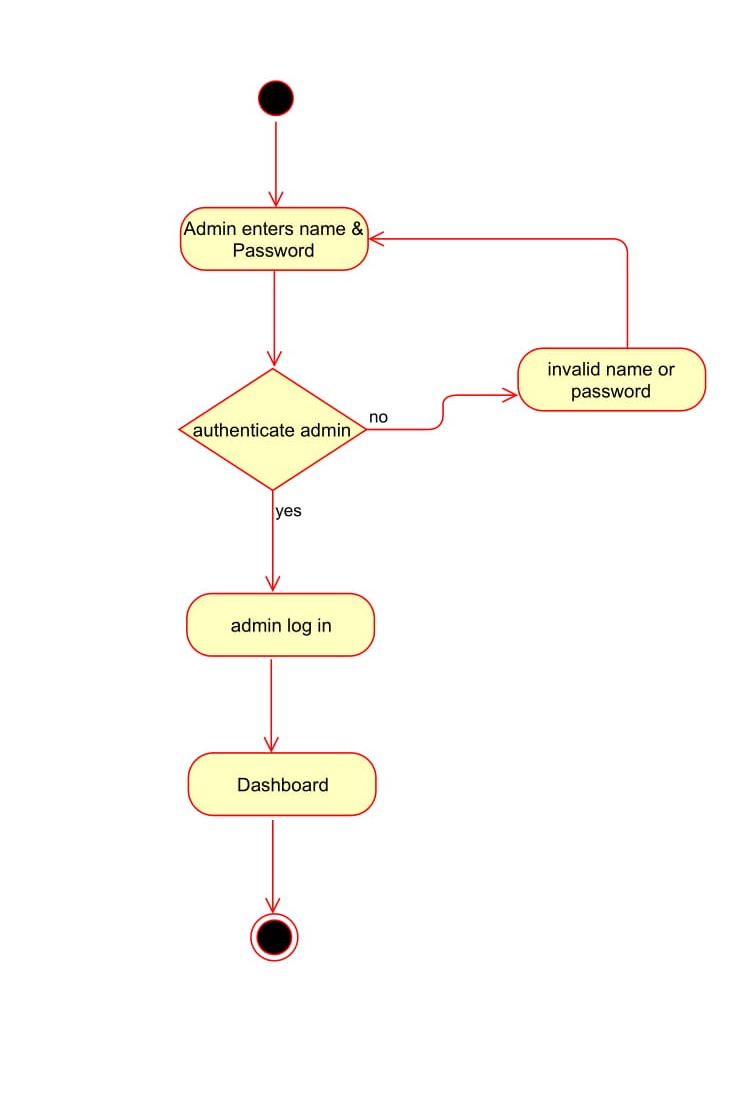
In this chapter requirements analysis, feasibility study, planning, forecasting, modeling, scheduling and design of the project is discussed. For developing any project, the major problem is requirement gathering. Asking questions from clients is straightforward than collecting requirements. We will also focus on functional and non-functional requirements.

The procedure for gathering requirements has its own defined procedure according to the complexity of the application. To define project schedule and processing, different models and techniques also focused on this chapter.

## Activity Diagram

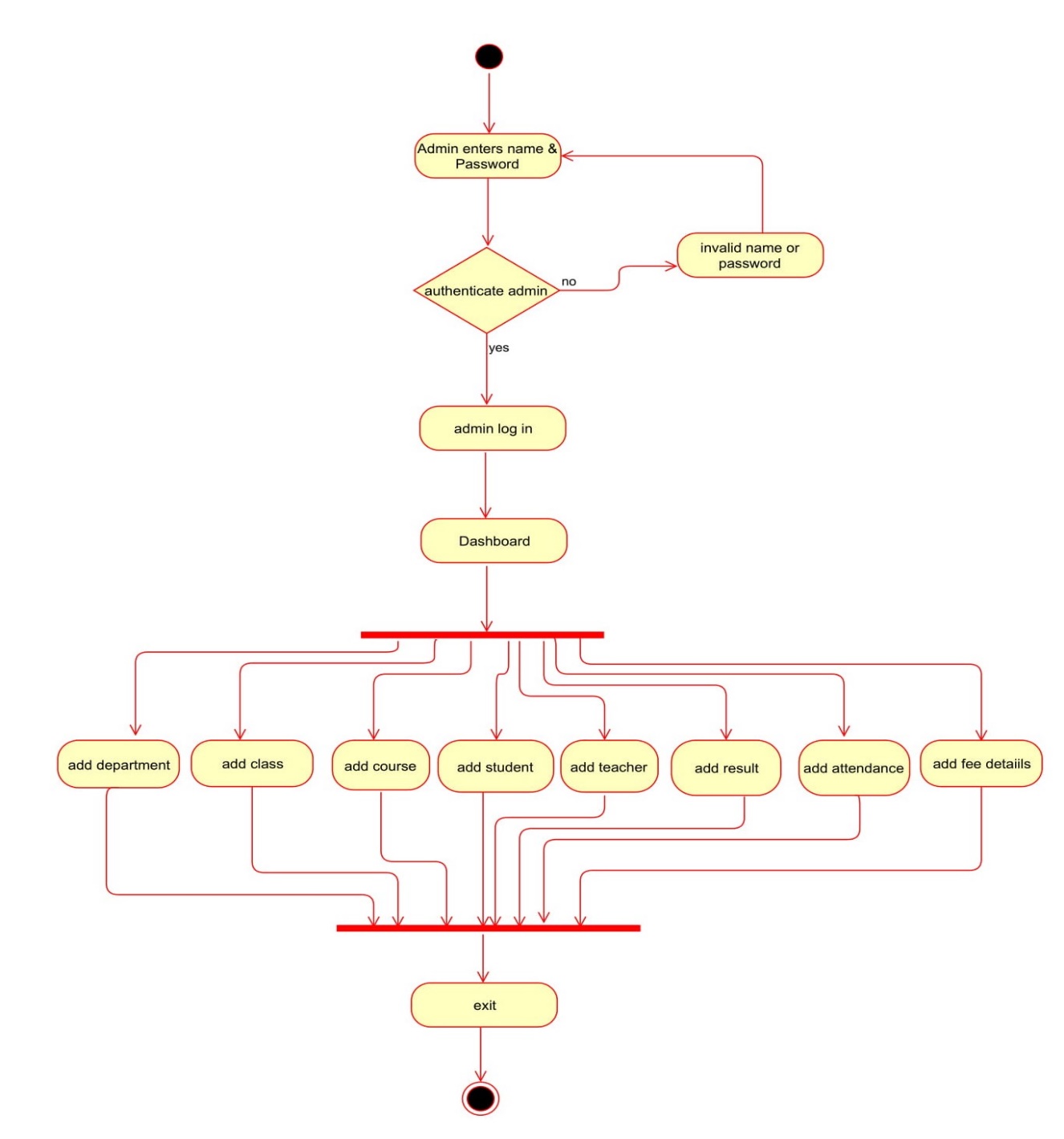
Activity diagrams are graphical representations of [workflows](https://en.wikipedia.org/wiki/Workflow) of stepwise activities and actions with support for choice, iteration and concurrency. In the [Unified Modeling Language](https://en.wikipedia.org/wiki/Unified_Modeling_Language), activity diagrams are intended to model both computational and organizational processes (i.e., workflows), as well as the data flows intersecting with the related activities.

## Login Activity Diagram for Admin



**Figure 4.1.1 Admin Login Activity Diagram**

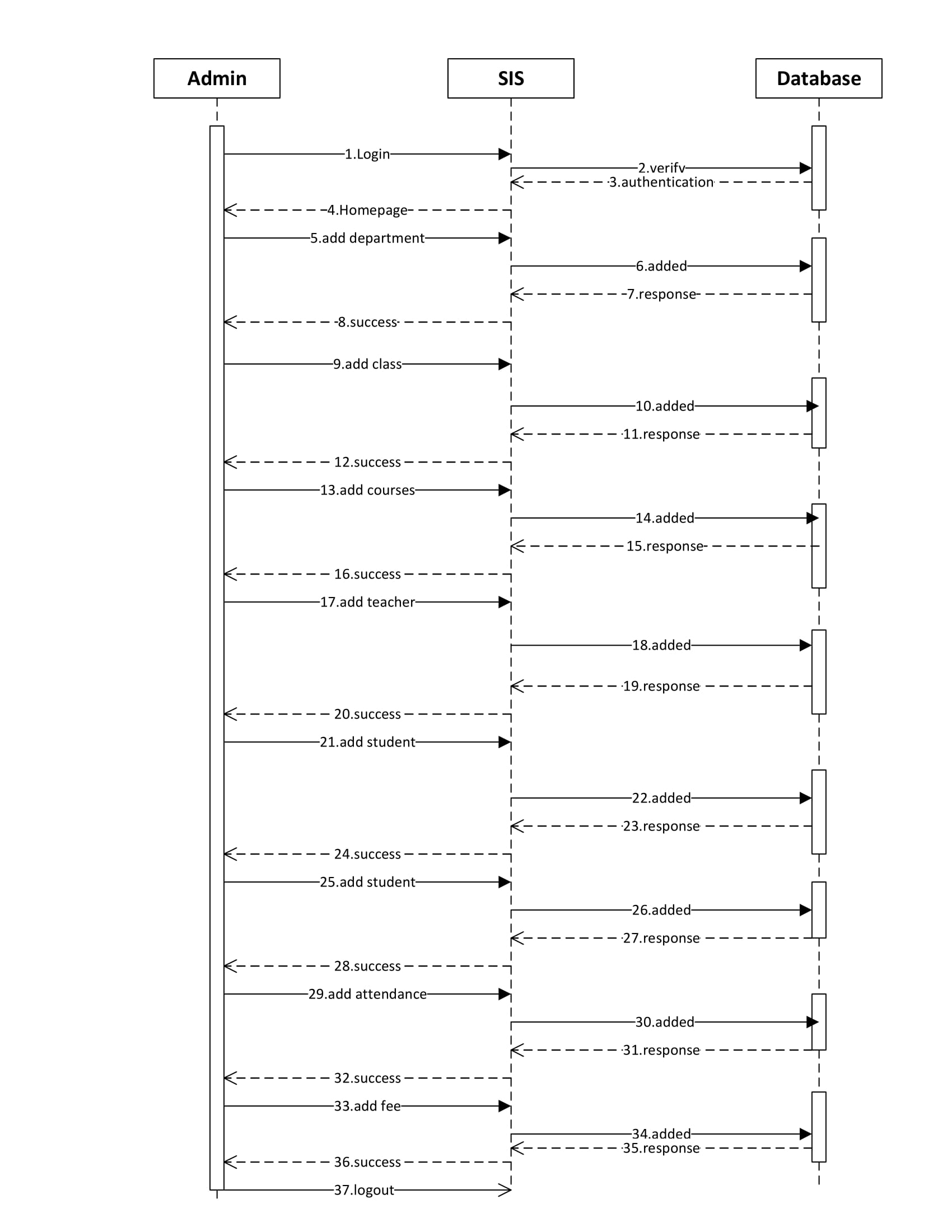
## Activity Diagram for Student Information System

****

**Figure 4.1.2 Student Information System Activity Diagram**

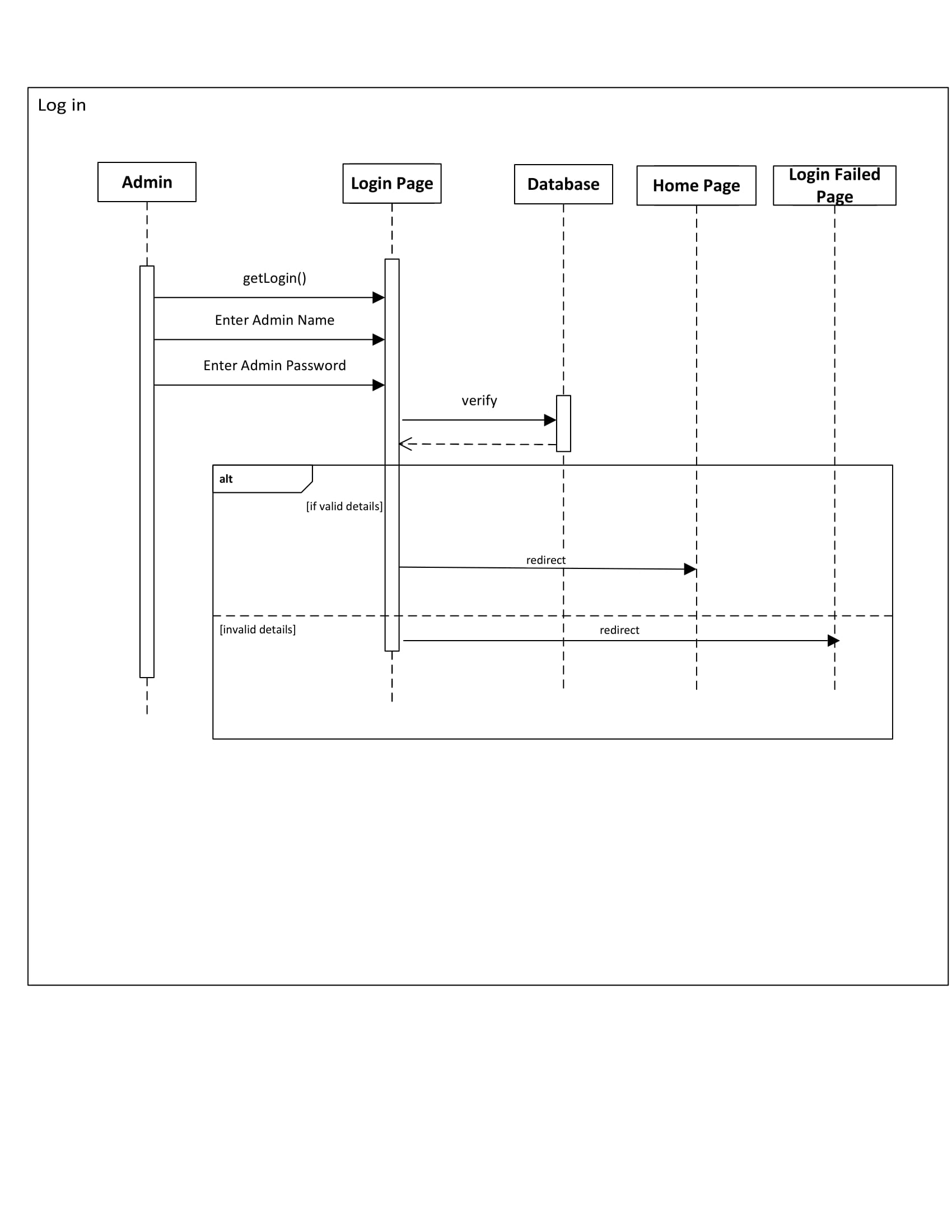
## Sequence Diagram

### 4.2.1 SIS Sequence Diagram



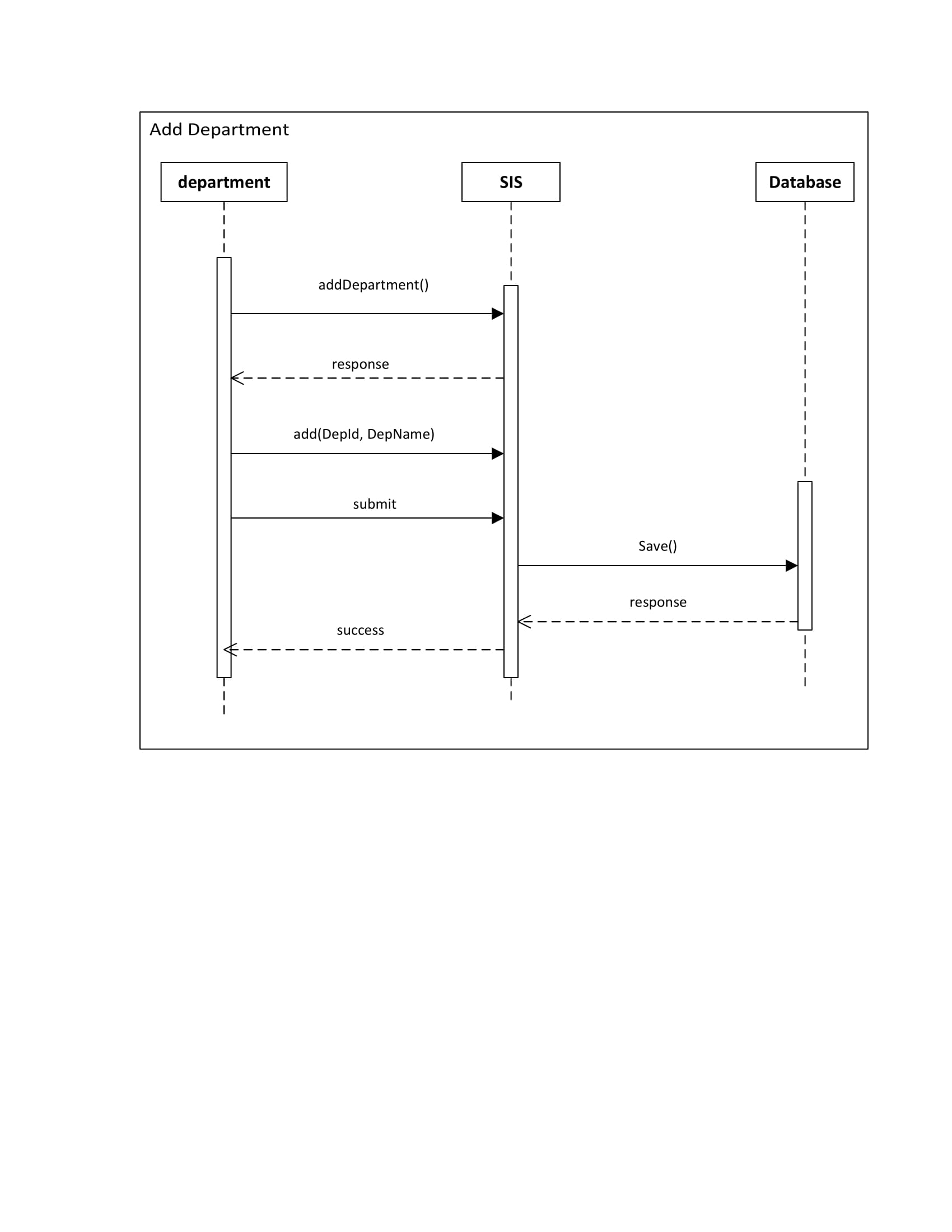
**Figure 4.2.1 Sequence Diagram**

### 4.2.2 User Login



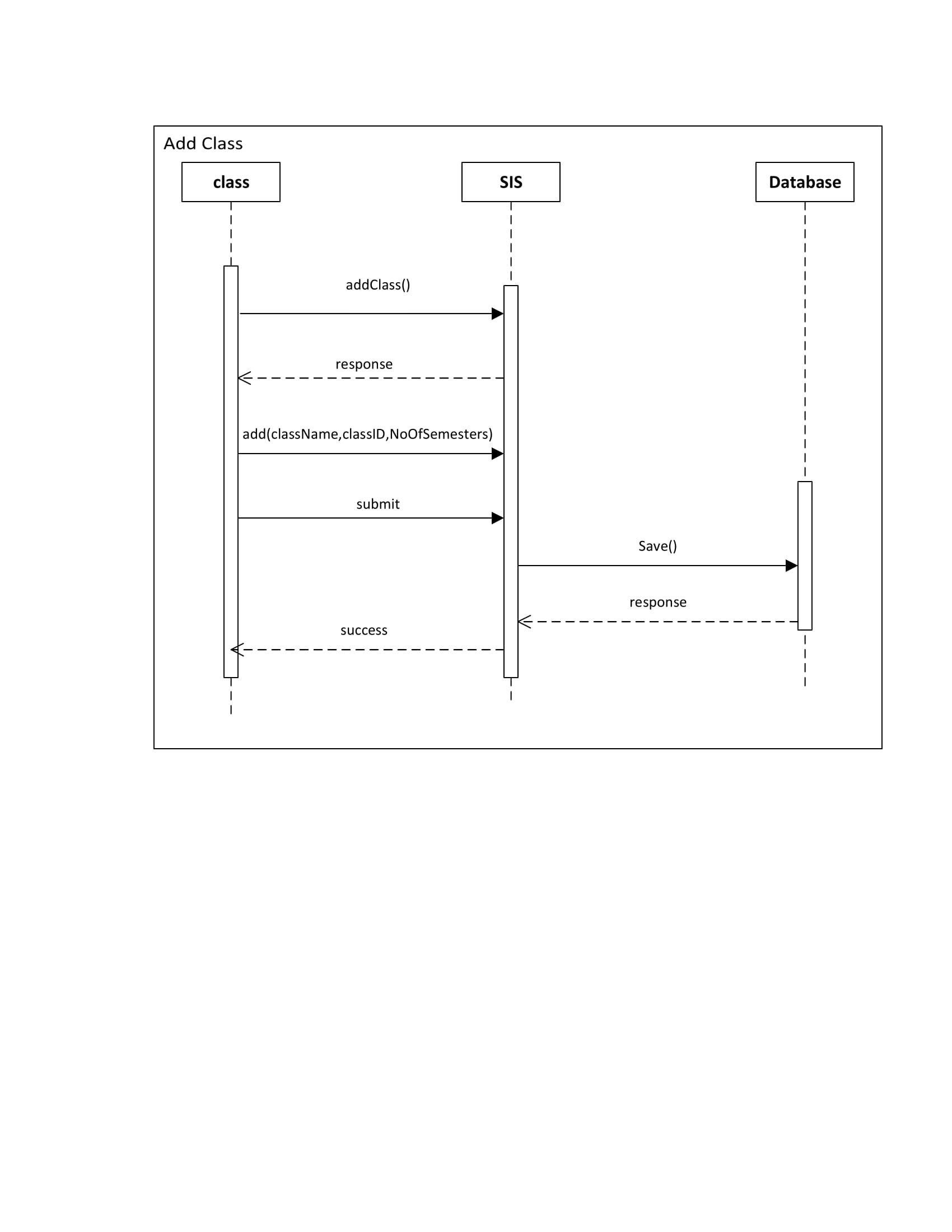
**Figure 4.2.2 User Login Sequence Diagram**

### 4.2.3 Add Department



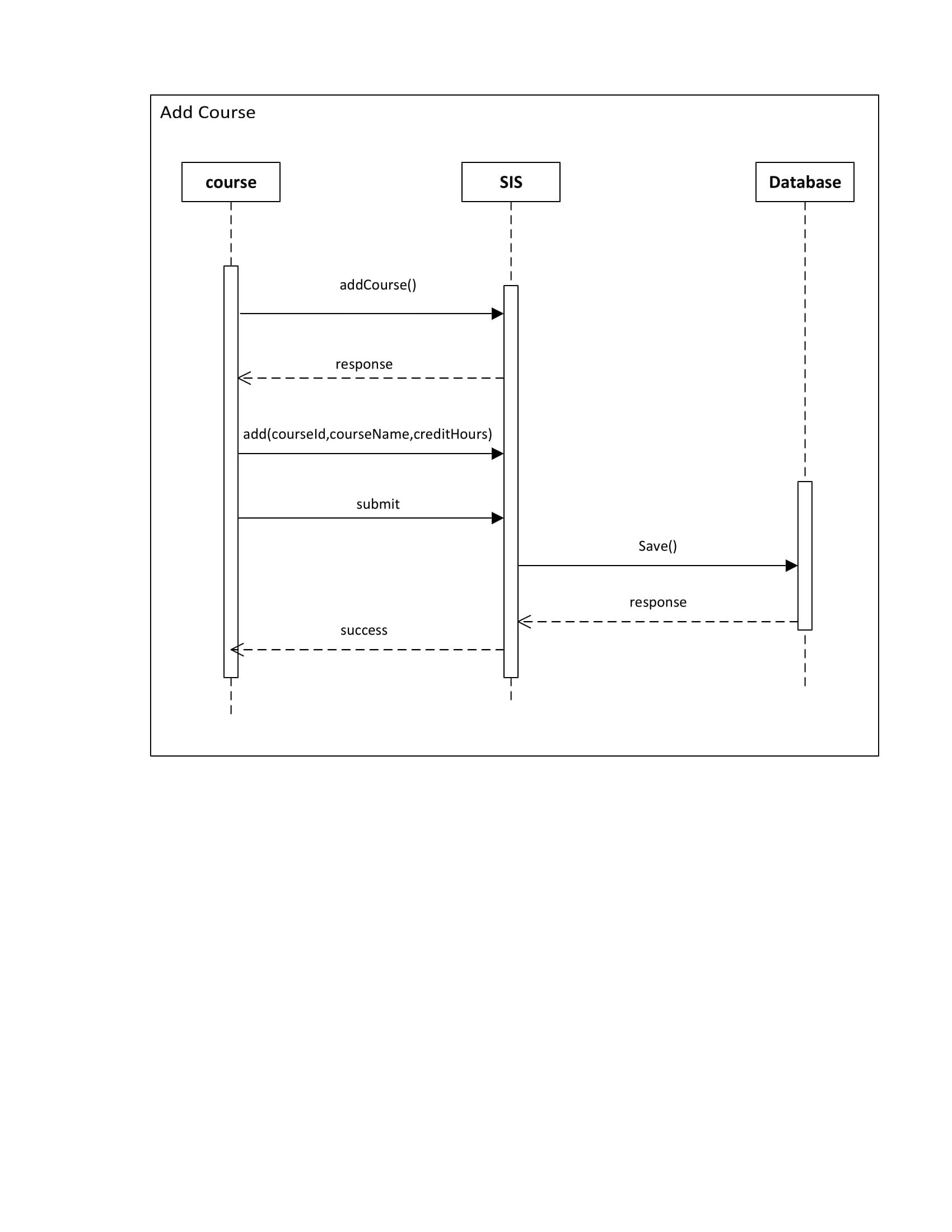
**Figure 4.2.3 Add Department Sequence Diagram**

### 4.2.4 Add Class



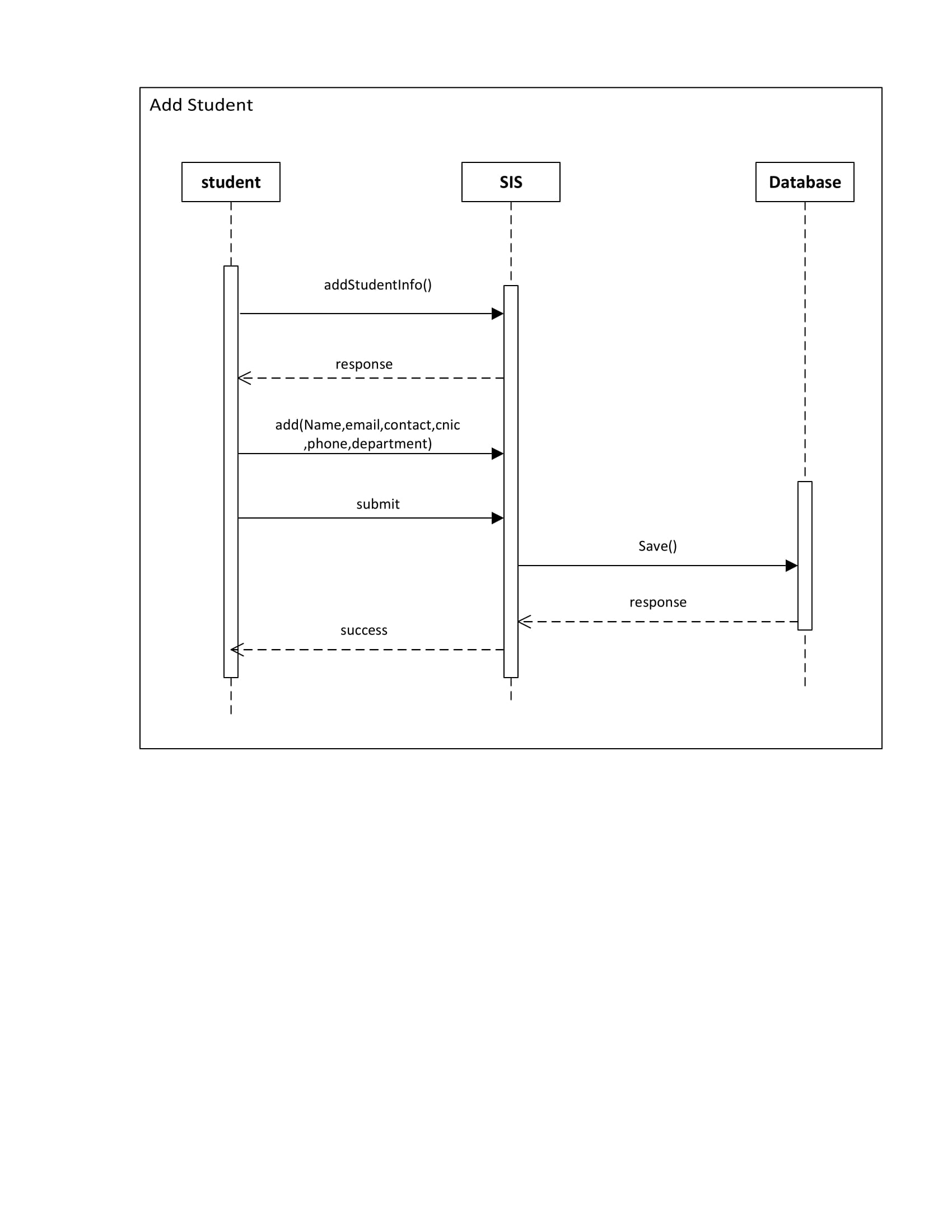
**Figure 4.2.4 Add Class Sequence Diagram**

### 4.2.5 Add Course



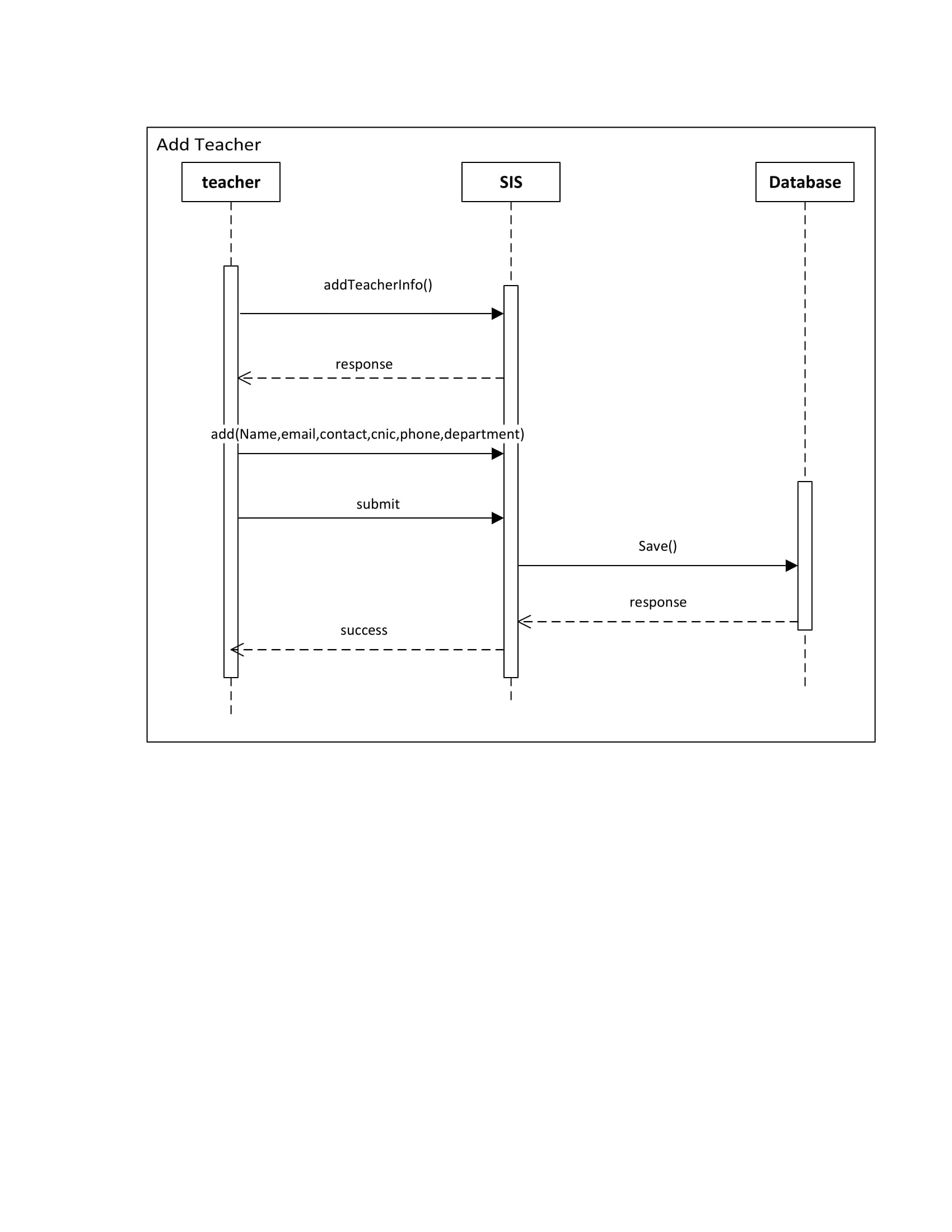
**Figure 4.2.5 Add Course Sequence Diagram**

### 4.2.6 Add Student Information



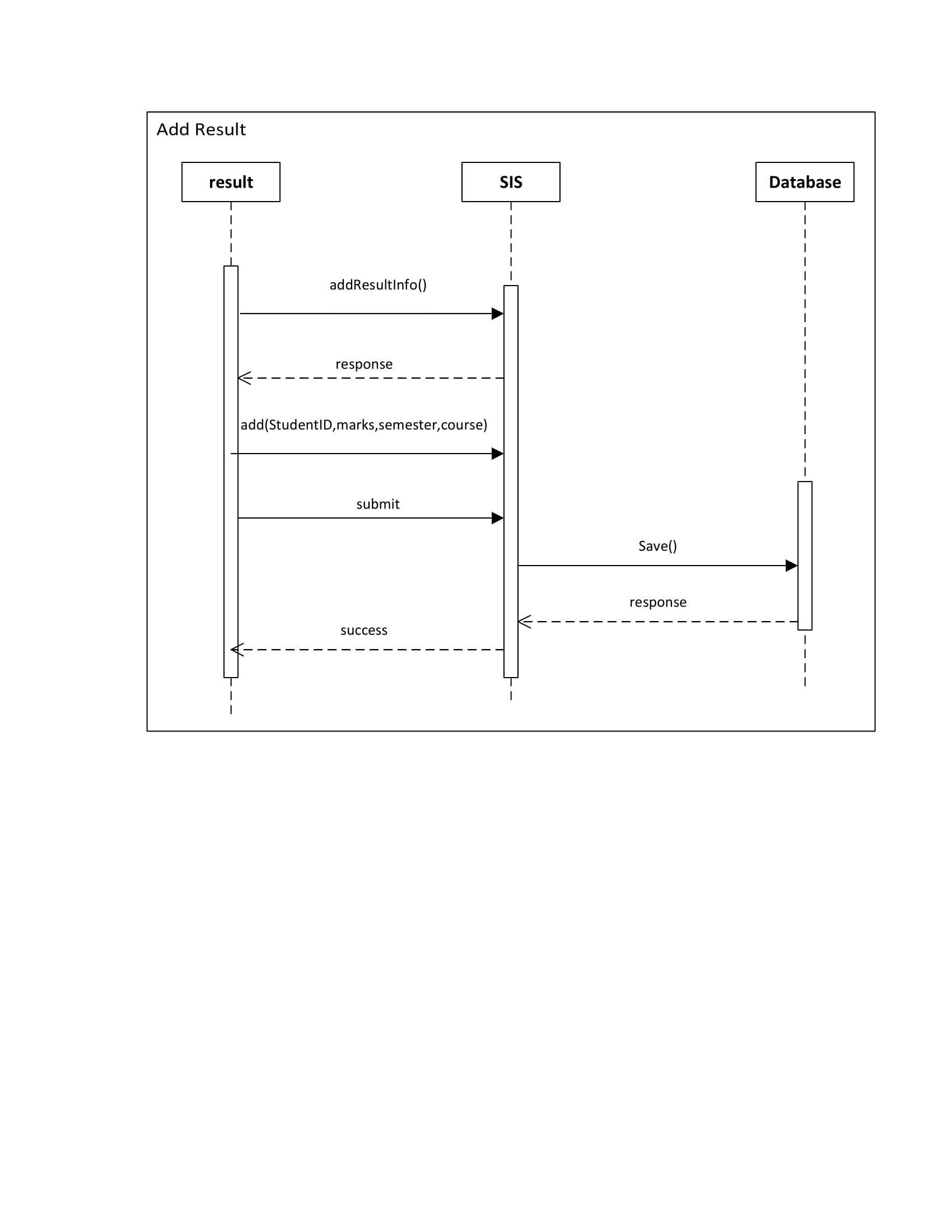
**Figure 4.2.6 Student Sequence Diagram**

### 4.2.7Add Teacher Information



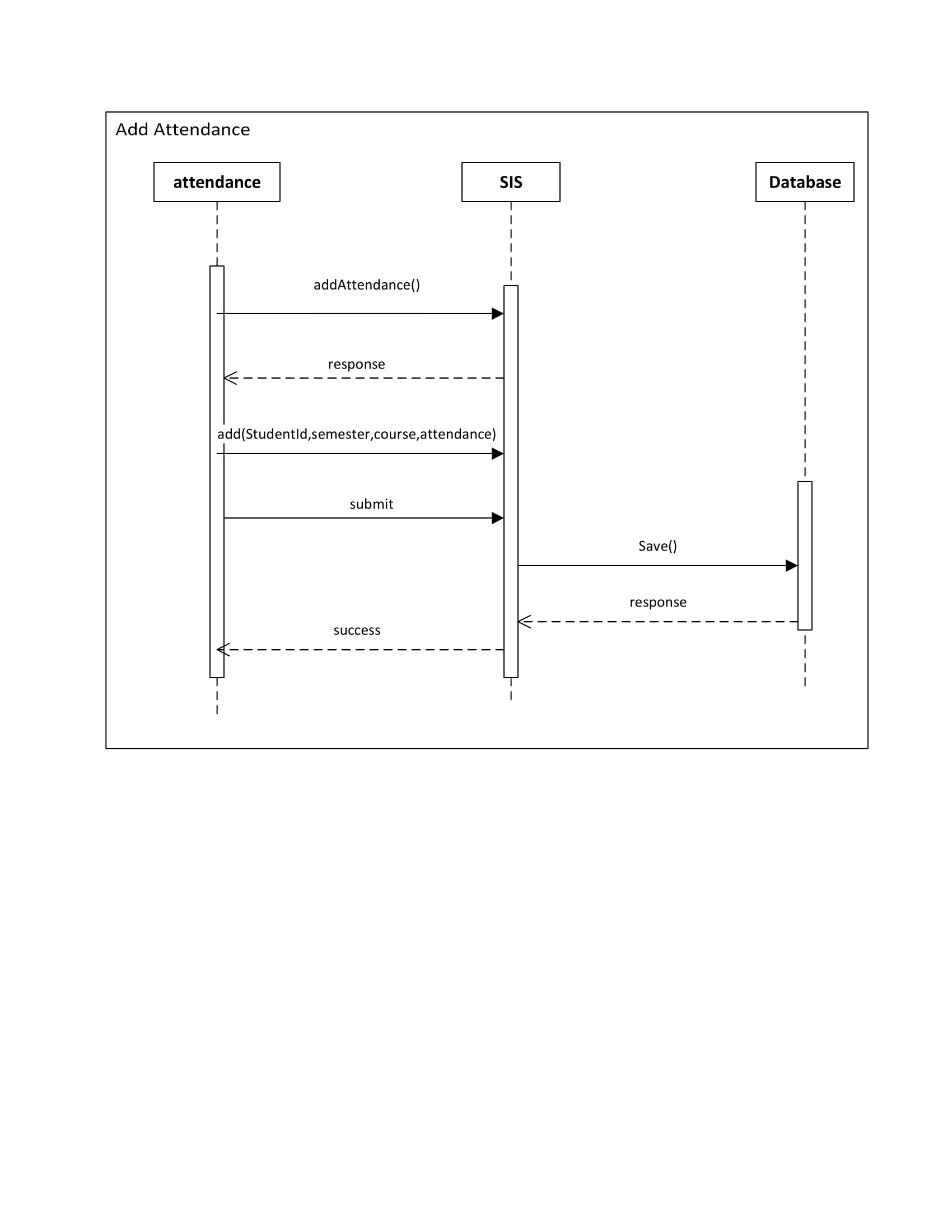
**Figure 4.2.7 Teacher Sequence Diagram**

### 4.2.8Add Student Result



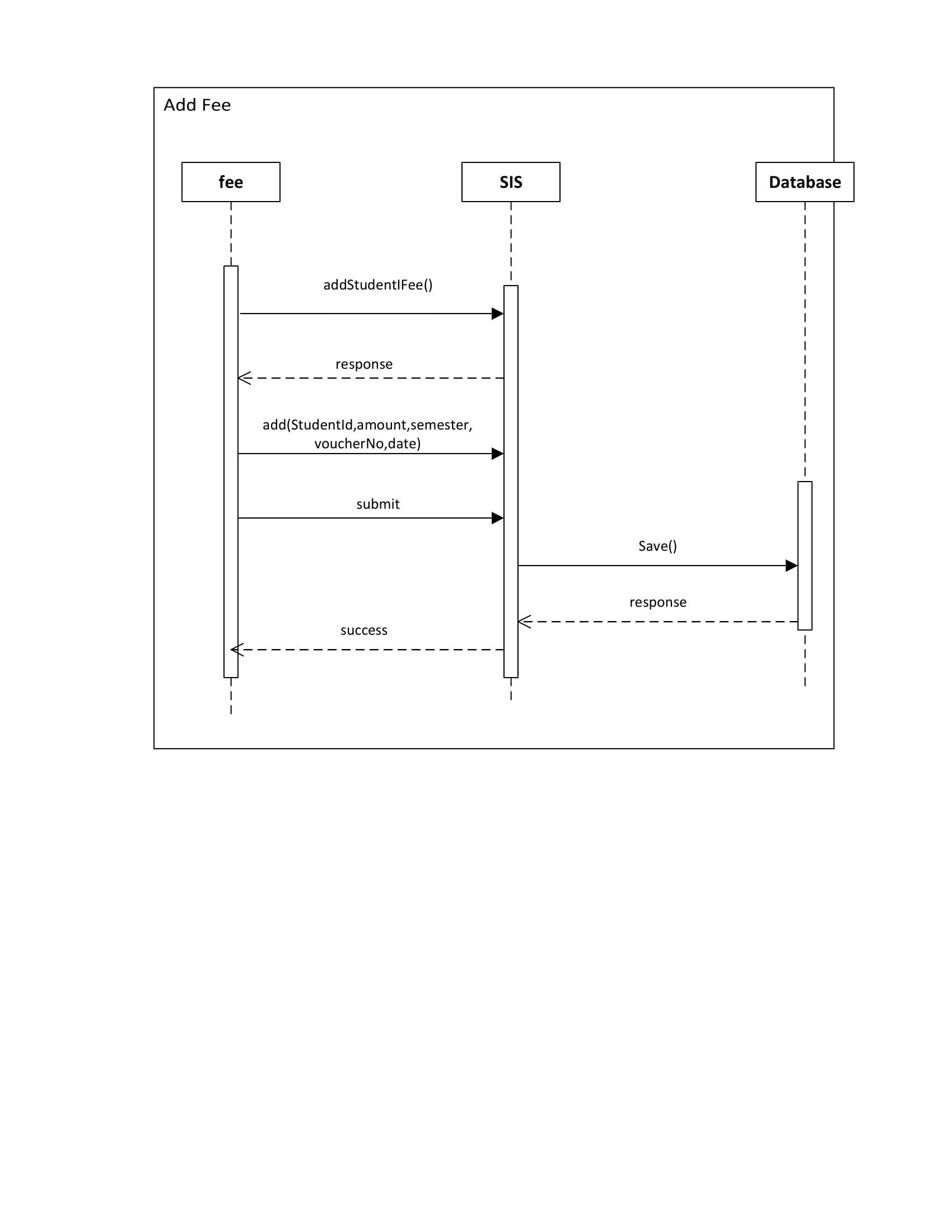
**Figure 4.2.8 Student Result Sequence Diagram**

### 4.2.9Add Student Attendance



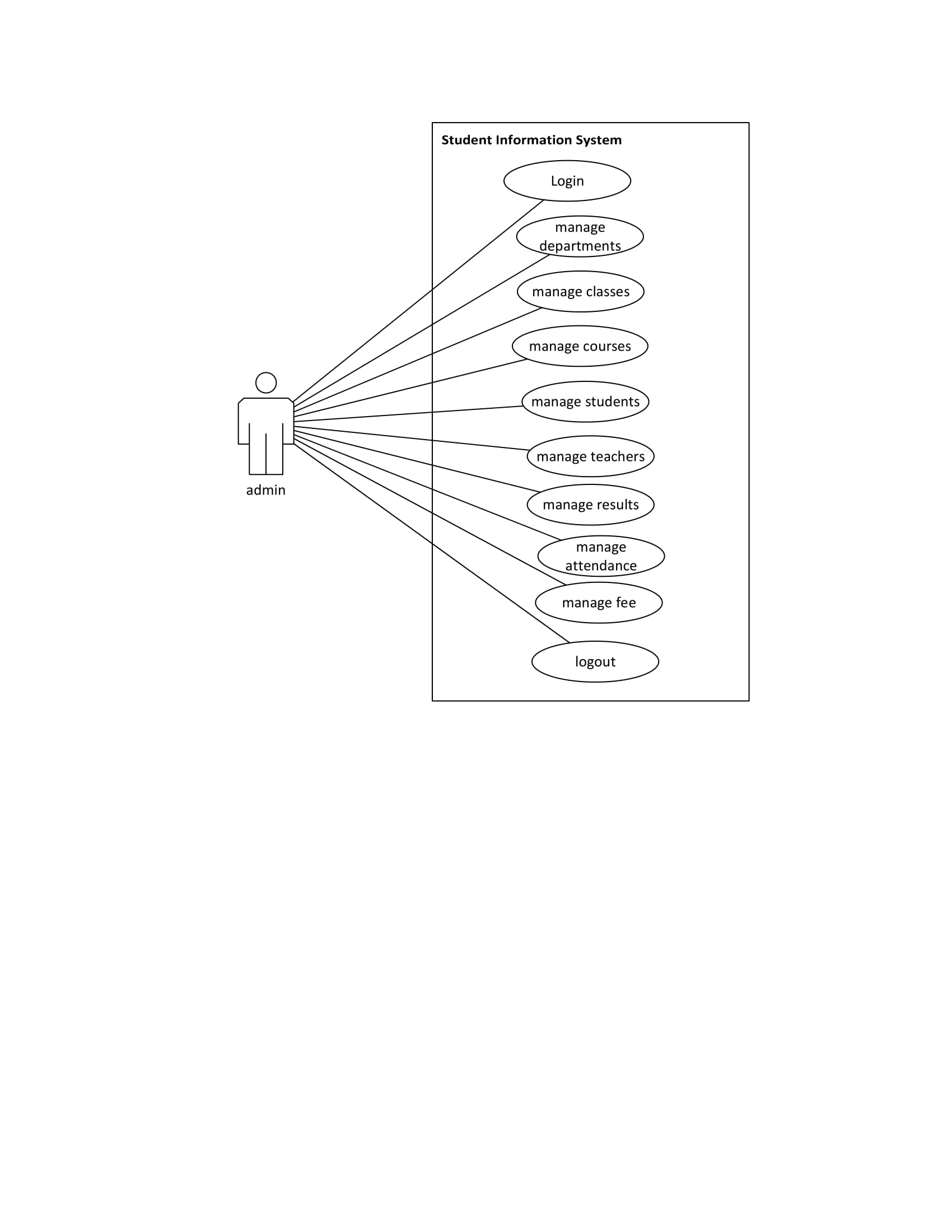
**Figure 4.2.9 Student Attendance Sequence Diagram**

### 4.2.10 Add Student Fee Details



**Figure 4.2.10 Student Fee Sequence Diagram**

## 4.3 Use Case Diagram



**Figure 4.3 Use case Diagram**

### 4.3.1 Use Case Description

|  |  |
| --- | --- |
| *Use Case UC1: Login User* | |
| Full Name | Login Admin |
| Level: | Administration |
| Primary Actor: | Administration |
| Secondary Actor | Administration |
| Description: | This use case describes how an actor logs into his/her SIS account. |
| Precondition: | Actor must have an active login account |
| Post condition: | Actor is log in and can use the system |
| Main Success scenario: | This use case starts when actor wishes to login his/her account.   1. The actor click smart login link. 2. The actor types his/her ID and Password. 3. The actor hits the login button 4. The system validates the actors password and logs him/her into the system |
| Extension: | There is no extension points associated with this use case. |
| Use Case Diagram 1:Login | |

|  |  |
| --- | --- |
| *Use Case UC2: Manage Department* | |
| Full Name | Manage Department |
| Level: | Admin |
| Primary Actor: | Admin |
| Secondary Actor | Admin |
| Description: | The Admin add, view, update and delete Department in SIS |
| Precondition: | 1. Admin must be log into system |
| Post condition: | Department id, name inserted into database |
| Main Success scenario: | This use case starts when admin wishes to add department in the SIS.   1. The admin click add department link. 2. The Admin type department name. 3. The Admin hits the save button 4. The system validates the department attributes, if the data meets the requirements it will be inserted into database. 5. The admin can also view, update and delete the departments. |
| Extension: | System will generate and error and provide some suggestion if the requirement will not fulfill. |
| Use Case Diagram 2:Manage\_Department | |
| *Use Case UC2: Manage Class* | |
| Full Name | Add Class |
| Level: | Admin |
| Primary Actor: | Admin |
| Secondary Actor | Admin |
| Description: | The Admin add Class in SIS |
| Precondition: | 1. Admin must be log into system |
| Post condition: | Class id, name, no of semesters inserted into database |
| Main Success scenario: | This use case starts when admin wishes to add class in the SIS.   1. The admin click add Class link. 2. The Admin type Class name, Alias and status. 3. The Admin hits the save button 4. The system validates the Class attributes, if the data meets the requirements it will be inserted into database. 5. The admin can also view, update and delete the classes. |
| Extension: | System will generate and error and provide some suggestion if the requirement will not fulfill. |
| Use Case Diagram 1:Manage\_Class | |

|  |  |
| --- | --- |
| *Use Case UC2: Add Course* | |
| Full Name | Add Course |
| Level: | Admin |
| Primary Actor: | Admin |
| Secondary Actor | Admin |
| Description: | The Admin add Course in SIS |
| Precondition: | 1. Admin must be log into system |
| Post condition: | Course id, code and cr. hours inserted into database |
| Main Success scenario: | This use case starts when admin wishes to add Course in the SIS.   1. The admin click add Course link. 2. The Admin type Course id, code and cr. hours. 3. The Admin hits the save button 4. The system validates the course attributes, if the data meets the requirements it will be inserted into database. |
| Extension: | System will generate and error and provide some suggestion if the requirement will not fulfill. |
| Use Case Diagram 1:Add\_Course | |

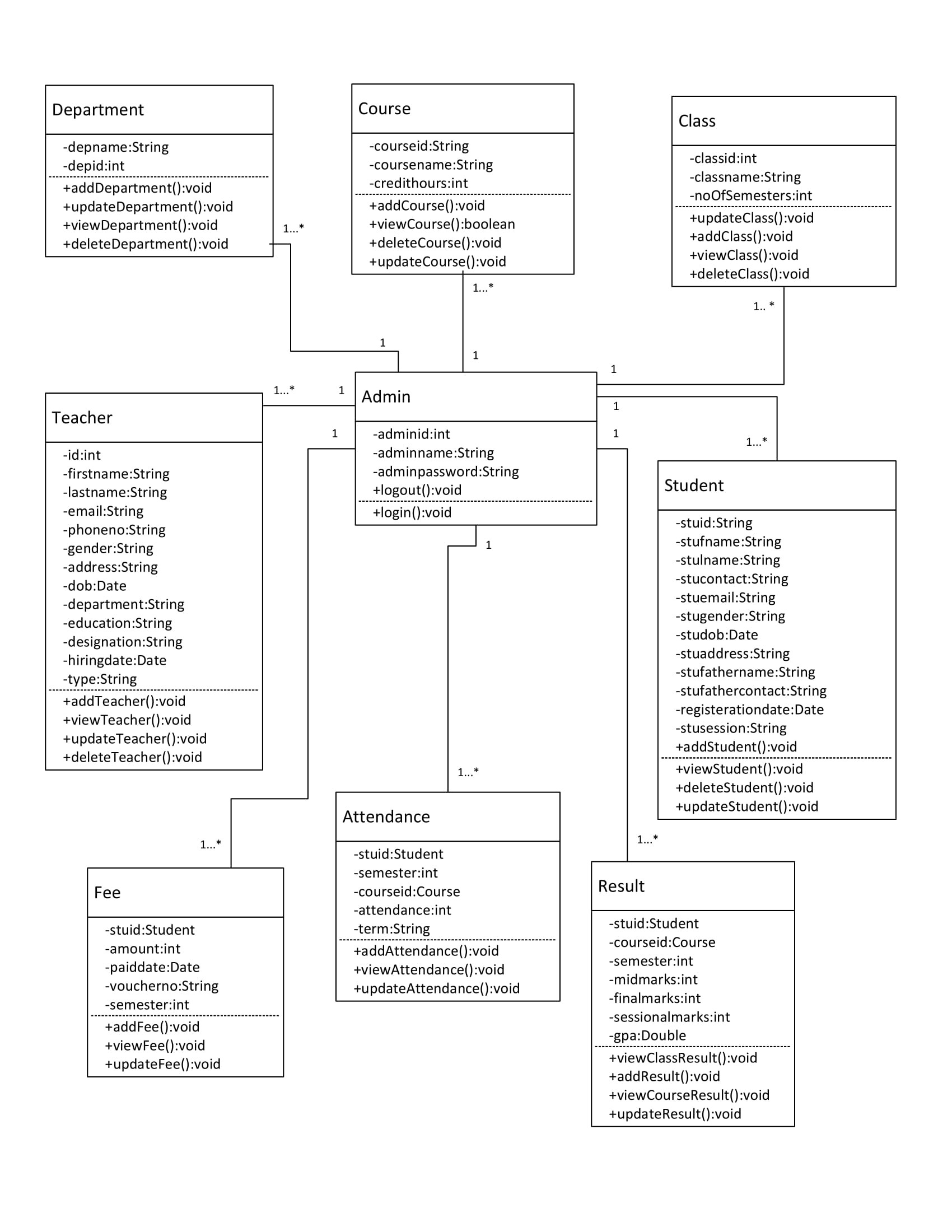
|  |  |
| --- | --- |
| *Use Case UC2: Add Student Information* | |
| Full Name | Add Student Information |
| Level: | Admin |
| Primary Actor: | Admin |
| Secondary Actor | Admin |
| Description: | The Admin add Student Information in SIS |
| Precondition: | 1. Admin must be log into system |
| Post condition: | Student information (name, father name, CNIC, cell no., email, Address, gender, date of birth, registration date, session, department, class) inserted into database. |
| Main Success scenario: | This use case starts when admin wishes to add Student Information in the SIS.   1. The admin click add Student Information link. 2. The Admin type Student Information Detail. 3. The Admin hits the save button 4. The system validates the Student Information attributes, if the data meets the requirements it will be inserted into database. |
| Extension: | System will generate and error and provide some suggestion if the requirement will not fulfill. |
| Use Case Diagram 1:Add\_Student | |

|  |  |
| --- | --- |
| *Use Case UC2: Add Student Result* | |
| Full Name | Add Student Result |
| Level: | Admin |
| Primary Actor: | Admin |
| Secondary Actor | Admin |
| Description: | The Admin add Student Result in SIS |
| Precondition: | 1. Admin must be log into system |
| Post condition: | Student result (student id, course id, semester, mid marks, final marks, sessional marks, gpa) inserted into database. |
| Main Success scenario: | This use case starts when admin wishes to add Student Result in the SIS.   1. The admin click add Student Result link. 2. The Admin type Student Result Detail. 3. The Admin hits the save button 4. The system validates the Student Result attributes, if the data meets the requirements it will be inserted into database. |
| Extension: | System will generate and error and provide some suggestion if the requirement will not fulfill. |
| Use Case Diagram 1:Add\_Result | |
| *Use Case UC2: Add Teacher Information* | |
| Full Name | Add Student Information |
| Level: | Admin |
| Primary Actor: | Admin |
| Secondary Actor | Admin |
| Description: | The Admin add Teacher Information in SIS |
| Precondition: | 1. Admin must be log into system |
| Post condition: | Teacher information (name, father name, CNIC, cell no., email, Address, gender, qualification, designation, type, hiring date) inserted into database. |
| Main Success scenario: | This use case starts when admin wishes to add Student Information in the SIS.   1. The admin click add Teacher Information link. 2. The Admin type teacher Information Detail. 3. The Admin hits the save button 4. The system validates the teacher Information attributes, if the data meets the requirements it will be inserted into database. |
| Extension: | System will generate and error and provide some suggestion if the requirement will not fulfill. |
| Use Case Diagram 1:Add\_Teacher | |

|  |  |
| --- | --- |
| *Use Case UC2: Add Student Attendance* | |
| Full Name | Add Student Attendance |
| Level: | Admin |
| Primary Actor: | Admin |
| Secondary Actor | Admin |
| Description: | The Admin add Student Attendance in SIS |
| Precondition: | 1. Admin must be log into system |
| Post condition: | Student information (student id, course id, course attendance, term, and semester) inserted into database. |
| Main Success scenario: | This use case starts when admin wishes to add Student Attendance Information in the SIS.   1. The admin click add Student Attendance link. 2. The Admin type Student Attendance Detail. 3. The Admin hits the save button 4. The system validates the Student Attendance Information attributes, if the data meets the requirements it will be inserted into database. |
| Extension: | System will generate and error and provide some suggestion if the requirement will not fulfill. |
| Use Case Diagram 1:Add\_Student\_Attendance | |
| **Use Case UC2: Add Student Fee** | |
| Full Name | Add Student Fee Information |
| Level: | Admin |
| Primary Actor: | Admin |
| Secondary Actor | Admin |
| Description: | The Admin add Student Fee Information in SIS |
| Precondition: | 1. Admin must be log into system |
| Post condition: | Student Fee information (student id, semester, amount, voucher no, and paid date) inserted into database. |
| Main Success scenario: | This use case starts when admin wishes to add Student Fee Information in the SIS.   1. The admin click add Student Fee Information link. 2. The Admin type Student Fee Detail. 3. The Admin hits the save button 4. The system validates the Student Fee Information attributes, if the data meets the requirements it will be inserted into database. |
| Extension: | System will generate and error and provide some suggestion if the requirement will not fulfill. |
| Use Case Diagram 1:Add\_Fee | |

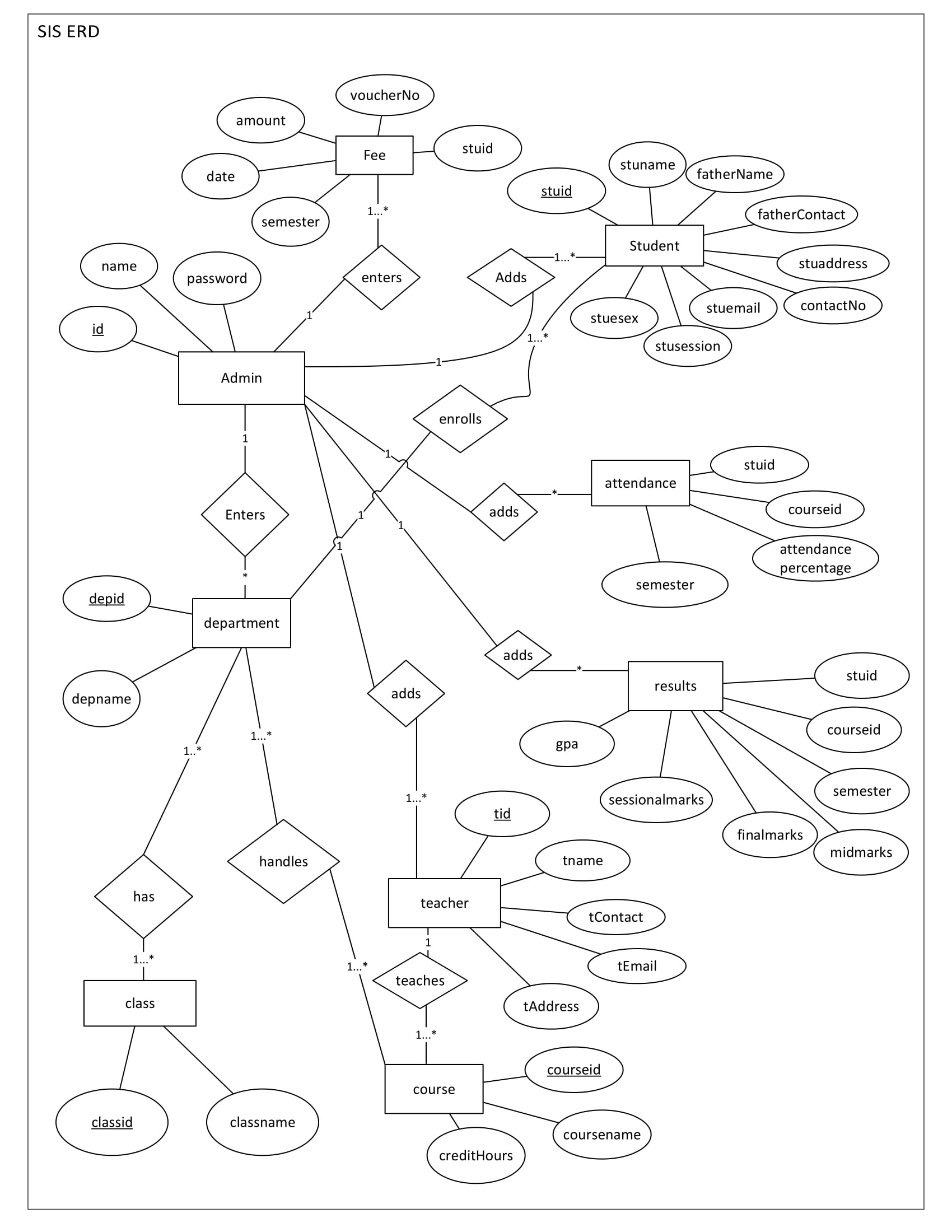
|  |  |
| --- | --- |
| *Use Case UC2: Logout* | |
| Full Name | Admin Logout |
| Level: | Admin |
| Primary Actor: | Admin |
| Secondary Actor | Admin |
| Description: | The Admin Logout. |
| Precondition: | 1. Admin must be log into system |
| Main Success scenario: | This use case starts when admin wishes to add Student Information in the SIS.   1. The admin click add Student Logout link. 2. The system logout the admin. |
| Extension: | System will generate and error and provide some suggestion if the requirement will not fulfill. |
| Use Case Diagram 1:LogOut | |

## Class Diagram



**Figure 4.4 Class Diagram**

## ERD Diagram



**Figure 4.5 Entity Relation Diagram**

# Chapter 5

# Implementation

## Introduction

In this chapter we will be discussing the fact that how Student Information Management System is implemented and which were the tools and technologies, that made the conceived idea a real artifact.

## Tools and Technology Used

### Tools

#### Eclipse

Eclipse is an [integrated development environment](https://en.wikipedia.org/wiki/Integrated_development_environment) (IDE) used in [computer programming](https://en.wikipedia.org/wiki/Computer_programming), and is the most widely used Java IDE. It contains a base [workspace](https://en.wikipedia.org/wiki/Workspace) and an extensible [plug-in](https://en.wikipedia.org/wiki/Plug-in_(computing)) system for customizing the environment. Eclipse is written mostly in [Java](https://en.wikipedia.org/wiki/Java_(programming_language)) and its primary use is for developing Java applications, but it may also be used to develop applications in other [programming languages](https://en.wikipedia.org/wiki/Programming_language) via plug-ins, including [Ada](https://en.wikipedia.org/wiki/Ada_(programming_language)), [ABAP](https://en.wikipedia.org/wiki/ABAP), [C](https://en.wikipedia.org/wiki/C_(programming_language)), [C++](https://en.wikipedia.org/wiki/C%2B%2B), [C#](https://en.wikipedia.org/wiki/C_Sharp_(programming_language)),[JavaScript](https://en.wikipedia.org/wiki/JavaScript),  [Perl](https://en.wikipedia.org/wiki/Perl), [PHP](https://en.wikipedia.org/wiki/PHP), [Prolog](https://en.wikipedia.org/wiki/Prolog), [Python](https://en.wikipedia.org/wiki/Python_(programming_language)), [R](https://en.wikipedia.org/wiki/R_(programming_language)), [Ruby](https://en.wikipedia.org/wiki/Ruby_(programming_language)), [Rust](https://en.wikipedia.org/wiki/Rust_(programming_language)), [Scala](https://en.wikipedia.org/wiki/Scala_(programming_language)), and [Scheme](https://en.wikipedia.org/wiki/Scheme_(programming_language)). Development environments include the Eclipse Java development tools (JDT) for Java and Scala, Eclipse CDT for C/C++, and Eclipse PDT for PHP, among others.

Eclipse [software development kit](https://en.wikipedia.org/wiki/Software_development_kit) (SDK) is [free and open-source software](https://en.wikipedia.org/wiki/Free_and_open-source_software), released under the terms of the [Eclipse Public License](https://en.wikipedia.org/wiki/Eclipse_Public_License), although it is incompatible with the [GNU General Public License](https://en.wikipedia.org/wiki/GNU_General_Public_License).

#### DBMS (SQL Server)

Microsoft SQL Server SQL Server is a relational database management system (RDBMS) from Microsoft that's designed for the enterprise environment. As a database server, it is a software product with the primary function of storing and retrieving data as requested by other software applications which may run either on the same computer or on another computer across a network (including the Internet).

### Technology

#### Language: Java

**Java** is a [general-purpose](https://en.wikipedia.org/wiki/General-purpose_language) [computer-programming language](https://en.wikipedia.org/wiki/Programming_language) that is [concurrent](https://en.wikipedia.org/wiki/Concurrent_computing), [class-based](https://en.wikipedia.org/wiki/Class-based_programming), [object-oriented](https://en.wikipedia.org/wiki/Object-oriented_programming), and specifically designed to have as few implementation [dependencies](https://en.wikipedia.org/wiki/Dependency_(computer_science)) as possible. It is intended to let [application developers](https://en.wikipedia.org/wiki/Application_developer) "[write once, run anywhere](https://en.wikipedia.org/wiki/Write_once,_run_anywhere)" (WORA), meaning that [compiled](https://en.wikipedia.org/wiki/Compiler) Java code can run on all platforms that support Java without the need for recompilation. Java applications are typically compiled to ["bytecode"](https://en.wikipedia.org/wiki/Java_bytecode) that can run on any [Java virtual machine](https://en.wikipedia.org/wiki/Java_virtual_machine) (JVM) regardless of the underlying [computer architecture](https://en.wikipedia.org/wiki/Computer_architecture). The language derives much of its original features from [SmallTalk](https://en.wikipedia.org/wiki/SmallTalk), with a [syntax](https://en.wikipedia.org/wiki/Syntax_(programming_languages)) similar to [C](https://en.wikipedia.org/wiki/C_(programming_language)) and [C++](https://en.wikipedia.org/wiki/C%2B%2B), but it has fewer [low-level](https://en.wikipedia.org/wiki/Low-level_programming_language) facilities than either of them. As of 2018, Java was according to [Github](https://en.wikipedia.org/wiki/Github) one of the most [popular programming languages in use](https://en.wikipedia.org/wiki/Measuring_programming_language_popularity), particularly for [client-server](https://en.wikipedia.org/wiki/Client%E2%80%93server) [web applications](https://en.wikipedia.org/wiki/Web_applications), with a reported 9 million developers.

Java was originally developed by a Canadian [James Gosling](https://en.wikipedia.org/wiki/James_Gosling) at [Sun Microsystems](https://en.wikipedia.org/wiki/Sun_Microsystems) ([which has since been acquired by Oracle](https://en.wikipedia.org/wiki/Sun_acquisition_by_Oracle)) and released in 1995 as a core component of Sun Microsystems' [Java platform](https://en.wikipedia.org/wiki/Java_(software_platform)).

## 5.2.3 Planning & Scheduling

### Gantt chart

A Gantt chart is a scheduling technique that is used commonly in project management. It is one of the useful and common ways of representation of activities displayed in contrast to time. On the vertical side of the chart activities list is displayed and on the top a time scale which is appropriate. A bar represents each activity’s length and also shows the start date, end date and duration of the activity.

This shows following information.

* “What the different activities are?”
* “When an activity starts and ends?”
* “How much long each activity is scheduled to finish?”
* “The start and finish date of the whole project.”

**Key Milestones of the Project with dates**

**Table 5‑1: Project Plan & Key Milestones**

|  |  |  |  |
| --- | --- | --- | --- |
| **Key Milestones of the Project with dates** | | | |
| **S. No** | **Elapsed time since start of the project** | **Milestone** | **Deliverable** |
| 1. | October 21,2018 - January 1,2019 | Analysis & Requirements | Complete Research Proposal |
| 2. | January 2,2019 - February 2 ,2019 | Planning, scheduling and purchase of hardware | Report |
| 3. | February 3,2019 – February 20,2019 | Modeling & design | Report |
| 4. | February 21,2019 - April 18,2019 | Coding & testing | Software System |
| 5. | April 20, 2019 | Deployment | Hardware & Software System |

# Chapter 6

# Database Design

### 6.1 Database Design

Database design is the organization of data according to a database model. The designer determines what data must be stored and how the data elements interrelate. With this information, they can begin to fit the data to the database model.

Database design involves classifying data and identifying interrelationships. This theoretical representation of the data is called ontology. The ontology is the theory behind the database's design.

### 6.2 Database

MYSQL is an application for storing information inside a “table” structure; let’s examine the reasons why you would use a Database rather than a spreadsheet or some other program for data storage. Imagine you’re creating an application for storing sales transactions. We’ll start by saving just a few columns of information such as the Student Id, name, address, phone, email, gender. One of the first storage options to consider is saving this information in a large text file. There are benefits to text file saves such as quick write times. The problem with text files is during a read, if the text file is large, it can take quite a bite of time to open and scan the contents of the file looking for what we want. Also, if we wanted to see all the record of a specific student, the entire text file would have to be read, and every line occurrence of the student name would need to be saved in some temporary place until we had them all. If we saved to a spreadsheet instead of a text file, we would have a Sort feature built in. So we may be able to find all the sales to a specific student quicker, but again, if the file was large, opening the spreadsheet could take a great deal of time.

**Why Not Use a Database?**

There are some problems with using a database. First, time must be taken to learn the new system. A database is not as intuitive as a spreadsheet. In addition, if there is only a small amount of data that doesn’t need to be changed over time, it’s probably simpler to save it in a file. Unfortunately, most business problems are neither simple nor small, so a database is usually the best tool for the job.

#### 6.2.1 Advantages of Database

* An organized and comprehensiveness of recording the result of the firms activities.
* A receiver of data to be used in meeting the information requirement of the MIS users.
* Reduced data redundancy.
* Reduced updating errors and increased consistency.
* Greater data integrity and independence from applications programs.
* Improved data access to users through use of host and query languages.
* Improved data security.
* Reduced data entry, storage, and retrieval costs.
* Facilitated development of new applications program.
* Standard can be enforced: Standardized stored data format is particularly desirable as an old data to interchange or migration (change) between the systems.
* Conflicting requirement can be handled.

#### 6.2.2 Constraints

SQL constraints are used to specify rules for the data in a table.

Constraints are used to limit the type of data that can go into a table. This ensures the accuracy and reliability of the data in the table. If there is any violation between the constraint and the data action, the action is aborted.

Constraints can be column level or table level. Column level constraints apply to a column, and table level constraints apply to the whole table.

The following constraints are commonly used in SQL:

* [NOT NULL](https://www.w3schools.com/sql/sql_notnull.asp) - Ensures that a column cannot have a NULL value
* [UNIQUE](https://www.w3schools.com/sql/sql_unique.asp) - Ensures that all values in a column are different
* [PRIMARY KEY](https://www.w3schools.com/sql/sql_primarykey.asp) - A combination of a NOT NULL and UNIQUE. Uniquely identifies each row in a table
* [FOREIGN KEY](https://www.w3schools.com/sql/sql_foreignkey.asp) - Uniquely identifies a row/record in another table
* [CHECK](https://www.w3schools.com/sql/sql_check.asp) - Ensures that all values in a column satisfies a specific condition
* [DEFAULT](https://www.w3schools.com/sql/sql_default.asp) - Sets a default value for a column when no value is specified
* [INDEX](https://www.w3schools.com/sql/sql_create_index.asp) - Used to create and retrieve data from the database very quickly

#### 6.2.3 Entity

An entity is any object in the system that we want to model and store information about. Entities are usually recognizable concepts, either concrete or abstract, such as person, places, things, or events which have relevance to the database. Entities are represented by rectangles (either with round or square corners):

Examples:

Customer, Staff, and Account.

#### 6.2.4 Attributes

An attribute is an item of information which is stored about an entity. For example, the entity 'Student' could have attributes such as student id, surname, forename, date of birth, telephone number, etc. An attribute can only appear in one entity, unless it is the key attribute in another entity. In a traditional filing system an attribute equates to a **field** in a record.

#### 6.2.5 Keys

A key is a data item that allows us to uniquely identify individual occurrences or an entity type. You can sort and quickly retrieve information from a database by choosing one or more fields (i.e. attributes) to act as *keys*. For instance, in a Student table you could use a combination of the last name and first name fields (or perhaps last name, first name and birth dates to ensure you identify each student uniquely) as a key field.

There are several types of key field:

1. Primary Key
2. Secondary Key
3. Foreign key
4. Composite key
5. **Primary Key**

A **primary key** consists of one or more attributes that distinguishes a specific record from any other. For each record in the table the primary key acts like a driver's license number or a national insurance number, only one number exists for each person.

For example, your student number

A primary key is **mandatory**. That is, each entity occurrence must have a value for its primary key.

1. **Secondary Key**

An entity may have one or more choices for the primary key. Collectively these are known as candidate keys. One is selected as the primary key. Those not selected are known as **secondary keys**.

For example, a student has student roll no, CNIC number and an email address. If the student roll no is chosen as the primary key then the CNIC number and email address are secondary keys. However, it is important to note that if any student does not have a CNIC number or email address (i.e. the attribute is not mandatory) then it cannot be chosen as a primary key.

1. **Foreign Key**

A **foreign key** is one or more attribute in one entity, which enables a link (or relationship) to another entity. That is, a foreign key in one entity links to a primary key in another entity. However, if the business rules permit, a foreign key may be optional.

For example, a student enrolls in a department. The department number column in the student entity is a foreign key, which links to the department entity.

1. **Composite Key**

A **composite key** consists of more than one attribute to uniquely identify an entity occurrence.

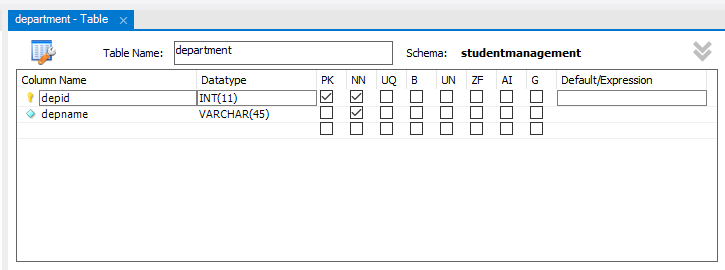
### 6.3 Database tables

#### 6.3.1 Admin Table

#### 

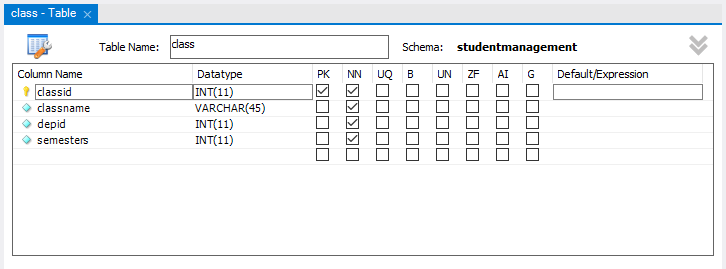
**Figure 6.3.1 Admin Database**

#### 6.3.2 Department Table



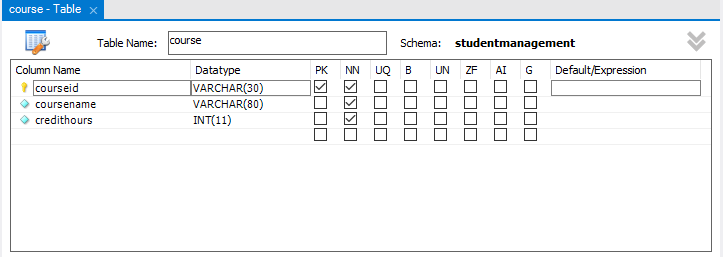
**Figure 6.3.2** **Department Database**

#### 6.3.3 Class Table



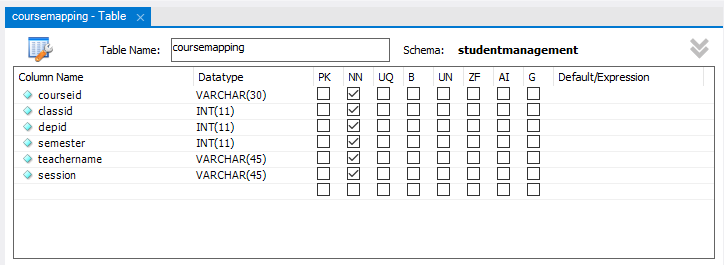
**Figure 6.3.3 Class Database**

#### 6.3.4 Course Table



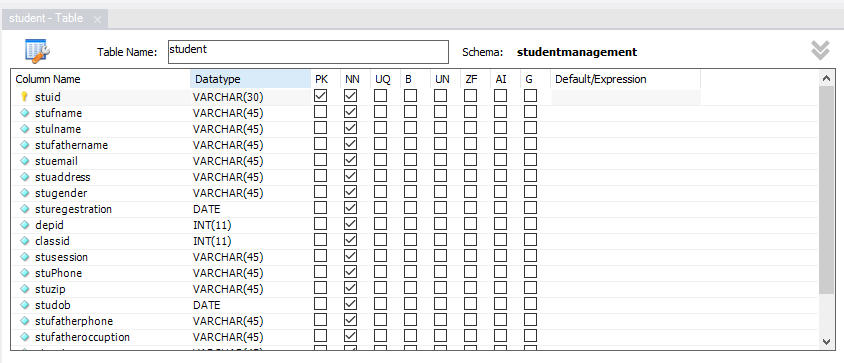
**Figure 6.3.4 Course Database**

#### 6.3.5 Course Mapping Table



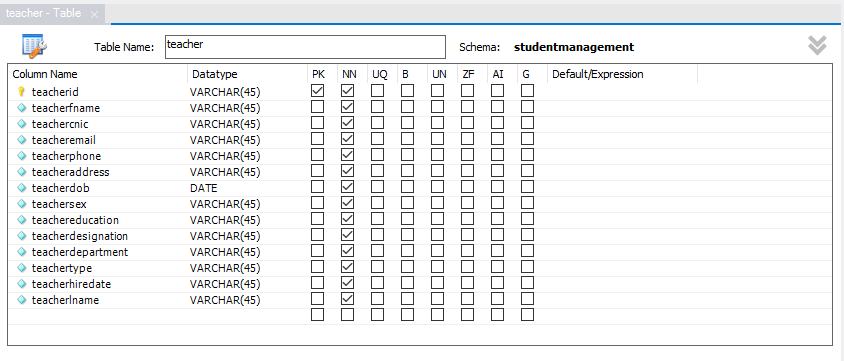
**Figure 6.3.5 Course Mapping Database**

#### 6.3.6 Student Table



**Figure 6.3.6 Student Database**

#### 6.3.7 Teacher Table



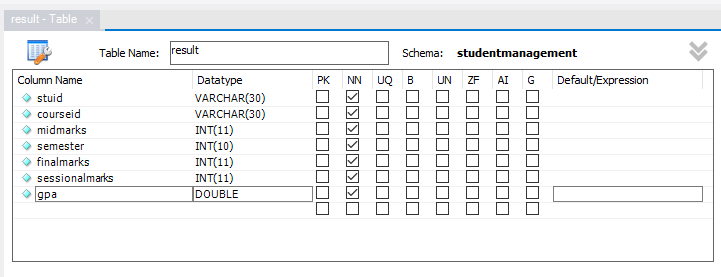
**Figure 6.3.7 Teacher Database**

#### 6.3.8 Student Attendance Table

#### 

**Figure 6.3.8 Student Attendance Database**

#### 6.3.9 Student Result Table



**Figure 6.3.9 Student Result Database**

#### 6.3.10 Student Fee Table

#### 

**Figure 6.3.10 Student Fee Database**

# Chapter 7

# User Interface

# 7.1 Introduction

A user interface, also called a "UI" or simply an "interface," is the means in which a person controls a software [application](https://techterms.com/definition/application) or hardware device. A good user interface provides a "user friendly" experience, allowing the user to interact with the software or hardware in a natural and intuitive way.

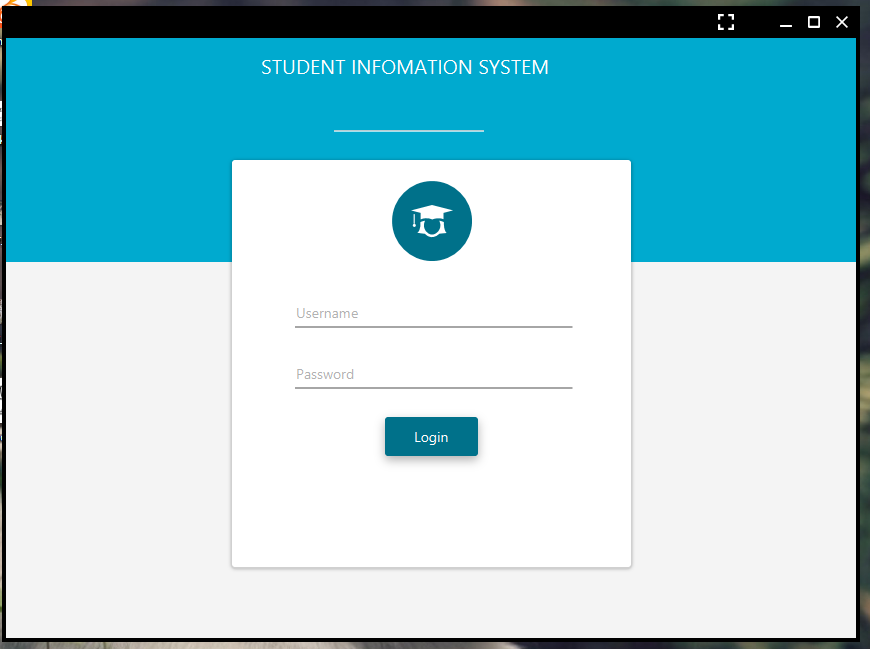
Nearly all software programs have a graphical user interface, or [GUI.](https://techterms.com/definition/gui) This means the program includes graphical controls, which the user can select using a mouse or keyboard. A typical GUI of a software program includes a [menu bar,](https://techterms.com/definition/menu_bar) [toolbar,](https://techterms.com/definition/toolbar) [windows,](https://techterms.com/definition/window) buttons, and other controls. The Macintosh and Windows operating systems have different user interfaces, but they share many of the same elements, such as a [desktop,](https://techterms.com/definition/desktop) windows, icons, etc. These common elements make it possible for people to use either operating system without having to completely relearn the interface. Similarly, programs like [word processors](https://techterms.com/definition/wordprocessor) and [Web browsers](https://techterms.com/definition/web_browser) all have rather similar interfaces, providing a consistent user experience across multiple programs.

Most [hardware](https://techterms.com/definition/hardware) devices also include a user interface, though it is typically not as complex as a [software](https://techterms.com/definition/software) interface. A common example of a hardware device with a user interface is a remote control. A typical TV remote has a numeric keypad, volume and channel buttons, mute and power buttons, an input selector, and other buttons that perform various functions. This set of buttons and the way they are laid out on the controller makes up the user interface. Other devices, such as digital cameras, audio mixing consoles, and stereo systems also have a user interface.

While user interfaces can be designed for either hardware of software, most are a combination of both. For example, to control a software program, you typically need to use a [keyboard](https://techterms.com/definition/keyboard) and [mouse,](https://techterms.com/definition/mouse) which each have their own user interface. Likewise, to control a digital camera, you may need to navigate through the on-screen menus, which is a software interface. Regardless of the application, the goal of a good user interface is to be user-friendly. After all, we all know how frustrating it can be to use a device that doesn't work the way we want it to.

### 7.2 User Interface Front End

## 7.2.1 Login

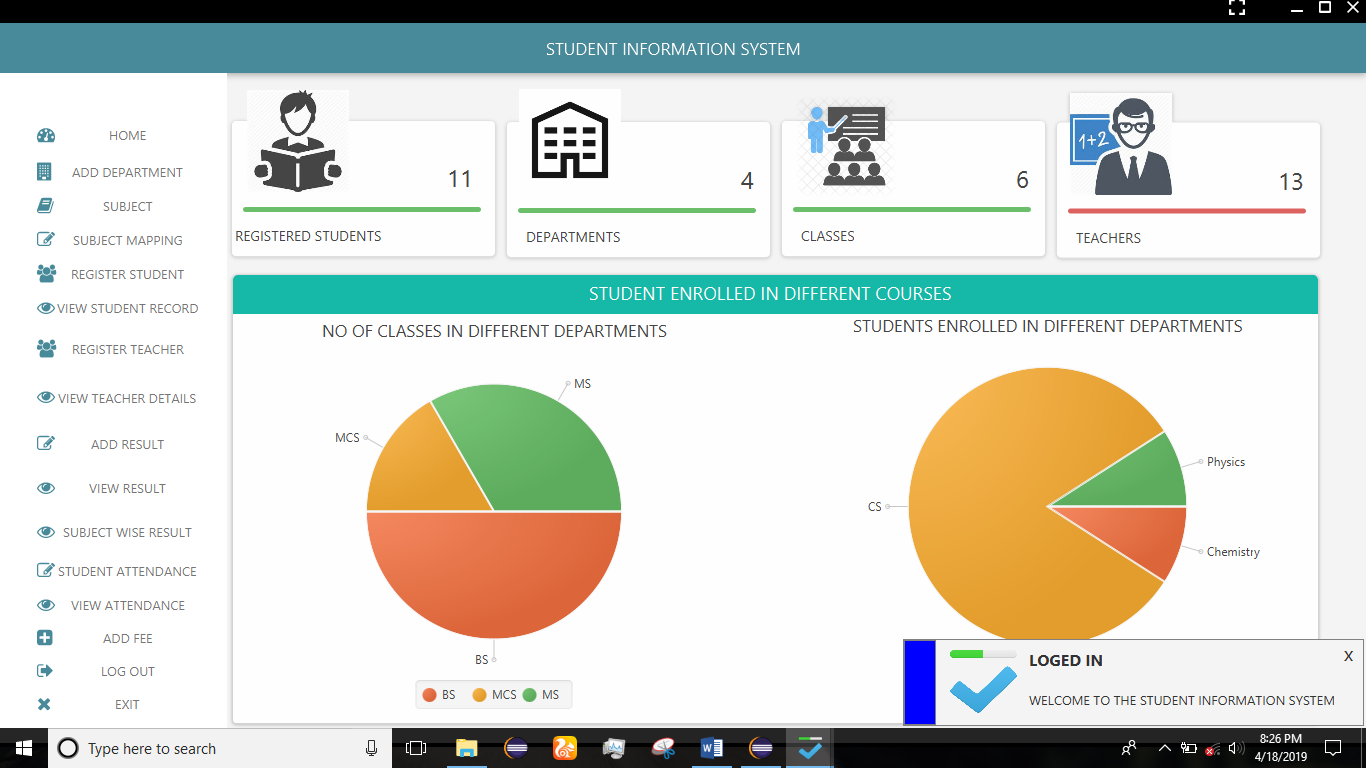


**Figure 7.2.1 Login**

**Description:**

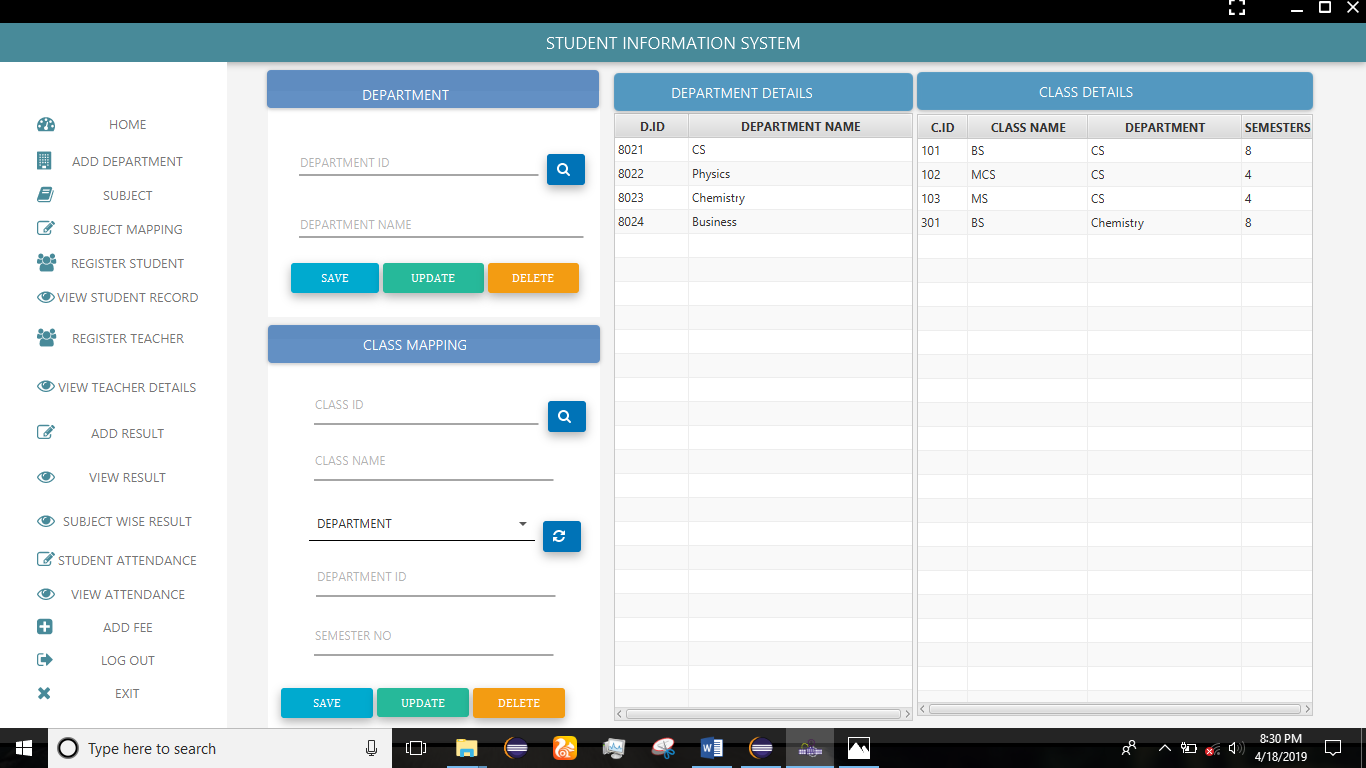
This is the Log in page of “Student Information System”. Here Admin can log by entering his name and password. If both match with the database value Admin will be logged in otherwise not.

## 7.2.2 Dashboard Page



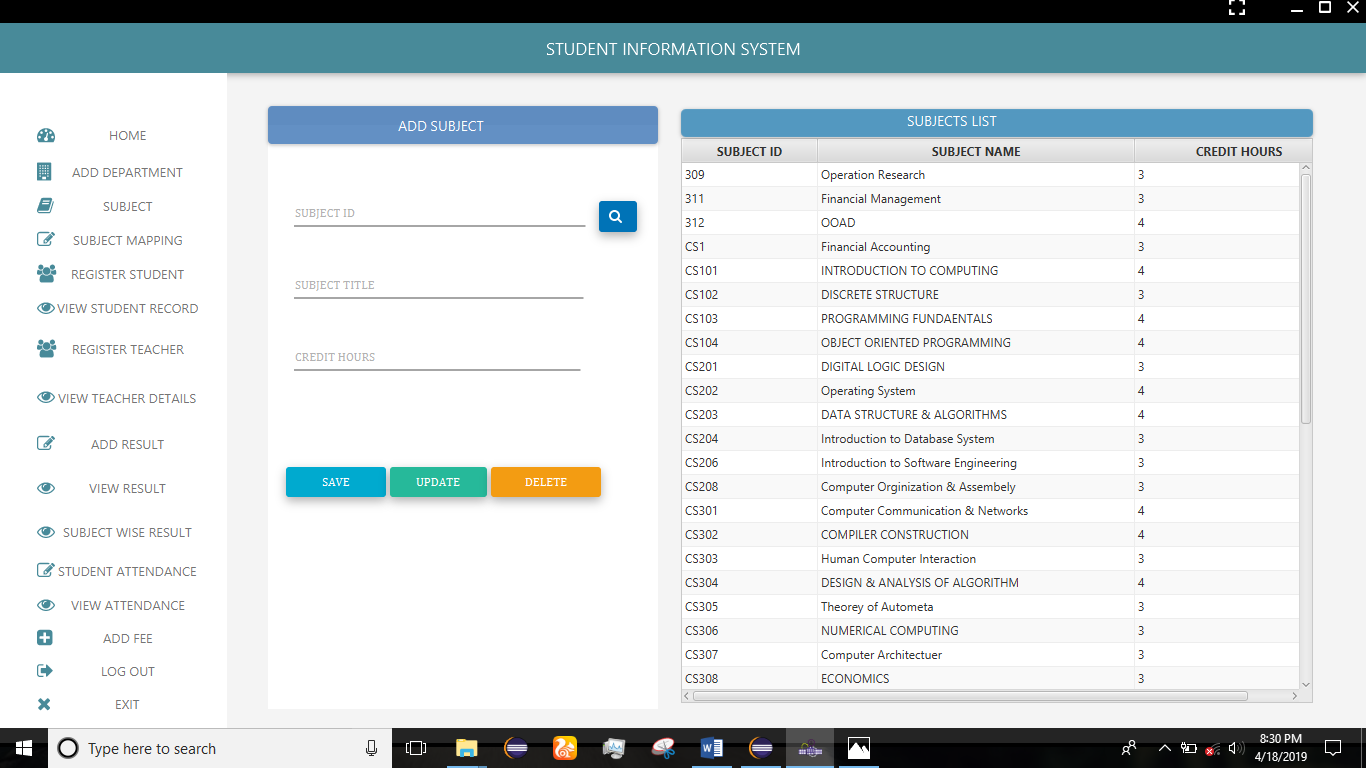
**Figure 7.2.2 Dashboard**

## 7.2.3 Department Page

****

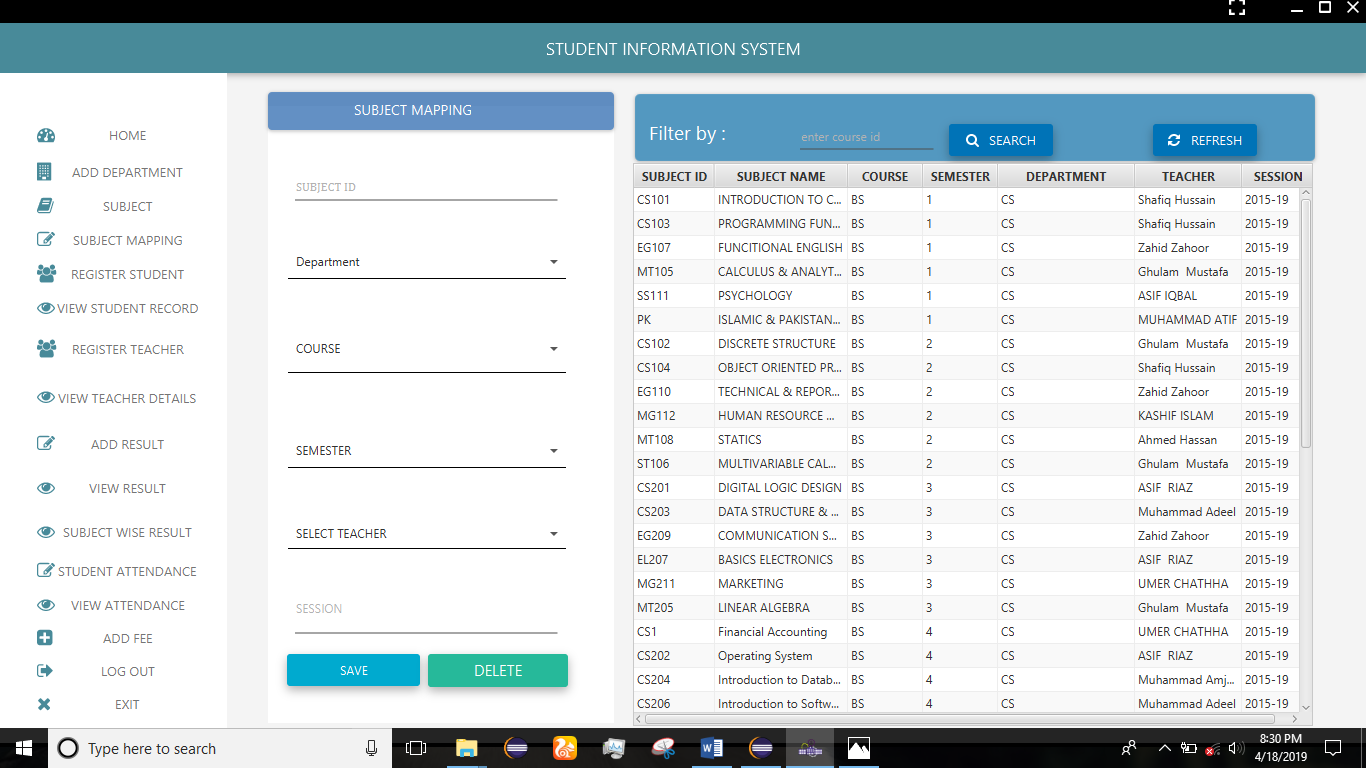
**Figure 7.2.3 Department Page**

## 7.2.4 Course Page



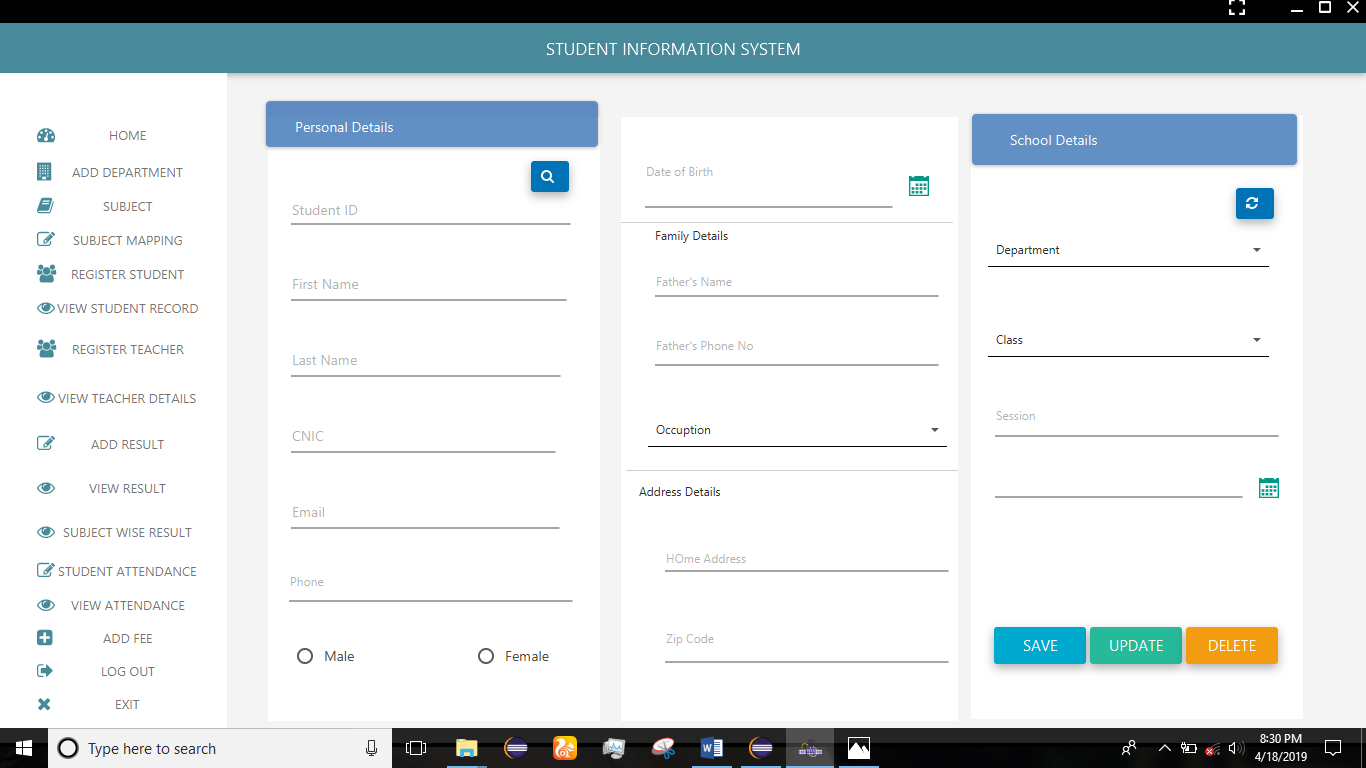
**Figure 7.2.4 Course Page**

## 7.2.5 Course Mapping Page



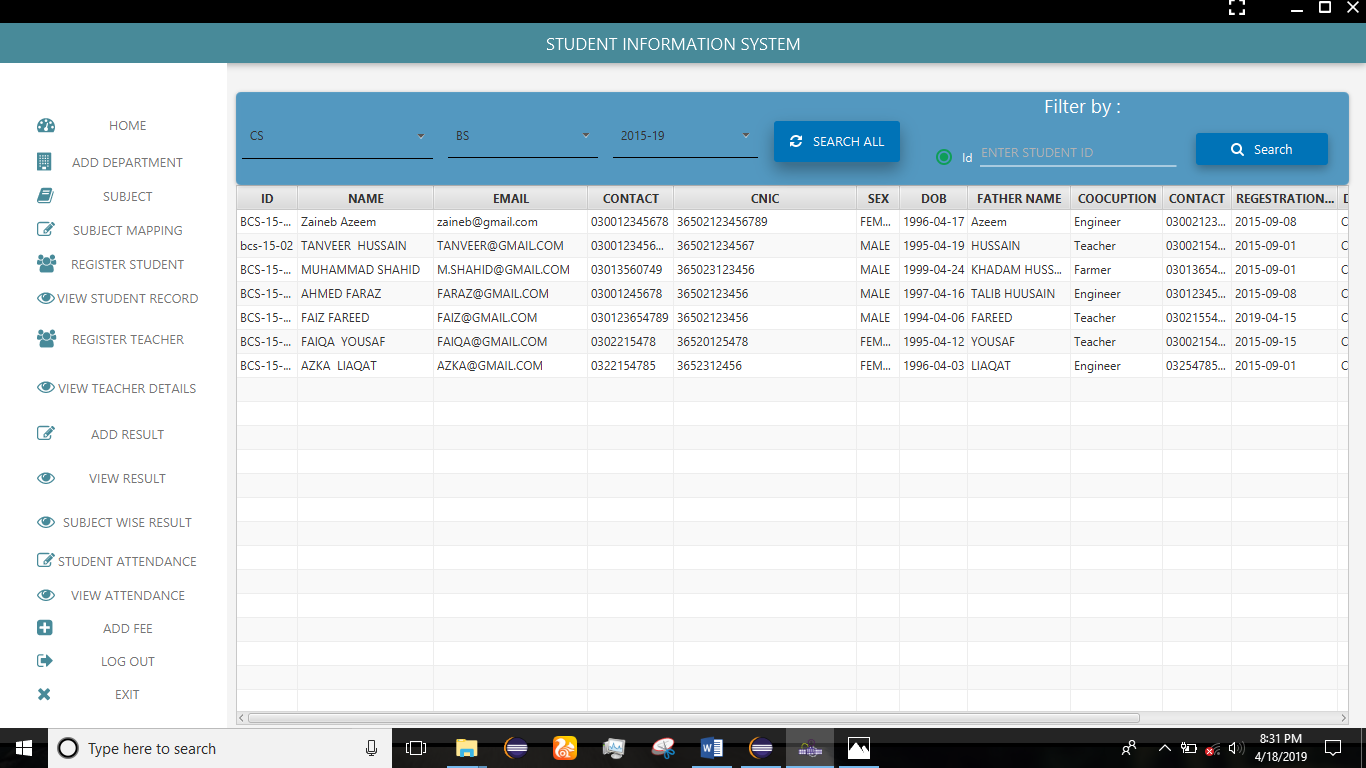
**Figure 7.2.5 Course Mapping Page**

# 7.2.6 Student Register Page

****

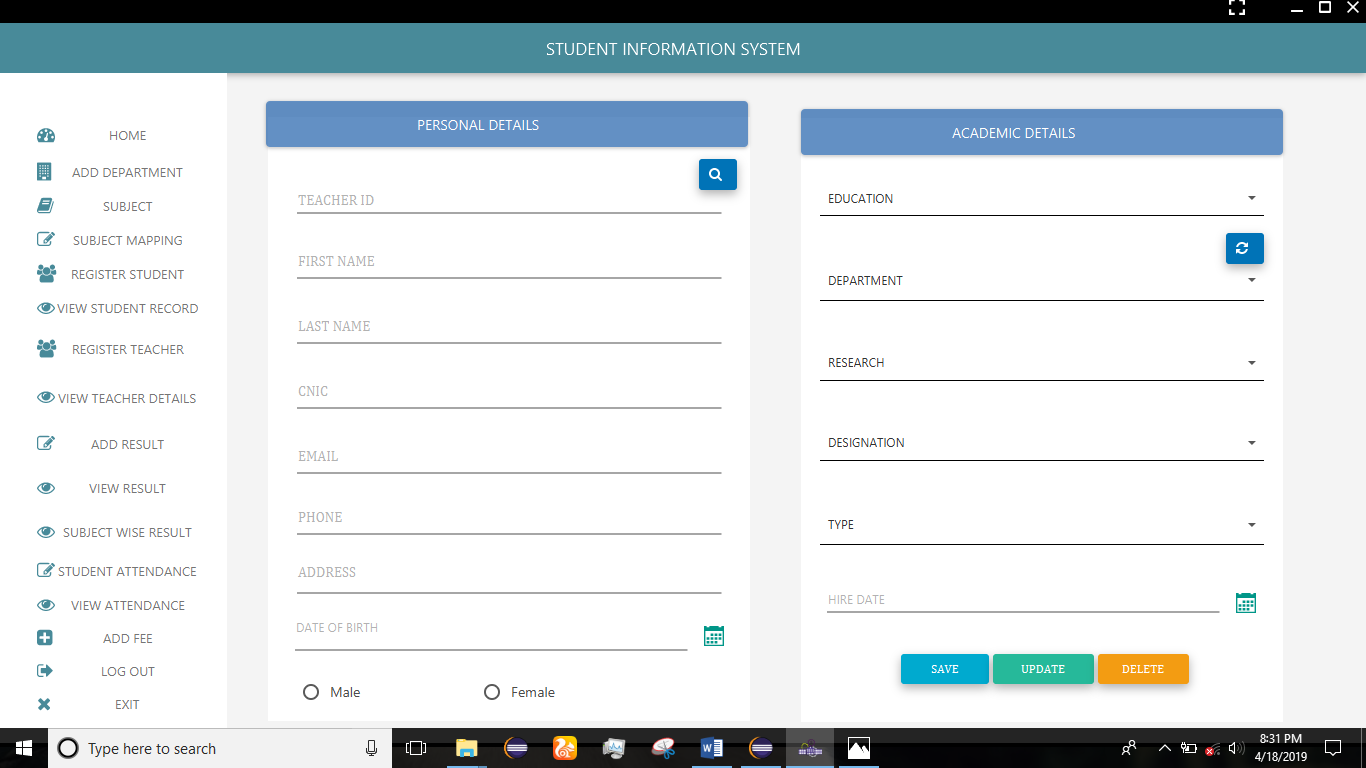
**Figure 7.2.6 Student Register Page**

# 7.2.7 Student View page

****

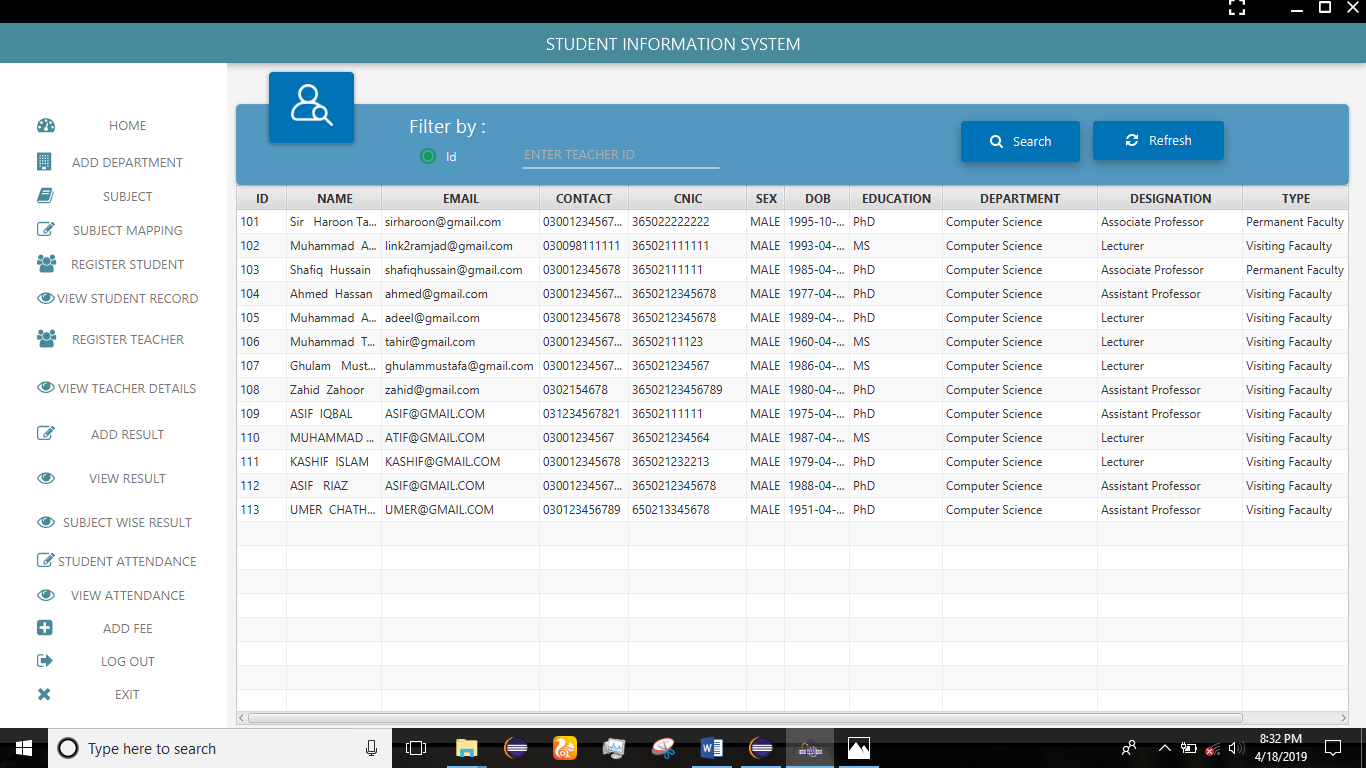
**Figure 7.2.7 Student View Page**

# 7.2.8 Teacher Register Page

****

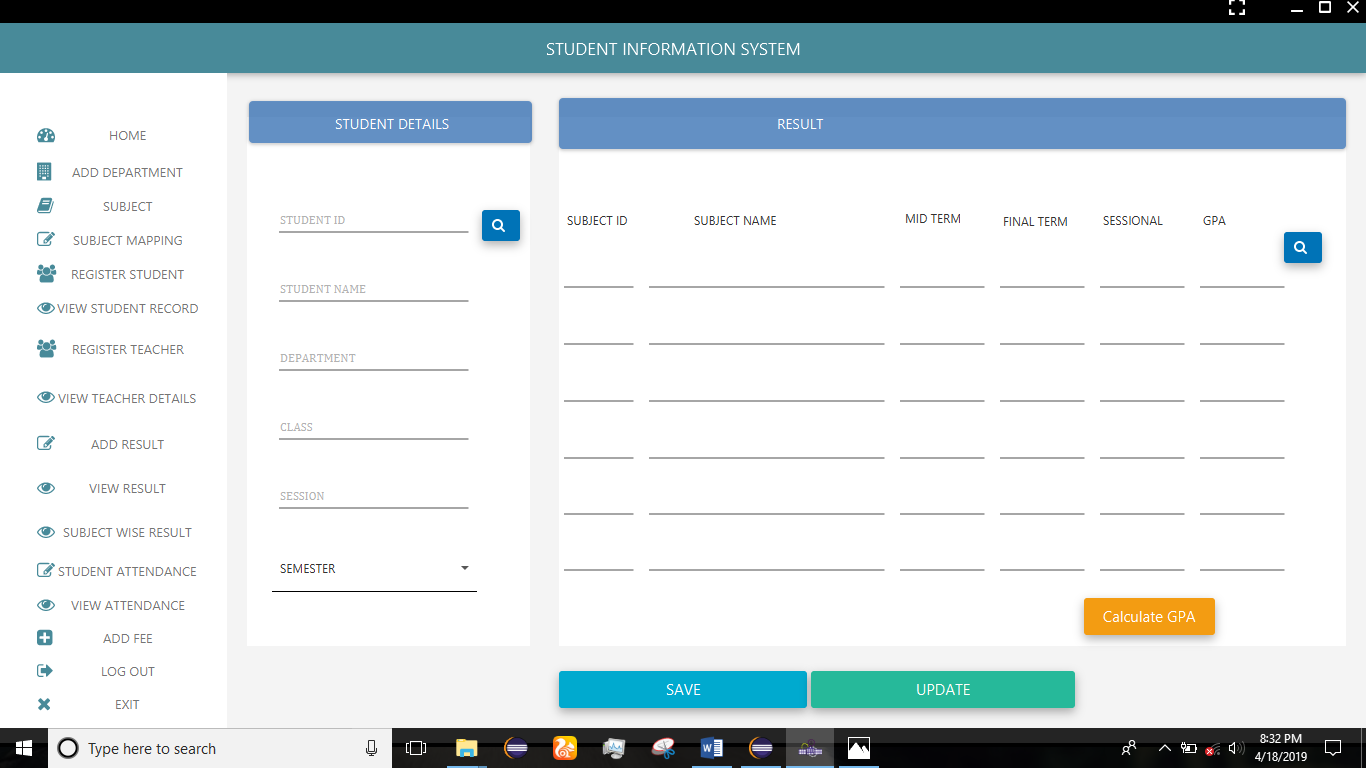
**Figure 7.2.8 Teacher Register Page**

# 7.2.9 Teacher View Page

****

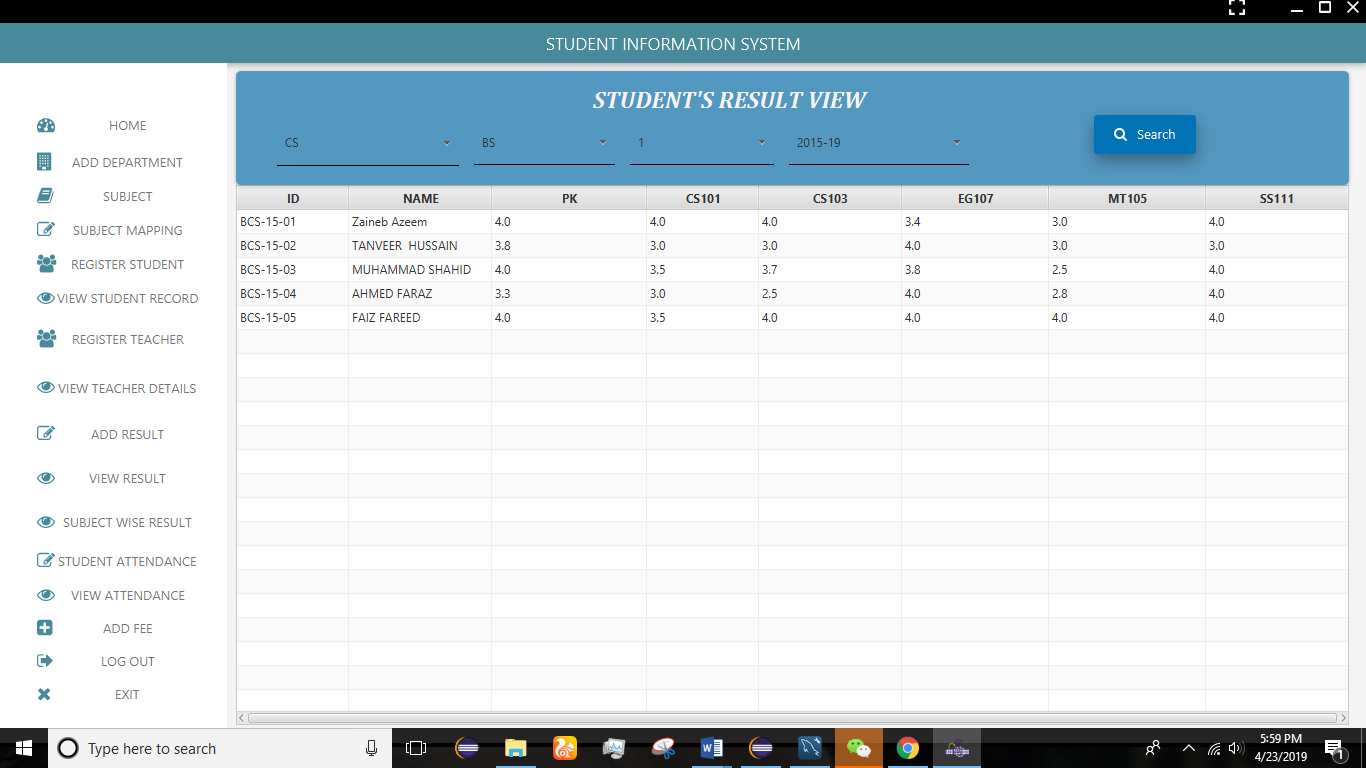
**Figure 7.2.9 Teacher View page**

# 7.2.10Add Result

****

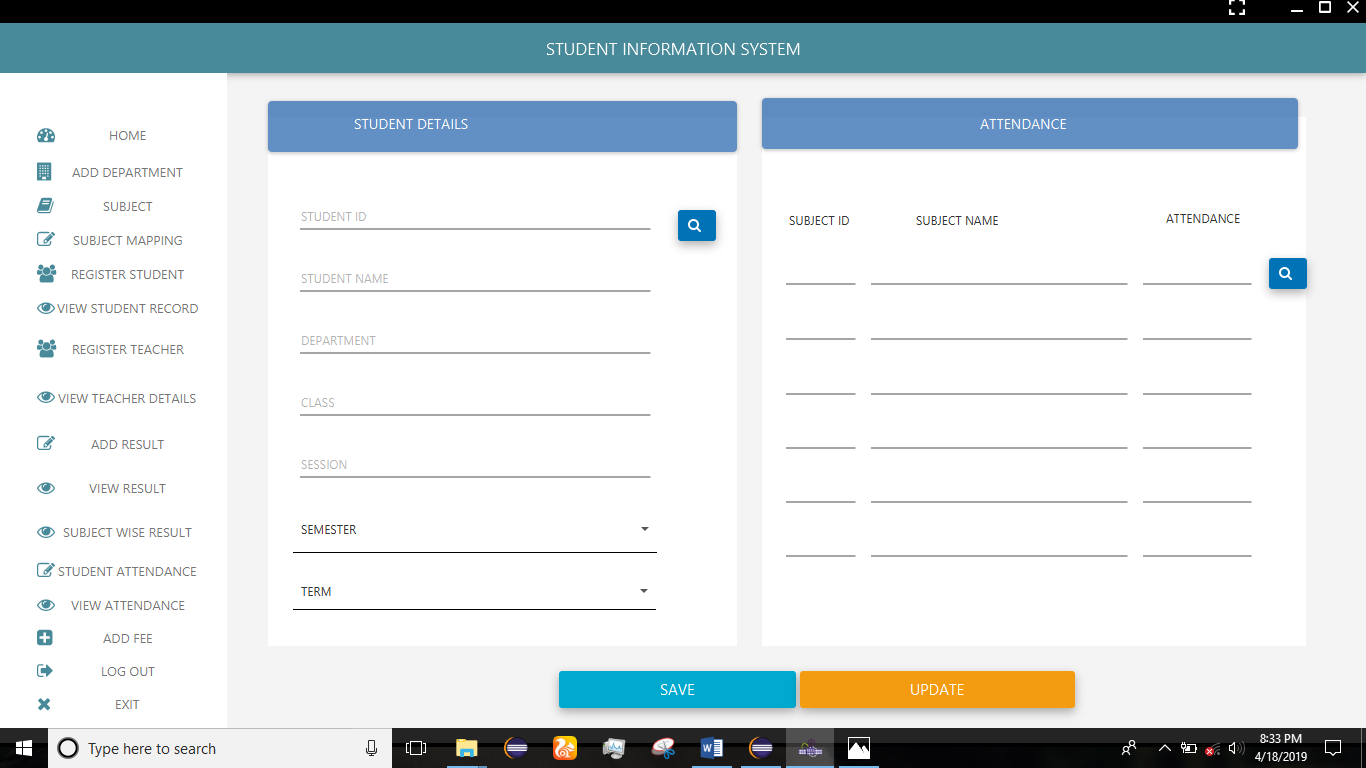
**Figure 7.2.10 Add Result Page**

# 7.2.11 View Result

****

**Figure 7.2.11 View Result**

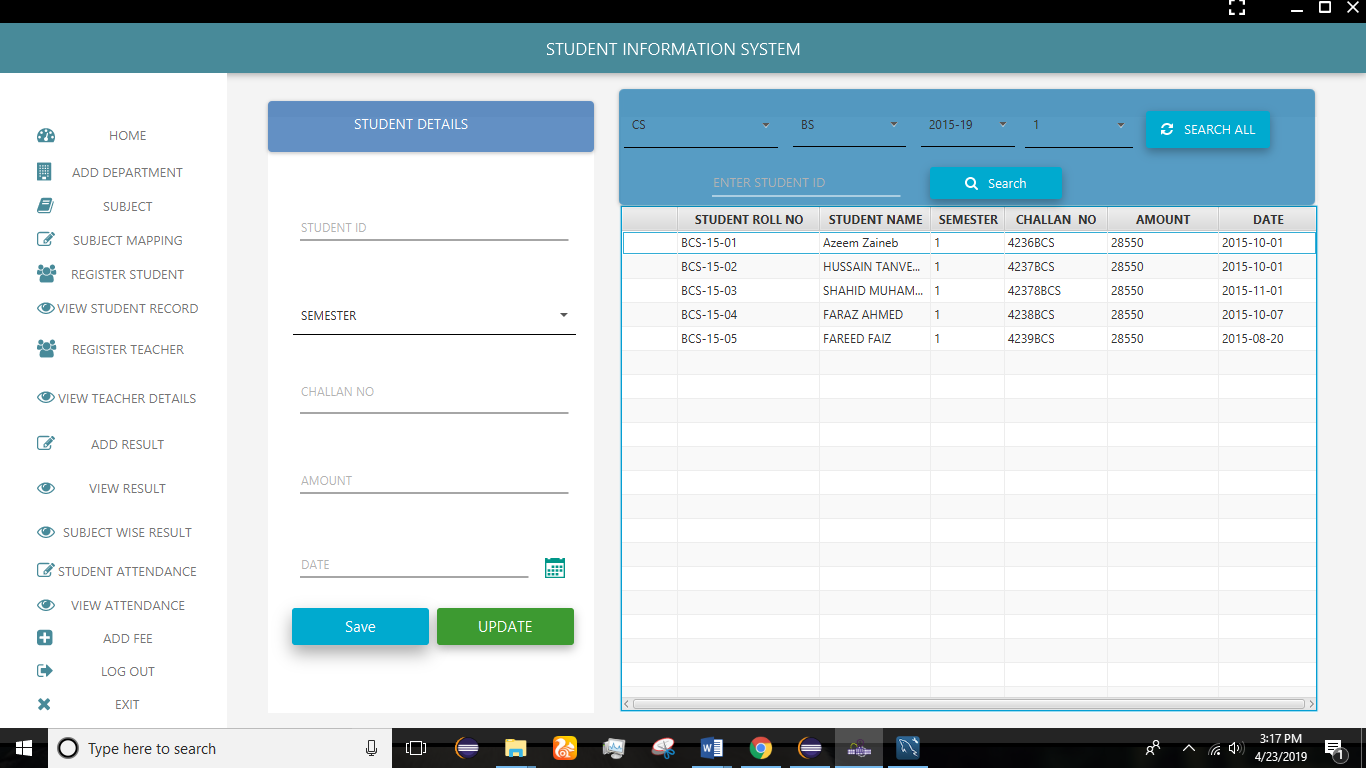
# 7.2.12 Add Attendance

****

**Figure 7.2.12 Attendance Page**

# 7.2.13 View Attendance

# 7.2.14 Fee Module

****

**Figure 7.2.14 Fee Page**

# Chapter 8

# SYSTEM TESTING

In this chapter, we will discuss the testing phase of developed application **“Student Information System”** in different manner to know that how much efficient and effective application is.

## 8.1 Introduction

A process of performing as application or program with the intention of finding errors and whether the application is fulfilling user needs. It can also be defined as the ability of a program in meeting the required or desired results.

In many methodologies of software engineering, a separate phase is called phase of testing which is performed after the completion of the implementation. There is a benefit in using this approach that it is hard to see one's own mistakes, and a fresh eye can find observable errors much faster than the person who has read the material many times.

## 8.2 Testing Plan

A process of performing as application or program with the intention of finding errors and whether the application is fulfilling user needs.

### 8.2.1 Unit Testing

The software units in an application are modules and routines that are assembled and integrated to perform a specific function. Unit testing focuses first on modules, independently of one another, to locate errors. This enables, to detect errors in coding and logic that are contained within each module. This testing includes entering data and ascertaining if the value matches to the type and size supported by Java. The various controls are tested to ensure that each performs its action as required.

Commonly used method is White-Box Testing method. Every time a component of the program is changed, it can be run for testing that is the biggest and famous benefit of this testing phase. Issues that are arise during this phase, allowing to be resolved as quickly as possible. Unit testing is familiar by software developers. It allows them to test their application units before move them to testers for formal testing.

### 8.2.2 System Testing

To test the complete application as a whole, system testing has been used. It is beneficial to check whether the application meets its requirements and fulfill Quality Standards.

### 8.2.3 Integration Testing

Integration testing allows the software developers to integrate all of the components/ units of the application within a program and then test them in a group. Basically, this testing level is used to catch the defects in the user interface between the functions/ modules. It is useful to determine how logically and efficiently all the units/ components are running together.

Here the streaming module and encoding module options are integrated and tested. This testing provides the assurance that the application is well integrated functional unit with smooth transition of data.

### 8.2.4 User Acceptance Testing

User acceptance of an application is the key factor for the success of any application. The application under consideration is tested for user acceptance by constantly keeping in touch with the application users at time of developing and making changes whenever required.

## 8.3 Test Cases

**Table 8‑1: Testing Cases**

|  |  |
| --- | --- |
| **Test Cases** | **Objectives** |
| 1 | To make sure that user can easily understand and can use the application |
| 2 | Make sure that user can easily login |
| 3 | Make sure that administrator can easily update, delete, save and view department. |
| 4 | Make sure that admin can save, view, update and delete class. |
| 5 | Make sure that admin can save, view, update and delete course. |
| 6 | Make sure that admin can easily assign view, and update courses to different classes. |
| 7 | Make sure that admin can save, view, update and delete Student. |
| 8 | Make sure that admin can save, view, update and delete Teacher. |
| 9 | Make sure that admin can save, view, and update student attendance. |
| 10 | Make sure that admin can save, view, and update student result. |
| 11 | Make sure that admin can save, view, and update student fee record. |
| 12 | Make sure that the application run at cross-platforms successfully. |

## 8.4 Testing Results

**Table 8‑2: Testing Result**

|  |  |  |  |
| --- | --- | --- | --- |
| **CRITERIA** | **Test Status** | | **REMARKS** |
| To make sure that user can easily understand and can use the application | Test successful | | None |
| Make sure that user can easily login | Test successful | | None |
| Make sure that administrator can easily update, delete, save and view department. | Test successful | None | |
| Make sure that admin can save, view, update and delete class. | Test successful | None | |
| Make sure that admin can save, view, update and delete course. | Test successful | None | |
| Make sure that admin can easily assign view, and update courses to different classes. | Test successful | None | |
| Make sure that admin can save, view, update and delete Student. | Test successful | None | |
| Make sure that admin can save, view, update and delete Teacher. | Test successful | None | |
| Make sure that admin can save, view, and update student attendance. | Test successful | None | |
| Make sure that admin can save, view, and update student result. | Test successful | None | |
| Make sure that admin can save, view, and update student fee record. | Test successful | None | |
| Make sure that the application run at cross-platforms successfully. | Test successful | None | |

# Chapter 9

# Conclusion & Future Work

In this chapter, we will discuss the results and discussions of this framework “**Student Information System”** with conclude remarks and will also discuss related future work of this application.

## 9.1 Conclusion

“Student Information System” is developed to provide assistance to clerical staff of the “University of Sahiwal”.This application provide ease to clerical staff in managing students’ records.

This application provide such environment that is more efficient and user friendly. It will improve the performance of clerical staff and decrease their efforts in managing the manual records.

## 9.2 Future Work

In next my first preference is to enhance this application by providing more new features that are as follows:

* Result cards of the student can be printed
* The timetable of every class can be printed
* The students result can be inserted after each term
* Teacher attendance and pay details can be added
* The performance of every student can be monitored and reports can be given to parents.

# REFERENCES:

1. UML Revision Task Force. OMG Unified Modelling Language Specification, Version 1.4 (final draft). February 2001
2. <https://docs.oracle.com/javafx/2/get_started/jfxpub-get_started.htm>
3. <https://www.javatpoint.com/javafx-tutorial>
4. <https://www.tutorialspoint.com/java/>
5. <https://www.w3schools.com/java/>
6. <https://www.w3schools.com/sql/default.asp>
7. <https://www.youtube.com/playlist?list=PLhs1urmduZ29LNYi_MaoU60JemQ6Aei6A>