TIMETABLE MAINTENANCE MANAGEMENT SYSTEM

NUR ATHIRAH MINHALINA BT MAHYIDDIN
4212002751

En. Azhar Bin Hamid
Faculty of Communication, Visual Art and Computing, University Selangor

A final year project submitted in partial fulfilment of the requirements for the award of the degree of Bachelor of Information Technology (Honor)

NOVEMBER 2023

ACKNOWLEDGE

Our God Almighty, for making all of this happen and serving as an inspiration for the future. Word cannot express my gratitude and my thanks to my lectures for his invaluable patience and feedback. I wish to express my sincere appreciation to my main project supervisor, Sir Azhar Bin for his encouragement, guidance, critics and advice. Furthermore, without his kind assistance, the project would not have been feasible.

I would like to thank the following people who helped me with this research project, my friends and my classmates for their support and brainstorming ideas for me. I would be remiss in not mentioning my family, especially my parents and all my family. Their belief in me has kept my spirits and motivation high during this process.

Lastly, I am also grateful to all the staff and some faculty in UNISEL helping me and sharing some data and information. Thanks for trusting me to complete this research or project without them. I think this project will never be finished successfully.

ABSTRACT

This research effort attempts to make managing timetables at educational institutions easier. Manual administration might be difficult due to unpredictability. The proposal suggests using an automated approach to address this. It requires to make operations more effective and less disrupting by providing real-time data and improved scheduling. The system architecture blueprint illustrates how various components relate together. UML is used to create and display components such as use cases, sequences, classes, and activities. In addition, the system uses Entity-Relationship Diagrams (ERDs) to organize data and produce a user-friendly interface. Thanks are extended to friends, classmates, and the UNISEL team for their assistance. Their assistance was critical in completing this assignment successfully.

ABSTRAK

Projek penyelidikan ini bertujuan untuk menjadikan pengurusan jadual waktu di institusi pendidikan lebih mudah. Pengurusan manual boleh menjadi sukar disebabkan perubahan yang tidak dijangka. Projek ini mencadangkan sistem automatik untuk menangani ini. Dengan memberikan data secara waktu sebenar dan penjadualan yang lebih baik, ia bertujuan untuk menjadikan operasi lebih lancar dan mengurangkan gangguan. Blueprints senibina sistem menunjukkan bagaimana bahagian yang berbeza berfungsi bersama. Ia menggunakan Bahasa Model Bersatu (UML) untuk mereka bentuk dan menunjukkan komponen seperti kes penggunaan, urutan, kelas, dan aktiviti. Sistem juga mengatur data menggunakan Carta Hubungan-Entiti (ERDs) dan membuat antara muka yang mesra pengguna. Terima kasih diberikan kepada keluarga, rakan-rakan, rakan sekelas, dan kakitangan UNISEL atas bantuan mereka. Sokongan mereka adalah penting dalam menyiapkan projek ini dengan berjaya.

Table of Content

CHAPTER 1

INTRODUCTION

1.0 Introduction

In real-life scheduling circumstances, things might change after a timetable is established, such as resources being unavailable. When it happens, the timetable must be changed without introducing too many significant modifications. A good schedule is considered strong if it is capable of handling these changes without sacrificing quality or accuracy. In universities, they first create a timetable based on what the academics and administrators require. Then they notify everyone, and anyone is able to ask for changes if necessary due to changes or issues. They adjust the timeline to reflect these modifications while attempting to remain as close to the original as possible. Then they notify everyone about the updated timetable, and students enroll for classes based on it and changes may after students sign up may interrupt it several variables might change once students enroll on to classes. New classes may be given, and existing ones may be discontinued or canceled. Also, new lecturers may show up.

In universities, they first create a timetable based on what the coordinator, Head of Department (HOD) and administrators required. Then they notify everyone, and anyone may make changes if needed due to changes or shortcomings. They adjust the timeline according to the changes while attempting maintain as accurate to the initial as possible. Then they inform everyone about the updated timetable, and students enroll for classes based on it.

In this project will discusses plenty of interruptions that might impact this process. several kinds of issues might change before students enroll for classes. In this study, we are thinking about many things that might disrupt timetables. For example, new classes may be implemented, while others may be removed. Also, some teachers can leave or establish the university. Sometimes a disturbance just makes it difficult for a lecturer to lecture at specific times. Alternatively, there may not be enough space for all of the students due to the increasing data.

A Timetable Maintenance Management System Project entails developing software that allows individuals to manage timetables, assign tasks, and observe how things are progressing as they occur. It usually allows you to create tasks, keep track of time, plan activities, and generate reports. One of the most beneficial aspects of this type of system is that it may improve people's productivity and time management skills.

1.1 Background of Study

A lot of universities must manually assign time slots to each course while according to a specific set of requirements. If they find any difficulties, they must solve them by attempting several solutions until one works. It takes a long time. Using software can make the process easier and faster. Software can help with timetable in several of ways. It also makes it easy to correct issues and modify schedules later on, making it easy to use and keep things operating smoothly. The class timetable organizes staff, facilities, and time for classes. The management of these resources has an impact on the courses that students can take. The number of professors available for each course is limited, as is the size of the available rooms. To avoid issues, thoroughly evaluate the syllabus and course schedule. Making the most of the period you have is important (JF Blakekesly, 1998).

Many studies show how helpful timetabling are. These studies evaluated several ways for dealing with timetable issues, which included operations research to university timetable. They focused on strategies that employ graph coloring to solve test timetabling difficulties.

Institutions should carefully consider why they are making these changes. When designing campus layouts, consider the duration it takes students to move between buildings. While specialized sessions can be spread out, scheduling core courses in different rooms is more difficult (JF Blakekesly, 1998).

1.2 Problem Statement

To address these issues, this study will develop a system. This system will improve processes, improve access to current information, and better communication of changes, enabling better timetable administration for educational institutions.

1. All changes, for example assigning lecturers, courses, and classrooms, are made by hand.
2. It is difficult to find updated timetable information.
3. Information changes happens unexpectedly without any notice.

1.3 Objectives

Manually monitoring timetable at educational institutions may be a requiring process, which frequently results in issues such as mistakes, inefficiency, and difficulty getting current information. This might cause interruptions and inconvenience to both staff members and students. To overcome these challenges, this work seeks to create an automated timetable management system with the following goals:

1. Develop a timetable maintenance management system for such as assigning lecturers, courses, and classrooms, minimizing the need for manual changes.
2. Make it easy for everyone to get the latest updated timetable information by integrating real-time changes into the system.
3. To generate the report for changes in timetable.

1.4 Scope of Study

manually organizing schedules might be difficult. This involves activities such as assigning lecturers, courses, and classrooms. However, the procedure gets significantly more challenging when changes happen unexpectedly and without previous notice. A lack of up-to-date data might cause confusion and disrupt the regular operation of the institution. As a result, there is an urgent need for a system that is automated capable of managing timetables effectively and providing users with real-time changes, resulting with better organization and no disruptions.

1.4.1 User Scope

User Scope	
Administrator	1. Provides real-time updates for thorough timetable. 2. Improves the general structure and accuracy of administrative activities, causing bigger use of resources.
Student	1. Accessing Timetable Information: Students should review their class schedules, which include course times, locations, and any changes.
Lecturer	1. Lecturers are able to focus on providing great education without being distracted by unexpected changes or issues in their timetable.

Table 1.4.1.1 User Scope

1.4.2 System Scope

System Scope	
User Authentication	Implement user authentication to make sure that only registered users, such as administrators, lecturers, and students, are allowed entry to the system.
Timetable Disruption management	Admin: • Administrators may assign courses to lecturers, choose which classrooms to utilize, and schedule class times. Lecturer: • Lecturers can review their teaching schedules, make modifications such as assigning courses or classrooms, and request timetable. Student: • Students may view their class schedules, including course descriptions, timings, and locations.
Real-time updates	1. Ensure that users receive timely notices and updates about any changes. 2. Ensure that users receive timely messages for timetable changes and room changes, allowing them to keep informed and prepare ahead.

1.5 Hardware & Software Requirement

Hardware	Purpose
Laptop • AMD Ryzen 5 4500U with Radeon Graphics 2.38 GHz • 8GB (GigaByte) • 64-bit operating system, x64-based processor	• use to make all proposals and report • use to develop the scheduling system

Table 1.5.1 Hardware Requirement

Software	Purpose
Microsoft Words	• use for writing the proposal
XAMPP	• as a localhost
Sublime (HTML, CSS)	• to code a system
PHP MyAdmin	• Manage databases for your web projects

Table 1.5.2 Software Requirement

1.6 Significance of Project

Managing timetables at educational institutions may be difficult, especially when unexpected changes happen after the plans have been formed. These changes, such as resource unavailability, necessitate immediate modifications to minimize interruptions. A successful timeline should be able to respond to these changes while maintaining quality and accuracy. Typically, colleges develop timelines based on academic and administrative needs, allowing for changes as needed. However, interruptions may still occur after adoption, demanding additional changes. This research will investigate these disturbances and emphasize the significance of creating appropriate management systems to overcome them.

This study has the potential to significantly assist educational organizations. Identifying and managing interruptions in timetable management allows institutions to enhance scheduling processes, distribute resources better, and improve overall operations. Implementing good management systems may result in more effective operations, less disruptions and better conditions for learning for students and lecturer. Students can also benefit from more planned and consistent timetable, which can help them study more effectively. Furthermore, having easy access to up-to-date information enables students to better manage their academic activities, lowering stress and improving overall learning outcomes.

Improving timetable management systems help both faculty members and lecturers. Faculty members can improve their lifestyles and fulfillment with their work by gaining more control over their schedules. Students can also benefit from more planned and consistent timetable, which can help them study more effectively.

Building on these outcomes, future academics could investigate new research directions, create creative solutions, and make improvements to current efforts to enhance timetable management systems. This study lays an outline for future development and creativity, which will benefit learning. institutions and stakeholders throughout the world.

1.7 Constraint & Critical Assumption

1.7.1 Constraints

There are two types of constraints: harsh constraints and soft constraints. Hard limitations are those that, under no circumstances, ought to be broken. On the other hand, soft restrictions are the desirable type of demands that permit optional conditions while still being able to be handled as minor ones. The schedule that has a realistic conclusion and a sufficient level of quality is the one that has the fewest total violations of the soft restrictions. In certain situations, the hard restrictions may also be viewed as the soft limitations to arriving at a workable solution.

1. Interviews can be difficult to schedule.
2. Only FCVAC students can participate
3. This opportunity is only available during specific semesters.
4. Users can access the feature until the 14th week of the semester.

1.7.2 Critical Assumption

2. Manual managing is generally difficult, especially when unexpected changes happen without notice.
3. The lack of current information could contribute to misunderstanding and limit the facility's regular tasks.
4. There is an unexpected need for an automated system able to managing timetables properly.
5. Developing such a system will probably to result in more efficiency and less disruptions.

1.8 Project Management

Project management refers to the group of people who are in charge of carrying out the project.

Coordinator

SIR ZAHRUL AZWAN BIN ABDL KAMARUL ADZHAR

Supervisor

SIR AZHAR BIN HAMID

Student

NUR ATHIRAH MINHALINA BT MAHYIDDIN

1.8.1 Gantt Chart

Every job that has to be completed, the estimated time for each task, the due dates for completing each assignment, and the connections between different tasks are all shown on a Gantt chart. Another tool that will be used as a guide as this project approaches its deadline is the Gantt chart. This Gantt chart at appendix A (refer page).

1.9 Summary

Managing manually, such as assigning lecturers, courses, and classrooms, is difficult. It becomes significantly more difficult when un-expectation changes happen without notice, producing uncertainty and disruptions. To address these issues, an automated method for successfully managing timetable needs to be developed. This solution would help improve an organization while minimizing disruptions by providing real-time data as well as improved scheduling management.

CHAPTER 5

DESIGN

5.1 Introduction

This chapter focuses on developing the various components of our system to satisfy our defined standards. This entails outlining the general layout of our system and arranging components to work together. Sub-systems are the smallest units of the system that we begin by dissecting in order to better organize our work. For this, we make use of a tool known as the Unified Modeling Language (UML). UML use diagrams, which resemble drawings, to illustrate how various system components will cooperate. To better comprehend how the system will operate and what it will accomplish, we employ diagrams like as use case diagrams, sequence diagrams, class diagrams, and activity diagrams.

Another important part of developing our system is determining how the data will be structured. We employ Entity-Relationship Diagrams (ERDs) for this. ERDs show us how different bits of information are connected to one another, much like a map of our data. Finally, we create the user interface, which will allow users to interact with our system. We want it to be straightforward and understandable for the user, therefore we build it with that in mind.

5.2 System Architecture

The system's architecture is similar to a blueprint, illustrating how various components fit together and function. It includes factors like how users view and uses the system, where information is kept (for example in databases), and how everything communicates.

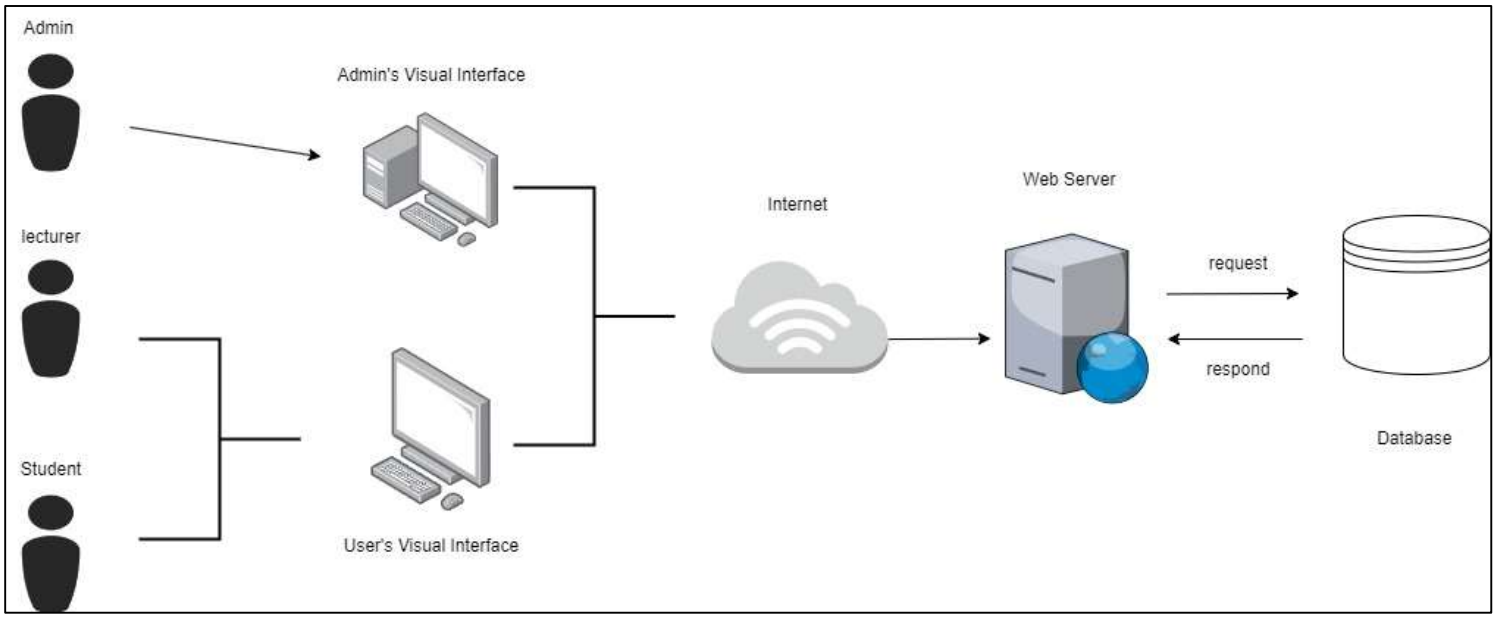


Figure 5.2.1 System Architecture

5.3 System Design

The Unified Modeling Language (UML) allows computer scientists to make diagrams to assist them design and discuss how computer programs will function. In UML, images are classified into four categories. First, utilize Case Diagrams depict how people or things will utilize a computer system to accomplish tasks. It's like creating a blueprint of how consumers will interact with the system. Next, Sequence Diagrams display the sequence of activities between various sections of a system. They allow you to observe how various elements communicate and collaborate.

Then, Class Diagrams demonstrate what is in the system, such as classes, characteristics, and connections. It's similar to drafting a diagram to outline how everything will be ordered and connected. Finally, Activity Diagrams, similar to flowcharts, depict the phases of a process or job. They aid in understanding how events occur in a sequential order. These UML diagrams assist teams in planning and understanding how to develop computer programs. They make it easy for everyone to communicate and establish effective processes.

5.3.1 Use Case Diagram

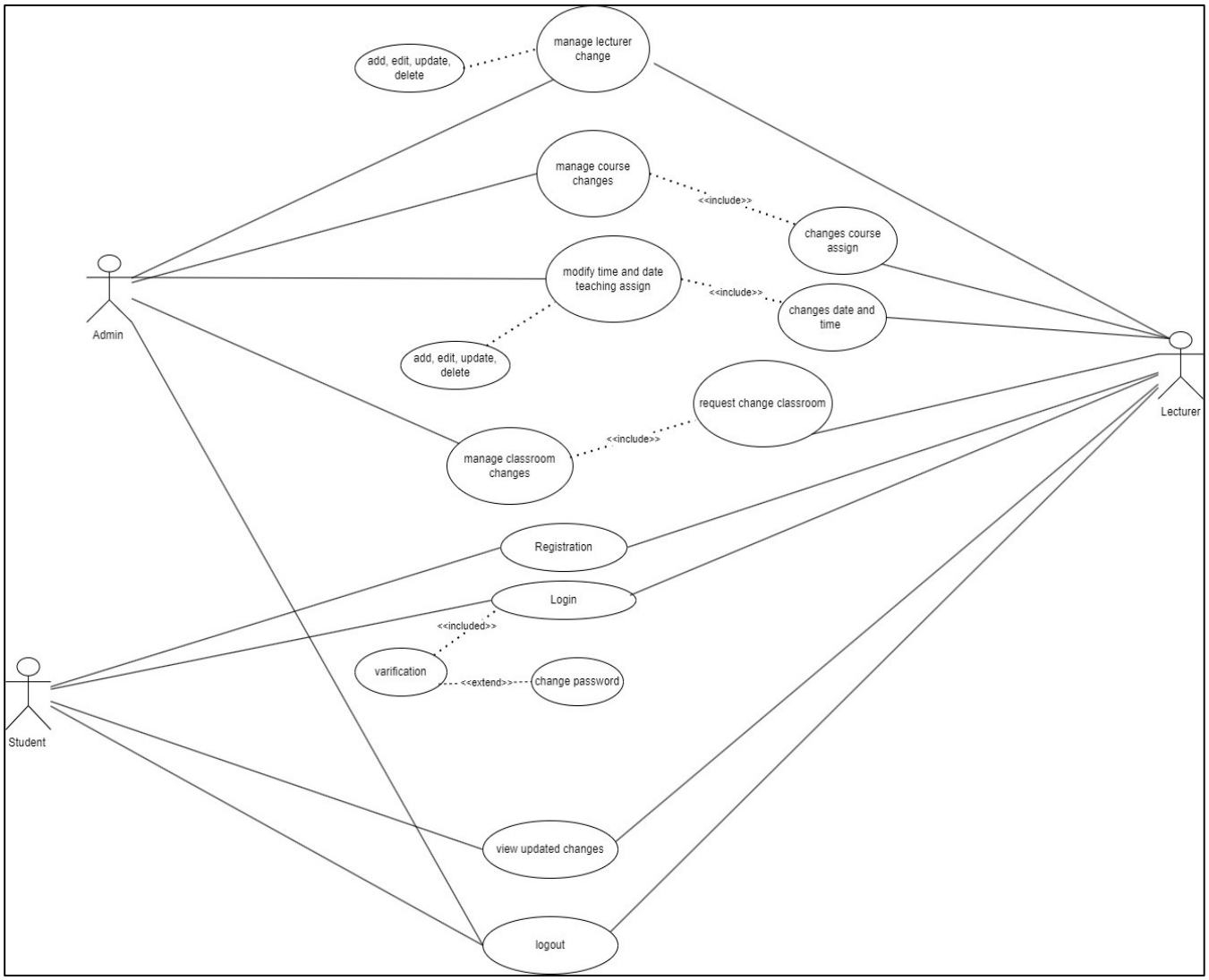


Figure 5.3.1 Use Case Diagram

5.3.1.2 Use case Registration

5.3.1.2.1 manage lecturer changes (add, edit, delete, update).

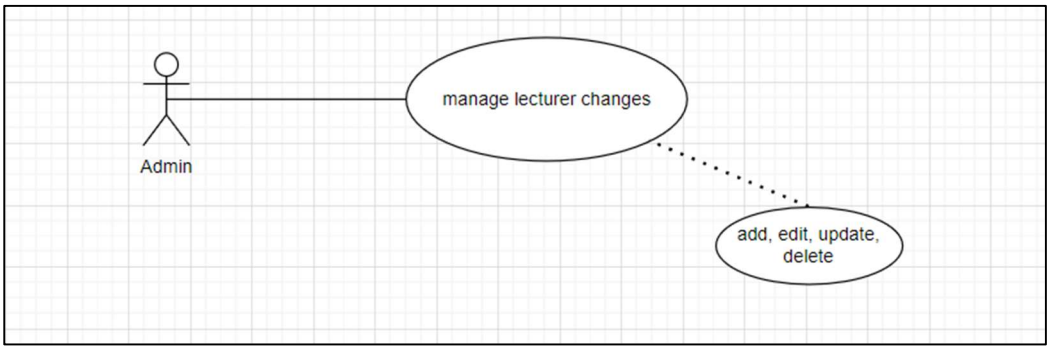


Figure 5.3.1.2.1 Manage Lecturer Change

Description	
Use case ID	001
Use case name	Manage lecturer change (ADD, EDIT, UPDATE AND DELETE)
Brief Description	This use case describes how an administrator will handle modifications pertaining to lecturers, such as adding, modifying, updating, and removing courses linked to lecturers.
Actors	Admin
Pre-Condition	To oversee actions pertaining to lecturers, the administrator needs to be signed into the system and have the required permissions. There ought to be instructors and courses already in place within the system.
Basic Flow	1. The interface for lecturers is accessed by the administrator. 2. The system offers options for managing lecturers, such as adding, modifying, updating, or eliminating courses linked to lecturers. 3. If the administrator decides to add a lecturer course: a. When adding a course, the administrator chooses the lecturer. b. The admin is prompted by the system to enter the information for the new course. c. The system adds the course for the chosen lecturer after the

	admin submits the data. 4. If the administrator chooses to modify or update a lecturer's course: a. The administrator chooses the lecturer whose course they wish to modify or update. b. The list of courses connected to the chosen lecturer is shown by the system. c. The administrator chooses the course that needs updating or editing. d. The admin is given options by the system to change the course details. e. The administrator submits the required adjustments after making them. f. The system makes the necessary updates to the course details. 5. If the administrator chooses to remove a lecturer's course: a. The administrator chooses the lecturer whose course they wish to remove. b. The list of courses connected to the chosen lecturer is shown by the system. c. The administrator chooses which course to remove. d. The administrator is prompted by the system to verify the deletion action. e. The administrator authorizes the removal. f. The chosen course is deleted by the system from the database. g. The program notifies the user that the course has been deleted and verifies that the deletion was successful.
Alternative Flow	In the event that the administrator chooses to stop any action in progress: a. The administrator chooses to stop the current action. b. The admin is taken back to the lecturer management interface by the

	system, which halted the current action. c. The administrator can then decide whether to log out of the system or carry out additional lecturer management tasks.
Exceptional Flow	If a technical problem arises while a lecturer is managing the system: a. An error message detailing the nature of the problem is displayed to the administrator. b. The admin is prompted to attempt the action again later by the system. c. The administrator may need to get in touch with technical support if the problem continues.
Post Condition	After a lecturer management action is successfully completed, the following happens: a. The system performs the designated action (addition, editing, updating, or deletion). b. The course linked to the chosen lecturer is eliminated from the system, specifically for deletion. c. The deleted course is no longer visible to the administrator in the lecturer management interface. d. The system is appropriately updated or cleared of any associated data or records pertaining to the deleted course.

5.3.1.2.2 manage course changes (add, edit, delete, update).

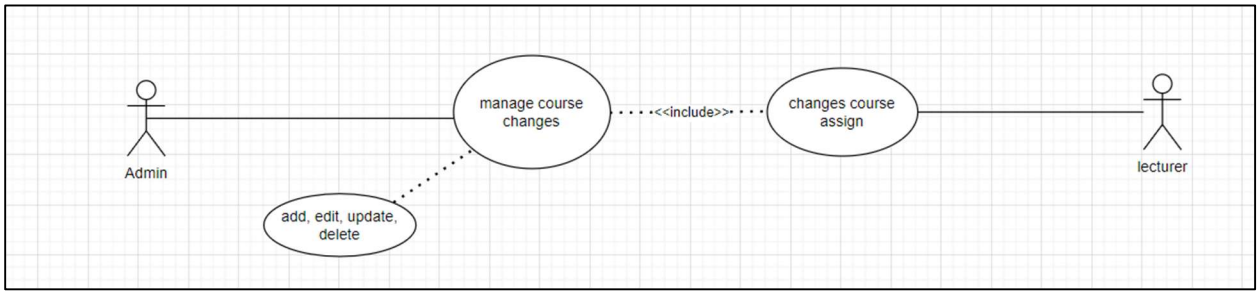


Figure 5.3.1.2.2 Manage Course

Description	
Use case ID	002
Use case name	Manage Course (ADD, EDIT, UPDATE AND DELETE COURSE)
Brief Description	This use case involves comprehensive course administration functionalities, encompassing addition, editing, updating, and deletion of courses. Additionally, it facilitates the capability to change course assignments for lecturer actors. This ensures that the system's courses remain accurate, current, and aligned with users' requirements.
Actors	Admin, Lecturer
Pre-Condition	The administrator needs to be logged into the system with appropriate permissions and valid credentials to manage courses. Additionally, the administrator should be able to add, edit, update, delete courses, and change course assignments for lecturers using the existing course structure within the system.
Basic Flow	1. The admin logs in with their details. 2. After logging in, they can see the course options. 3. They can pick from options like adding, changing, removing courses, or changing who teaches them. 4. If they want to remove a course: a. They see a list of courses and pick one to remove.

	b. They confirm that they want to delete it. c. The system takes it off the list. d. The system tells them it's done. 5. If they want to do something else, like adding or changing courses, or changing who teaches them, they can do that too. 6. After they finish, they can log out or keep doing things in the system.
Alternative Flow	• If they decide to stop removing a course: a. They choose to cancel. b. The system doesn't delete anything and takes them back to the options. c. They can then decide what to do next.
Exceptional Flow	• If something goes wrong when doing anything with courses, like adding, changing, or removing them, or changing who teaches them: a. They get an error message. b. The system tells them to try again later. c. If it keeps happening, they might need to ask for help.
Post Condition	• After they finish doing anything with courses, like adding, changing, or removing them, or changing who teaches them: a. The course list gets updated. b. If they deleted a course, it's not there anymore for people to choose. c. They won't see the deleted course in the list. d. The system makes sure everything's correct and updated. e. If they changed who teaches a course, it shows up in the system.

5.3.1.2.3 modify time and date teaching assign (add, edit, delete, update).

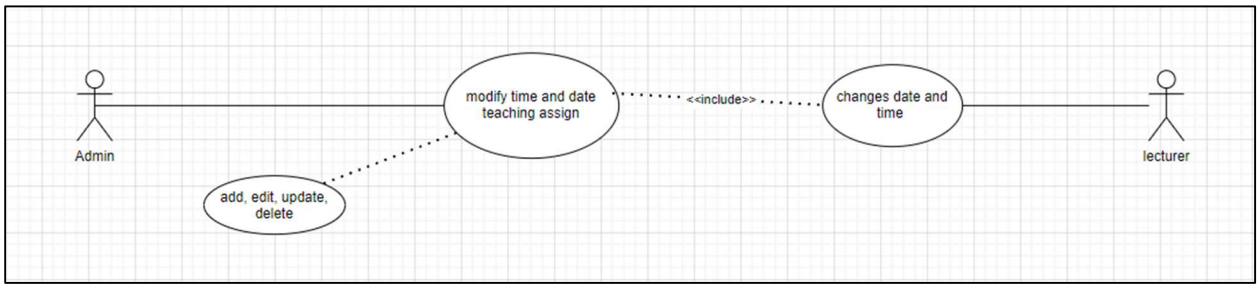


Figure 5.3.1.2.3 Modify Time & Date

Description	
Use case ID	003
Use case name	Modify Time and Date Teaching Assign (ADD, EDIT, UPDATE AND DELETE)
Brief Description	This use case outlines the process for administrators to manage course schedules by adding, editing, updating, and deleting them. Additionally, it involves modifying the date and time of teaching assignments, which includes the capability to change the assigned lecturer for a particular time and date.
Actors	Admin, lecturer
Pre-Condition	The administrator must be logged into the system with the necessary permissions, and there should be existing courses available in the system.
Basic Flow	• The administrator accesses the teaching assignment management interface. • The system presents options to add, edit, update, or delete course schedules, including modifying date and time assignments. • If the administrator chooses to add a course schedule: a. They select the course requiring a schedule addition. b. The system prompts for details such as date, time, duration, and

	location. c. After submission, the schedule is added to the chosen course. • If the administrator wants to modify or update a course schedule: a. They choose the course needing a schedule change. b. The system displays current schedules for the selected course. c. The admin selects the schedule to modify. d. System provides options to change schedule details, including date, time, and assigned lecturer. e. After adjustments, the admin submits changes. f. The system updates the schedule details accordingly, including the assigned lecturer. • If the administrator decides to remove a course schedule: a. They choose the course with the schedule to remove. b. System shows current schedules for the chosen course. c. The admin selects the schedule to delete. d. System verifies deletion action. e. After confirmation, the system deletes the chosen schedule. f. Confirmation of successful deletion is displayed.
Alternative Flow	If the administrator chooses to stop any action in progress: a. The administrator chooses to stop the current action. b. The admin is brought back to the teaching assignment management interface by the system, which halts the current action. c. The administrator can then decide whether to log out of the system or carry out additional teaching assignment management tasks.
Exceptional Flow	If the system encounters technical issues during any teaching assignment management action: a. The system notifies the administrator of the error and describes the nature of the problem if it encounters technical difficulties while handling any teaching assignment management action. b. The admin is prompted to attempt the action again later by the

	system. c. The administrator may need to get in touch with technical support if the problem continues.
Post Condition	Following the successful completion of any teaching assignment management action, the following happens: a. The system performs the designated action (addition, editing, updating, or deletion). b. In the case of deletion, the system deletes the schedule connected to the chosen course. c. The teaching assignment management interface's deleted schedule is no longer visible to the administrator. d. The system is appropriately updated or cleared of any associated data or records pertaining to the deleted schedule.

5.3.1.2.4 manage classroom changes (request change classroom).

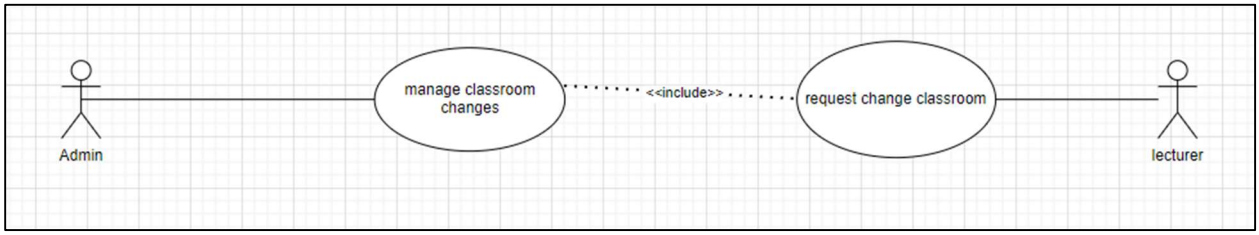


Figure 5.3.1.2.4 Manage Classroom Change

Description	
Use case ID	004
Use case name	Manage Classroom Changes (request change classroom).
Brief Description	This use case explains how an administrator can add, edit, update, and remove classroom data, among other changes related to the classroom.
Actors	Admin
Pre-Condition	To handle classroom changes, the administrator needs to be logged into the system and have the required permissions. In addition, the system ought to contain current classroom data.
Basic Flow	• The interface for classroom management is accessed by the admin. • The system offers choices for handling modifications to the classroom, such as adding, modifying, updating, or removing classroom data. • Should the administrator decide to add a classroom: a. The administrator gives the new classroom the information it needs, including the room number, capacity, amenities, etc. b. After confirming the supplied data, the system incorporates the new classroom. • If the administrator chooses to update or modify a classroom: a. From the list of existing classrooms, the administrator chooses the

	classroom they wish to modify. b. The system shows the most recent data for the chosen classroom. c. The administrator updates the classroom's pertinent details. d. The system incorporates the most recent modifications into the classroom data. • If the administrator decides to remove a classroom, they must first: a. choose the desired classroom from the list of active classrooms. b. The administrator is prompted by the system to verify the deletion action. c. The administrator authorizes the removal. d. The chosen classroom is deleted by the system from the database. e. The system notifies the user that the classroom has been deleted and verifies that the deletion was successful.
Alternative Flow	In the event that the administrator chooses to stop any action in progress: a. The administrator chooses to stop the current action. b. The admin is taken back to the classroom management interface by the system, which halted the current action. c. The administrator can then decide whether to log out of the system or carry out additional classroom management tasks.
Exceptional Flow	In the event that a technical problem arises while a classroom management action is being performed: a. The system notifies the administrator of the error and describes the problem in detail. b. The admin is prompted to attempt the action again later by the system. c. The administrator may need to get in touch with technical support if the problem continues.
Post Condition	Following the successful completion of any classroom management action, the following happens:

	a. The system performs the designated action (addition, editing, updating, or deletion). b. The classroom is deleted specifically from the system. c. The deleted classroom is no longer visible to the administrator in the classroom management interface. d. The system is appropriately updated or cleared of any associated data or records pertaining to the deleted classroom.

5.3.1.2.5 Registration (lecturer and Student)

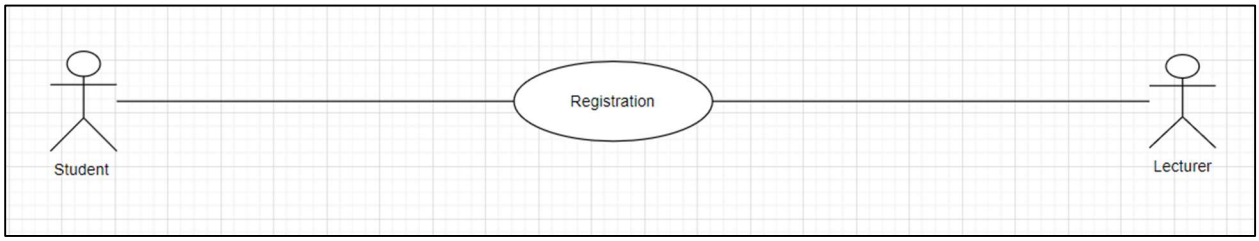


Figure 5.3.1.2.5 Registration

Description	
Use case ID	005
Use case name	Registration
Brief Description	This use case outlines the procedure by which both lecturers and students register for system access.
Actors	Lecturer, student
Pre-Condition	For both lecturers and students to have access to the system, they either need to be sent registration credentials or an invitation to register.
Basic Flow	• The system provides a registration interface that both lecturers and students can access. • The actor (either lecturer or student) is prompted by the system to enter their registration credentials, which usually consist of private data like name, password, and email address. • The actor completes the necessary registration fields. • The accuracy and completeness of the information are checked by the system. • The system generates a new account (either lecturer or student) and provides access to the system if the data submitted is legitimate. • A confirmation message indicating successful registration is sent to the actor.
Alternative Flow	In the event that the actor (lecturer or student) enters their registration

	information incorrectly: a. The system notifies the actor to check and update any inaccurate or missing data. b. The actor makes the necessary revisions to the registration details and resubmits them. c. The system confirms the data that was submitted once more. d. The system creates the account and grants access to the system if the information supplied is now valid. e. The actor might need to get help from support if mistakes continue.
Exceptional Flow	In the event that there are technical difficulties with the registration process: a. An error message detailing the nature of the technical problem (such as a server error or a failed database connection) is displayed by the system. b. It is advised that the actor (lecturer or student) try registering again at a later time. c. The actor may need to get in touch with technical support if the problem continues.
Post Condition	Following a successful registration: For lecturers: a. The system creates the lecturer's account. b. The lecturer uses the given credentials to log into the system. c. The lecturer is now able to use the features of the system as allowed by their role and permissions. For students: a. The system creates the student's account. b. The student uses the given credentials to log into the system. c. The student is now able to use the features of the system as allowed by their role and permissions.

5.3.1.2.6 manage classroom changes (request change classroom).

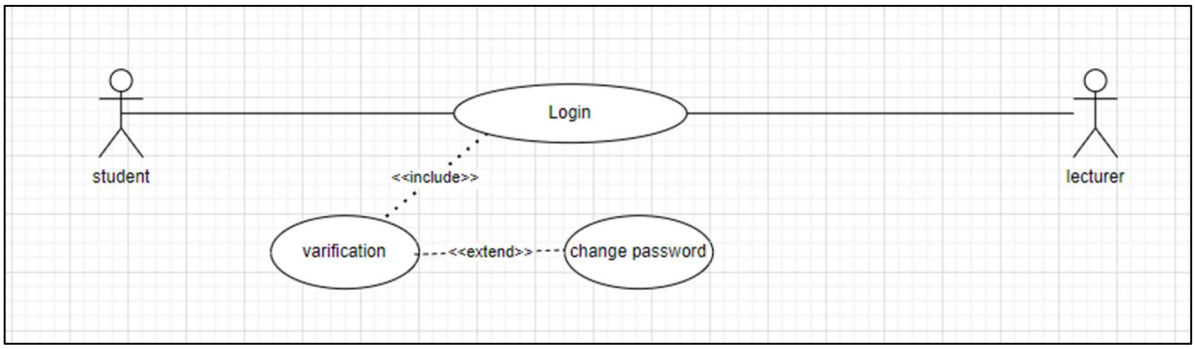


Figure 5.3.1.2.6 Manage Classroom Change

Description	
Use case ID	006
Use case name	Login (Verification, Change Password)
Brief Description	This use case explains how both lecturers and students log into the system, confirm their identity, adjust their passwords as needed, and deal with login errors.
Actors	Lecturer, student
Pre-Condition	The actor (either lecturer or student) needs to already have a registered account in the system.
Basic Flow	• The actor (either lecturer or student) accesses the page provided by the system for login. • The actor is prompted by the system to enter their login information, which usually consists of a password and username (such as an email address). • The actor inputs their login details. • The system compares the entered credentials to the data that has been stored. • If the credentials provided are valid, the system allows access to the actor and sends them to the dashboard or specified landing

	page. • If the actor wants to modify their password: a. Go to the system's user settings or profile and select the "Change Password" option. b. The actor is prompted by the system to enter their existing password and then choose a new one. c. The actor fills out the form and submits it with the necessary data. d. After confirming the modifications, the system modifies the actor's password. • If the actor encounters an error during the login process, the system will display an error message informing them of the login error (e.g., invalid username or password), and prompt them to try entering their credentials again.
Alternative Flow	In the event that an actor (lecturer or student) needs to reset their password: a. Click the "Forgot Password" link that appears on the login screen. b. The actor is prompted by the system to enter their email address upon registration. c. The actor's email address receives a link to reset their password from the system. d. The actor clicks on the link to go to a page where they can change their password. e. After confirming the request for a password reset, the system modifies the password.
Exceptional Flow	In the event that a technical problem arises when logging in: a. An error message indicating the nature of the problem (e.g., server error, database connection failure) is displayed by the system. b. A note is sent to the actor telling them to try logging in later. c. If the problem continues, the actor might need to get in touch with technical support for more help.

Post Condition	• After logging in successfully: • The actor (lecturer or student) can access the features and functionalities of the system. • After the actor's identity has been confirmed, they are able to act in accordance with the roles and permissions that have been granted to them by the system. • When a password is changed: • The system updates the actor's password. • For upcoming sessions, the actor can log in with the new password. • When an actor gets a login error, they are informed of it and asked to try logging in again using the right credentials. If necessary, the actor can choose to recover their password or try to log in again.

5.3.1.2.7 View updated changes

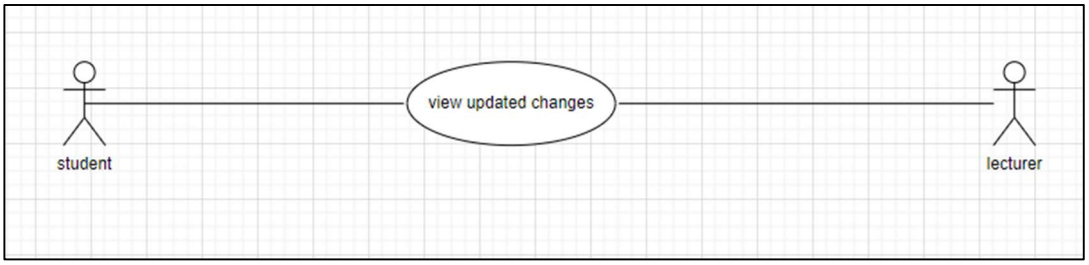


Figure 5.3.1.2.7 View Update Changes

Description	
Use case ID	007
Use case name	View Update Changes
Brief Description	The method by which a lecturer examines, and updates system changes is described in this use case.
Actors	Lecturer, student
Pre-Condition	It is necessary for the lecturer to have the necessary permissions to view and update changes in the system.
Basic Flow	• The actor (either lecturer or student) accesses the system's update changes interface. • The system presents a list of updates or modifications available to view and/or modify. • The actor selects a particular update or modification from the list. • If the update or change needs to be modified: a. The actor chooses to update or edit the selected item. b. The system displays the current details of the selected item. c. The actor makes the required changes to the item. d. The actor updates the system with the new data. e. After confirming the modifications, the system updates the item. • If the modification or update is solely for visual purposes: a. The actor chooses to view the selected item.

	b. The system displays the current details of the selected item for the actor to review.
Alternative Flow	In the event that the actor (lecturer or student) chooses to reverse any changes made along the way: a. The actor decides to reverse the current modification. b. The system brings the actor back to the update changes interface, halting the modification process. c. The actor can then decide whether to leave the system or take other actions.
Exceptional Flow	In the event that there are technical difficulties with the system while viewing or modifying: a. An error message describing the nature of the problem is displayed by the system (e.g., server error, database connection failure). b. The actor is advised to attempt accessing or altering the modifications at a later time. c. The actor may need to get in touch with technical support if the problem continues.
Post Condition	Following a successful viewing or modification of the changes: a. The actor (lecturer or student) has updated or reviewed the required system changes. b. The system updates itself in accordance with the actor's modifications. If any changes were made, the system saves and applies the updated changes. If nothing more than viewing took place: a. The actor learns more about the specifics or current state of the chosen item.

5.3.1.2.8 Logout

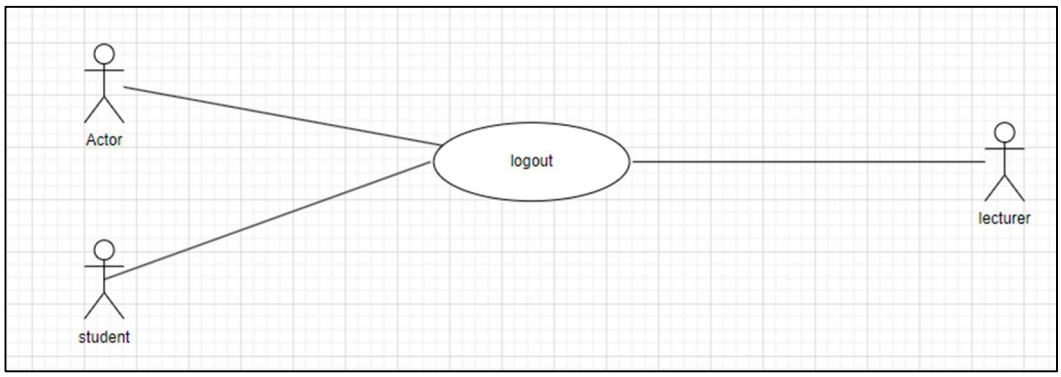


Figure 5.3.1.2.8 Logout

Description	
Use case ID	008
Use case name	Logout
Brief Description	This use case describes how an admin, lecturer, or student logs out of the system.
Actors	Admin, lecturer, student
Pre-Condition	It is required that the user log out the system.
Basic Flow	1. The actor (admin, lecturer, or student) navigates to the logout feature within the system interface. 2. The system confirms the actor's intention to log out. 3. The actor validates the action of logging out. 4. The system ends the actor's session and redirects them to the logout confirmation page.
Alternative Flow	To maintain security, the system automatically logs out the actor if they are not active for a predetermined amount of time. When an actor logs out due to inactivity, they are taken to the login page.
Exceptional Flow	In the event that there are technical difficulties with the logout process: a. An error message detailing the nature of the problem is displayed by the system (e.g., server error, database connection failure).

	b. The actor is advised to close the browser window and attempt to log out at a later time. c. The actor may need to get in touch with technical support if the problem continues.
Post Condition	Following a successful logout: The actor's session ends, and their system authentication is removed. Either the login page or the logout confirmation page is displayed to the actor. The actor discards any data or actions that are not saved during the session. Until they log in once more, the actor is unable to utilize system features.

5.3.2 Sequence Diagram

A sequence diagram is a form of interaction diagram that illustrates how objects interact in a specific scenario from a use case or system activity.

5.3.2.1 Sequence Diagram (Admin Login)

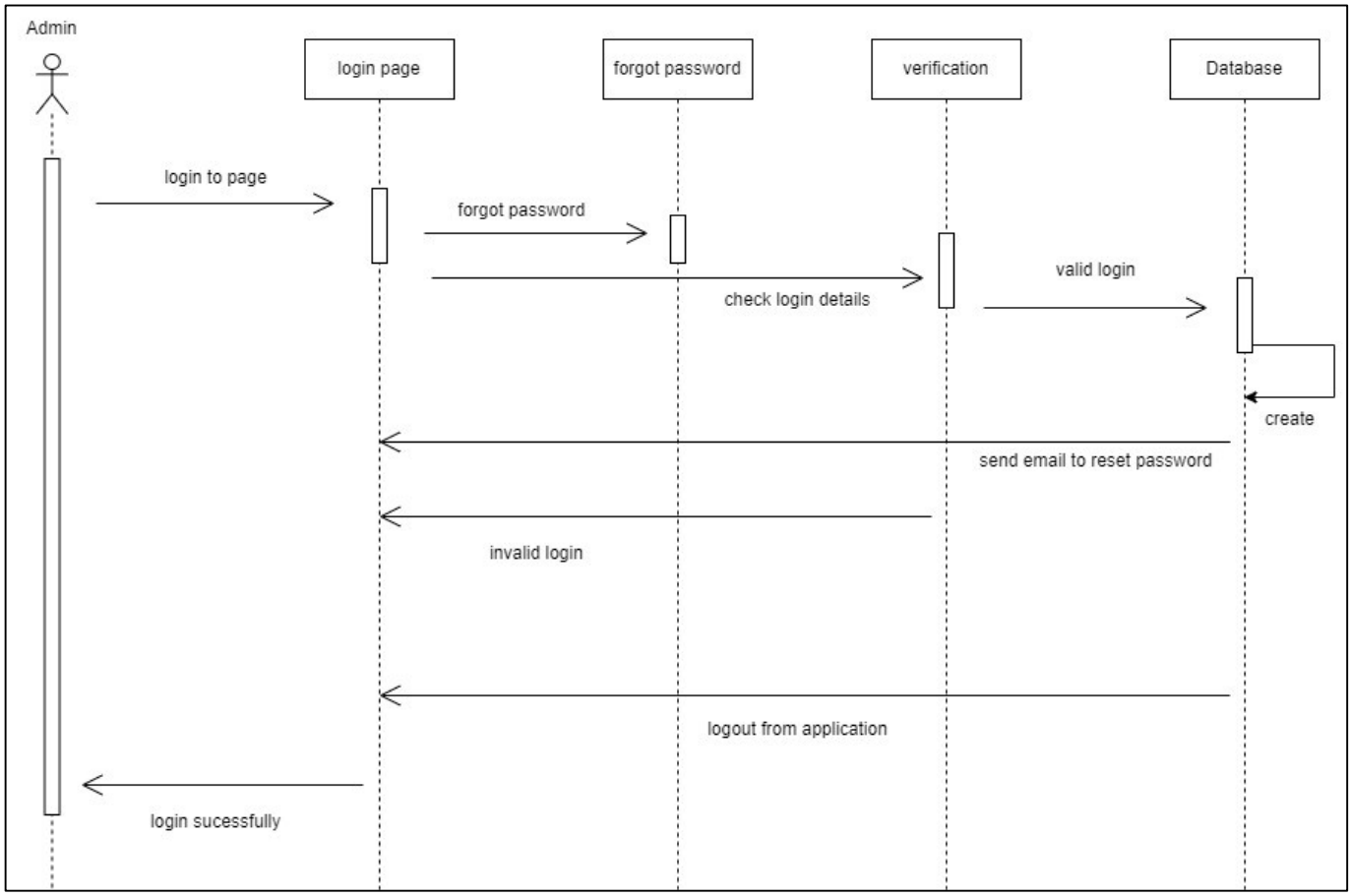


Figure 5.3.2.1 Admin Login

5.3.2.2 Sequence Diagram (Admin manage lecturer)

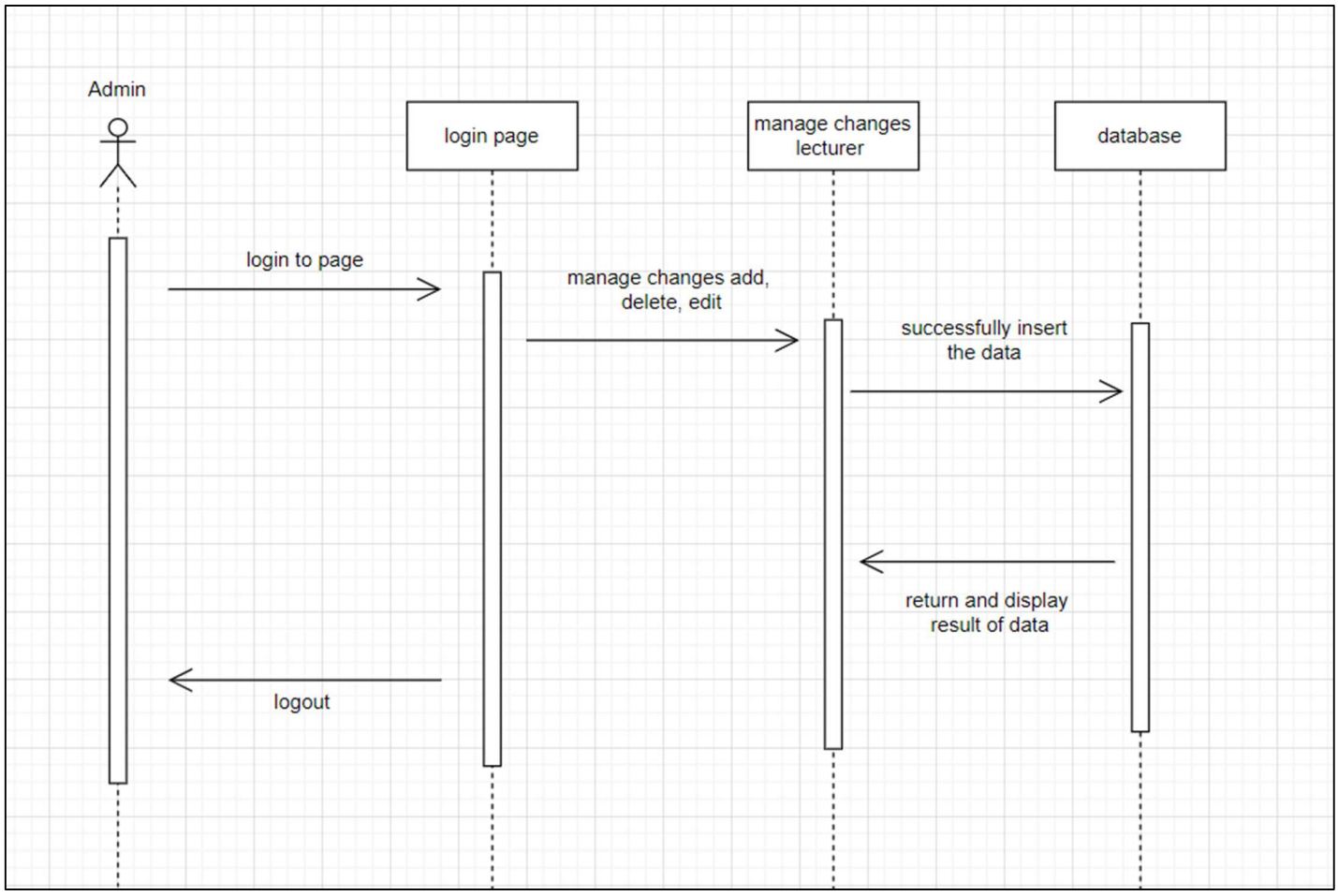


Figure 5.3.2.2 Admin Manage Lecturer

5.3.2.3 Sequence Diagram (Admin manage course, classroom and modify date & time)

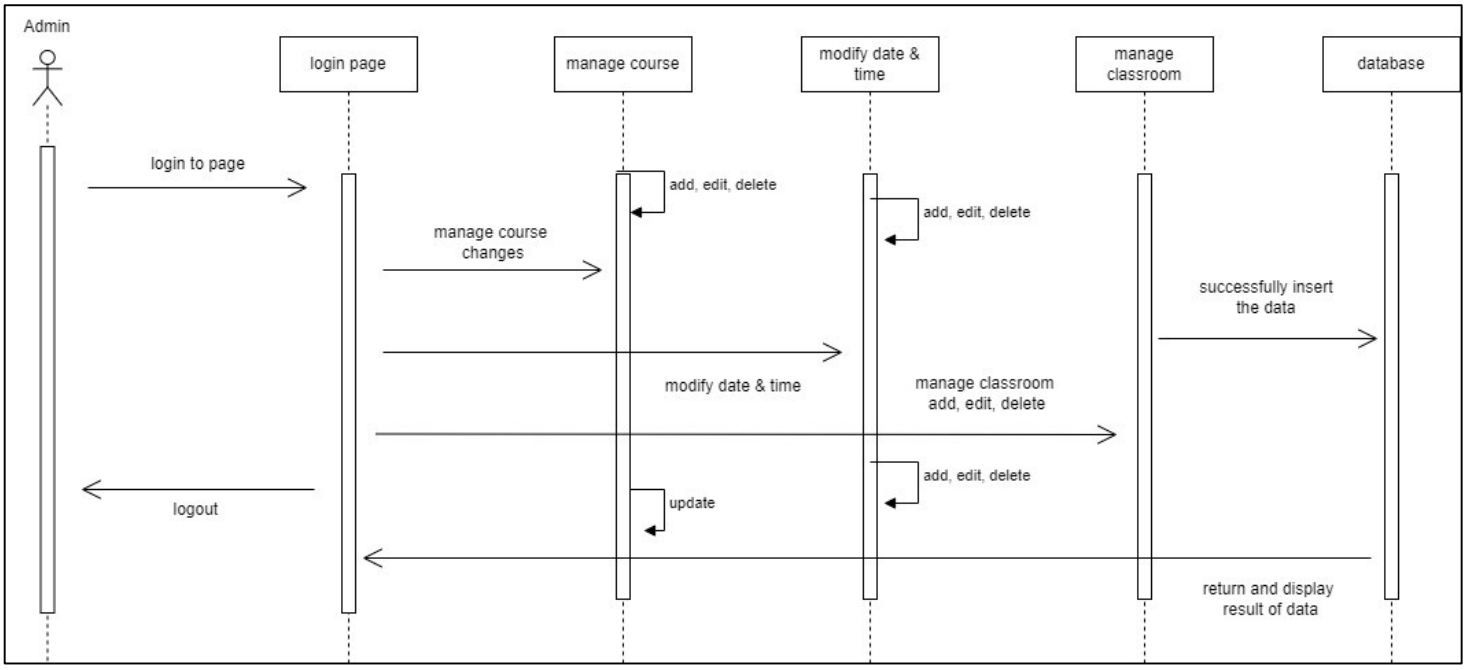


Figure 5.3.2.3 Admin Page

5.3.2.4 Sequence Diagram (student and lecturer register page)

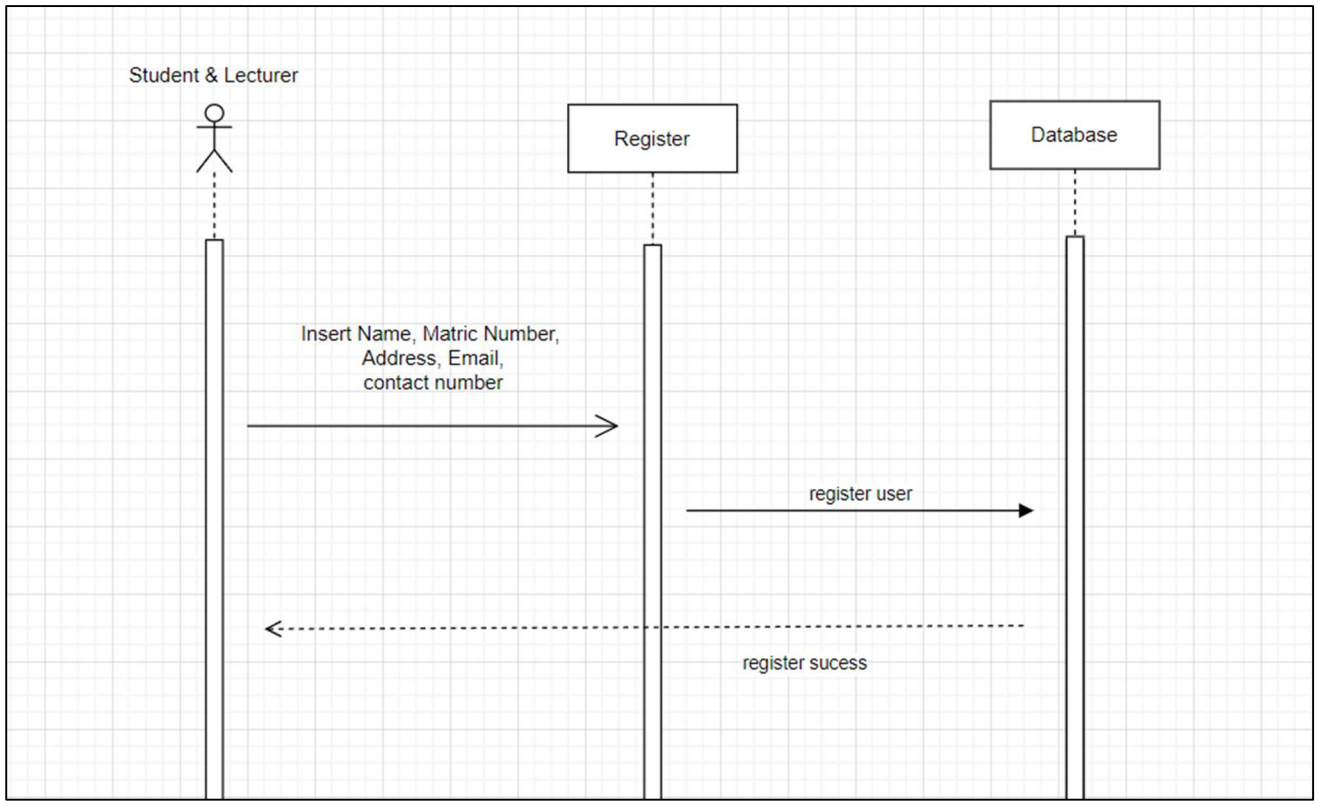


Figure 5.3.2.4 Register Page

5.3.2.5 Sequence Diagram (student and lecturer view updated page)

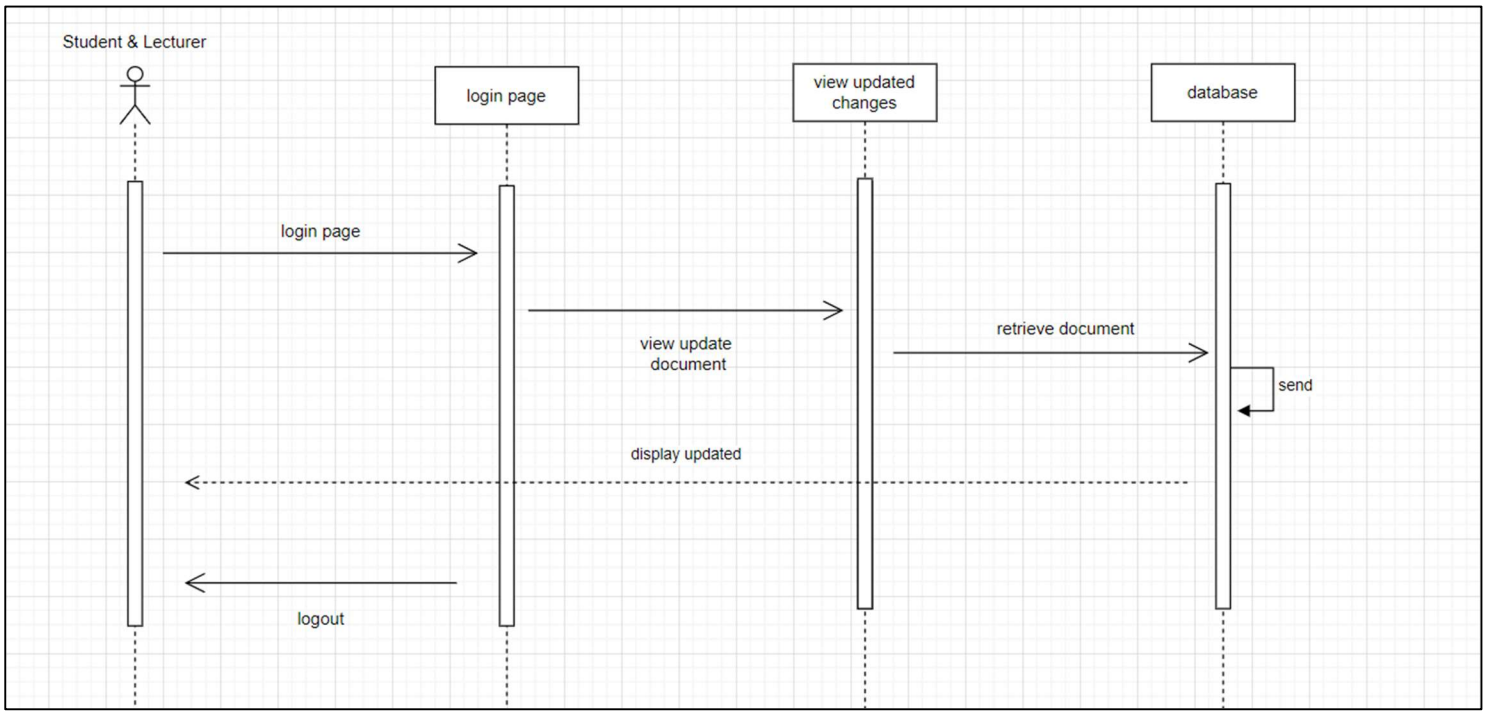


Figure 5.3.2.5 View Update Page

5.3.2.6 Sequence Diagram (lecturer page)

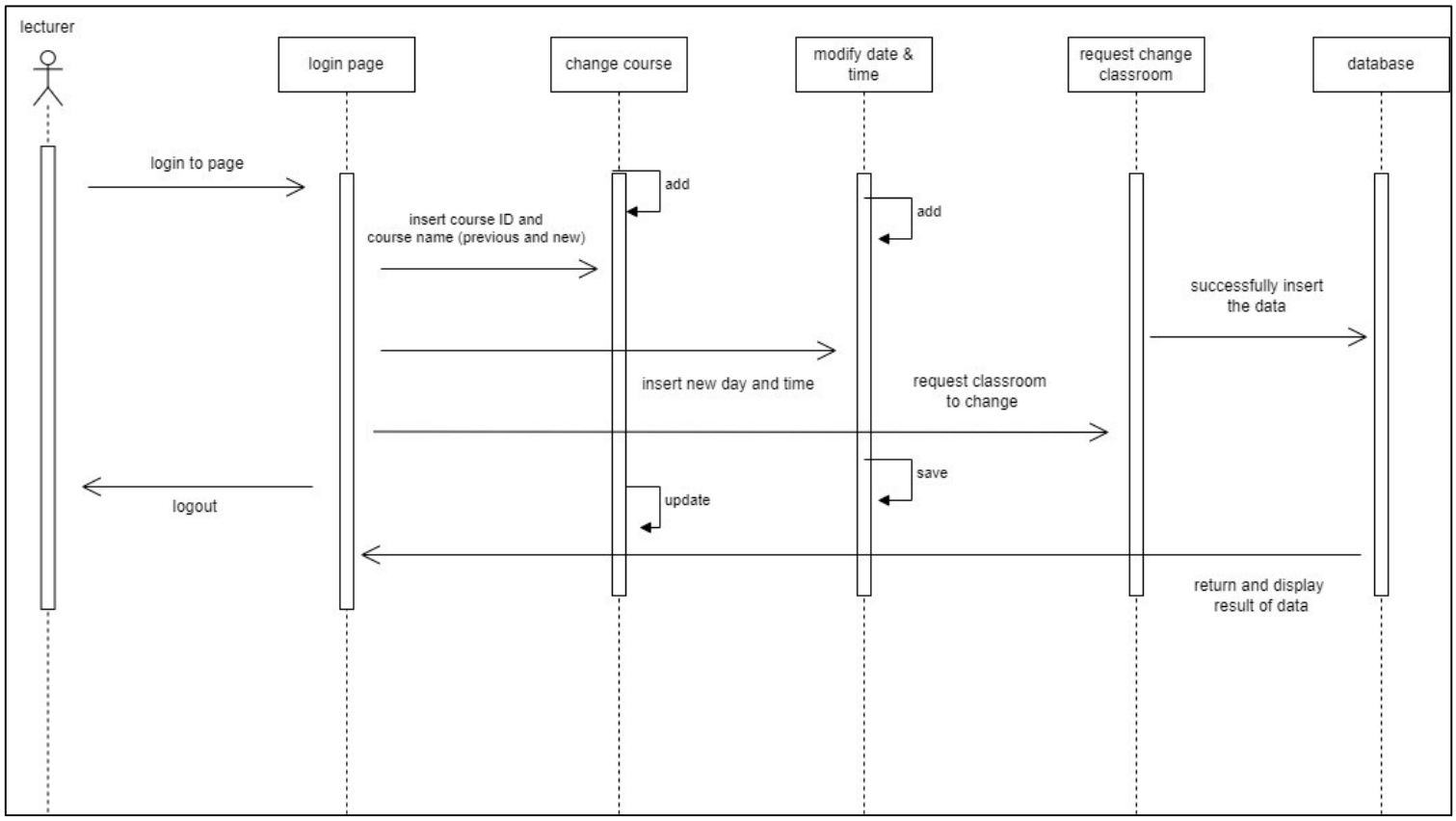


Figure 5.3.2.6 Lecturer Page

5.3.3 Activity Diagram

An activity diagram is a sort of behavioral diagram that describes the flow of control or data between activities in a system.

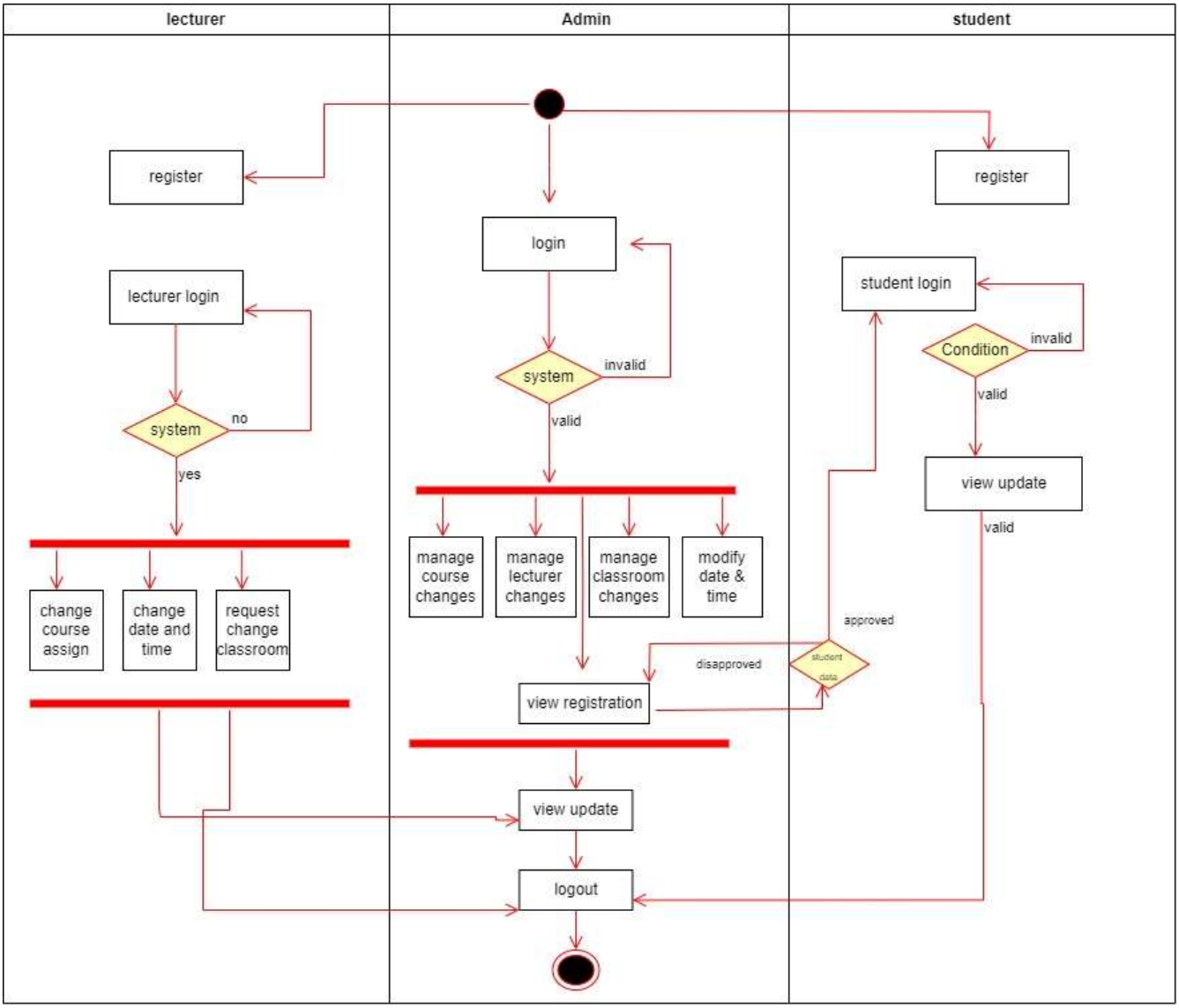


Figure 5.3.3.1 Activity Diagram

5.3.4 Class Diagram

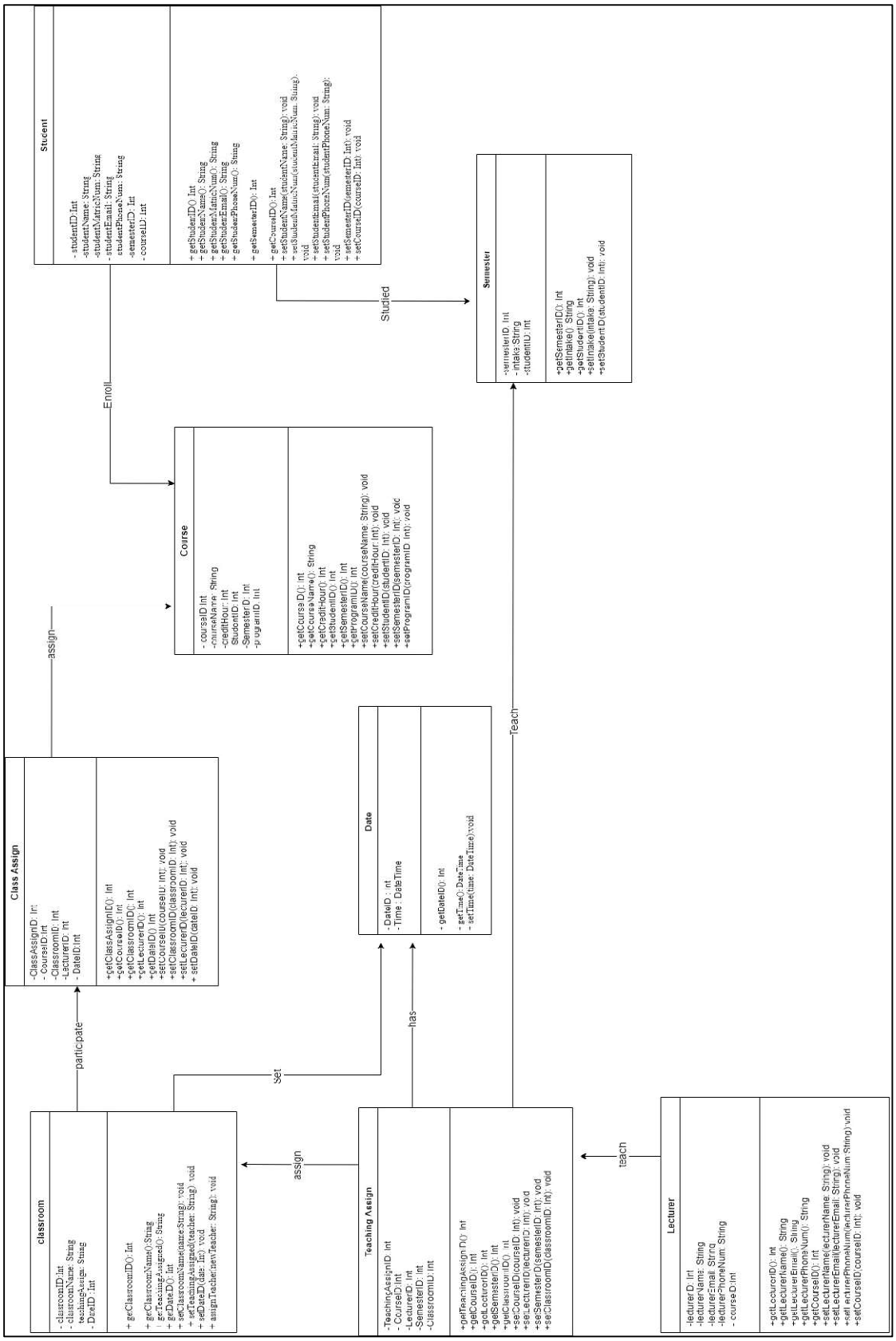


Figure 5.3.4 Class Diagram

5.4 Database Design

There are several stages involved in database design, including as implementation, normalization, schema design, and conceptual modeling.

5.4.1 ERD Diagram

A database's entities and connections are shown visually in an entity-relationship diagram (ERD).

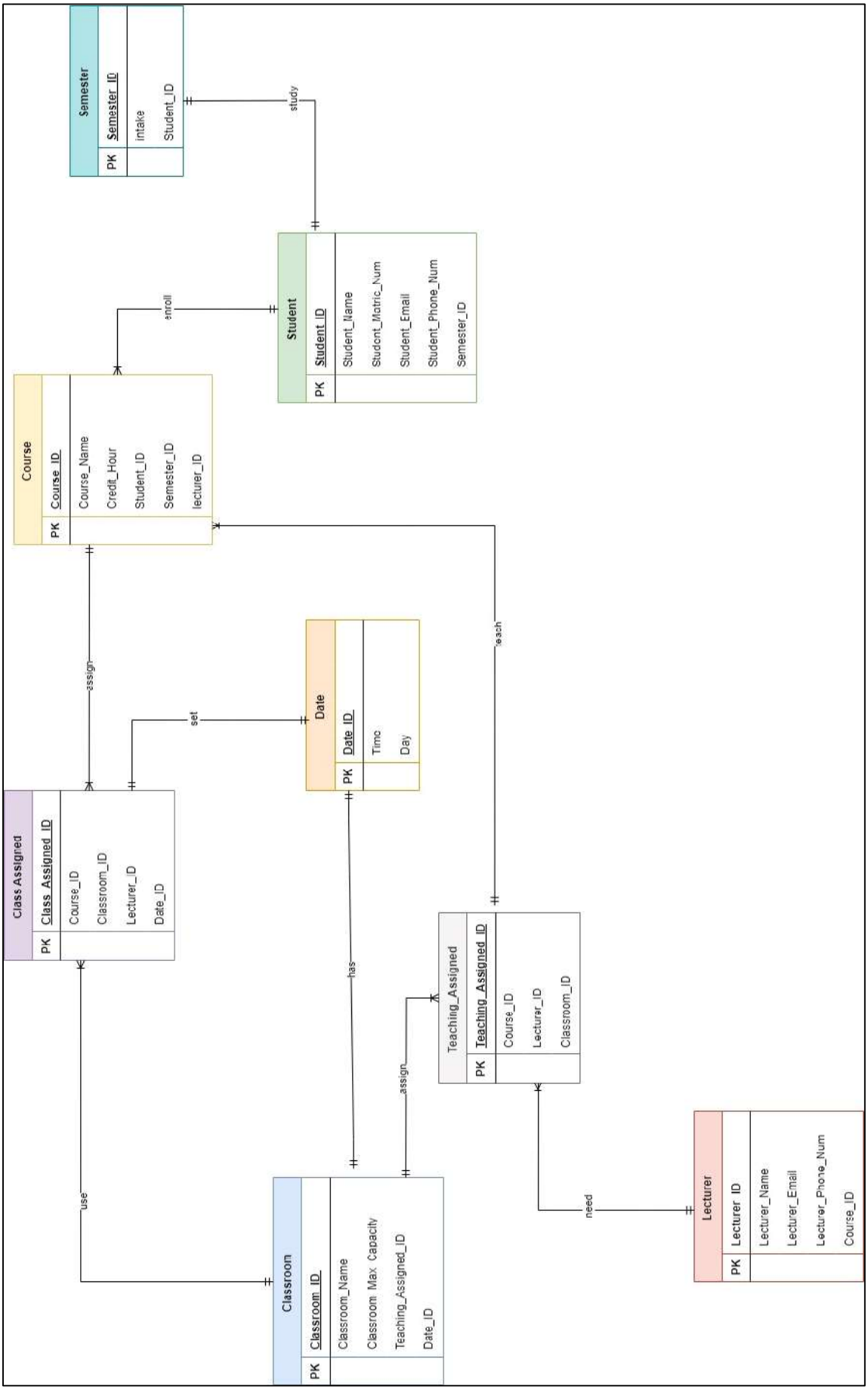


Figure 5.4.1.1 ERD Diagram

5.4.2 Data Dictionary

Classroom

Field Name	Field Type	Field Size	Unique
Classroom_id	Int	50	Primary key
Classroom_name	Varchar	100	Null
Teaching_assigned_id	Int	100	Foreign Key
Date_id	Int	100	Foreign Key

Class Assigned

Field Name	Field Type	Field Size	Unique
Class_assigned_id	Int	50	Primary key
Course_id	Int	50	Foreign key
Classroom_id	Int	50	Foreign key
Lecturer_id	Int	50	Foreign key
Date_id	Int	50	Foreign key

Date

Field Name	Field Type	Field Size	Unique
Date_id	Int	100	Primary key
Time	DateTime	200	Null
Day	Varchar	20	Null

Course

Field Name	Field Type	Field Size	Unique
Course_id	Int	50	Primary key
Course_name	Varchar	50	Null
Credit_hour	Int	20	Null
Student_id	Int	50	Foreign key
Semester_id	Int	50	Foreign key

Student

Field Name	Field Type	Field Size	Unique
Student_id	Int	50	Primary key
Student_name	String	50	Null
Student_matric_num	String	55	Null
student_email	String	100	Null
Student_phone_num	String	100	Null
Semester_id	String	50	Foreign key
Course_id	Int	50	Foreign key

Semester

Field Name	Field Type	Field Size	Unique
Semester_id	Int	50	Primary key
Intake	Varchar	50	Null
Student_id	Int	50	Foreign key

Teaching assigned

Field Name	Field Type	Field Size	Unique
Teaching_assigned_id	Int	50	Primary key
Course_id	Int	50	Foreign key
Lecturer_id	Int	50	Foreign key
Semester_id	Int	50	Foreign key
Classroom_id	Int	50	Foreign key

Lecturer

Field Name	Field Type	Field Size	Unique
Lecturer_id	Int	50	Primary key
Lecturer_name	Varchar	100	Null
Lecturer_email	Varchar	100	Null
Lecturer_phone_num	Varchar	100	Null
Course_id	Int	50	Foreing key

5.5 Interface Design

Designing a user interface (UI) requires creating visual elements and interactive components that allow users to engage with a software program or system.

5.5.1 Admin Login

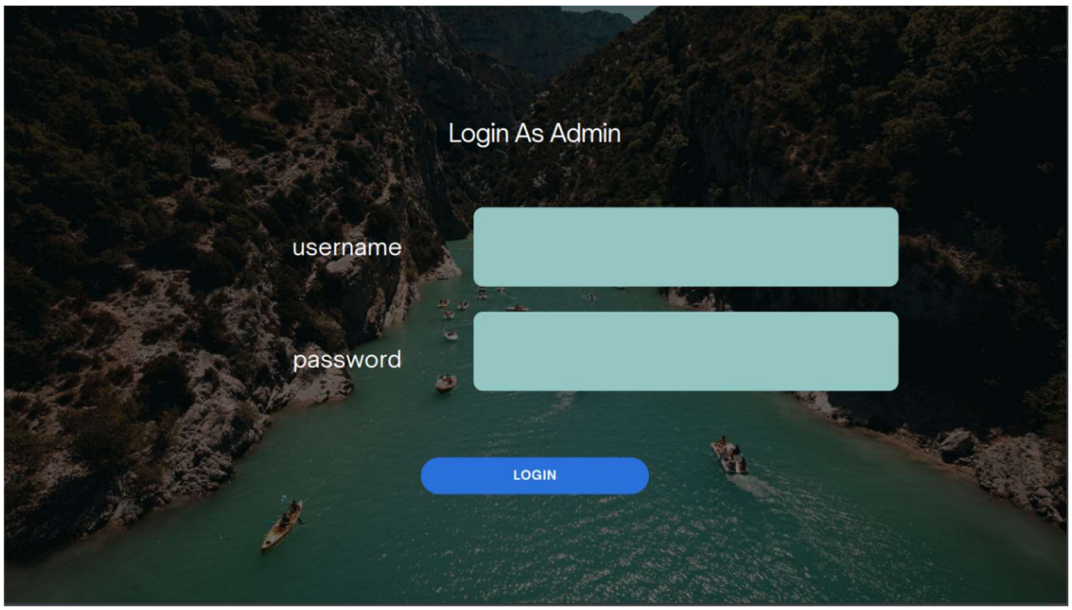


Figure 5.5.1 Admin Login

5.5.2 Admin Dashboard

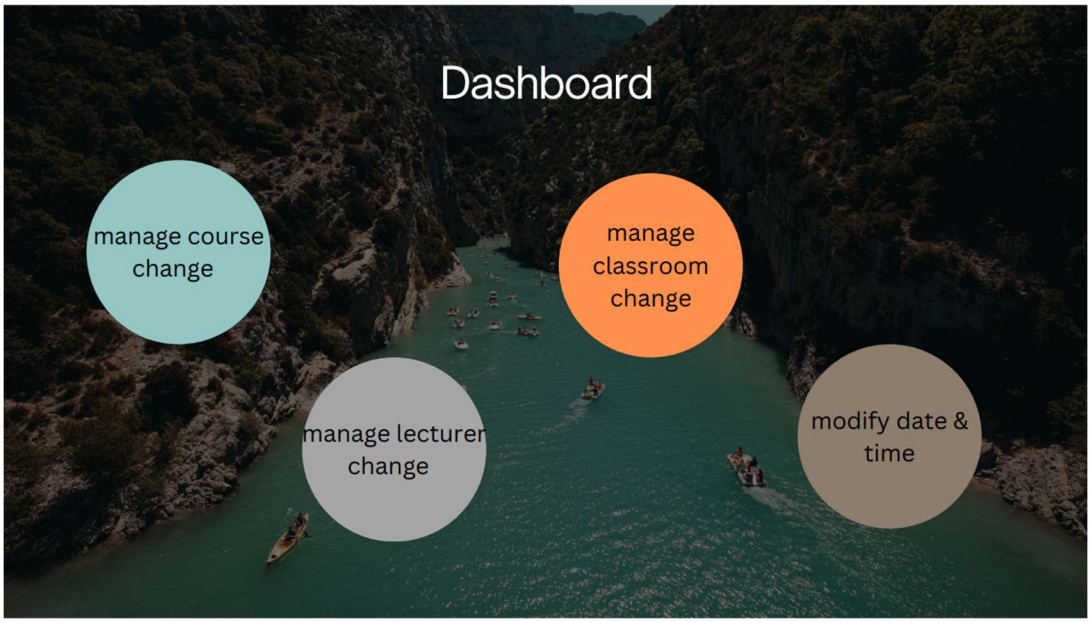


Figure 5.5.2 Admin Dashboard

5.5.3 Admin Manage Course Change

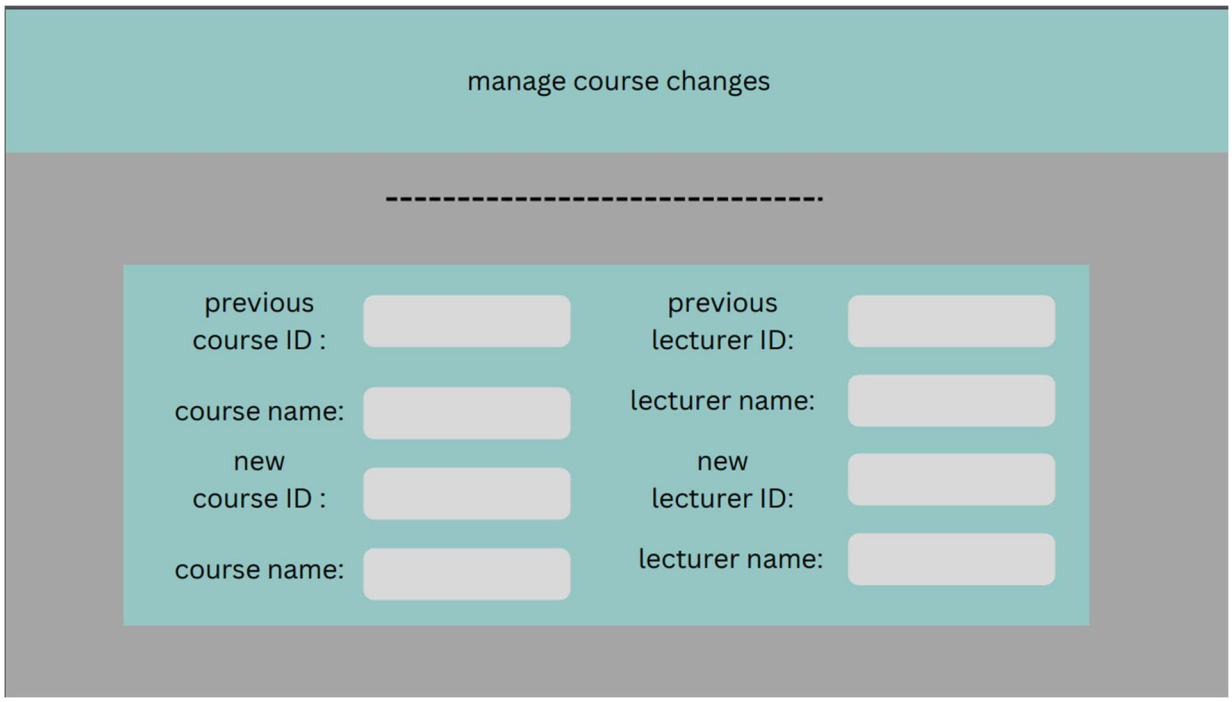


Figure 5.5.3 Admin Manage Course

5.5.4 Admin Manage Lecturer Change

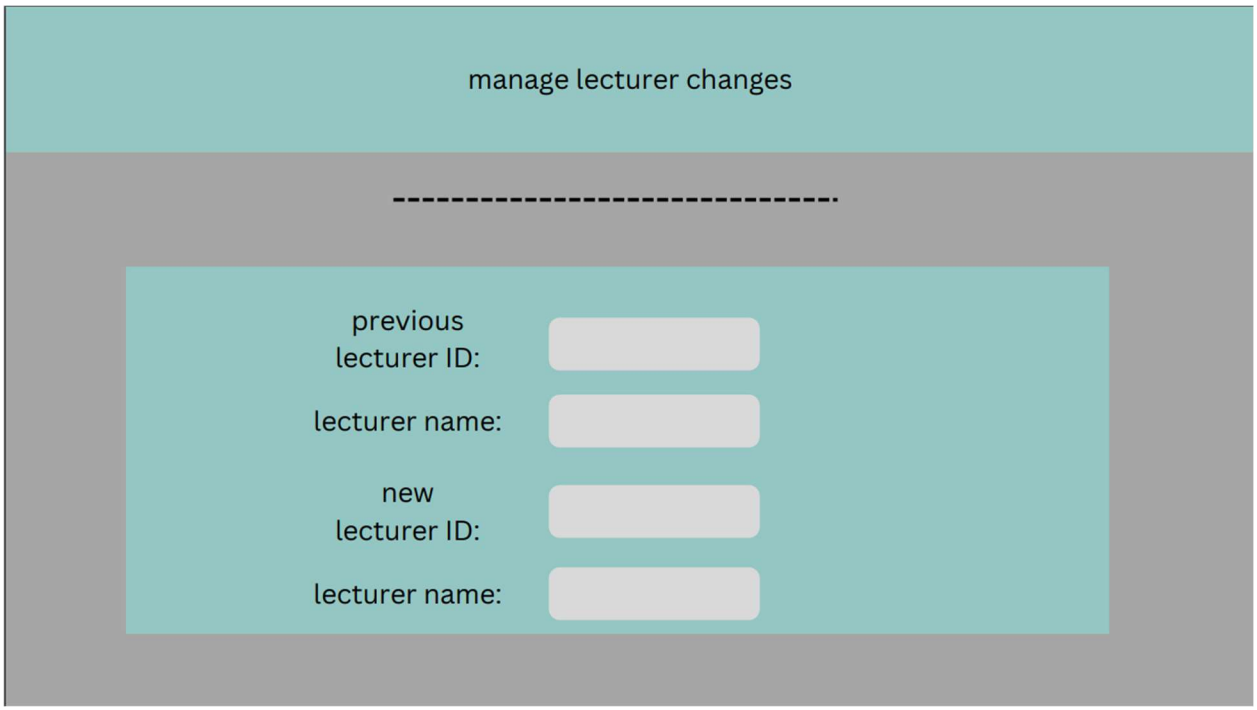


Figure 5.5.4 Admin Manage Lecturer

5.5.5 Admin Manage Classroom

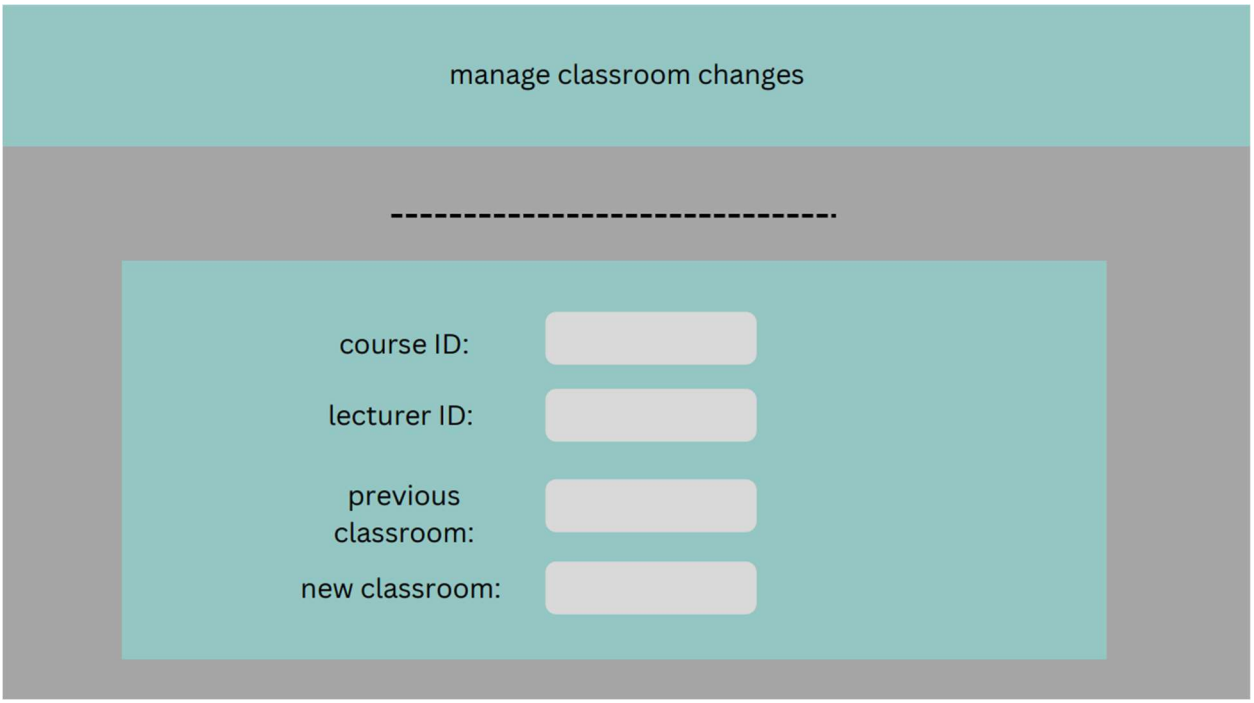


Figure 5.5.5 Admin Manage Classroom

5.5.6 Admin Modify Date & Time Changes

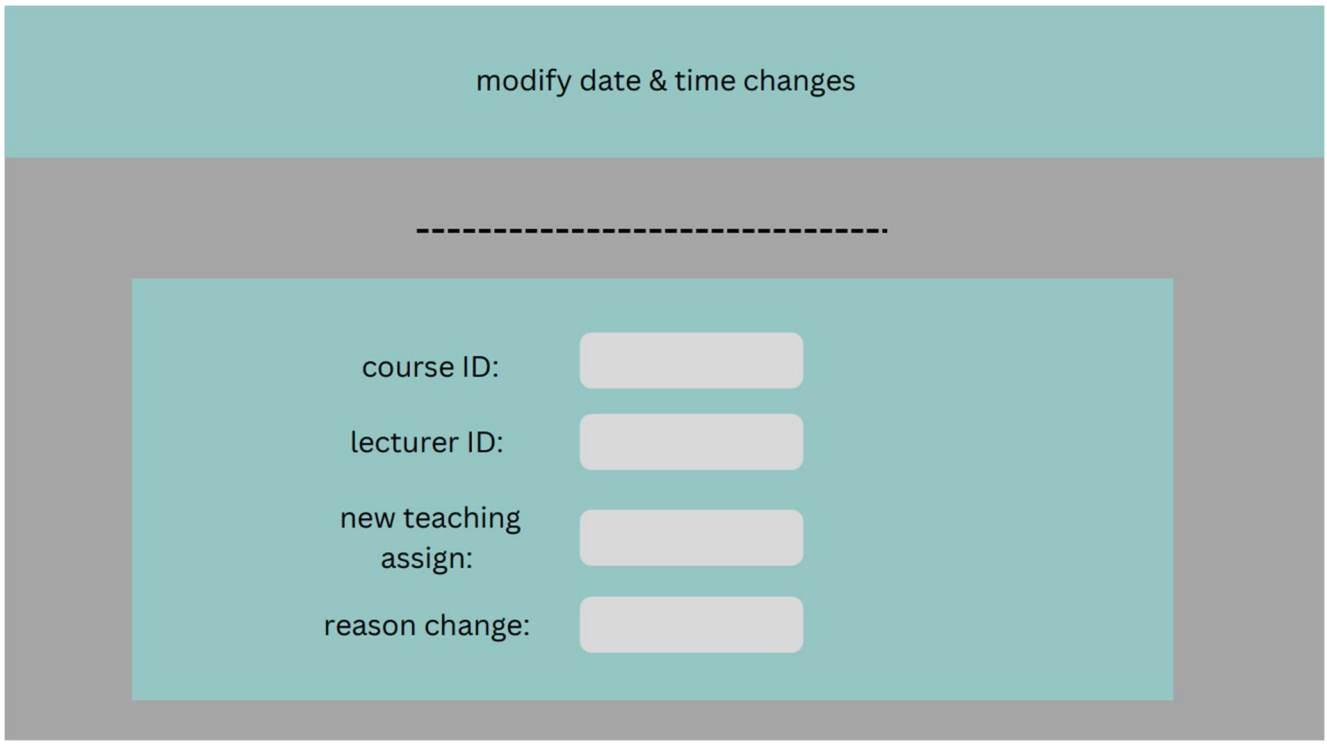


Figure 5.5.6 Admin Modify Date & Time

5.5.7 Login page (Student and Lecturer)

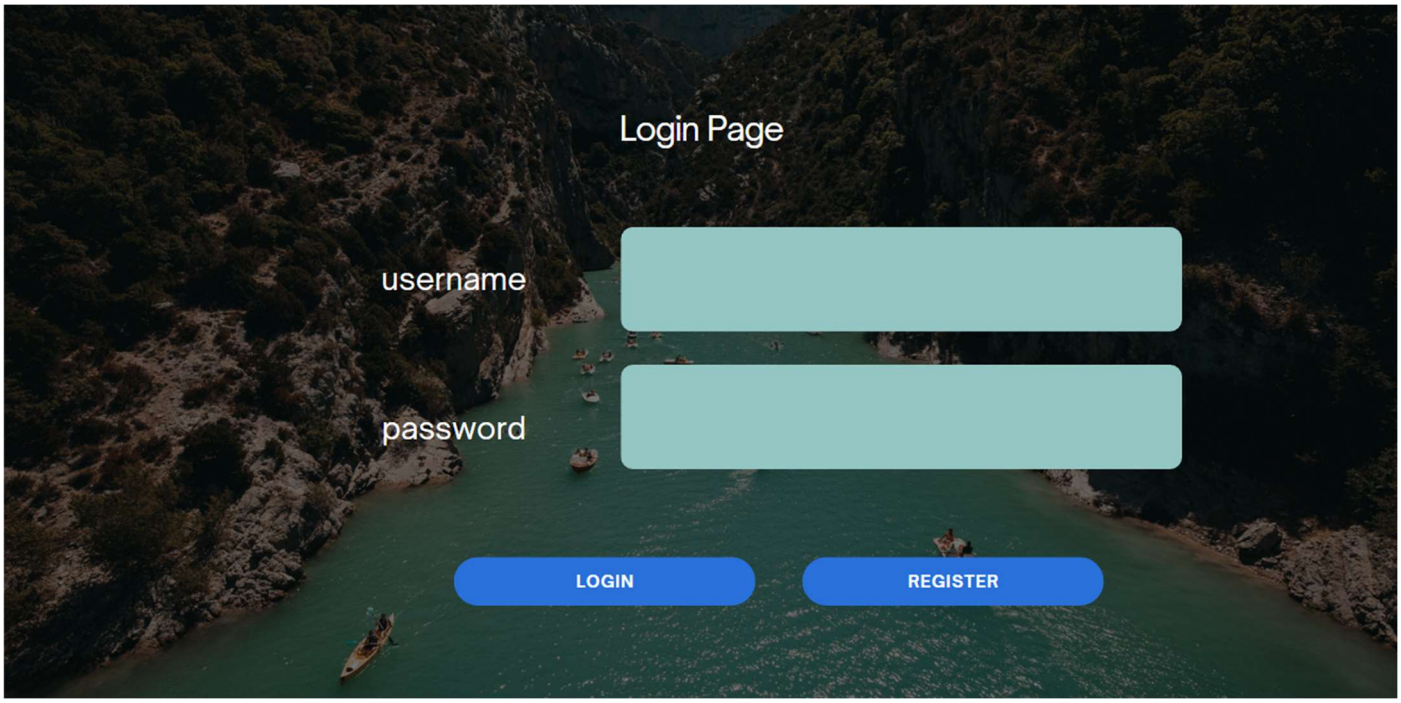


Figure 5.5.7 Login Page

5.5.8 Dashboard (Lecturer)

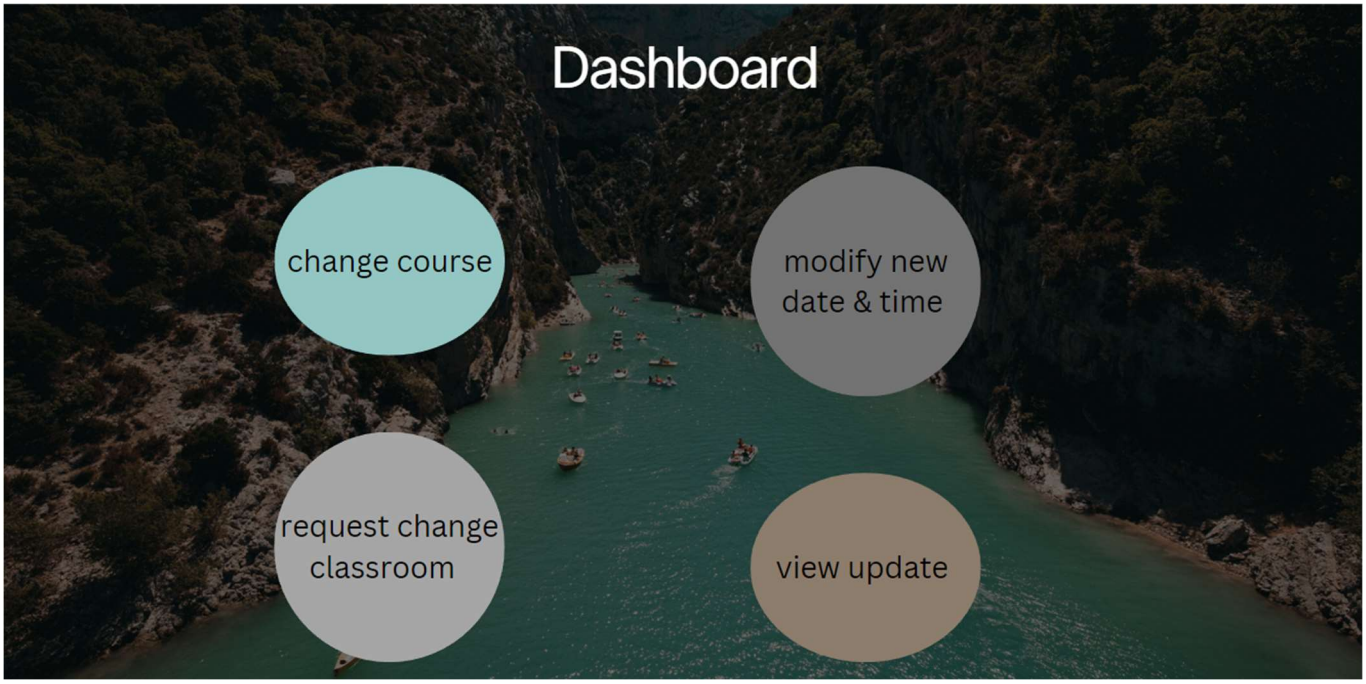


Figure 5.5.8 Dashboard

5.5.9 Change course (lecturer)

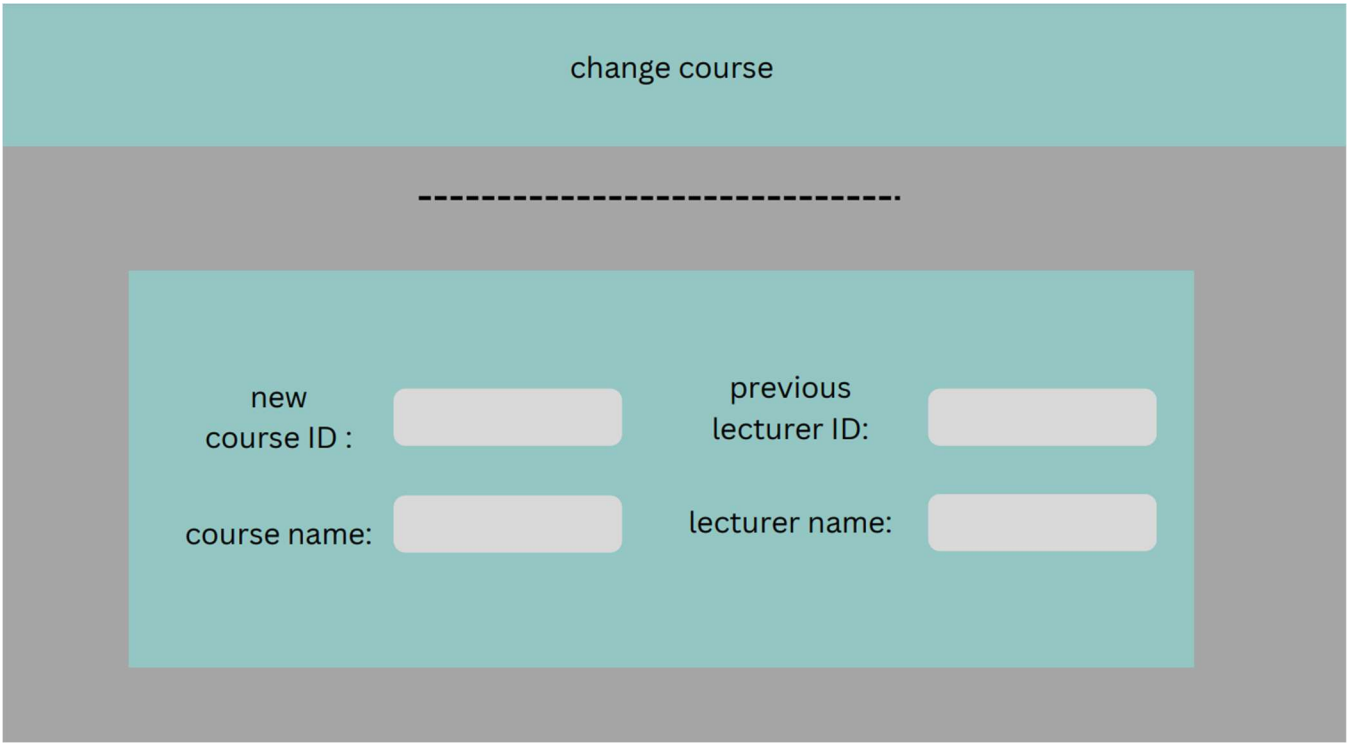


Figure 5.5.9 Change Course

5.5.10 Request Change Classroom

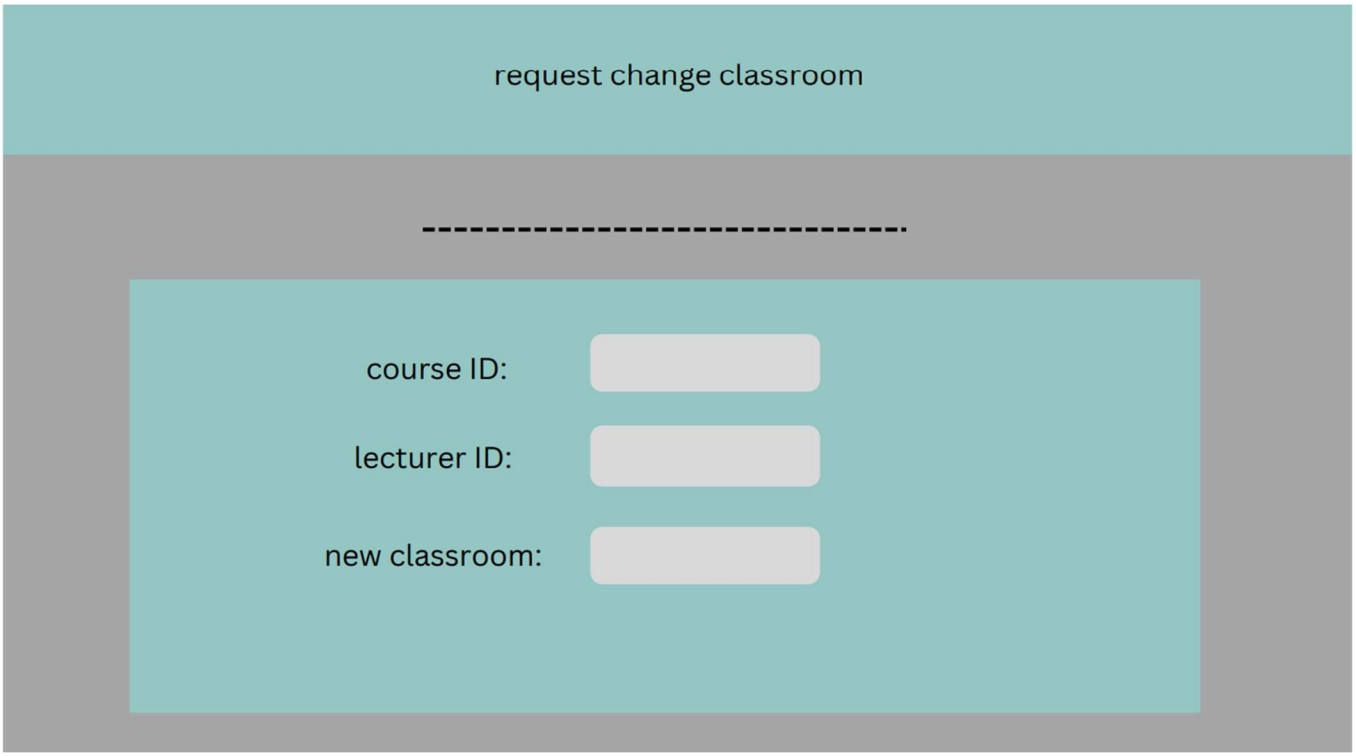


Figure 5.5.10 Request Change Classroom

5.5.11 Modify New Date & Time

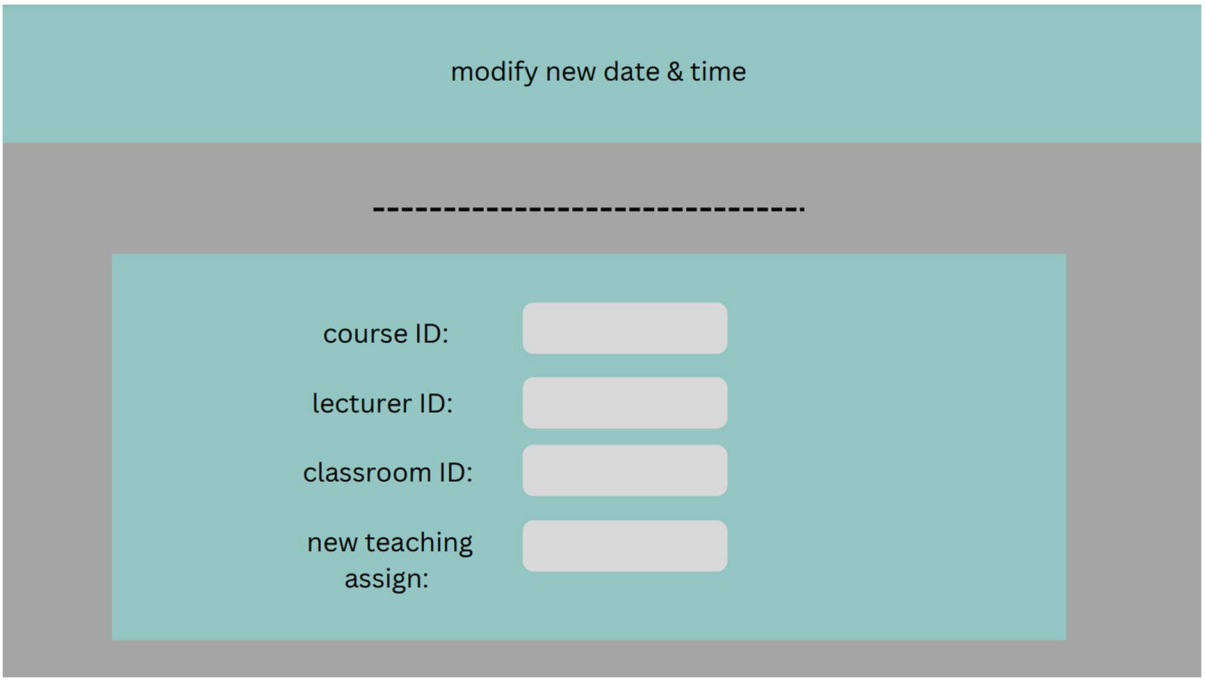


Figure 5.5.11 Modify New Date & Time

5.5.12 Registration (student and lecturer)

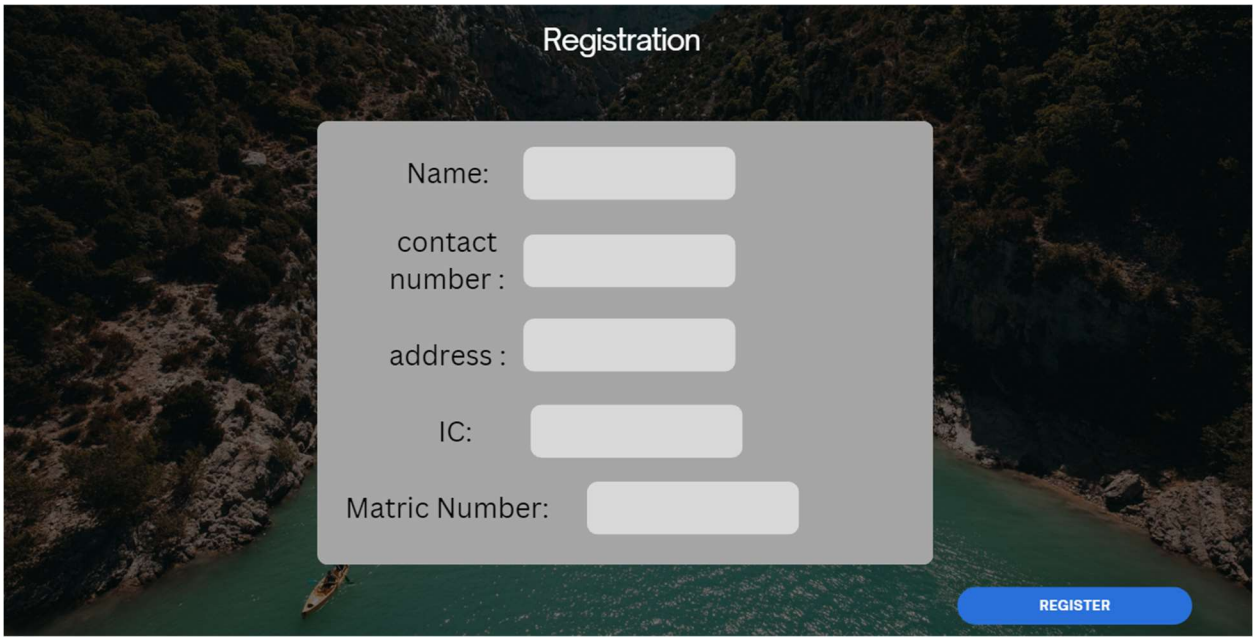


Figure 5.5.12 Registration

5.5.13 View Updated Changes (student and lecturer)

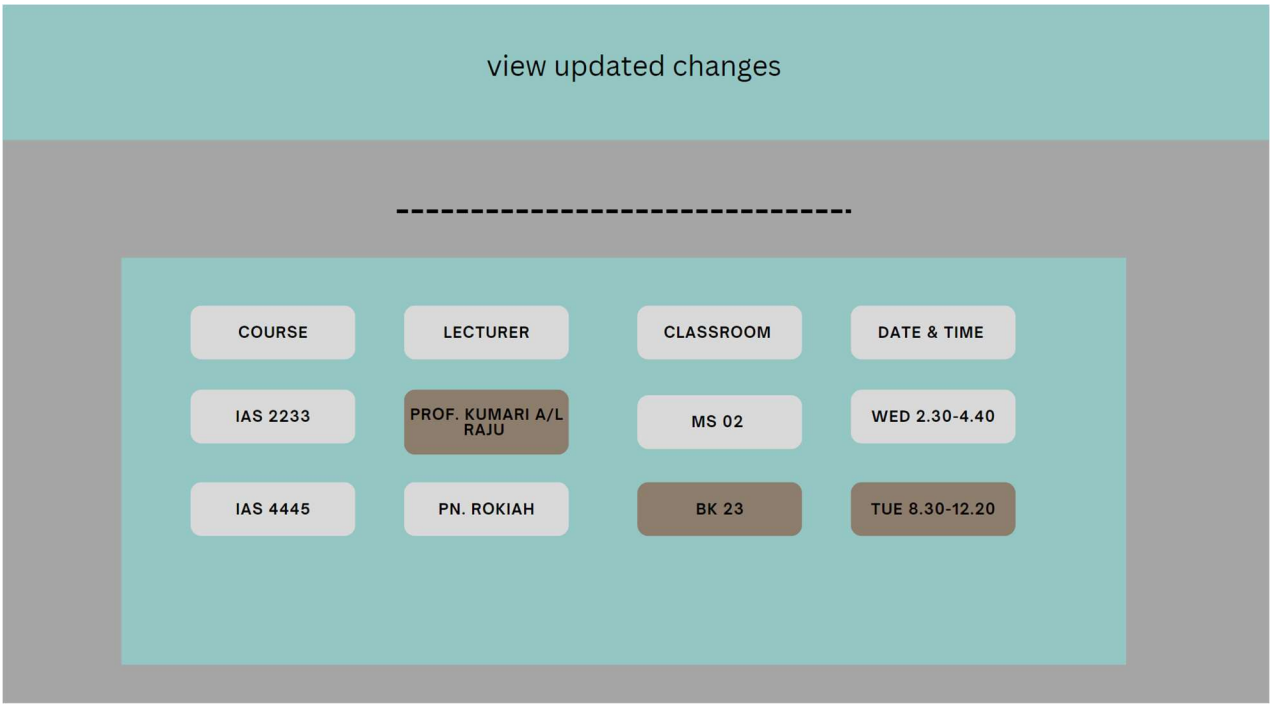


Figure 5.5.13 View Update Changes

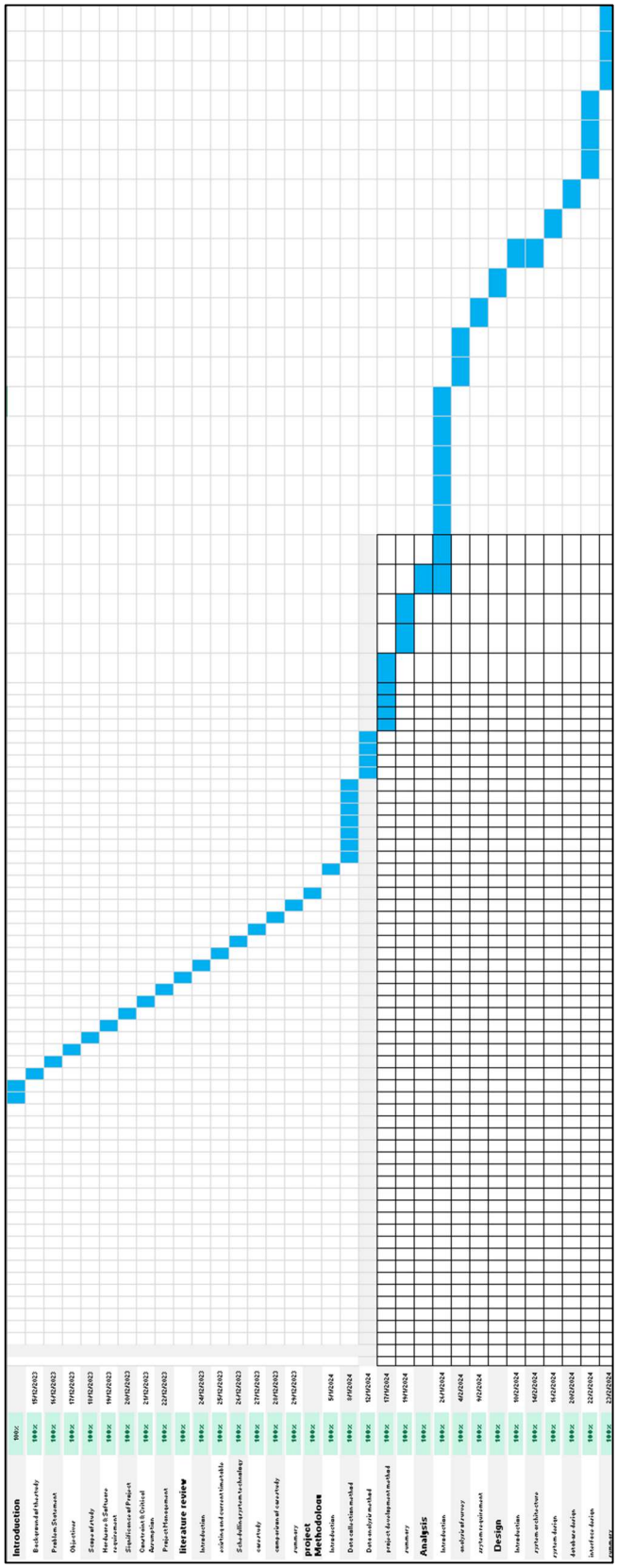
5.6 Summary

This chapter methodically develops the database's basic architecture using entity-relationship modeling, defining entities, attributes, and relationships. With a strong emphasis on one-to-many links, the schema develops as a comprehensive representation of the system's complexities. The resulting class diagram acts as a visual blueprint, clarifying the system's design and allowing for controlled data manipulation via integrated methods. Finally, Chapter 5 emphasizes the importance of database architecture as a driver of efficiency and innovation in information management.

References

1. JF Blakekesly, Keith S. Murray, Frederick H. Wolf and Dagmar Murray. 1998. Academic scheduling

Appendix A



Appendix A Gantt Chart