Woldia University

Institute of Technology

School of Computing

Department of Computer Science

TITLE OF PROJECT: WEB BASED CLASS SCHEDULING SYSTEM FOR WOLDIA UNIVERSITY

A project submitted in partial fulfillment of the requirements for the degree of B.Sc. in Computer Science (Regular)

Submitted by:

Gebreamlak	**Mamo**	**1305915**
Mulugeta	**Abay**	**149650**
Nahom	**Birehane**	**146980**
Samuel	**Molla**	**147162**
Seyoum	**Tarik**	**147234**
Zewdnesh	**Ephrem**	**147645**

Advised by:

Tesfaye B. (MSc)

May, 2025

Woldia, University

As partial fulfilment of the requirements for degree of

Bachelor of Science

In

Computer Science

From

Woldia University

Examiner	**Examiner**
Name_____	Name_____
Signature_____	Signature_____
Designation_____	Designation_____
Department of Computer Science	Department of Computer Science
Date _____	Date _____

DECLARATION

This is to declare that the project entitled web based class scheduling system for WDU is an original work done by us, undersigned students in the Department of Computer Science, School of Computing, Institute of Technology, Woldia University. The reports are based on the project work done entirely by us and not copied from any other source.

Advisor

Tesfaye B. (MSc)

Signature____

Department of Computer Science Date_____

			Signature
Gebreamlak	Mamo	1305915	_____
Mulugeta	Abay	149650	_____
Nahom	Birhane	146980	_____
Samuel	Molla	147162	_____
Seyoum	Tariku	147234	_____
Zewdnesh	Ephrem	147645	_____

ABSTRACT

At Woldia University, the current ASC Timetables software used for class scheduling presents multiple challenges, including a complicated interface, the need for manual data entry by department heads, and no direct access for instructors or students, leading to inefficiencies, delays, and frequent scheduling conflicts. Students and staff currently receive schedules via printouts or Telegram, which are unreliable. To address these issues, a web-based class scheduling system is proposed that is user-friendly, flexible, and installation-free. It will allow instructors to submit availability online, enable department heads to generate and modify conflict-free schedules, and allow students to view or download schedules in real time. Built with modern technologies like Next.js, Tailwind CSS, Node.js, and PostgreSQL, the system will feature role-based access, automated notifications, feedback mechanisms, and report generation capabilities. This digital solution will streamline communication, reduce manual errors, enhance transparency, and significantly improve the scheduling experience for students, instructors, and administrators across all departments.

ACKNOWLEDGMENT

First of all we would like to praise almighty God who helps us to accomplish this documentation successfully. Next, our deepest gratitude goes to our advisor Mr. Tesfaye. B for his excellent advice and active guidance throughout this project.

We would also like to express our thanks to Woldia University, particularly the Department of computer science, for providing us with access to laboratory facilities, free internet services, and helpful materials. Their willingness to assist us through interviews, documents, and patience in answering our many questions played a crucial role in the success of this project.

Table of Contents

Contents

Catalog

List of Figures

List of Tables

List of Acronyms

ASC ---------- Academic scheduling software

CSS ----------Cascading Style Sheets

OOA ----------Object-Oriented Analysis

OOD ----------Object-Oriented Design

OOSE ----------Object-Oriented Software Engineering

UI----------User Interface

UML----------Unified Modeling Language

WDU ----------Woldia University

CHAPTER ONE

1. INTRODUCTION

1.1. Background of the organization

Woldia University is a public higher education institution in Ethiopia, officially established in 2011 by the Council of Ministers Regulation No. 223/2011. The foundation for its construction was laid in 2008 by regional and national education leaders, marking the beginning of a significant development in the country's education sector. Located in Woldia town in the North Wollo Zone of the Amhara Region, the university began its academic activities in December 2011 with an initial enrollment of 599 students across four faculties and 12 departments. Within a year, it expanded to six faculties and 24 departments, reflecting rapid growth and increasing demand for higher education.

Over the years, Woldia University has grown substantially. Today, it consists of two institutes, six colleges, and two schools, operating across three campuses: the main campus in Woldia, and satellite campuses in Mersa and Lalibela. The university now serves over 12,000 students and employs more than 2,300 academic and administrative staff members. It has identified key focus areas such as tourism, agriculture, technology, and health, aligning its programs with national development goals.

Woldia University is committed to innovation and sustainability, exemplified by its partnership to establish a solar photovoltaic mini-grid project with a capacity of up to 100 megawatts. This initiative aims to supply stable, renewable energy to the university and surrounding communities and supports plans for future expansion, including a public-private medical school.

The university’s mission is to produce competent graduates, conduct research that solves real-world problems, and provide meaningful community services. Its vision is to become one of Ethiopia’s top five universities in teaching and research excellence by 2022 E.C. Woldia University is guided by core values including quality, teamwork, customer focus,

inclusiveness, and competitiveness. Strategically located at a major crossroads, the university plays a crucial role in the educational and socio-economic development of the region and the country at large.

1.2. Background of the project

The Online Class Scheduling System is a project designed to improve how academic schedules are managed. Currently, institutions rely on ASC software systems for scheduling classes. While these software has several problems, including difficulty for students and teachers to easily view schedules or receive notifications. It does not update in real-time, which makes managing last-minute changes challenging. Additionally, the software lacks good online access, making it hard for users to check schedules across different devices. The need for manual updates further leads to delays and inefficiencies.

The new system will replace the ASC software systems process with an online solution. It will make scheduling faster, reduce errors, and allow students and teachers to view their schedules anytime, from anywhere. By using technology, the system will help institutions use their resources better, improve transparency, and make the academic experience smoother for everyone. Key Processes of the System will handle the following tasks:

- Assigning classrooms based on course needs and teacher availability.
- Generate conflict-free class schedules
- Show schedules online in real time for students and teachers.
- Browser access available on desktops without setup or installation.
- Auto detect and fix scheduling conflicts (e.g., two classes in one room)
- Allow admins to create and manage user accounts.

The new system will bring many benefits, including:

- Saves time by automating the entire scheduling process
- Reduces printing costs with digital schedules
- Improves communication with instant updates
- Easy to use for all users.
- Increases efficiency and improves the academic experience.

1.3. Statement of the problem

The current ASC software system used for class scheduling at WDU University has many problems that make slow planning and frustrating. Each department prepares its own schedule without seeing what other departments are doing. This often causes issues like two classes being assigned to the same room or a teacher being scheduled for two classes at the same time. If a change is needed like moving a class to a new day or time staff have to make updates manually, which takes a lot of time and can lead to mistakes. There is no single online place where students, teachers, and staff can check their updated schedules from any device, which makes it hard to stay informed.

Another major issue is the lack of automatic notifications. When a schedule changes, students and instructors don't get alerts, so they might not even know something has been updated. Also, the system does not let users give feedback or ask for changes directly. Instructors and students must go through department heads or admins, which causes delays and extra work. Fixing schedule conflicts is also done by hand and can take a long time. These problems make the system inefficient and stressful for everyone who uses it, especially students, instructors, and scheduling staff.

1.4. Objective of the project

1.4.1. General Objective

The general objective of the project is to develop web based class scheduling system for WDU.

1.4.2. Specific Objective

- To design a user-friendly interface for easy navigation.
- To develop a module for editing class schedules.
- To implement optimization techniques for improved schedule generation.
- To assign accurate classrooms for classes without time conflicts.
- To implement an automated notification system.

- To help department heads manage schedules digitally.
- To enable students to access their department's schedule online.
- To allow students and instructors to interact with the system directly.

1.5. Scope of the Project

The scope of the Online Class Scheduling System defines the specific features and limitations of what the system will do. The system is designed for Woldia University (WDU) and focuses on improving how class schedules are created, accessed, and managed. The scope of the system includes:

- The system will be developed specifically for use by WDU students, instructors, and administrators, and will support scheduling for all departments within the university.
- Notifications and reminders to students and instructors about upcoming classes, with instructors able to request schedule changes for administrator review and approval.
- Students and instructors can view, download (in PDF), and give feedback on their schedules to improve future planning.
- A central dashboard for administrators to monitor and manage scheduling activities in real-time, with archiving of past schedules for future reference and reporting.
- Support for generating automated reports and statistics on class utilization, instructor workloads, and room occupancy.
- The system will be accessible through modern web browser without the need for installation, ensuring full compatibility with all desktop computers.

1.6. Methodology of the Project

1.6.1. Data gathering methodology

On this project, we use the following methods to gather data:

- **Observation**: We closely observed how the current ASC scheduling software is being used at Woldia University. This helped us see step by step how department heads create class schedules, assign teachers, and manage classroom availability. During our

observation, we noticed that the system requires a lot of manual data entry, which takes time and can lead to mistakes. We also saw that instructors and students do not interact with the system directly—they receive their schedules later through printouts or Telegram groups. By watching how the process works in real situations, we were able to clearly understand the difficulties users face, such as scheduling conflicts, lack of real-time updates, and poor communication. This observation helped us identify what needs to be improved in the new system.

- **Interview**: To better understand how class scheduling is currently done at Woldia University, we spoke directly with the department heads of Computer Science and Mathematics. These interviews gave us a clear, real-world picture of the challenges they face and how the existing system works on a daily basis. Talking with the people who actually prepare the schedules helped us design a system that truly fits their needs instead of making assumptions.

We chose the interview method because it allowed us to ask detailed questions, listen to their experiences, and get honest feedback. This approach gave us valuable insights that we wouldn't have been able to get through observation or documents alone. The interviews were especially helpful for the following reasons:

- ✓ **To understand the actual process:** We wanted to learn how department heads currently collect teacher availability, assign classrooms, and organize class times.
- ✓ **To discover the main problems:** They told us about issues like overlapping classes, double-booked rooms, and how time-consuming it is to make changes manually.
- ✓ **To gather useful information for system development:** Their input helped us understand what features the new system should have, such as conflict detection, easier updates, and online access for students and instructors.

- **Document Analysis**: We reviewed existing class schedules and related documents from various departments to understand how timetables were prepared in the current system. By going through these documents, we were able to see the structure of the schedules, how classes were assigned to rooms and instructors, and how time slots were distributed.

This helped us identify both the strengths and weaknesses of the current scheduling method. For example, we noticed that while the documents showed clear organization in some areas, there were also cases of overlapping classes and repeated manual corrections. Analyzing these records gave us useful information about the challenges of the existing system and helped us decide what features and improvements should be included in the new system.

1.6.2. System Analysis and Design methodology

We used the Object-Oriented Software Engineering (OOSE) methodology to develop our class scheduling system. This approach is widely used in modern software development because it relies on object-oriented principles and visual modeling to represent system components in a clear, modular, and maintainable way. OOSE promotes better communication among team members and stakeholders, increases system reliability, and improves software quality through the use of models such as use case diagrams, class diagrams, and sequence diagrams. Specifically, we applied Object-Oriented Analysis (OOA) to capture system requirements and represent them as interacting objects. This helped us organize the system logically and design it in a way that mirrors real-world scheduling activities.

OOA has several advantages. It helps address complex problem domains, ensures consistency across all stages of development, and promotes the use of standard methodologies such as Object-Oriented Design (OOD), design patterns, and the Unified Modeling Language (UML). It also supports code reusability and the clear representation of shared behaviors and structures among system components.

However, relying only on OOA is not always sufficient, especially when building a system that must address user experience, business logic, and real-time feedback. To complement OOA, we also used Structured Analysis to understand the flow of data and processes before converting them into objects. This method helped us break down the scheduling tasks into simpler, functional parts and identify key inputs, outputs, and processes early in the project.

Additionally, we used a Prototyping technique during the design phase. By creating interactive mockups of the user interface, we were able to gather feedback from department heads, instructors, and students. This feedback helped us refine system features and ensure that the final product would meet the users' actual needs. Furthermore, we employed Use Case-Driven Modeling, which allowed us to focus on the interactions between users and the system. This ensured that all functionalities were user-centered and aligned with real usage scenarios.

By combining Object-Oriented Analysis with Structured Analysis, Prototyping, and Use Case Modeling, we developed a scheduling system that is not only technically robust but also practical, user-friendly, and aligned with the operational needs of Woldia University.

1.6.3. System Development Model

We chose the Iterative System Development Model for our project because it allows us to build the system step-by-step by developing small parts of the project in repeated cycles. In each iteration, we design, develop, and test a portion of the scheduling system, allowing us to identify problems early and make improvements gradually. This method helps reduce errors, manage risks better, and ensures that each part of the system is functional before moving to the next. It is a practical approach for complex systems like this propose system, where understanding and refining each part gradually improves the final outcome.

The model also supports continuous feedback from users such as students, instructors, and administrators after each version. This means we can quickly respond to user needs and make changes without starting from scratch. As the project progresses, the system becomes more complete with each iteration, and by the final phase, we have a reliable and user-accepted product. The Iterative model is well-suited for our class scheduling system because it supports flexibility, step-by-step progress, and continuous improvement.

1.6.4. System Testing Methodology

Unit Testing

It involves testing individual components or modules of the class scheduling system in isolation to ensure they function correctly. For example, we test functions like scheduling a class or validating user inputs to catch errors early. If a problem occurs, it's easier to pinpoint and fix since each unit is tested separately.

Integration Testing

Integration testing checks if different modules of the system work together as expected after unit testing is complete. It verifies that components, like the UI, backend APIs, and database, integrate smoothly to meet functional requirements. It ensures that a schedule created in the UI is correctly saved to the database.

System Testing

System testing evaluates the entire class scheduling system as a whole by involving external users to test its functionality and usability. It collects feedback on how users interact with features like viewing or editing schedules to ensure the system meets their needs. This testing confirms the system is reliable, user-friendly.

1.6.5. Development Environment and Programming Tools

Table 1.1: Software Development Tools

Category	Technology	Purpose
Frontend Development	next.js, tailwindCSS	Building the user interface
Backend Development	Node.js with express	Handling server-side logic and API development
Database	PostgreSQL	Storing and managing data (e.g. schedules, user accounts, resources).
Code Editor	Visual Studio Code	Writing, debugging, and managing code
UML Design Tools	Draw.io, Edraw Max	Drawing different diagrams for system design and documentation
Documentation	Microsoft Word	Writing project documentation and reports
Presentation	Microsoft PowerPoint	Creating presentations for project updates, pitches, or briefings

Table 1.2: Hardware Development Tools

Category	Device	Purpose
Development Machines	Laptop/Desktop	Writing code and running the development environment
Networking	Ethernet Cable	Ensuring stable internet Connectivity for development
Storage Devices	USB Flash Drive	Helps transfer project files easily between different devices for sharing purposes.
Printer	Shared Printer	Printing reports, schedules, and other documents for testing purposes.

1.7. Significance of the Project

This project significantly contributes to the operational efficiency of Woldia University by improving how academic schedules are managed.

The Significance of the project is listed below: -

- Helps the university better achieve its academic and strategic goals.
- Provides a modern, user-friendly, and fast service accessible from anywhere.
- Avoids class scheduling time clashes automatically.
- Reduces misunderstandings between instructors, students, and administrators.
- Reduces the time and effort required to prepare and manage class schedules.
- Automatically detects and avoids schedule clashes for rooms, instructors, and students.
- Allows students and instructors to give feedback or raise complaints directly through the system.
- Prevents loss of schedules and scattered information by using one centralized platform.
- Helps department heads and administrators make better decisions using reports and usage data.
- Allows easier updates across multiple departments without restarting the whole system.

Target beneficiaries of Class scheduling Management System are: -

- **Students:** Easy access to real-time class schedules from any device without needing printed copies or Telegram updates. They can download timetables, receive instant notifications about changes, and submit feedback directly through the system.
- **Lecturers:** Can view their assigned teaching schedules, provide availability, and request schedule changes online. This streamlines communication with department heads and helps avoid conflicts in teaching assignments.
- **Department Heads:** Gain tools to generate, review, approve, and adjust department schedules efficiently. They can detect and resolve conflicts, manage instructor loads, and receive feedback directly from users for better planning.

1.8. Feasibility study

1.8.1. Technical Feasibility

Technical feasibility from the user's perspective, the system is designed to be easily accessible through standard web browsers without requiring users to install additional software. The interface is simple, responsive, and compatible with desktop devices, ensuring students, instructors, and administrators can use the system with minimal training. The design considers varying levels of digital literacy by using clear navigation, simple forms, and helpful tool-tips. By providing role-based dashboards and user-specific access, the system reduces complexity and ensures that each user only sees what is relevant to them. Regular user testing and feedback during development will also help identify and fix usability issues early, making sure the final product is both functional and user-friendly.

1.8.2. Operational Feasibility

To make sure the system is operationally feasible, it will be designed to match the skills and needs of its users. Most students, instructors, and staff already have basic computer knowledge, so the system will use a familiar and easy to understand layout. The main features will be clearly labeled, and users will only see tools and options that are meant for their role, which helps avoid confusion and misuse.

The system will also be flexible, meaning it can work well for different types of users with different schedules and responsibilities. Simple instructions, help messages, and guided steps will be included to support users as they navigate the system. In addition, we will gather feedback from real users during testing to improve how the system works in everyday situations. This will help make sure that the final version is practical, easy to use, and fits well into the normal operations of Woldia University.

1.8.3. Economic Feasibility

The proposed online scheduling system fully replaces the manual and semi-automated processes currently used at Woldia University. By digitizing data entry, schedule generation,

and distribution, the system improves efficiency and significantly reduces recurring operational costs. Below are some key economic benefits explained with practical examples:

Reduced Printing and Paper Costs

In the current system, each department prints class schedules multiple times per term.

Example: If 10 departments each print 100 schedule sheets per term at 5 birr per page, and there are 3 terms per year.

10 × 100 × 5 × 3 = 15,000 ETB/year saved by going fully digital.

No Need for Paid Licensing or Installations

The system is developed using free and open-source technologies, requiring no paid software licenses.

Example: Commercial scheduling software may cost thousands of birr per year in licensing this cost is fully avoided.

1.8.4. Legal Feasibility

Legal feasibility ensures that the proposed class scheduling system complies with all relevant laws, regulations, and institutional policies at both the university and national levels. This includes adhering to data privacy and protection laws, such as ensuring that personal information of students, instructors, and staff (e.g., names, schedules, contact details) is securely stored and not shared without proper authorization. The system must also follow university IT and data management policies, including rules about who can access or modify scheduling data and how user accounts are managed.

Furthermore, since the system deals with academic records and internal operations, it must comply with academic regulations set by the Ministry of Education and Woldia University's internal guidelines. For instance, any changes to class schedules must follow established approval procedures, and the system must ensure that only authorized users (e.g.,

department heads, admins) can make such changes. It should also respect user consent, especially when sending notifications or storing feedback.

By ensuring legal feasibility, we prevent potential issues such as unauthorized access, data misuse, and conflicts with institutional or national laws. This not only protects the university from legal liability but also builds trust among users, as they know their data and rights are respected within the system.

1.8.5. Scheduling Feasibility

Schedule feasibility is making sure whether potential time frames and completion data can be meeting or not. The project team members discussed on it to work hardly and also estimate project to be completed on time without any delay. For organization replacing current systems with automated one is advisable for better performance. To replace the current system, we should make the whole plan before starting to develop a computerized system. In order to develop this project, we have planned all the necessary tasks. In order to accomplish time effectively and efficiently, we used the schedule time table in all phases of project we do. It's important for group teams in case of managing time to complete this project as scheduled.

Table 1.3: Timeline Table

No	Activity	Start	Finish	Year 2025 G.C							
				Month							
				Mar	Apr	May	Jun	July	Aug	Sep	Oct
1	Proposal	01/03/2025	13/03/2025								
2	Requirement Gathering	22/03/2025	15/04/2025								
3	System Analysis	16/04/2025	24/05/2025								
4	System Design	20/04/2025	01/05/2025								
5	Implementation	17/05/2025	22/09/2025								
6	Testing	19/09/2025	29/10/2025								

1.9. Team Composition

The project built through collaborative teamwork. Each phase involves all team members to ensure comprehensive development and successful completion.

Table 1.4: Team Composition

Phases	Work Breakdown	Deliverables
Requirement Gathering	All Members	Collects data and information required
System Analysis	All Members	Determines what the system does
System Design	All Members	Models the system
Implementation	All Members	Working code modules
Testing	All Members	Verifies that each module and the overall system work as expected through unit, integration, and system testing
Documentation	All Members	Team works together to document system processes and designs, ensuring proper knowledge transfer and future reference

CHAPTER TWO

2. EXISTING SYSTEM

2.1. Introduction

Academic scheduling software (ASC) is a specialized offline application that must be installed on a local computer or network. It automates the complex process of organizing class timetables by considering constraints such as room availability, faculty schedules, and student enrollments. The existing system typically relies on rule-based algorithms to generate conflict-free schedules while optimizing resource utilization. This section explores the operational workflow of ASC software, including data input, scheduling logic, and output generation. Subsequent subsections will analyze its key functionalities, limitations, and practical challenges in implementation.

2.2. Description of the existing system

At Woldia University, class schedules are currently created using a software program called ASC Timetables. This tool helps organize classrooms and match courses with available times. However, it is not easy to use. Only department heads can operate the system. Teachers do not use it directly. Instead, they send their available times and preferences through paper forms, emails, or by talking to the department heads. Then, staff have to manually enter this information into the system, which takes time and can lead to mistakes.

While the ASC software does help reduce some schedule conflicts, it doesn't allow full customization based on each teacher's needs or unexpected changes. Many times, the schedules it creates still need to be fixed manually to fit real situations. One big problem is that departments work separately, with no shared system to coordinate. This can lead to double-booked rooms or overlapping class times because there's no central place where everyone can see what's happening across the university.

Also, students and teachers don't get direct access to the system. Final schedules are usually printed or shared in Telegram groups, which can be confusing or outdated. There's no

official online space where users can check their schedule, give feedback, or request changes. Because of these issues, schedule management becomes slow, confusing, and frustrating. That's why the university needs a new, web-based system that is easier to use, lets everyone stay updated, and connects all departments in one place.

2.3. Limitation of the Existing System

- Lack of real time availability integration.
- There is no built-in support for feedback, conflict reporting, or schedule requests.
- It doesn't automatically send updates when changes are made to the schedule.
- Requires manual work to fix automated schedules.
- No centralized web platform.
- Need desktop installation

2.4. Model of the Existing System

2.4.1. Actor

The core contributors in the existing model include:

- **Department Head:** Directly interacts with the ASC (Automated Scheduling and Conflict-resolution) software. They gather class schedules, course details, and room requirements from instructor and input this information into the system. They are also responsible for adjusting entries when conflicts are identified.
- **Instructor:** Does not directly interact with the system. Instead, instructor submit their course schedules, preferred times, and availability to the Department Head through forms or verbal communication. They receive finalized schedules through printouts or Telegram messages.
- **Student:** Has no direct interaction with the scheduling system. Students simply receive the final class schedules after they are approved, distributed via printed documents or Telegram channels.

2.4.2. Essential use-case

The current class schedule system includes several participants, each with defined tasks and responsibilities. The class schedule system includes several participants, each with defined tasks and responsibilities. Those are: -

- **Department Head:** Gathers teaching schedules and course information from instructors within their department. Inputs all necessary data into the ASC software and handles department-level schedule adjustments to resolve internal conflicts.
- **Instructor:** Responsible for providing their availability, course assign, and preferred teaching times to the Department Head. They do not use the scheduling system directly but are notified of the final schedules once they are approved.
- **Student:** Receives finalized class schedules after administrative processing is complete. Students have no involvement in the scheduling process and cannot make change requests.

2.5. Business rule

- Every user must log in with a valid username and password to access role-specific features in the system.
- Only admin have the authority to create, update, or delete user accounts (students, instructors, and department heads).
- Each course, instructor, room, and department must have a unique ID to ensure accurate scheduling and data integrity.
- The system requires key data inputs such as course lists, instructor assignments, room details, year, section, and student numbers before schedule generation.
- Only department heads are allowed to generate and approve class schedules for their respective departments.
- Department heads must assign instructors to sections before generating the schedule.
- All schedule change requests from instructors must be reviewed and approved by the department head before being applied.

- Notifications about schedule changes are sent only to the users affected by the update (e.g., assigned instructors and students).
- Class schedules must be prepared and finalized before the beginning of each semester to ensure timely access for all users.
- Feedback or complaints submitted by students or instructors through the system will only be visible to department heads and admin for review and response, ensuring privacy and proper issue resolution.

CHAPTER THREE

3. PROPOSED SYSTEM

3.1. Overview of the proposed system

After gathering requirements and identifying problems in the existing system, our project developed a solution that both addresses current issues and adds valuable new functionality for users. The proposed web-based Class Scheduling System offers a comprehensive solution to the limitations of the current ASC software by introducing a hierarchical, automated approach with enhanced user accessibility. Key improvements include complete process automation, role- based access control, student centric design features, global scheduling coordination, and integrated workflows. The solution maintains the existing system's strengths while addressing its weaknesses through intelligent scheduling algorithms, transparent processes, and improved accessibility across all user levels, particularly benefiting both students and teachers who gain direct schedule viewing and management capabilities.

Following sections will detail functional requirements, non-functional requirements, system requirements, and user workflows to demonstrate how this proposed system overcomes current challenges while delivering additional functionality that significantly enhances the scheduling experience for all stakeholders.

3.2. Requirement specification

3.2.1. Functional Requirement

- Manage User Role: Admins can create, modify, or deactivate user accounts and assign roles/permissions.
- Automated Schedule Generation**:** Department Heads can generate optimized timetables based on instructor availability, room capacity, and course requirements.
- Download Schedules**:** Students and staff can download their schedules as easy to read PDF files to save or print.

- User Feedback**:** stakeholders can send feedback or report problems about their schedules, which will go straight to the right staff members for quick follow-up.
- Real Time Notifications**:** The system will send timely alerts to users when schedules change or important updates occur.
- Generation Report**:** Department heads and admins can generate reports summarizing schedules, room usage, or instructor allocation for analysis and decision-making.
- Schedule Conflict Alerts: When submitting schedule changes, users will be immediately notified if their request change conflicts with existing schedules.
- Schedule View: Students can view their class schedules Instructors can view their assigned teaching schedules and availability.
- Instructor Assigning Registration: Allows admin to efficiently assign courses to instructors based on their expertise, availability, and workload, ensuring conflict-free schedules.
- Course Registration: Streamlines course management, enabling admin to create and manage courses, students to enroll seamlessly, and instructors to access course rosters.
- Room Registration: Facilitates the organized registration and management of buildings and rooms, tracking their availability, capacity, and maintenance status.
- Class Section Registration: Enables admin to create and manage individual class sections, associating them with specific courses, instructors, and schedules for structured and conflict-free organization.
- Batch Registration: Streamlines the process of defining academic batches by grouping students based on their year of admission, program, and specialization, linking them to relevant schedules and resources for better coordination.
- Building Registration: Allows administrators to register and manage details of buildings, including their building number, and associated rooms, ensuring a structured overview of available facilities for scheduling and resource allocation.

3.2.2. Non-functional requirement

Nonfunctional requirements describe aspects of the system that are not directly related to the functional behavior of the system. Nonfunctional requirements include a broad variety of

requirements that apply to many different aspects of the system, from usability to performance.

- **Usability** The system will offer a user-friendly interface with simple and intuitive navigation. It's designed so users can easily learn how to interact with the system, increasing their efficiency.
- **Performance:** The system will use efficient algorithms to handle scheduling tasks quickly and accurately. It's designed to provide fast response times and high processing speed while ensuring the information is always valid.
- **Error Handling:** The systems can able handle exceptions that may happen while user uses the system. It handles exceptions of data duplication to save memory space error related to scheduling such as assigning more than one class for instructor at a time.
- **Reliability:** We can say our system is reliable as we mentioned before the system will consistently perform its intended function. Unless there is an internet connection problem occurs our system is available at any time. In the quality assessing the users will be involved by feedback mechanism in which they can give comment on the system. Feather change or to maintain the system if error may happen on the system.
- **Security:** The system will have strong security features when a registered user wants to log in, they'll first select their role (Admin, Scheduler, or User), and then enter their username and password. This approach prevents unauthorized access and keeps the system secure.
- **Portability:** To use the system user should to have any browser on his computer that enables his/her to connect to the internet.
- **Modifiability** Authorized users will be able to modify the system when needed, making it flexible and adaptable to future changes.

3.2.3. System Requirement

The system must fulfill the following core functionality and infrastructure requirements to meet user and institutional needs:

- The system shall operate as a web-based application, accessible via modern web browsers.
- It shall support multi-user access with role-based authentication (admin, instructor and student).
- A central server is required to host the application and database for schedule management and user data.
- The system must be compatible with standard internet-enabled devices.
- It must ensure secure login, password encryption, and optional notifications.
- The system should be scalable, supporting multiple departments, instructors, and concurrent users.
- It must provide real-time schedule generation, conflict detection, and error handling features.
- It shall allow PDF export for offline access.

3.3. User interface specification and description

The user interface (UI) is designed to be intuitive, user-friendly, and accessible through web browser such as Google Chrome, Mozilla Firefox or Microsoft Edge. The UI adapts to the role of the user (student, instructor, department head/admin) and allows easy interaction with the class scheduling functionalities.

- Login Interface: All users will log in securely using a username and password to access their respective role-based functionalities.
- Dashboard Interface: This designed to show each user only what they need, in a simple and organized way. Students and instructors can easily view their class schedules, upcoming sessions, and receive updates right from their dashboard. For department heads and administrators, the dashboard provides tools to manage schedules, monitor classrooms, and track important data like the number of classes, batches, and rooms.
- Conflict Notification Interface: Real-time alerts will notify users of scheduling conflicts, such as overlapping classes or double assigned rooms/instructors.
- Student View Interface: Students can search for, view, download, or print their class schedules in a list format.

- Printable Schedule Output: Users can export the schedule as a PDF or print it directly from the system.
- Feedback and Compliance Interface**:** The system provides a dedicated feedback section where users can easily submit comments, concerns, or complaints about their schedules. All submissions are securely stored and routed to the appropriate administrators for timely review and response. This feature ensures the system complies with user rights to voice concerns and supports continuous improvement based on user input.
- Help/Support Menu: A help section includes FAQs, guides, and tips to assist users in navigating and using the system effectively.

CHAPTER FOUR

4. SYSTEM MODELING

4.1. Introduction

This chapter presents a general overview of the system model using UML, which helps visually describe how the system works. It includes the use case model to show interactions between users and the system, the object model to represent the structure and relationships of system components, and the dynamic model to explain how the system behaves over time. These models together give a clear picture of how the system functions, making it easier to design and understand.

4.2. Functional/Use-case model

4.2.1. Use Case Diagram

The use case description illustrates how different users interact with the scheduling system to manage class schedules efficiently.

Actors

- **Student:** Views class schedules, receives schedule notifications, downloads schedules for offline access, and requests opinions or raises concerns about their schedules for review by academic staff.
- **Instructor:** Views assigned teaching schedules, requests changes if necessary, and provides availability information for scheduling.
- **Department Head:** Generates departmental schedules, approves schedules, resolves conflicts, and manages overall academic timetable.
- **Admin:** Manages user roles and permissions, configures system settings, updates schedule data, and oversees overall system management

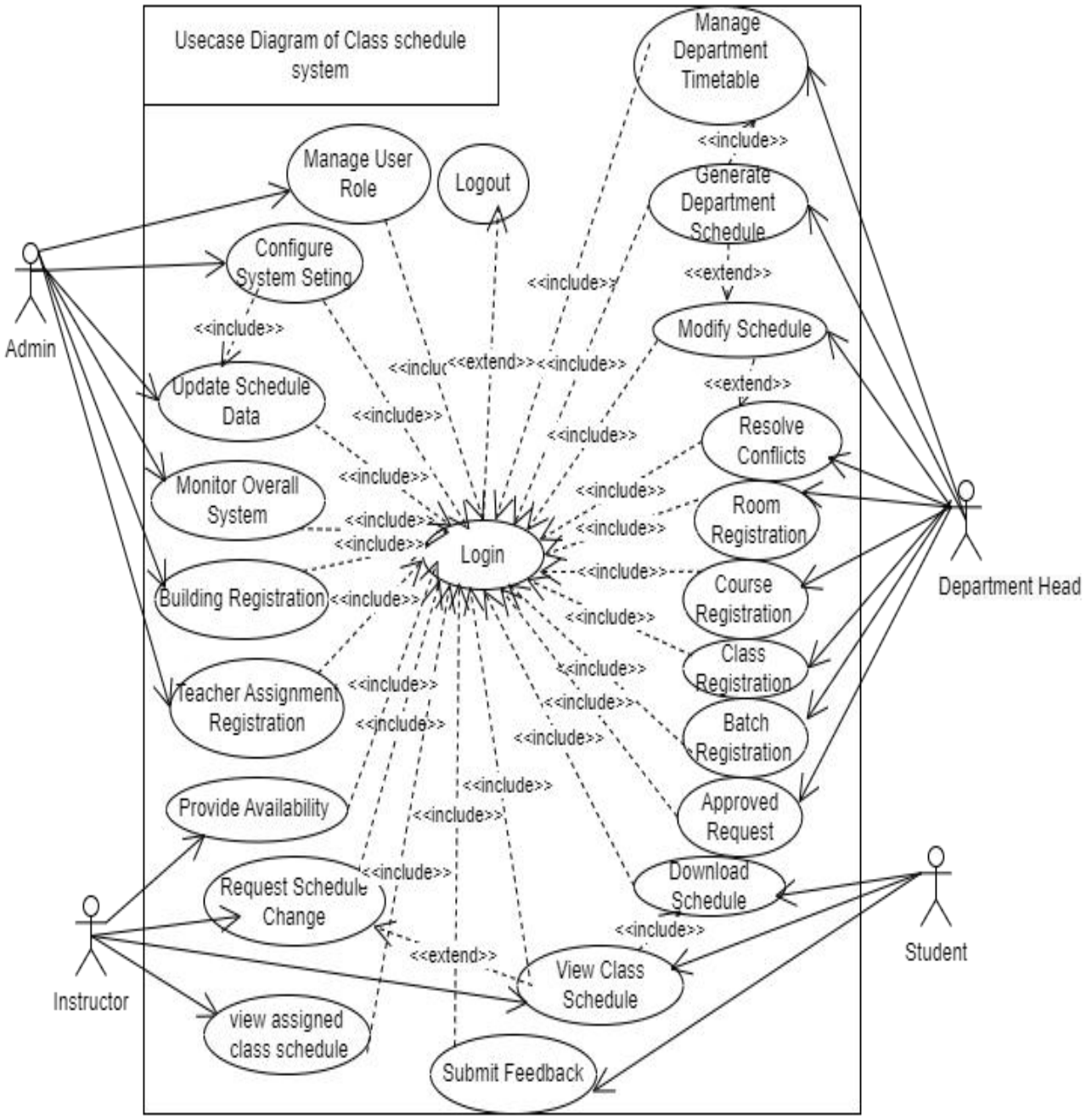

Figure 4.1: Use Case Diagram

Table 4.1 Use Case Description for Login

Field	Description	
Use Case ID	UCID-01	
Use Case Name	Login	
Actors	Student, Instructor, Department Head, Admin	
Description	This use case allows any actor (Student, Instructor, Department Head, or Admin) to log into the system by entering their username and Password. The system verifies the credentials and grants access if valid.	
Precondition	The user must have a registered account and must be on the login page to enter credentials.	
Normal Flow	**Actor action**	**System response**
	1.User navigates to the login page 2.User enters their username and password	3. System validates the entered credentials 4.System grants access and redirects to dashboard
Alternative Flows	1a. If the credentials are incorrect, the system prompts the user to re-enter their details. 2a. If the user has forgotten their password, the system provides a password recovery option.	
Postconditions	The user is successfully logged in and redirected to their respective dashboard based on role.	
Business Rules	- Valid username and password are required. - The system must validate credentials before granting access. - Each role is redirected to its own specific dashboard. - Password recovery option is available.	
Assumptions	The user has internet access, a registered role-based account, and correct login credentials.	

Table 4.2. Use Case for View Class Schedule

Field	Description	
Use Case ID	UCID-02	
Use Case Name	View Class Schedule	
Actors	Student, Instructor	
Description	This use case allows students to view their assigned class schedule for the current term.	
Precondition	The student and instructor must be logged in and enrolled in a program.	
Normal Flow	**Actor action**	**System response**
	1.Student logs into the system 2.Navigates to "My Schedule"	3.System displays the schedule
Alternative Flows	No schedule yet	
Postconditions	The student's and instructor's class schedule is displayed.	
Business Rules	Only authenticated students and instructor can view schedules.	
Assumptions	The schedule data is already available and up-to-date.	

Table 4.3. Use Case for Receive Schedule Notifications

Field	**Description**	
Use Case ID	UCID-03	
Use Case Name	Receive Schedule Notifications	
Actors	Student	
Description	This use case allows students and instructo to receive notifications when their schedule is updated.	
Precondition	Student and instructo are subscribed to schedule notifications and has a valid device or email address.	
Normal Flow	**Actor action**	**System response**
	3.Student and instructor receives the notification	1.Schedule is updated by an authority 2.System triggers a notification
Alternative Flows	1a. Notification fails due to internet or device issues.	
Postconditions	The student and instructor receives the notification.	
Business Rules	Notifications must be sent immediately after any schedule changes.	
Assumptions	The student and instructor have internet access and an active notification subscription.	

Table 4.4. Use Case Description for Download Schedule

Field	Description	
Use Case ID	UCID-04	
Use Case Name	Download Schedule	
Actors	Student	
Description	Enables students to download their class schedule in a printable or shareable file format.	
Precondition	The student is logged in, has access to view their schedule, and is on the schedule page.	
Normal Flow	**Actor action**	**System response**
	1.Student views schedule on-screen 2.Clicks “Download Schedule” button	3.System generates the schedule file 4.File download is initiated
Alternative Flows	1a. If the file generation fails, the system displays an error message and offers a retry button. 2a. If the network disconnects during download, the system notifies the student and allows them to restart.	
Postconditions	Student download Schedule.	
Business Rules	-Downloaded schedules must adhere to the institution’s formatting standards. - Only authenticated students may initiate a download. - File names should include student ID and term.	
Assumptions	-The student’s browser supports file downloads. - The student has sufficient storage space on their device.	

Table 4.5 . Use Case Description for Submit Feedback or Raise Concerns

Field	**Description**	
Use Case ID	UCID-05	
Use Case Name	Submit Feedback or Raise Concerns	
Actors	Student, Instructor	
Description	Allows students and instructor to submit feedback about their schedule or raise concerns regarding scheduling conflicts, timing, or other related issues.	
Precondition	The student is logged in and on the “Feedback” or “Help” page.	
Normal Flow	**Actor action**	**System response**
	1.Navigates to the “Feedback” Page 2.Fills out feedback form or selects “Raise Concern” 3.Submits the form	4.System logs the entry and notifies staff
Alternative Flows	1a. If form validation fails (e.g., empty mandatory fields), the system highlights errors and prompts correction. 2a. If submission fails due to a network error, the system alerts the student and allows retry.	
Postconditions	The feedback or concern is recorded in the system and routed to the appropriate departmental staff or administrator for review and action.	
Business Rules	-All submissions must include a timestamp and student identifier. - Feedback entries are confidential and viewable only by authorized staff. - The system must acknowledge receipt to the student immediately.	
Assumptions	-The student has internet access. - Relevant staff monitors incoming feedback regularly.	

Table 4.6. Use Case Description for View Assigned Teaching Schedule

Field	Description	
Use Case ID	UCID-06	
Use Case Name	View Assigned Teaching Schedule	
Actors	Instructor	
Description	This use case allows instructors to view their assigned teaching schedule, including class timings, locations, and any updates or changes.	
Precondition	The instructor is logged in and has valid access to their teaching schedule.	
Normal Flow	**Actor action**	**System response**
	1.Instructor logs into the system 2.Navigates to "Teaching Schedu section 3.Views schedule details	4.System displays the teaching schedule
Alternative Flows	1a. If the schedule is not available due to a system error, the system displays a message indicating the issue. 2a. If the instructor has no assigned schedule, the system displays an appropriate message indicating no teaching assignments.	
Postconditions	The instructor successfully views the complete teaching schedule for the current term or any updates made to it.	
Business Rules	Only instructors with assigned courses can view their teaching schedule. The schedule must be updated automatically if any changes are made to assignments.	
Assumptions	- The instructor has internet access. - The teaching schedule is up to date in the system. - The instructor has the necessary permissions to view their schedule.	

Table 4.7. Use Case Description for Request Schedule Changes

Field	**Description**	
Use Case ID	UCID-07	
Use Case Name	Request Schedule Changes	
Actors	Instructor	
Description	This use case allows instructors to request changes to their assigned teaching schedule, such as rescheduling classes, adjusting timings, or changing locations.	
Trigger/Precondition	The instructor is logged in, has an assigned teaching schedule, and needs to request a change.	
Normal Flow	**Actor action**	**System response**
	1. Instructor logs into the system 2. Navigates to "Request Schedule Change" 3. Fills out the change request form 4. Submits the request	5. System displays the teaching schedule
Alternative Flows	1a. If the form validation fails (e.g., missing required information), the system displays an error and prompts the instructor to fix the issue. 2a. If the request submission fails due to network issues, the system alerts the instructor and allows retry.	
Postconditions	The instructor successfully views the complete teaching schedule for the current term or any updates made to it.	
Business Rules	- Requests must include a valid reason and proposed change details. - The system must route the request to the appropriate administrator or department head for review.	
Assumptions	- The instructor has internet access. - The instructor has valid teaching assignments in the system.	

Table 4.8. Use Case Description for Provide Availability Information

Field	Description	
Use Case ID	UCID-08	
Use Case Name	Provide Availability Information	
Actors	Instructor	
Description	This use case allows instructors to provide their availability for teaching, including preferred times and dates for scheduling classes.	
Precondition	The instructor is logged in and is on the "Provide Availability" page in the system.	
Normal Flow	**Actor action**	**System response**
	1.Logs into the system 2.Navigates to "Provide Availability" page 3.Enters available days and times 4.Submits the availability	5.System updates instructor's availability
Alternative Flows	1a. If the input is invalid (e.g., conflicting availability), the system displays an error message prompting correction. 2a. If the instructor does not input any availability, the system notifies them to fill out the form.	
Postconditions	The system records the request, and the relevant department or administration is notified for review and approval.	
Business Rules	- Availability must be in valid time slots. - The system must prevent conflicts in availability submissions. - Only authorized instructors can submit availability information.	
Assumptions	- The instructor has internet access. - The system has a valid schedule structure and available time slots for submission.	

Table 4.9. Use Case Description for Generate Department Schedule

Field	Description	
Use Case ID	UCID-09	
Use Case Name	Generate Department Schedule	
Actors	Department Head	
Description	Enables the Department Head to create or regenerate the class schedule by combining course requirements, room availability, batch, and section details	
Precondition	Department Head is logged in and has access to scheduling tools.	
Normal Flow	**Actor action**	**System response**
	1.Logs in 2.Navigates to “Generate Schedule” 3.Selects courses, rooms, batches, sections 4.Clicks “Generate”	5.System processes and generates provisional schedule
Alternative Flows	1a. If required data (e.g., room availability or batch details) is missing, the system prompts for completion. 2a. If a conflict (e.g., room overlap) is detected, the system flags it and suggests adjustments.	
Postconditions	A provisional department schedule is generated and presented for review.	
Business Rules	- Room capacity must meet or exceed batch size. - Assign sections must not overlap time slots. - Courses must follow predefined sequencing rules. - Conflicts must be resolved before finalizing the schedule.	
Assumptions	- All prerequisite data (e.g., courses, rooms, batches, sections) is accurate and up-to-date. - The scheduling algorithm handles conflicts effectively. - The system’s scheduling engine is operational.	

Table 4.10. Use Case Description form Modify Schedule

Field	Description	
Use Case ID	UCID-10	
Use Case Name	Approve/Modify Schedule	
Actors	Department Head	
Description	Allows the Department Head to review, approve, or adjust a generated department schedule before publishing it.	
Precondition	A draft department schedule exists and the Department Head is logged in.	
Normal Flow	**Actor action**	**System response**
	1.Logs in 2.Opens the draft schedule 3.Reviews and adjusts as needed 4.Clicks “Approve”	5.System saves and notifies relevant stakeholders
Alternative Flows	1a. If changes introduce conflicts, the system highlights problematic entries. 2a. Department Head rejects the draft, returning it to the scheduling engine for regeneration.	
Postconditions	The schedule is marked as approved and, if modified, changes are saved and propagated to downstream systems.	
Business Rules	- Only one approved version may be active at a time. - All modifications must maintain no-overlap constraints. - Approval timestamps and approver ID must be recorded.	
Assumptions	- The draft schedule is up-to-date. - The Department Head has authority to approve or request changes.	

Table 4.11. Use Case Description for Resolve Scheduling Conflicts

Field	Description	
Use Case ID	UCID-11	
Use Case Name	Resolve Scheduling Conflicts	
Actors	Department Head	
Description	Enables the Department Head to identify and resolve conflicts in the timetable, such as instructor overlaps or room double-bookings.	
Precondition	A schedule (draft or approved) contains one or more conflict alerts.	
Normal Flow	**Actor action**	**System response**
	1. Logs in 2. Navigates to "Conflict Resolution" 3. Views list of conflicts 4 Selects and resolves conflicts Saves changes	6.System updates the schedule
Alternative Flows	1a. If no suitable resolution is found, the Department Head can escalate the conflict to the Super Admin for manual intervention.	
Postconditions	Conflicts are resolved and the schedule is updated to eliminate overlaps.	
Business Rules	- All conflicts must be resolved before final approval. - Resolution choices must adhere to room capacity and instructor availability. - Each resolution step must be logged.	
Assumptions	- The system accurately detects all conflicts. - Department Head has the authority to make resolution decisions.	

Table 4.12. Use Case Description for Manage Departmental Timetables

Field	Description	
Use Case ID	UCID-12	
Use Case Name	Manage Departmental Timetables	
Actors	Department Head	
Description	Allows the Department Head to view, update, and publish individual course timetables within the department.	
Precondition	The Department Head is logged in and a department schedule (draft or approved) exists.	
Normal Flow	**Actor action**	**System response**
	1. Logs in 2. Navigates to “Manage Timetables” 3. Selects and edits timetables 4. Saves and publishes	5.System notifies stakeholders
Alternative Flows	1a. If an edit introduces a conflict, the system flags it and prompts corrective action. 2a. If publish fails, the system alerts and allows retry.	
Postconditions	Selected timetables are updated and published; stakeholders are notified of any changes.	
Business Rules	- Only one version of each timetable may be published at a time. - All updates must maintain no-overlap constraints. - Publication timestamps and editor ID must be recorded.	
Assumptions	- The department schedule is up to date. - Department Head has publish rights.	

Table 4.13. Use Case Description for Manage User Roles and Permissions

Field	Description	
Use Case ID	UCID-13	
Use Case Name	Manage User Roles and Permissions	
Actors	Admin	
Description	Enables the Admin to create, edit, or revoke user accounts, assign roles (Student, Instructor, Department Head), and configure permissions.	
Precondition	The Admin is logged in and has access to the user management interface.	
Normal Flow	**Actor action**	**System response**
	1. Logs in 2. Navigates to "User Management" 3. Selects or creates user 4. Assigns or updates role and permissions 5. Saves changes	6. System sends notification to user
Alternative Flows	1a. If invalid permissions are selected, the system displays an error and blocks the change. 2a. If the update fails, system alerts and allows retry.	
Postconditions	User roles and permissions are updated in the system; affected users are notified of changes.	
Business Rules	- Only Admins may manage roles and permissions. - Role changes must follow institution policy. - Audit logs must record every change.	
Assumptions	- Admin has internet access. - User directory is synchronized and current.	

Table 4.14. Use Case Description for Configure System Settings

Field	Description	
Use Case ID	UCID-14	
Use Case Name	Configure System Settings	
Actors	Admin	
Description	Allows the Admin to adjust global system parameters such as term dates, notification rules, default time slots, and formatting standards.	
Precondition	The Admin is logged in and on the system settings page.	
Normal Flow	**Actor action**	**System response**
	1. Logs in 2. Navigates to "System Settings" 3. Adjusts parameters 4. Saves changes	5.System validates and applies settings
Alternative Flows	1a. If validation fails (e.g., end date before start date), the system highlights errors and prompts correction. 2a. If save fails, system alerts and allows retry.	
Postconditions	System parameters are updated and applied immediately or upon the next scheduled update cycle.	
Business Rules	- Dates must follow academic calendar policies. - Changes in notification rules must comply with privacy regulations. - All changes recorded in audit log.	
Assumptions	- Admin understands the impact of each setting. - System configuration module is operational.	

Table 4.15. Use Case Description for Update Schedule Data

Field	Description	
Use Case ID	UCID-15	
Use Case Name	Update Schedule Data	
Actors	Admin	
Description	Enables the Admin to update system-wide schedule data, including course offerings, instructor assignments, and room assigning, across all departments.	
Precondition	The admin is logged in and has access to the scheduling data management interface.	
Normal Flow	**Actor action**	**System response**
	1. Logs in 2. Navigates to "Schedule Data Management" 3. Updates course/instructor /room data 4. Saves updates	5.System propagates and notifies stakeholders
Alternative Flows	1a. If data validation fails (e.g., conflict between room and instructor), the system highlights the issue and prompts correction. 2a. If save fails, system alerts and allows retry.	
Postconditions	Updated schedule data is applied to the system, and affected departments are notified of changes.	
Business Rules	- Changes must comply with course scheduling policies. - Conflicting assignments (e.g., double-booked room or instructor) must be flagged. - All changes are logged with timestamps and actor ID.	
Assumptions	- All required scheduling data is available and up-to-date. - Admin has the necessary authority to update schedule data.	

Table 4.16. Use Case Description for Monitor Overall System

Field	Description	
Use Case ID	UCID-16	
Use Case Name	Monitor Overall System	
Actors	Admin	
Description	Allows the Admin to monitor the overall health and status of the system, including checking for errors, system performance, and user activity logs.	
Precondition	The Admin is logged in and has access to the system monitoring interface.	
Normal Flow	**Actor action**	**System response**
	1.Logs in 2.Navigates to “System Monitoring” 3.Views system status	4. System provides resolution suggestions if needed
Alternative Flows	1a. If no issues are detected, the system displays a "healthy" status. 2a. If critical issues are detected, the system alerts the Admin and recommends corrective actions.	
Postconditions	System status is updated; if issues are identified, alerts are generated, and corrective actions can be initiated.	
Business Rules	- All system logs must be recorded for auditing purposes. - System performance must be above predefined thresholds. - Alerts must be triggered for critical failures.	
Assumptions	- System monitoring tools are active and functioning. - Admin is familiar with system troubleshooting.	

Table 4.17. Use Case Description for Course Registration

Field	Description	
Use Case ID	UCID-17	
Use Case Name	Course Registration	
Actors	Department Head, Admin	
Description	Allows the registration of new courses in the system.	
Precondition	Admin or department head is logged in.	
Normal Flow	**Actor action**	**System response**
	1. Logs in 2. Navigates to registration 3. Enters details 4. Saves changes	5. System stores data
Alternative Flows	1a. If the course code already exists, the system displays a conflict.	
Postconditions	Course is added to the system database.	
Business Rules	Each course must have a unique identifier.	
Assumptions	User has all necessary course information.	

Table 4.18 Use Case Description for Room Registration

Field	Description	
Use Case ID	UCID-18	
Use Case Name	Room Registration	
Actors	Department Head	
Description	Allows the registration of rooms with attributes such as capacity and type.	
Precondition	Admin is logged in.	
Normal Flow	**Actor action**	**System response**
	1. Logs in 2. Navigates to registration 3. Enters details 4. Saves changes	5.System stores data
Alternative Flows	1a. If room ID already exists, the system prompts for a unique identifier.	
Postconditions	Room details are saved to the system database.	
Business Rules	Room capacity must be greater than zero.	
Assumptions	Admin has the room details ready for entry.	

Table 4.19. Use Case Description for Building Registration

Field	Description	
Use Case ID	UCID-19	
Use Case Name	Building Registration	
Actors	Admin	
Description	Enables the registration of buildings within the system.	
Precondition	Admin is logged in.	
Normal Flow	**Actor action**	**System response**
	1. Logs in 2. Navigates to registration 3. Enters details 4. Saves changes	5.System stores data
Alternative Flows	1a. If the building ID already exists, the system flags a duplicate.	
Postconditions	Building details are stored in the system.	
Business Rules	Each building must have a unique identifier.	
Assumptions	Admin has all necessary building details.	

Table 4.20. Use Case Description for Class Section Registration

Field	Description	
Use Case ID	UCID-20	
Use Case Name	Class Section Registration	
Actors	Department Head	
Description	Facilitates the registration and management of class sections.	
Precondition	Class information, including course and instructor details, is available.	
Normal Flow	**Actor action**	**System response**
	1. Logs in 2. Navigates to registration 3. Enters details 4. Saves changes	5.System stores data
Alternative Flows	1a. If section data conflicts with existing entries, the system prompts for correction.	
Postconditions	Class section and batch information are saved to the system.	
Business Rules	Each section must be linked to a unique batch.	
Assumptions	User has the class and batch details ready for entry.	

Table 4.21. Use Case Description for Batch Registration

Field	Description	
Use Case ID	UCID-21	
Use Case Name	Batch Registration	
Actors	Department Head	
Description	Allows for the registration and management of academic batches.	
Precondition	Batch information, including year, program, and specialization, is available	
Normal Flow	**Actor action**	**System response**
	1. Logs in 2. Navigates to registration 3. Enters details 4. Saves changes	5.System stores data
Alternative Flows	1a. If batch data conflicts with existing entries, the system prompts for correction.	
Postconditions	Teachers are assigned to the selected courses.	
Business Rules	Only available teachers can be assigned to courses.	
Assumptions	Teacher and course data are up-to-date.	

Table 4.22. Use Case Description for Teacher Assignment Registration

Field	Description	
Use Case ID	UCID-22	
Use Case Name	Teacher Assigning Registration	
Actors	Admin	
Description	Allows the department head to assign teachers to specific courses.	
Precondition	Department head is logged in and the course data exists in the system.	
Normal Flow	**Actor action**	**System response**
	1. Logs in 2. Navigates to registration 3. Enters details 4. Saves changes	5.System stores data
Alternative Flows	1a. If the teacher is unavailable, the system shows an error.	
Postconditions	Teachers are assigned to the selected courses.	
Business Rules	Only available teachers can be assigned to courses.	
Assumptions	Teacher and course data are up-to-date.	

4.3. Dynamic Model

4.3.1. Sequence Diagram

Sequence diagrams visually map how actors interact with the scheduling system over time, showing the step-by-step flow of actions and responses.

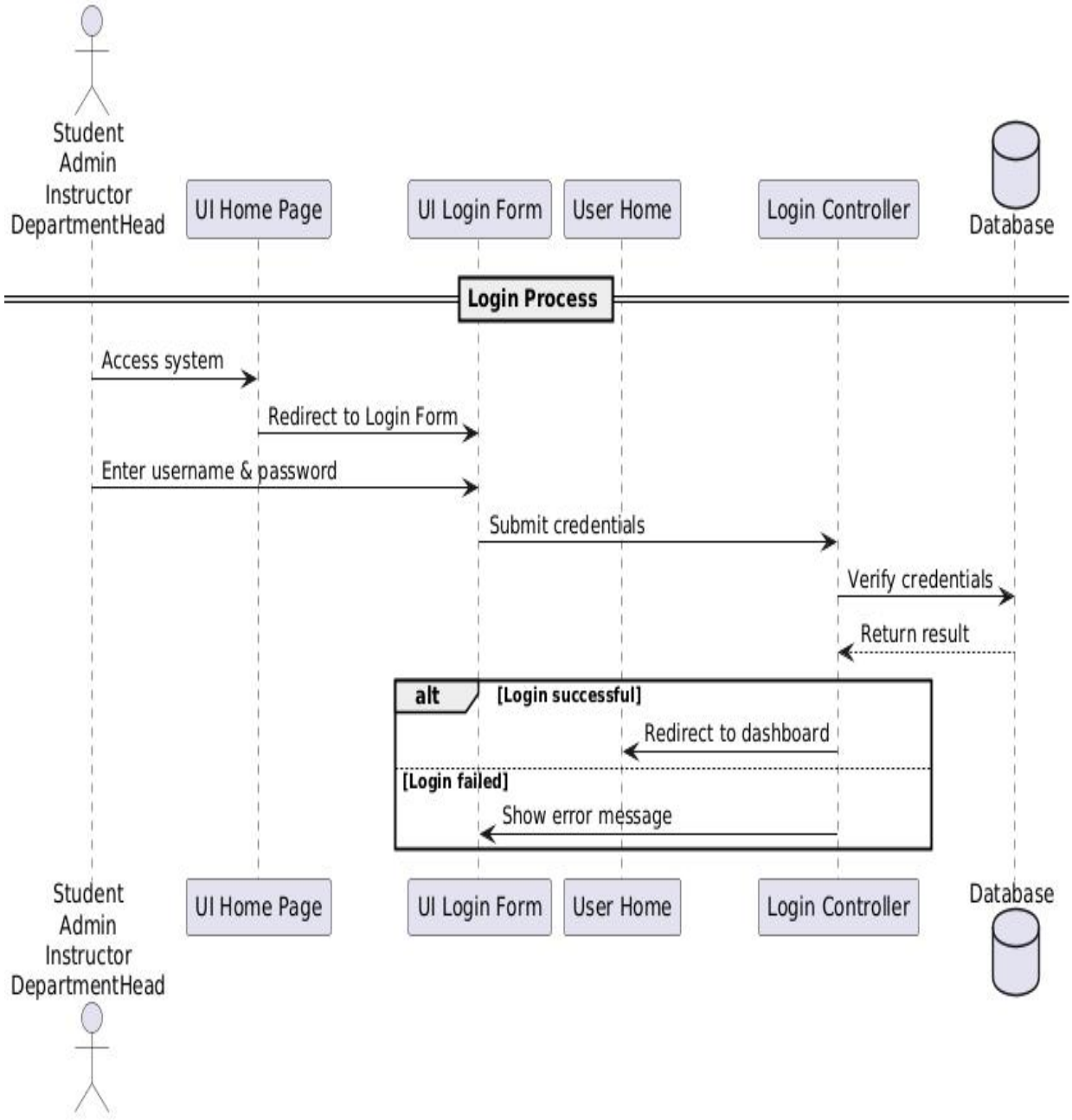

Figure 4.2: Sequence diagram for login

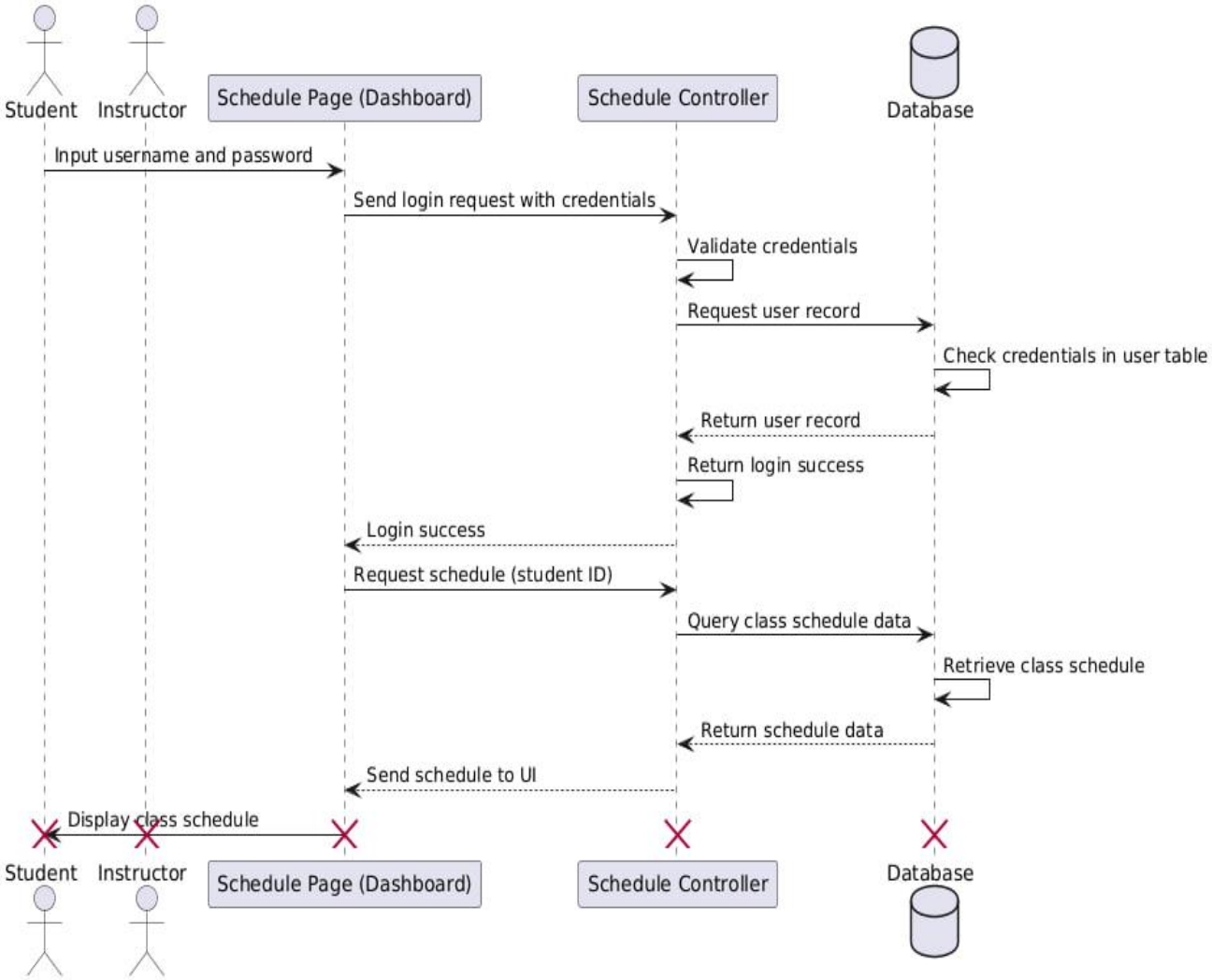

Figure 4. 3. Sequence Diagram for View class schedule

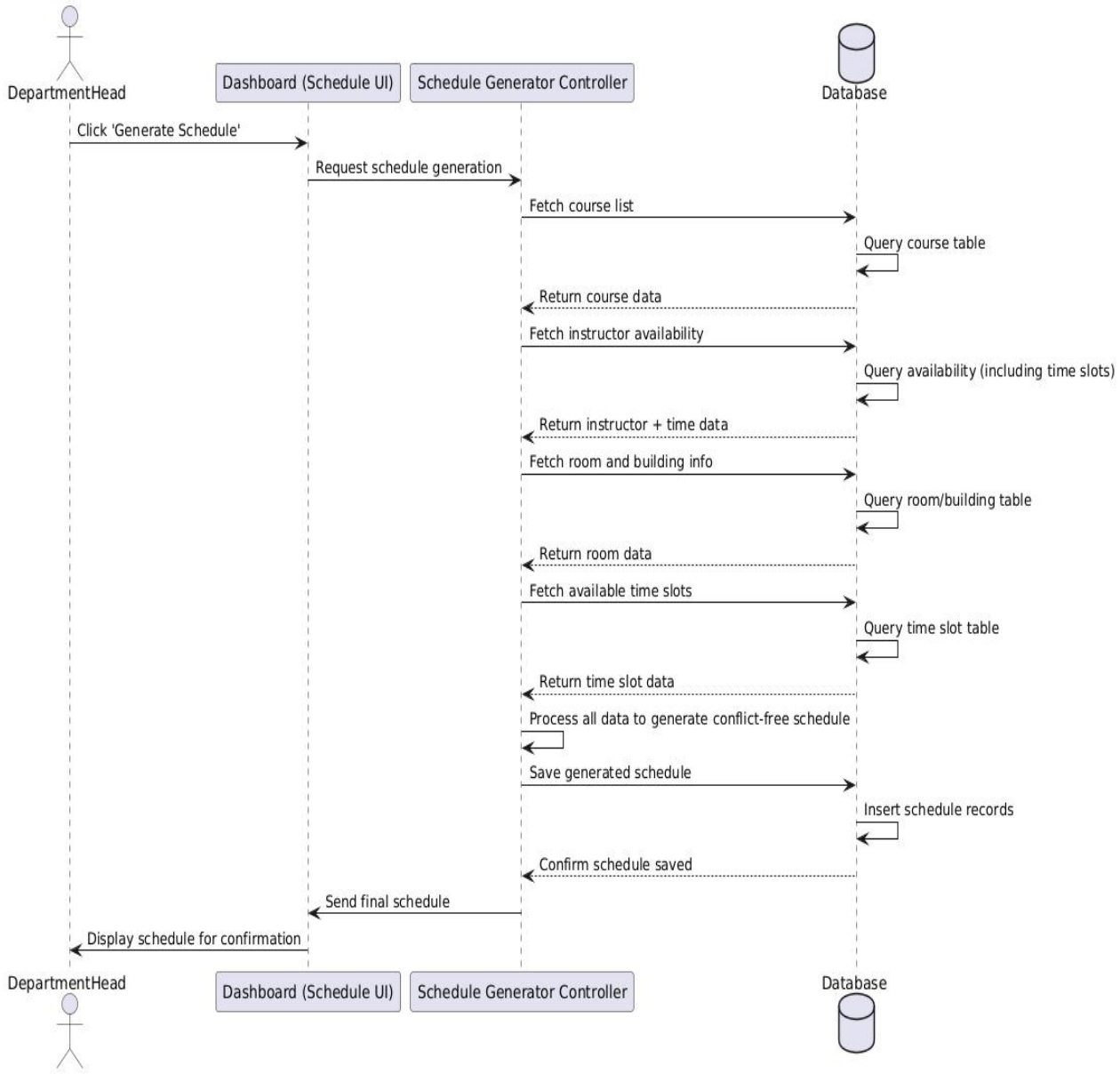

Figure 4. 4: Sequence Diagram for Generate Department Schedule

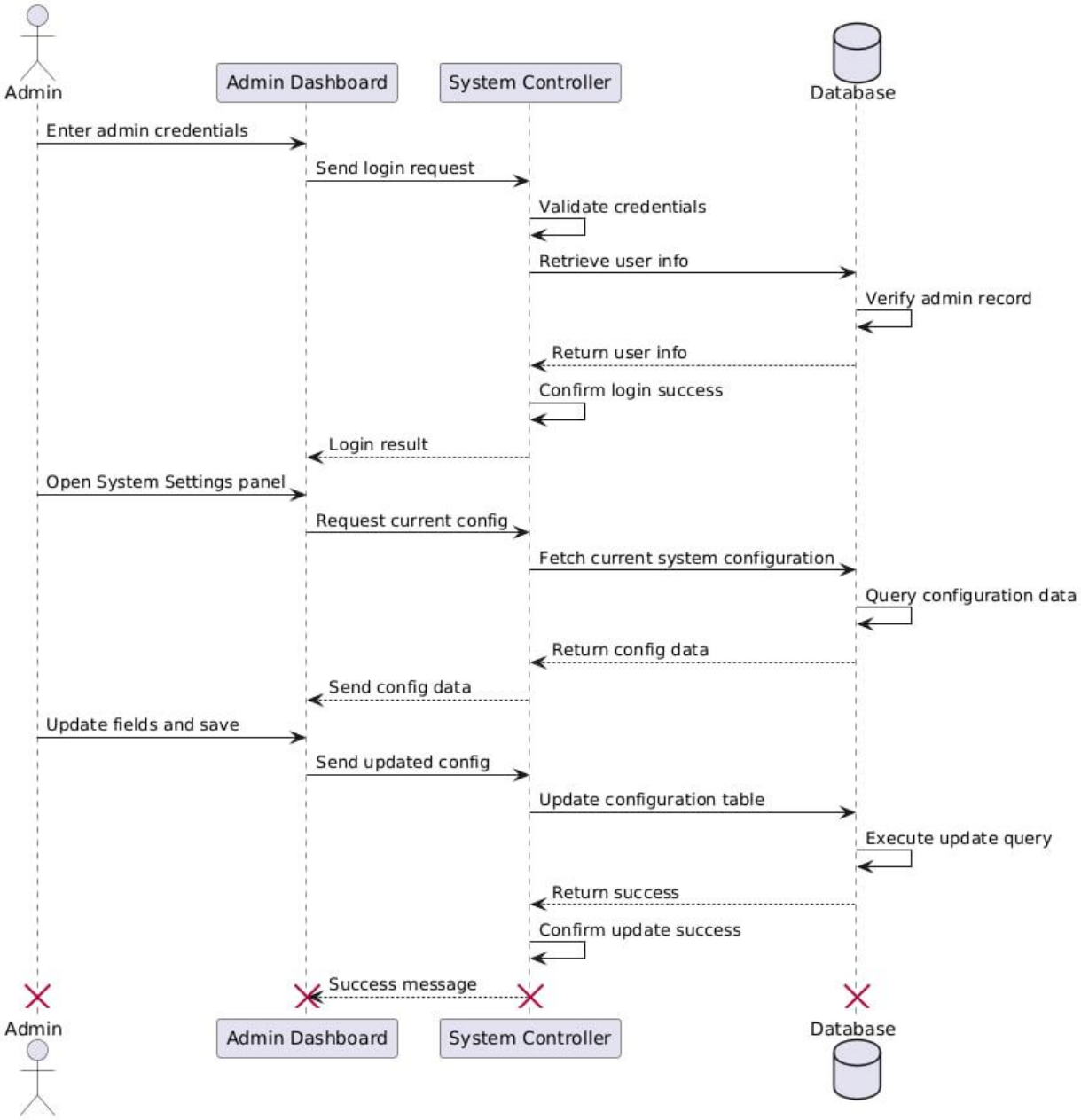

Figure 4. 5: Sequence Diagram for Configure System Settings

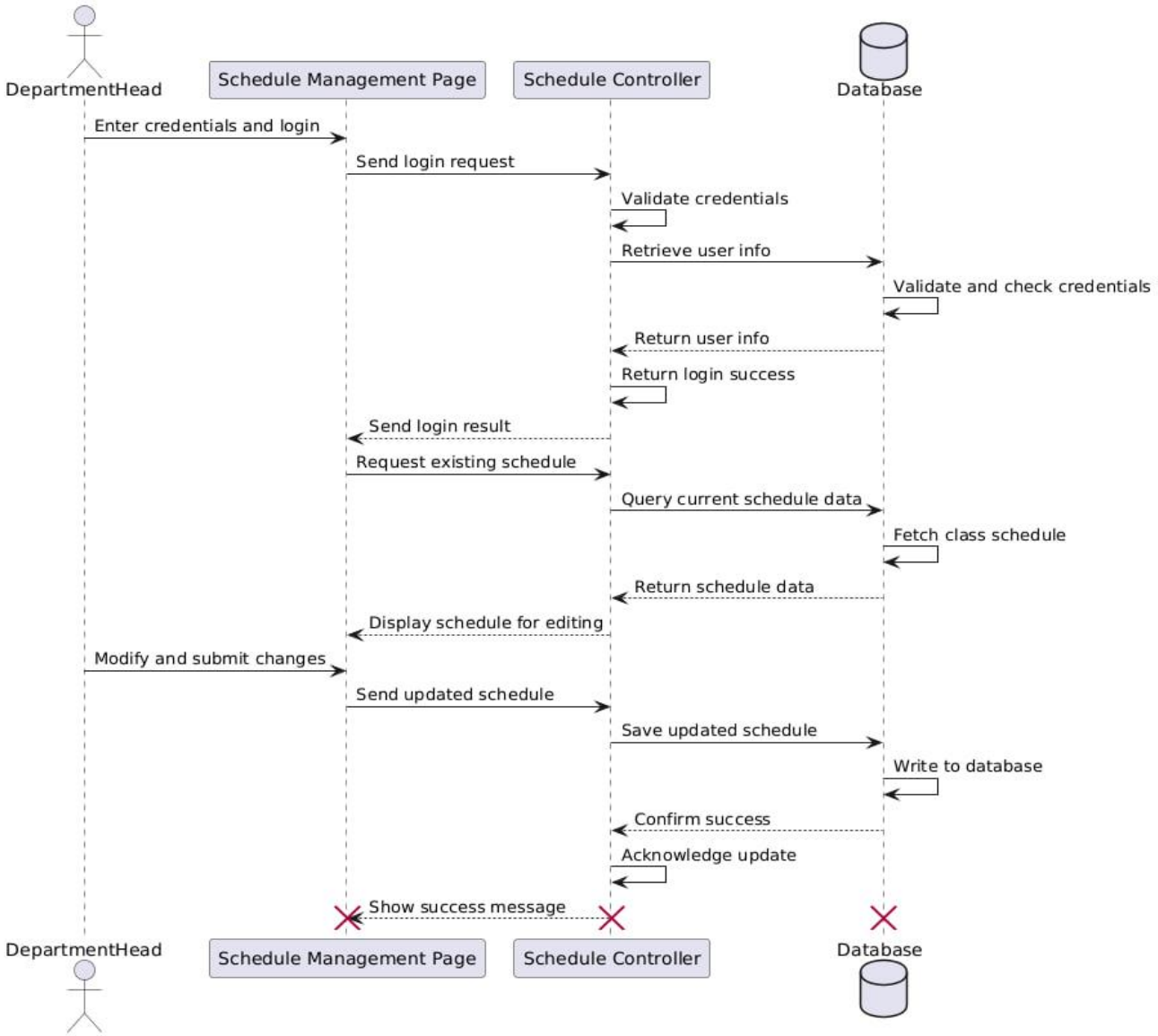

Figure 4.6: Sequence Diagram for Modify schedule

4.3.2. Activity Diagram

Activity diagrams model the step-by-step workflows of your scheduling system, showing how actors navigate processes like schedule generation, conflict resolution, and notifications.

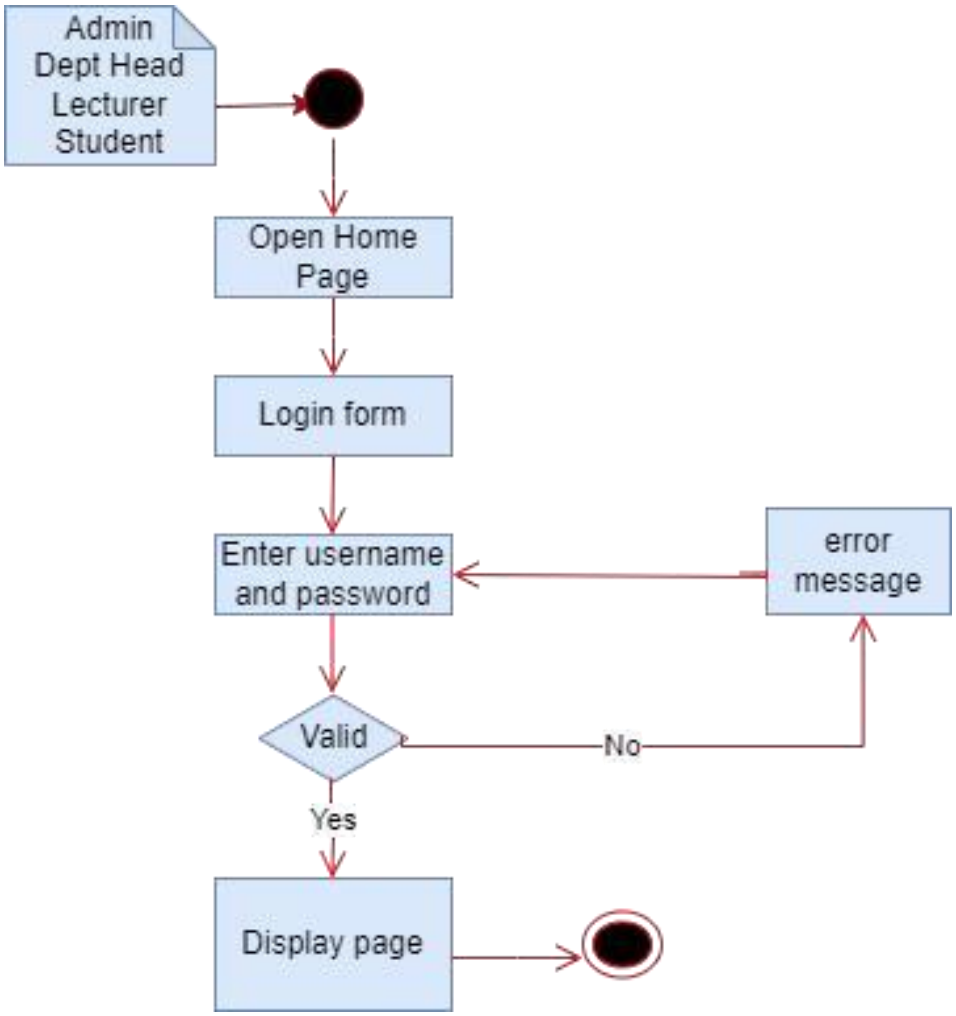

Figure 4. 7: Activity Diagram for Login

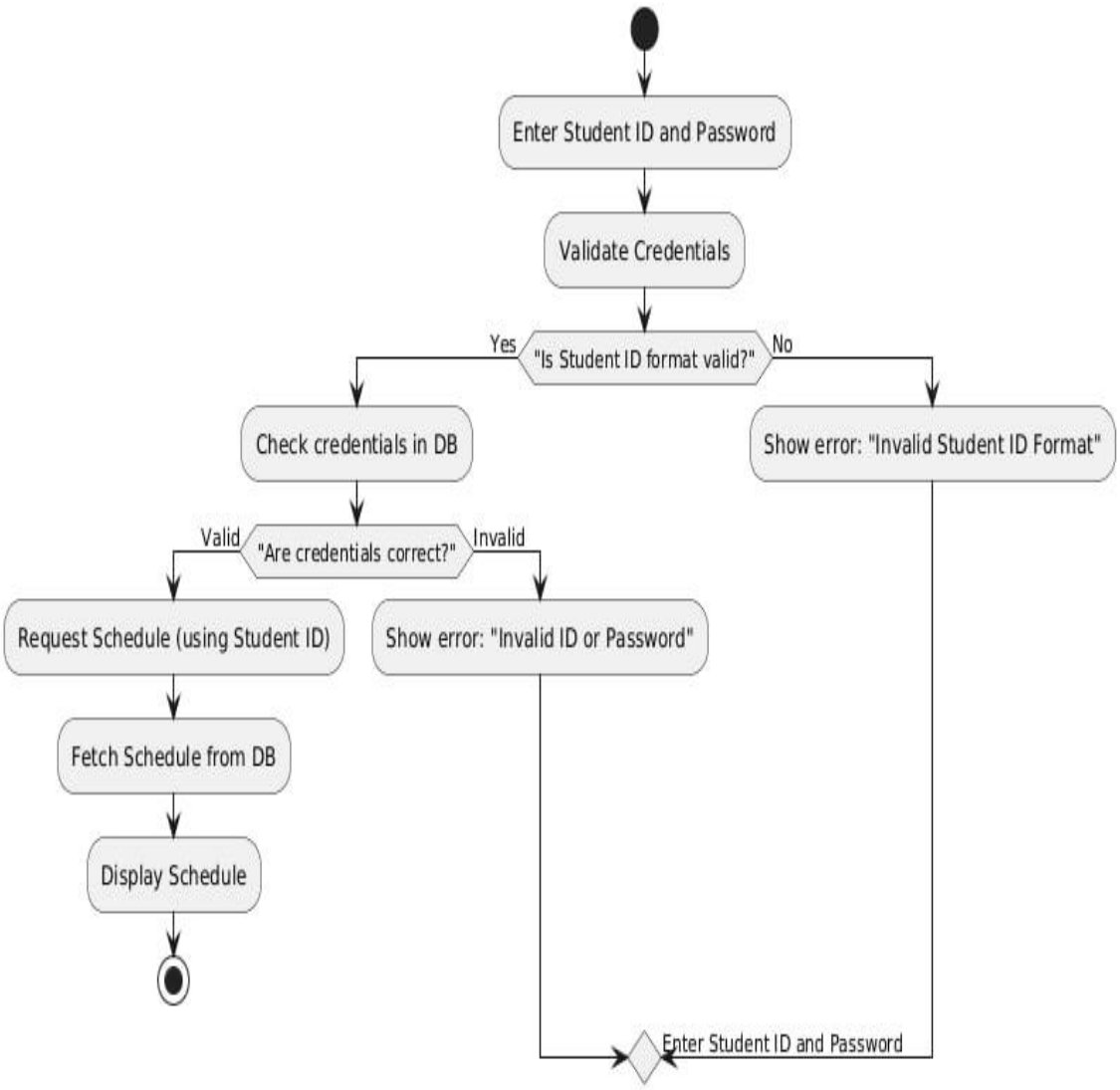

Figure 4.8: Activity Diagram for View class schedule

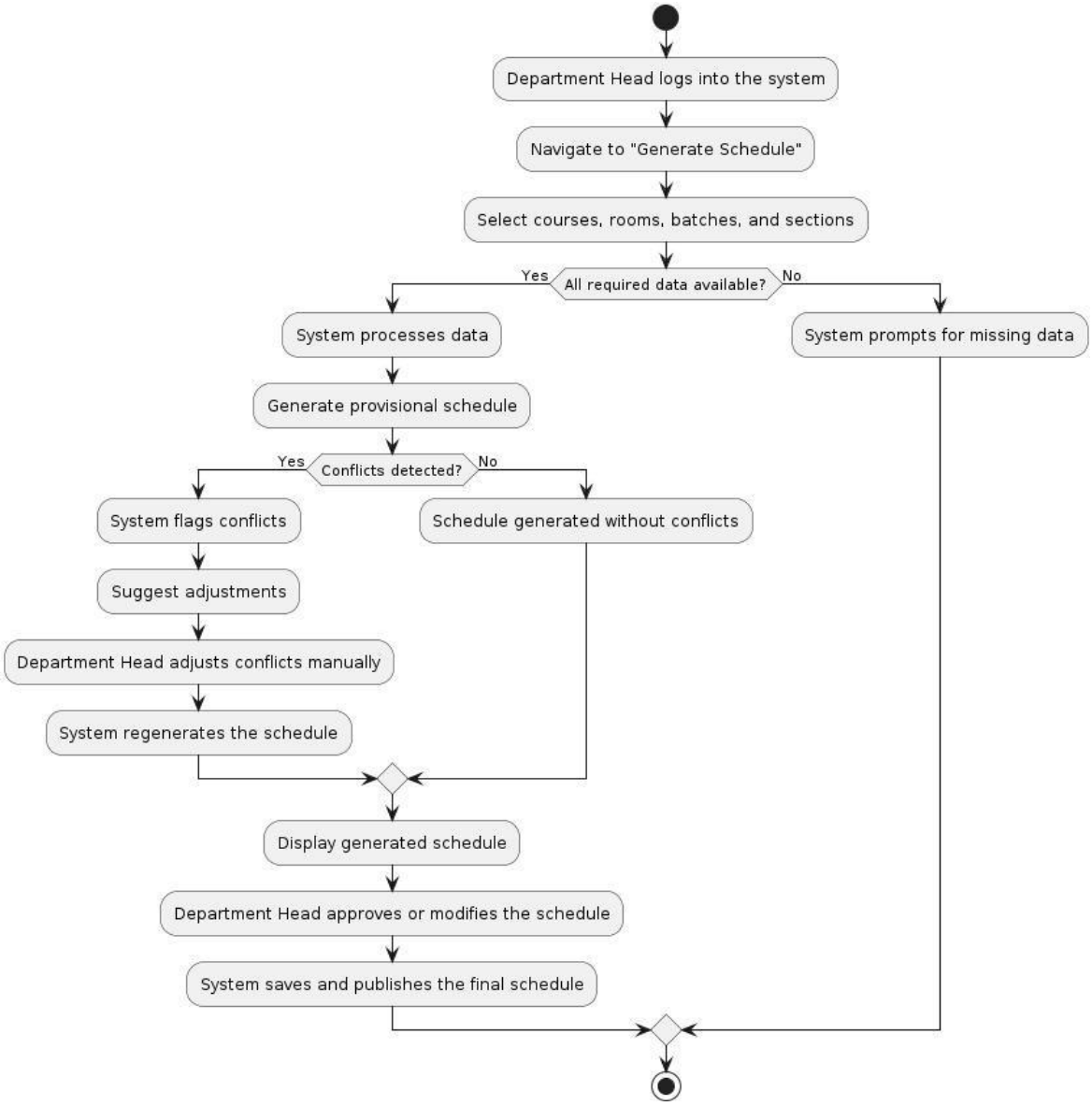

Figure 4. 9: Activity Diagram for Generate Department class Schedule

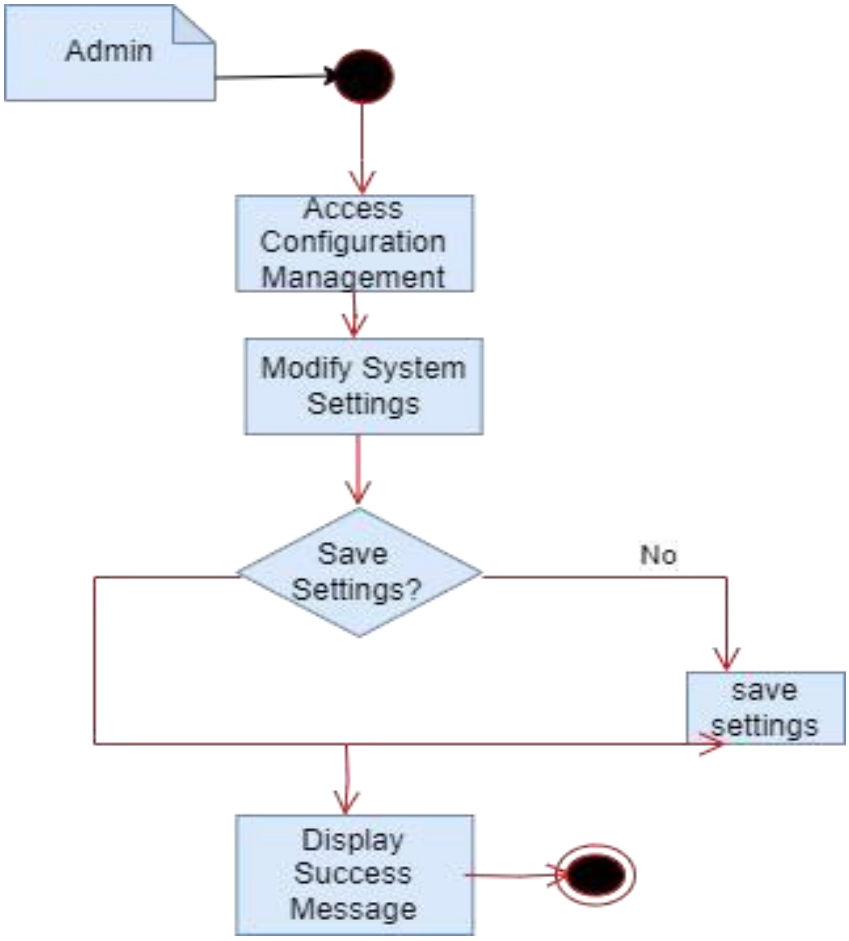

Figure 4. 10: Activity Diagram for Configure System Settings

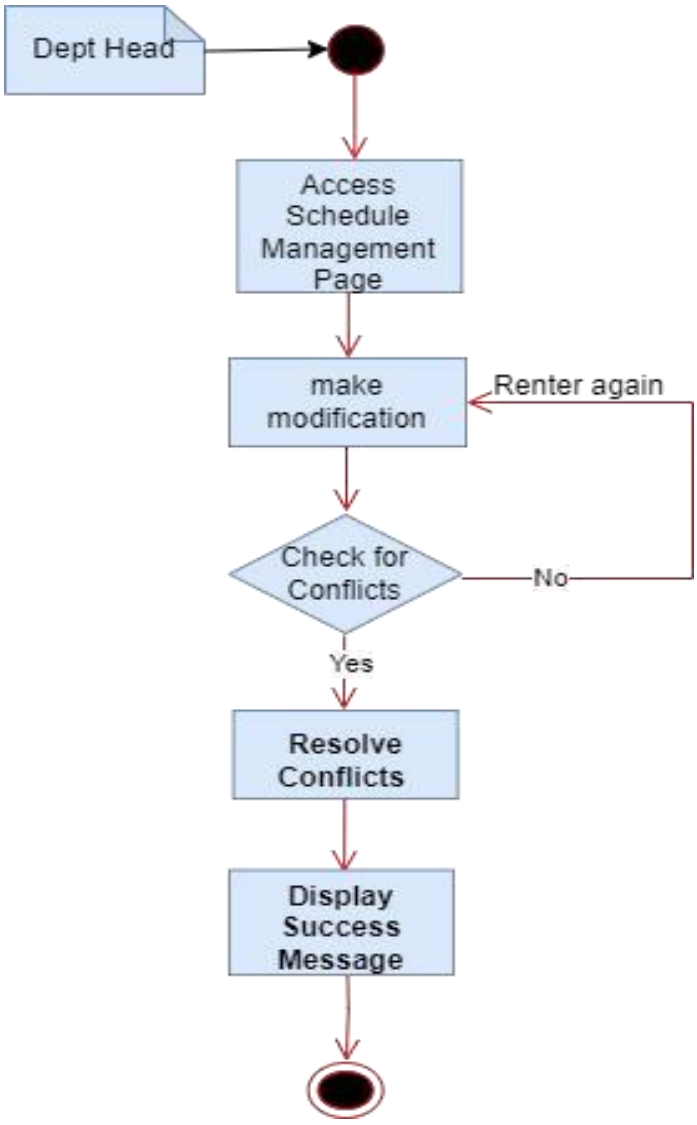

Figure 4. 11: Activity Diagram for Modify Class schedule

4.3.3. State Chart Diagram

State Chart Diagrams capture the dynamic behavior of system objects by mapping their lifecycle states and the events that trigger transitions.

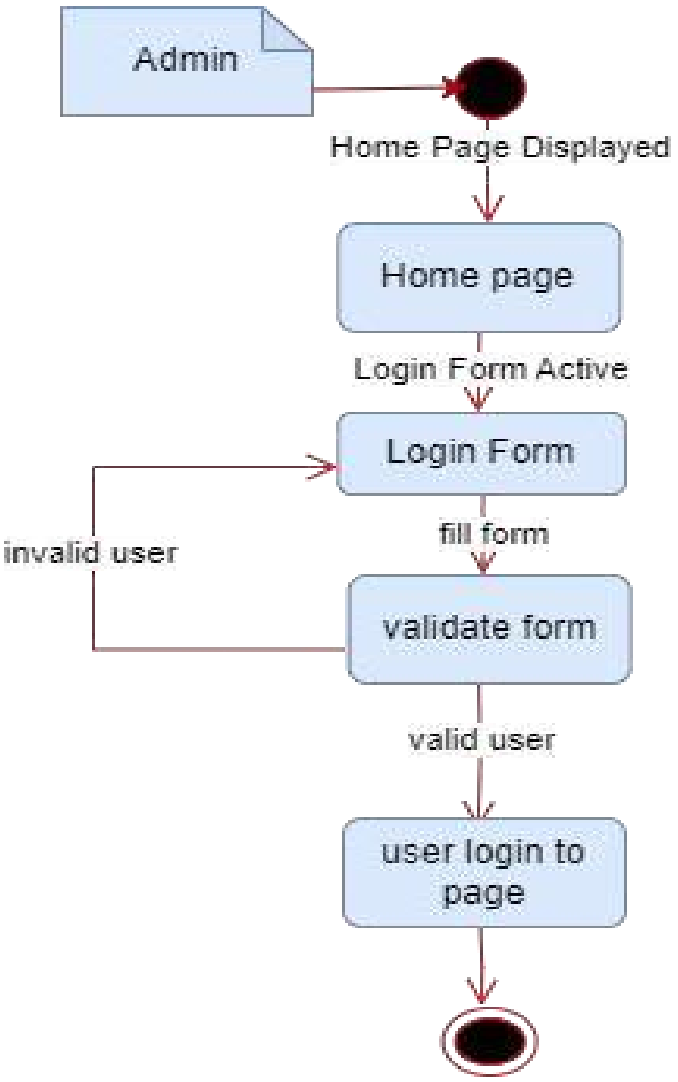

Figure 4. 12: state chart Diagram for login

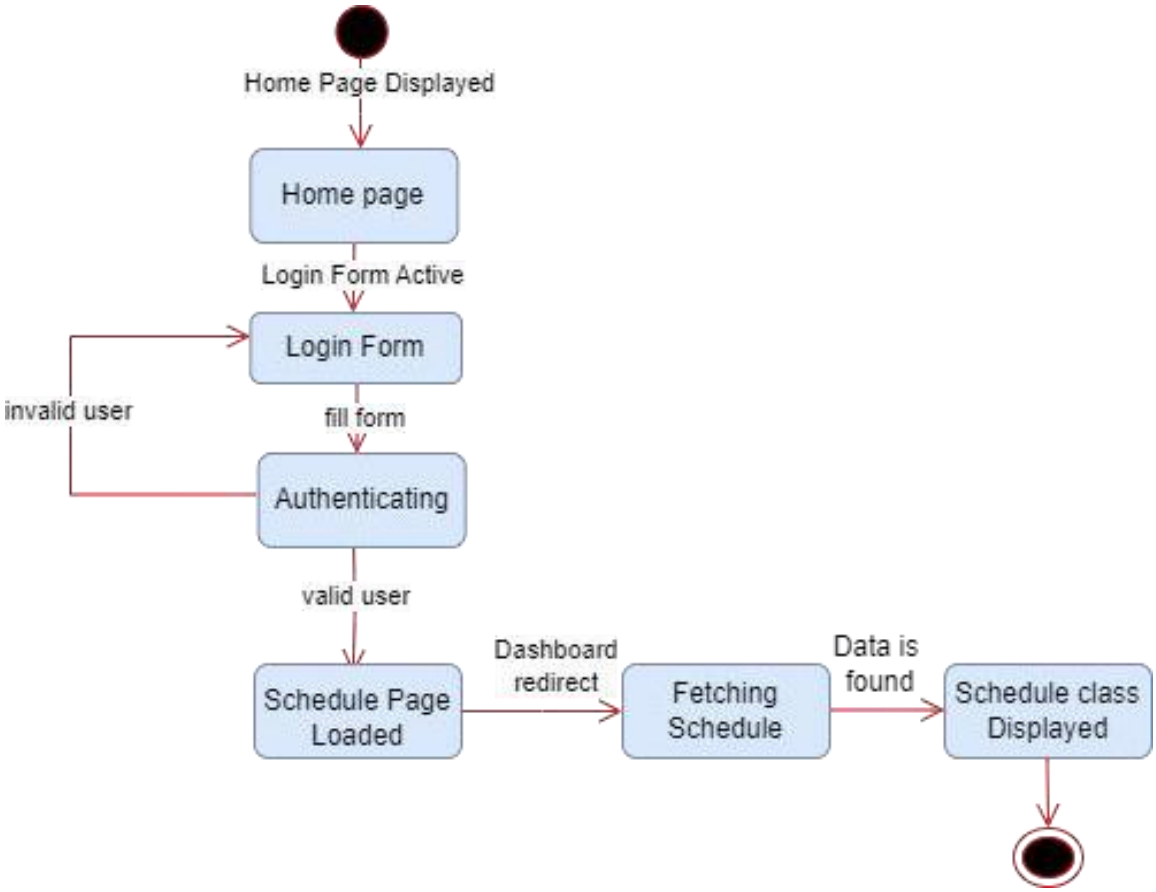

Figure 4. 13: state chart Diagram for View Class Schedule

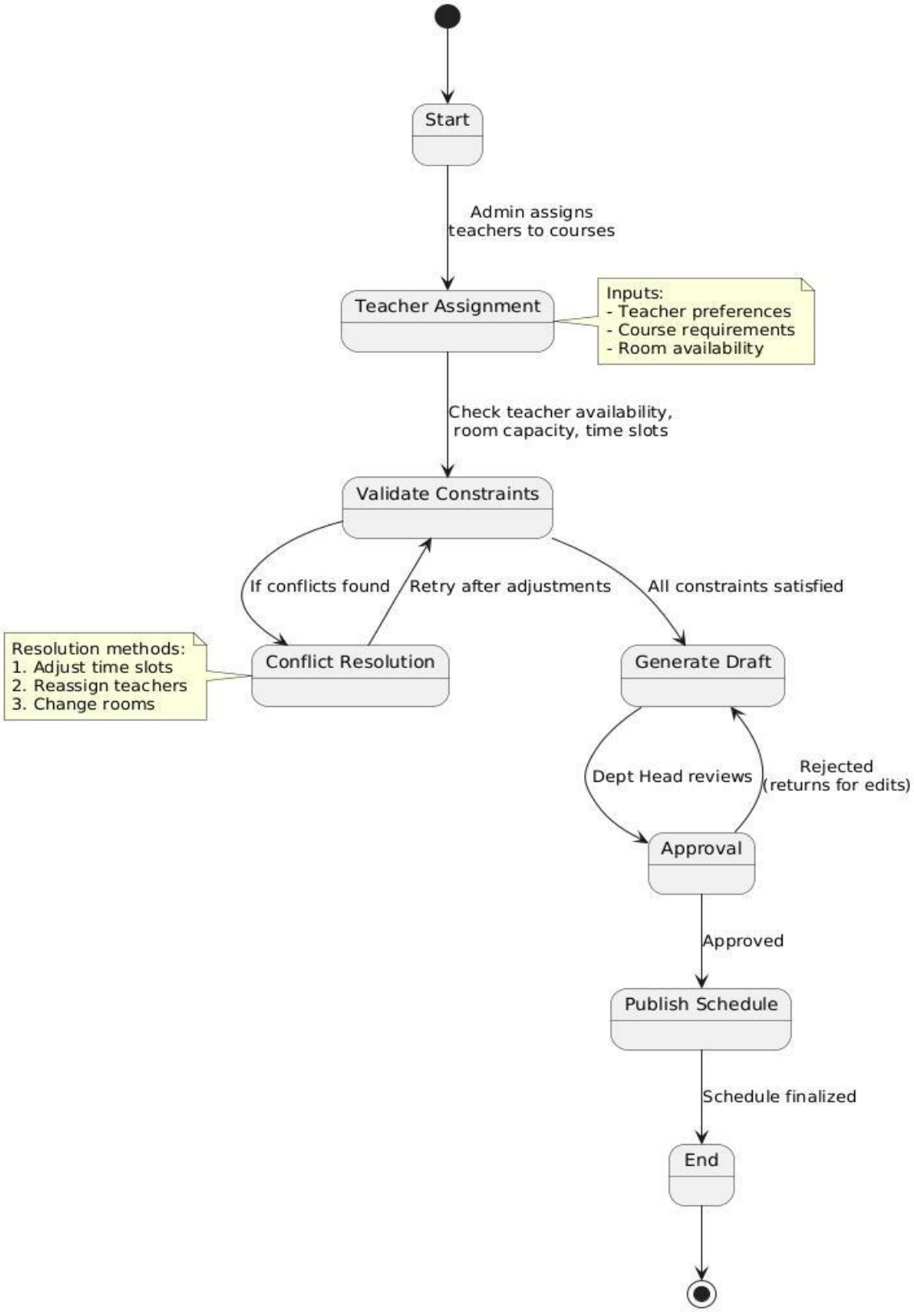

Figure 4. 14: State chart Diagram for Generate Class Schedule

4.4. Object Model

4.4.1. Class Diagram

A class diagram models system structure using classes (attributes/methods) and their relationships (inheritance, associations). It serves as a blueprint for object-oriented design and database schema, representing real-world entities and their interactions.

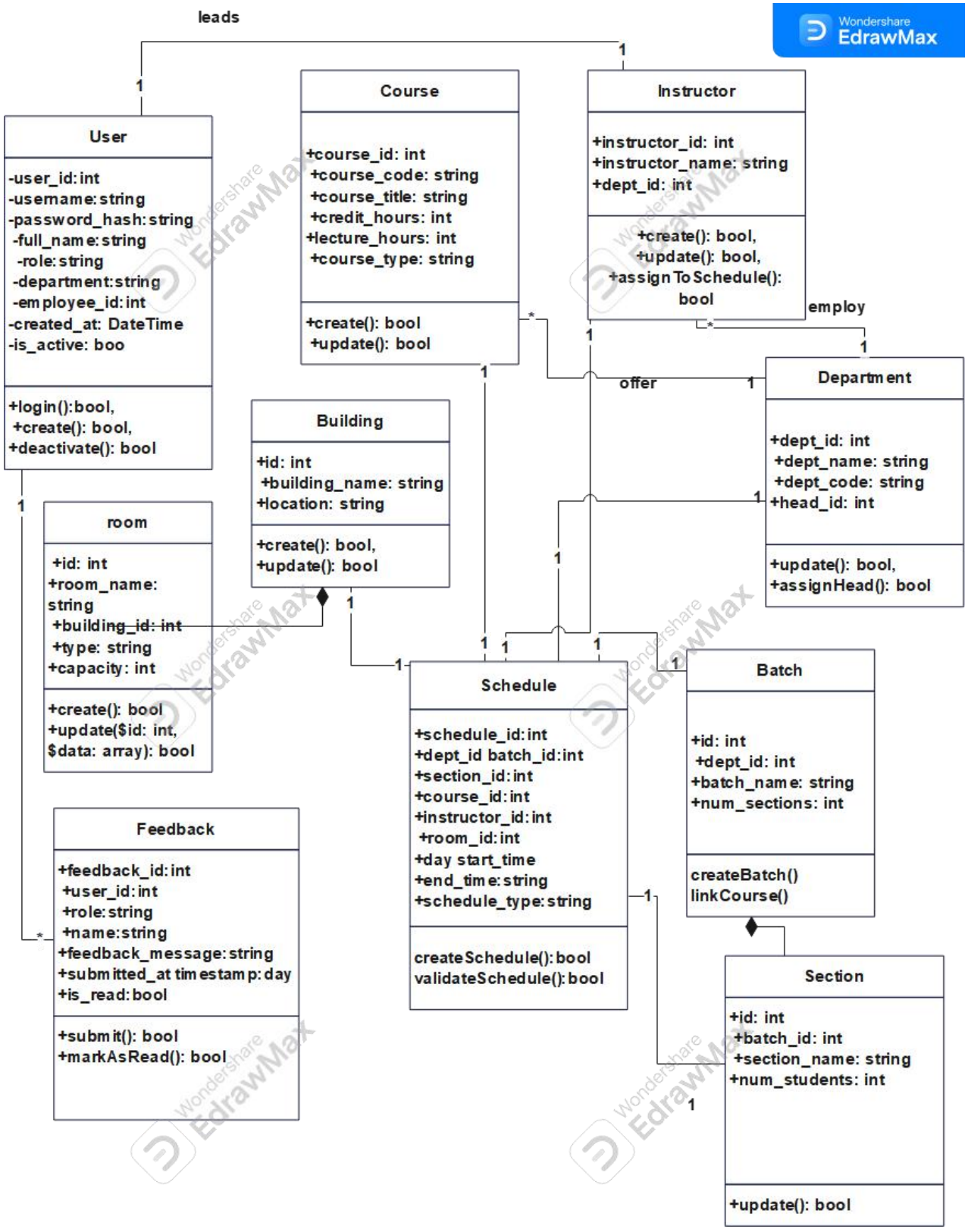

Figure 4. 15: Class Diagram

4.5. Database Specification

4.5.1. E-R Diagrams

Entity relationship model is a graphical representation of entities and their relationships to each other, typically used in computing in regard to the organization of data within databases or information systems. An entity is a piece of data-an object or concept about which data is stored. A relationship is how the data is shared between entities.

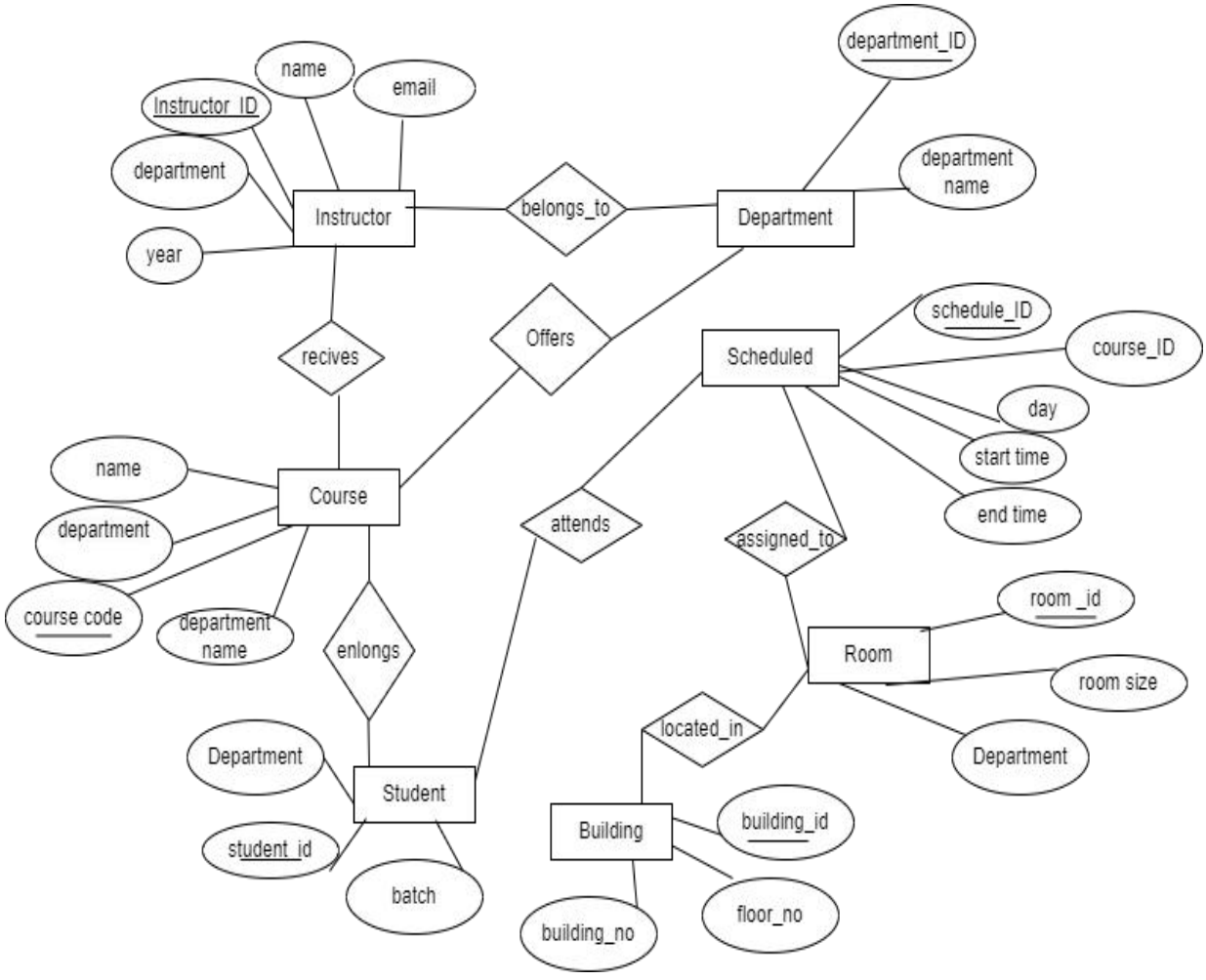

Figure 4. 16: ER_Diagram

REFERENCES

1. R. S. Pressman and B. R. Maxim, Software Engineering: A Practitioner's Approach, 8thed. New York, NY, USA: McGraw-Hill Education, 2014.
2. Object Management Group (OMG), "Unified Modeling Language (UML) Specification," Version 2.5.1, Dec. 2017. [Online]. Available: https://www.omg.org/spec/UML
3. TrustRadius, "ASC Timetable Reviews & Ratings 2025," 2025. [Online]. Available: https://www.trustradius.com/products/asc-timetable/reviews
4. Woldia University, "Woldia University Official Website," [Online]. Available: https://wldu.edu.et
5. Schwalbe, K. (2015). *Information Technology Project Management* (8th ed.). Cengage Learning.
6. Wiegers, K., & Beatty, J. (2013). *Software Requirements* (3rd ed.). Microsoft Press.

APPENDIX

Interview and Questioners

For Department Heads

Understanding the Current System

1. Can you describe how you currently prepare the class schedule for your department?
2. What are the biggest challenges you face when using the existing ASC timetable software?
3. How do you currently collect availability from instructors?
4. Have you ever experienced room or time conflicts? If so, how do you resolve them?
5. How do you inform students and teachers when a schedule changes?

Identifying Problems and Needs

6. What do you wish the current scheduling system could do better?
7. Is it difficult for you to make schedule changes once the timetable is finalized?
8. Do you receive feedback or complaints from students or teachers about schedules?
9. Would having real-time notifications help in your daily work?

Feedback on the Proposed Web-Based System

10. Would it help if instructor could enter their availability directly online?
11. How useful would it be if you could see possible schedule conflicts before finalizing the timetable?
12. Do you think students and instructor being able to see their schedule online will reduce confusion?
13. Would having an automatic notification system make your job easier?

Expectations and Improvements

14. How much time do you think could be saved by using an automated scheduling tool?

15. Would you be willing to use a web-based system that doesn't need installation and can be accessed from anywhere?
16. How important is it for you to generate reports about room usage and instructor workload?
17. Would you prefer a system where you can test a schedule before finalizing it?