Zewail City of Science, Technology and Innovation
University of Science and Technology
School of Computational Sciences and Artificial Intelligence

CSAI 203 - Fall 2025

Introduction to Software Engineering

Unify

Phase 2: Software Requirements Specification

Team Number: #27

Team Members:

Ahmed Moatasem 202300917
Jana Mahmoud 202301597
Mohamed Hatem 202301610
Karim Wael 202202212
Ali Mohab 202300786

Representative Contact info: s-ahmed.momtaz@zewailcity.edu.eg

Software Requirements Specification (SRS) Document Outline:

1. Introduction

1.1 Purpose:

The purpose of this project is to develop **Unify**, an integrated, AI powered student portal designed to centralize academic services, streamline student workflow, and enhance productivity across universities. The system provides secure authentication, personalized schedule optimization, academic planning, course management, AI-based assistance, and real-time reminders all through a unified web interface.
Unify aims to replace scattered academic tools with a single intelligent platform that supports students, instructors, and administrators in managing courses, tasks, communication, and academic progress efficiently.

1.2 Scope

The Unify system is a **university wide web platform** designed for students, instructors, and administrative staff. It centralizes academic information and integrates intelligent tools to improve learning efficiency and academic management. The system provides features such as user authentication, AI powered schedule optimization, course enrollment, academic tracking, note summarization, natural language academic queries, calendar synchronization, task management, and communication tools.

The platform supports **Zewail City** as the primary use case but is designed to scale for **all universities**, allowing multi institution deployment with customizable modules, course structures, and role-based access.
Unify operates through an HTML/CSS/JavaScript front-end and a Flask based backend, with optional integration to external services such as Google Calendar.

1.3 Definitions, Acronyms, and Abbreviations

1.3.1 **AI**: Artificial Intelligence
1.3.2 **NLP**: Natural Language Processing
1.3.3 **TA**: Teaching Assistant
1.3.4 **LMS**: Learning Management System
1.3.5 **SRS**: Software Requirements Specification
1.3.6 **API**: Application Programming Interface
1.3.7 **RBAC:** Role-Based Access Control

1.4 References

1. University Academic Policy Manual, 2024 Edition.

2. HTML5, CSS3, and JavaScript Web Standards Documentation, World Wide Web Consortium (W3C), 2024.

3. Crawley et al. (2022), System Architecture and Product Development for Complex Systems.
4. Kossiakoff & Biemer (2020), Systems Engineering Principles and Practice.
5. TM Forum AI Maturity Model (2023).
6. Internal design specification document: Self-Service Student Tracker v1.0 Architecture Draft.
7. Zewailcity Self-Service.

1.5 Overview

Section 2 describes the overall system, its context, and constraints. Section 3 details specific requirements, use case models, and diagrams. Section 4 includes appendices with the data dictionary and glossary.

2. Overall Description

2.1 Product Perspective:

Unify is a standalone, web based student productivity platform that integrates multiple academic modules schedule optimization, task management, note summarization, enrollment, and transcript generation under a single unified interface.
It optionally synchronizes with external systems such as the **Google Calendar API** and the university **LMS** for automatic deadline and event updates.

2.2 Product Functions:

Unify will provide both traditional student portal features and intelligent AI-driven academic tools:

- **Secure User Authentication:** Log in, log out, and manage user sessions.
- **Dashboard Overview:** Display schedule, deadlines, and quick student actions.
- **AI Schedule Optimizer:** Automatically generates conflict-free academic schedules.
- **Course Catalog & Enrollment:** Browse, search, register, and drop courses.
- **Academic Calendar Sync:** Integrate deadlines and classes with Google Calendar.
- **Natural Language Query Assistant:** Answer academic questions using plain English input.

- **AI Note Summarizer:** Summarize lectures, documents, and study materials.
- **Task & Reminder Manager:** Create tasks, set reminders, and view daily planner.
- **Transcript Generator:** Produce an unofficial transcript on demand.
- **In-App Messaging:** Enable communication between students, instructors, and TAs.
- **Admin Management Tools:** Manage users, courses, roles, and academic data.

2.3 User Classes and Characteristics

Students (Primary Users): Access schedules, upload notes, manage tasks, receive reminders, and communicate with peers or TAs.
Teaching Assistants (Secondary Users): Review class schedules, post notes or announcements, and support student queries.
Instructors (Supervisory Users): Approve enrollments, update course content, and monitor academic progress.
Administrators (Tertiary Users): Manage user accounts, roles, and overall system maintenance.

2.4 Operating Environment

Frontend: HTML, CSS, JavaScript
Backend: Python (Flask Framework)
Database: MongoDB or MySQL
Integration: Optional Google Calendar API and LMS API
AI Engine: TensorFlow and PyTorch
Supported Browsers: Chrome, Edge, Firefox, Safari (latest versions)
Operating Systems: Cross-platform (Windows, macOS, Linux)

2.5 Design and Implementation Constraints

Technology Stack: The system must use HTML, CSS, JavaScript (front-end) and Flask/Python (backend).
Browser Support: Must run on Chrome 90+ and Firefox 88+.
Data Privacy: Must follow institutional privacy rules and **GDPR** and **data protection** laws.
AI Constraints: AI features depend on available ML libraries and computation resources.
Network Dependency: Features require stable internet connectivity.
Scalability: Must be scalable to support **at least 10,000 concurrent users**.

2.6 User Documentation

- Online user manual (HTML help section).
- In-app tooltips and guided walkthrough for new users.
- Video guides for AI features.

2.7 Assumptions and Dependencies

- Users have stable internet access.
- AI features depend on data volume and model accuracy.
- Deployment requires a hosting environment capable of running a Python/Flask application and a relational database.
- Integration with external LMS depends on available APIs.

3. Specific Requirements

3.1 Functional Requirements

FR1: System shall use Machine Learning to generate conflict-free and optimized course schedules based on user preferences.

FR2: System shall summarize uploaded lecture notes or recordings using NLP pipelines and store summaries in a Notes Dashboard.

FR3: System shall support voice and text natural language queries to retrieve user data (e.g., schedules, grades).

FR4: System shall integrate with Google Calendar and LMS to sync academic deadlines and personal events.

FR5: System shall send smart reminders for exams, assignments, and events, with countdown and snooze options.

FR6: System shall include a Pomodoro style Focus Timer to manage study sessions and record productivity statistics.

FR7: System shall allow users to add, edit, delete, and organize tasks with priority and deadline sorting.

FR8: System shall provide secure login and signup functionality with password hashing and optional two-factor authentication.

FR9: System shall enforce Role-Based Access Control (RBAC) with permissions for Students, TAs, Instructors, and Admins.

FR10: System shall allow browsing of the course catalog with filters (course title, instructor, credits, schedule time).

FR11: System shall enable students to automatically enroll in courses after approval and notify instructors/admins.

FR12: System shall generate and export unofficial academic transcripts in PDF format.

FR13: System shall provide secure in-app messaging between students, TAs, and instructors, supporting individual and group chats.

FR14: System shall provide accessibility options including dark/light modes, color-blind themes, and dyslexia-friendly fonts.

3.2 Use Case Model

3.2.1 Use Case Diagram

Diagram 1:

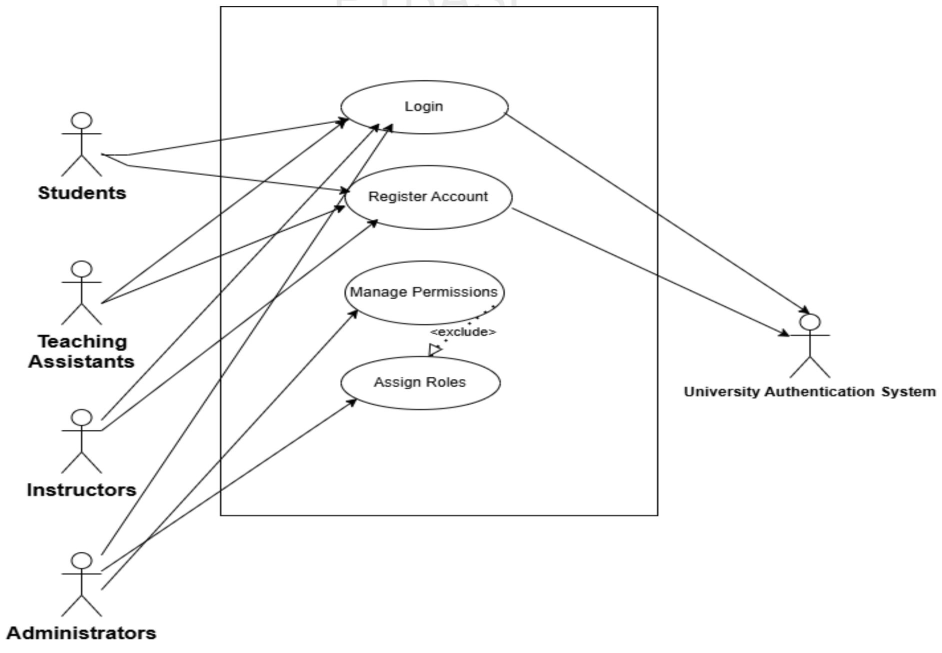


Diagram 2 :

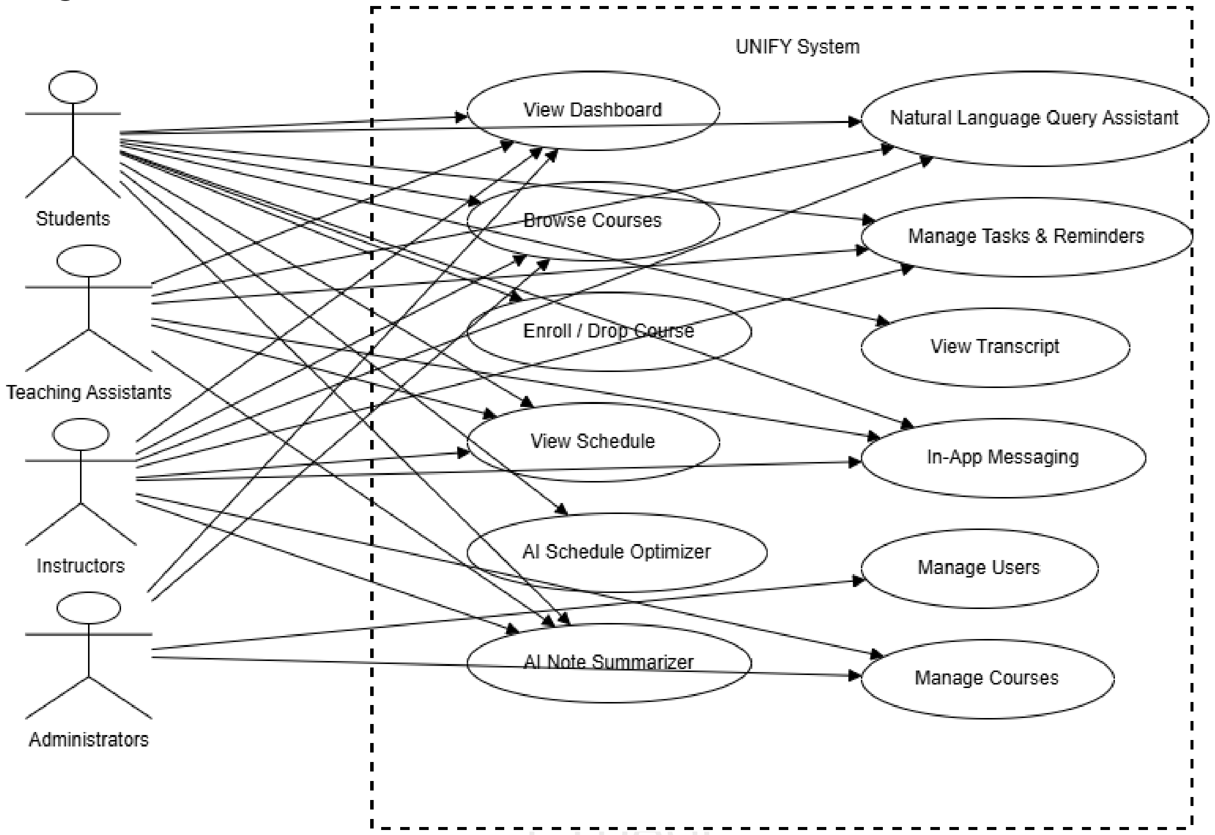
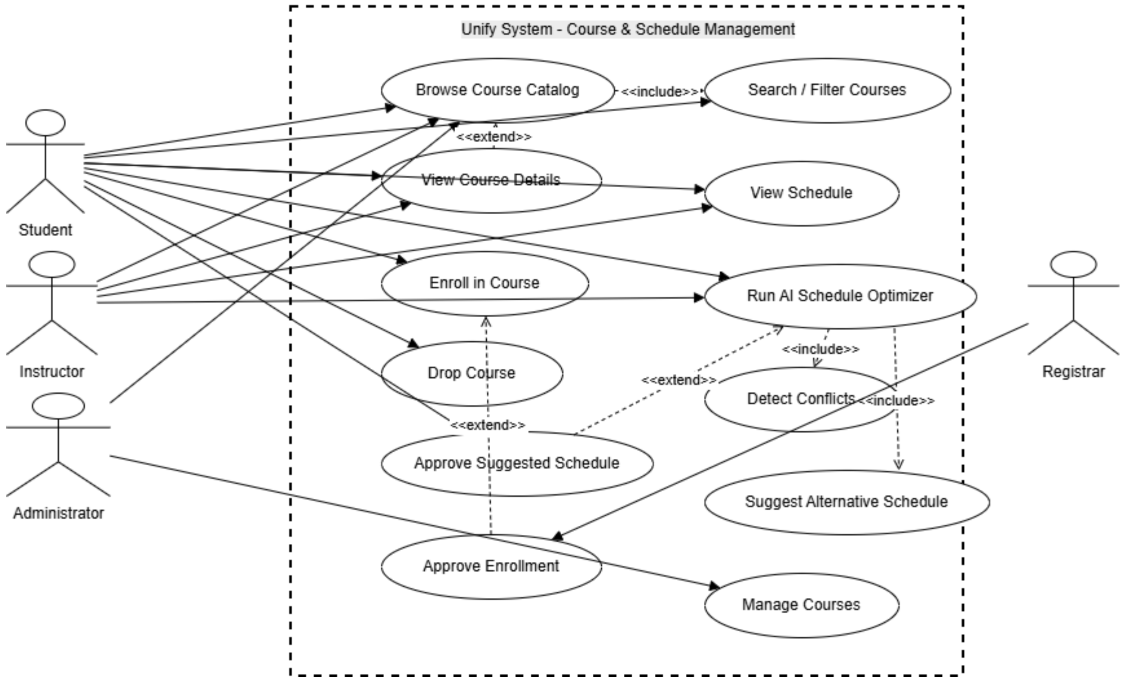


Diagram 3 :



3.2.2 Use Case Descriptions

3.2.2 Use Case Descriptions (IEEE Template)

Diagram 1: Authentication & Authorization

This use case diagram illustrates the processes involved in user authentication, registration, and high-level access control within the UNIFY system. It shows how different user types (Students, Teaching Assistants, Instructors, Administrators) gain access and how an external University Authentication System is used to validate credentials. It also covers the administrative task of configuring roles and permissions.

The following use cases are defined within this diagram:

Register Account

- Description:
  This use case allows a new user (Student, Teaching Assistant, Instructor, or Administrator) to create an account in UNIFY and link it to their university identity.
- **Relationships:**
  - Initiated by Students, Teaching Assistants, Instructors, and Administrators.
  - May consult the University Authentication System to verify the user's institutional identity.

Login

- **Description:**
  This use case allows an existing user to log in to UNIFY using their university credentials.
- **Relationships:**
  - Initiated by Students, Teaching Assistants, Instructors, and Administrators.
  - Interacts directly with the University Authentication System, which validates the credentials and returns an authentication result.

Manage Permissions

- **Description:**
  This is the primary use case for an Administrator to configure and control access rights within the system, defining what each role is allowed to do.
- **Relationships:**
  - Initiated by the Administrator.
  - <> Assign Roles as part of the permission-management process.

Assign Roles

- **Description:**
  This use case allows the Administrator to assign or change roles (e.g., Student, Teaching Assistant, Instructor, Administrator) for a user, thereby granting a predefined set of permissions.
- **Relationships:**
  - Initiated by the Administrator.
  - <>: Included as a mandatory part of Manage Permissions.

Diagram 2: UNIFY Core System Features

This use case diagram presents the main academic and productivity features of the UNIFY system. It shows how Students, Teaching Assistants, Instructors, and Administrators interact with dashboards, course browsing, AI-assisted tools, communication, and administrative management functions.

View Dashboard

- Description:
  Displays a personalized dashboard containing key information such as enrolled courses, upcoming events, reminders, and recent notifications.
- Relationships:
  - Initiated by Students, Teaching Assistants, Instructors, and Administrators.

Natural Language Query Assistant

- Description:
  Allows users to ask questions in natural language (e.g., about schedules, grades, or courses) and receive answers generated from UNIFY data.
- Relationships:
  - Initiated by Students, Teaching Assistants, Instructors, and Administrators.

Browse Courses

- Description:
  Enables users to browse the list of available courses and sections for a specific term, including basic information like title, code, and instructor.
- Relationships:
  - Initiated primarily by Students; Teaching Assistants and Instructors may also use it for reference.

Enroll / Drop Courses

- Description:
  Allows a Student to enroll in or drop courses through the UNIFY interface, subject to institutional rules such as deadlines, capacity, and prerequisites.
- Relationships:
  - Initiated by Students.

View Schedule

- Description:
  Shows a time-based view (daily/weekly/term) of classes and related academic events for the logged-in user.
- Relationships:
  - Initiated by Students, Teaching Assistants, and Instructors.

AI Schedule Optimizer

- Description:
  Uses AI to propose an optimized class schedule for a Student based on selected courses and personal preferences (e.g., days, time windows, workload).
- Relationships:
  - Initiated by Students.

AI Note Summarizer

- Description:
  Generates concise summaries and key points from lecture notes or uploaded documents to support study and revision.
- Relationships:
  - Initiated by Students, Teaching Assistants, and Instructors.

Manage Tasks & Reminders

- Description:
  Provides tools for users to create, edit, and track tasks and reminders related to coursework, grading, and other responsibilities.
- Relationships:
  - Initiated by Students, Teaching Assistants, and Instructors.

View Transcript

- Description:
  Allows a Student to view their academic record, including completed courses, grades, and summary metrics (e.g., GPA).
- Relationships:
  - Initiated by Students.

In-App Messaging

- Description:
  Enables users to send and receive messages within the UNIFY system for course-related or administrative communication.
- Relationships:
  - Initiated by Students, Teaching Assistants, Instructors, and Administrators.

Manage Users

- Description:
  Allows Administrators to create, update, deactivate, or review user accounts and essential profile information.
- Relationships:
  - Initiated by Administrators.

Manage Courses

- Description:
  Enables Administrators and Instructors to create and maintain course offerings and sections, including schedule, capacity, and assigned instructor.
- Relationships:
  - Initiated by Administrators and Instructors.

Diagram 3: Course & Schedule Management

This use case diagram focuses on the detailed management of courses and schedules within UNIFY. It involves Students, Instructors, Administrators, and the Registrar and addresses course catalog browsing, enrollment, AI-based optimization, and approval workflows.

Browse Course Catalog

- Description:
  Allows users to browse the full catalog of courses and sections available for a selected term.

- Relationships:
    - Initiated by Students, Instructors, and Administrators.
    - <> Search / Filter Courses to refine the list.
    - <> View Course Details when a specific course is selected.

Search / Filter Courses

- Description:
  Provides the ability to search and filter courses based on criteria such as department, level, time, day, and instructor.
- Relationships:
    - Initiated by users while browsing the catalog.
    - <>: Used as a mandatory part of Browse Course Catalog whenever filters are applied.

View Course Details

- Description:
  Shows detailed information about a selected course or section, including description, prerequisites, schedule, instructor, and capacity.
- Relationships:
    - Initiated when a user selects a course from the catalog.
    - <>: Acts as an optional extension to Browse Course Catalog.

Enroll in Course

- Description:
  Allows a Student to request or perform enrollment in a specific course section after viewing its details.
- Relationships:
    - Initiated by Students.
    - May lead to Approve Enrollment if manual approval is required.

Drop Course

- Description:
  Enables a Student to drop a course they are currently enrolled in, subject to drop deadlines and institutional policies.
- Relationships:
  - Initiated by Students.

View Schedule

- Description:
  Displays the Student's current schedule of enrolled courses and may also present teaching schedules for Instructors.
- Relationships:
  - Initiated by Students and Instructors.
  - Run AI Schedule Optimizer
- Description:
  Generates an optimized schedule for a Student based on selected courses and constraints, and helps resolve conflicts.
- Relationships:
  - Initiated by Students.
  - <> Detect Conflicts as part of its analysis.
  - <> Suggest Alternative Schedule to offer improved options.

Detect Conflicts

- Description:
  Analyses a proposed or current schedule to detect time clashes, overloads, and policy violations.
- Relationships:
  - Invoked by the system as part of Run AI Schedule Optimizer.
  - <>: Mandatory sub-function of the optimizer.

Suggest Alternative Schedule

- Description:
  Suggests alternative schedules that remove conflicts and better match Student preferences.
- Relationships:
  - Triggered after conflicts are found or when the Student requests alternatives.
  - <>: Optional extension of Run AI Schedule Optimizer.

Approve Suggested Schedule

- Description:
  Allows Instructors, Administrators, or the Registrar to review and approve or reject a Student’s proposed (often AI-generated) schedule.
- Relationships:
  - Initiated by Instructors, Administrators, or the Registrar.

Approve Enrollment

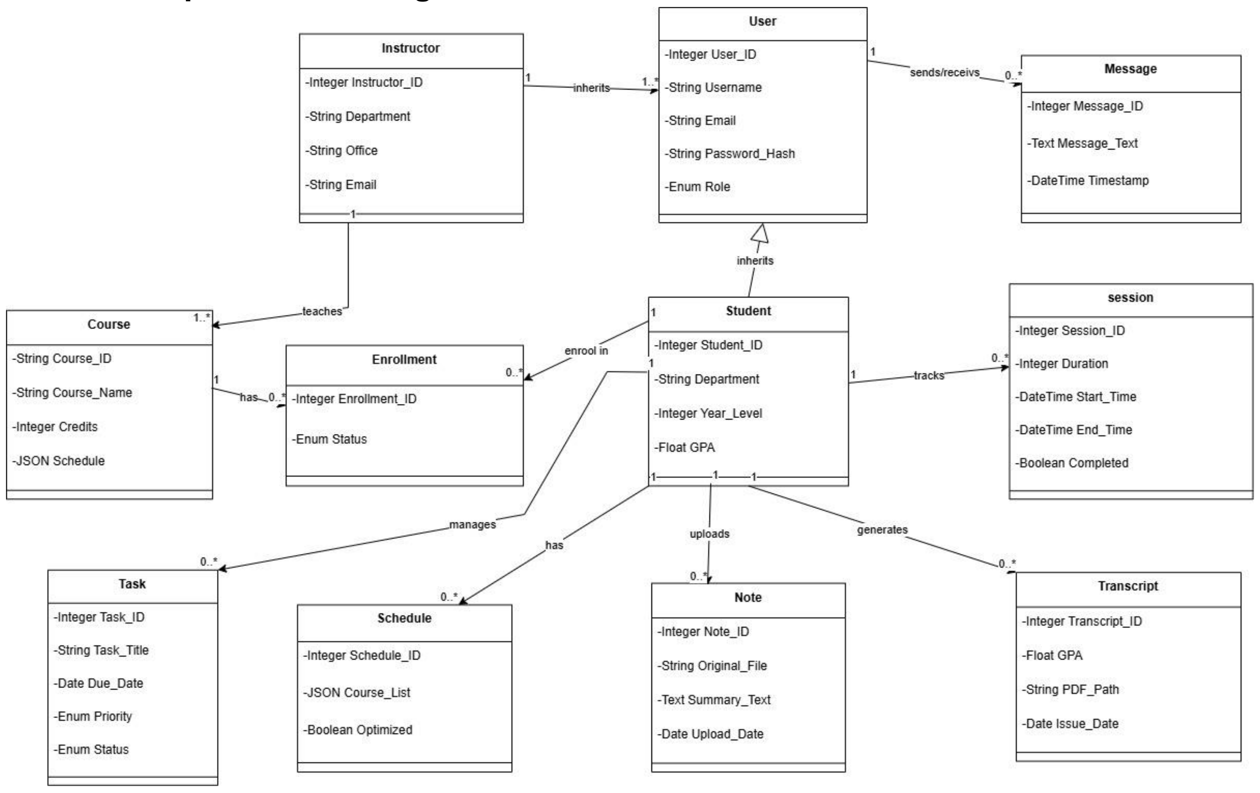
- Description:
  Enables academic staff to approve or deny individual enrollment requests that require manual review (e.g., overrides, capacity issues).
- Relationships:
  - Initiated by Instructors, Administrators, or the Registrar.
  - Often follows from Enroll in Course when approval is required.

Manage Courses

- Description:
  Provides detailed control over course and section configuration within the Course & Schedule Management subsystem (times, rooms, capacity, rules).
- Relationships:
  - Initiated by Administrators and Instructors.

3.3 Domain Model

3.3.1 Conceptual Class Diagram



3.3.2 Class Descriptions

Class Name	Description	Key Attributes	Associations
User	Represents a registered user account in the Unify system with authentication credentials and role-based access control.	User_ID, Username, Email, Password_Hash, Role	Inherits to Student, Inherits to Instructor, Sends/Receives Message
Student	Represents a registered user of the system who can enroll in courses, manage tasks, upload notes, and track academic progress.	Student_ID, Department, Year_Level, GPA	Enrolls in Course, Manages Task, Has Schedule, Uploads Note, Schedules Calendar_Event, Receives Reminder, Generates Transcript, Tracks Focus_Session
Instructor	Represents a teaching member who manages courses and grades assignments.	Instructor_ID, Department, Office, Email	Teaches Course

Course	Represents a course offered by an instructor and attended by students.	Course_ID, Course_Name, Credits, Schedule	Has many Students (via Enrollment), Taught by Instructor
Enrollment	Represents the association between a student and a course with enrollment status tracking.	Enrollment_ID, Status	Links Student to Course
Task	Represents a piece of work or assignment managed by students with priority and deadline tracking.	Task_ID, Task_Title, Due_Date, Priority, Status	Managed by Student
Schedule	Represents a student's personalized timetable with enrolled courses	Schedule_ID, Course_List, Optimized	Belongs to Student

	and AI-optimized time slots.		
Note	Represents uploaded lecture notes or recordings with AI-generated summaries for student reference.	Note_ID, Original_File, Summary_Text, Upload_Date	Uploaded by Student
Message	Represents communication between users within the secure in-app messaging system.	Message_ID, Message_Text, Timestamp	Sent by User, Received by User
Transcript	Represents an academic transcript containing course grades and GPA, exportable in PDF format.	Transcript_ID, GPA, PDF_Path, Issue_Date	Generated for Student

Session	Represents a Pomodoro-style study session tracked for productivity analysis and time management.	Session_ID, Duration, Start_Time, End_Time, Completed	Tracked by Student

3.4 Non-Functional Requirements

NFR1: System shall respond to user actions within **2 seconds** (Performance).

NFR2: System shall maintain 99% uptime. (Reliability).

NFR3: System shall **encrypt all passwords** and sensitive data using **SHA-256** (Security).

NFR4: System shall support up to **500 concurrent users** without performance degradation (Scalability).

NFR5: System shall ensure **data synchronization accuracy** of at least **99%** between local and external calendars (Integrity).

NFR6: System shall be compatible across **desktop and mobile browsers** (Usability).

NFR7: System shall ensure **backup and recovery** of all user data daily (Availability).

3.5 External Interface Requirements:

3.5.1 User Interface:

The Unify system provides a responsive, web-based graphical user interface (GUI) designed for ease of use across devices (desktop, tablet, and mobile).

- Design Style: Clean, minimal UI using modern HTML5, CSS3, and JavaScript.
- Navigation: Sidebar and top navigation bar for quick access to modules (Schedule, Notes, Tasks, Messages, Profile).

- Accessibility: Supports dark/light modes, large-text scaling, and screen-reader compatibility.

- **User Roles:**
- Students: View schedules, tasks, summaries, and messages.
- TAs/Instructors: Manage courses, post materials, and send announcements.
- Admins: Manage users, permissions, and system configurations.

3.5.2 Hardware Interface:

Any device capable of running a modern web browser (PC, laptop, tablet, or smartphone).

3.5.3 Software Interface:

- GitHub API (commits, branches).
- Flask framework (Python) communicating with MongoDB.

3.5.4 Communication Interface:

- Protocols: HTTPS for secure web communication.
- Data Transfer: JSON format for REST API requests/responses.
- Real time Messaging: WebSocket protocol for in-app chat between users.

4. Appendices

4.1 Appendix A: Data Dictionary

User

Attribute Name	**Type**	**Description**
User_ID	int (PK)	Unique user identifier
Username	string	Login/display name
Email	varchar(100)	Used for authentication and communication
Password_Hash	Varchar(255)	Encrypted password (SHA-256)
Role	enum (Student / TA / Instructor / Admin)	Use access level

Student

Attribute Name	**Type**	**Description**
Student_ID	Integer (PK)	Unique student identifier
User_ID	Integer (PK)	
Department	Varchar(100)	Academic department
Year_Level	Integer	Current study level
GPA	Float	Cumulative GPA

Instructor

Attribute Name	Type	Description
Instructor_ID	Integer (PK)	Unique instructor identifier
User_ID	Integer (PK)	Linked to User entity
Department	Varchar(100)	Instructor’s department
Office	Varchar(100)	Office location
Email	Varchar(100)	Instructor contact email

Course

Attribute Name	Type	Description
Course_ID	Varchar(20) (PK)	Unique course code
Course_Name	Varchar(100)	Official course title
Credits	Varchar(100)	Number of credit hours
Instructor_ID	Integer(PK)	Linked to Instructor entity
Schedule	JSON	Days and times of lectures

Enrollment

Attribute Name	Type	Description
Enrollment_ID	Integer (PK)	Unique enrollment identifier
Student_ID	Integer (PK)	Linked to Student entity
Course_ID	Varchar(20)	Linked to Course entity
Status	Enum (Pending / Approved / Completed)	Enrollment Status

Task

Attribute Name	type	description
Task_ID	Integer (PK)	Unique task identifier
Student_ID	Integer (FK)	Linked to Student entity
Task_Title	Varchar(255)	Task or assignment title
Due_Date	Date	Deadline for completion
Priority	Enum (High / Medium / Low)	Task importance
Status	Enum (Pending / Completed)	Current progress state

Schedule

Attribute Name	type	description
Schedule_ID	Integer (PK)	Unique schedule identifier
Student_ID	Integer (FK)	Linked to Student entity
Course_List	JSON	List of enrolled courses and time slots
Optimized	Boolean	Indicates if AI optimizer

		applied

Note

Attribute Name	Type	description
Note_ID	Integer (PK)	Unique note identifier
Student_ID	Integer (FK)	Linked to Student entity
Original_File	Varchar(255)	Path of uploaded note file
Summary_Text	Text	NLP-generated summary content
Upload_Date	Date	Date of upload

Message

Attribute Name	Type	Description
Message_ID	Integer (PK)	Unique message identifier
Sender_ID	Integer (FK)	Linked to User entity
Receiver_ID	Integer (FK)	Linked to User entity
Message_Text	Text	Message content
Timestamp	Datetime	Time of sending

Transcript

Attribute Name	Type	Description
ipt_ID	Integer (PK)	Unique transcript identifier
Student_ID	Integer (FK)	Linked to Student entity
GPA	Float	Calculated grade point average

PDF_Path	Varchar(255)	File path of exported transcript
Issue_Date	Date	Date

Calendar_Event

Attribute Name	Type	Description
Event_ID	Integer (PK)	Unique event identifier
Student_ID	Integer (FK)	Linked to Student entity
Title	Varchar(100)	Event name
Date	Date	Event date
Time	Time	Event time
Source	Enum (Google / LMS / Manual)	Event origin

Reminder

Attribute Name	Type	Description
Reminder_ID	Integer (PK)	Unique reminder identifier
Student_ID	Integer (FK)	Linked to Student entity
Event_ID	Integer (FK)	Linked to Calendar_Event
Reminder_Time	Datetime	Scheduled reminder time
Status	Enum (Active / Snoozed / Sent)	Reminder status

Focus_Session

Attribute Name	Type	Description
Session_ID	Integer (PK)	Unique session identifier
Student_ID	Student_ID	Linked to Student entity
Duration	Duration in minutes	Datetime
Start_Time	Datetime	Datetime
End_Time	Datetime	Session end time
Completed	Boolean	Indicates session completion

4.2 Appendix B: Glossary

● **User**: Any person accessing the Unify system.

● **Dashboard**: Main page showing schedule, deadlines, and tasks.
● **Course Catalog**: List of available university courses.
● **Enrollment**: Registering or dropping a course.
● **Schedule**: Weekly view of classes and academic events.
● **AI Schedule Optimizer**: Tool that generates a conflict-free schedule.
● **Conflict Detection**: Identifies overlapping classes.
● **Note Summarizer**: AI feature that summarizes uploaded notes.
● **NLP Assistant**: Answers user questions written in natural language.
● **Task**: A user-created academic or personal item to complete.
● **Reminder**: Notification sent before a task or deadline.
● **Transcript**: Academic record showing completed courses and grades.
● **Registrar**: External role that approves student enrollment requests.
● **Admin Panel**: Interface for managing users and courses.
● **Authentication System**: University system used for login verification.
● **Frontend**: User interface built with HTML, CSS, and JavaScript.
● **Backend**: Server logic implemented using Flask (Python).
● **Database**: Storage system holding users, courses, and academic data.