POLYTECHNIC UNIVERSITY (MAUBIN)
FACULTY OF COMPUTING

SMART DETECT LEARNING MANAGEMENT SYSTEM

Object-Oriented Design and Development
+
Advanced Database System

<u>Submitted By</u>
Fourth Year (CS)
Semester (VIII)
Projected by Group (4)

2024-2025 Academic Year

PROJECT GROUP MEMBERS

Name	Roll No.
Mg Aung Khant Kyaw	MUB-1266
Mg Aung Kaung Sett	MUB-1573
Ma Eain Chit Poe	MUB-1206
Ma Kay Zin Phoo Pwint	MUB-1269
Ma Su Pan Pan Oo	MUB-1315

ABSTRACT

Smart Detect Learning Management System (SDLMS) is an end-to-end web learning application with intelligent content sensing supported by fundamental learning management features. Developed using recent state-of-the-art full-stack technologies such as React.js frontend, Express.js backend, and PostgreSQL database, SDLMS caters to modern e-learning challenges using AI-based automated content sensing and real-time collaboration features.

The platform supports three different roles of users—administrators, teachers, and students—each with custom interfaces aiding course design, assignment workflow, quiz mechanism, and academic year design. Among the innovations is the pre-installed AI detection mechanism that identifies potential AI-generated work in students' submissions automatically, encouraging academic integrity in combination with educational feedback rather than punishment.

Technical supremacy is exhibited through Docker containerization, Socket.IO real-time messaging, Redis performance optimization caching, and Prometheus metrics monitoring. It features exhaustive security in JWT authentication, role-based access control, and secure file handling. Some of its other features include automatic backup systems, material management, and department-wise chat capability.

SDLMS effectively authenticates the way through which emerging web technologies can reshape the learning experience, offering a secure, scalable environment that accommodates institutional needs with consideration for usability and academic integrity in the digitalized learning space.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1
INTRODUCTION

1.1 Introduction

The **Smart Detect Learning Management System (SDLMS)** is an intelligent, modern platform designed to facilitate enhanced teaching and learning experiences. By integrating AI-driven insights, seamless user management, and dynamic course delivery, SDLMS bridges traditional education with smart technology. Built with modular design principles, it supports adaptive learning paths, security-conscious access, and insightful analytics tailored for educators and learners alike.

1.2 Objectives

- Deliver a **robust, user-friendly platform** for managing courses, quizzes, and learning content.
- Implement **role-based access control** (admins, teachers, students) to ensure tailored experiences and secure data access.
- Facilitate **scalable deployment** suitable for educational institutions of varying sizes.
- Establish a foundation for future enhancements, such as AI-powered detection system and analytics dashboards.

1.3 Motivation

Conventional Learning Management Systems often suffer from static content delivery and limited insights. This project was motivated by the need to shift toward **smart, data-driven learning tools** that respond to learner behavior. By embedding intelligent detection and feedback mechanisms, SDLMS aims to enhance engagement, improve outcomes, and empower educators with actionable analytics, helping bridge the gap between traditional teaching and modern education trends.

1.4 System Specifications

- User Roles: Administrator, Teacher, Student—each with custom dashboards and workflows.
- Core Features: Course creation and management, quiz handling (create, grade, review), and content organization.
- Smart Detection: assignment ai detection system and dashboard pattern analysis.
- Technology Stack: Modular architecture using React, Nodejs. Scalability through RESTful APIs, secure with token-based authentication.
- UI/UX Design: Clean, responsive interface via React, optimized for all devices.

1.5 Methodology

- Requirement Analysis
- Database Design and Testing
- Backend Development and Testing

- Frontend UI Development
- Integration of Frontend and Backend
- AI Detective System Research and Training
- System Testing

CHAPTER 2
UML DIAGRAM DESCRIPTION

2.1 System Overview (System Flow Diagram)

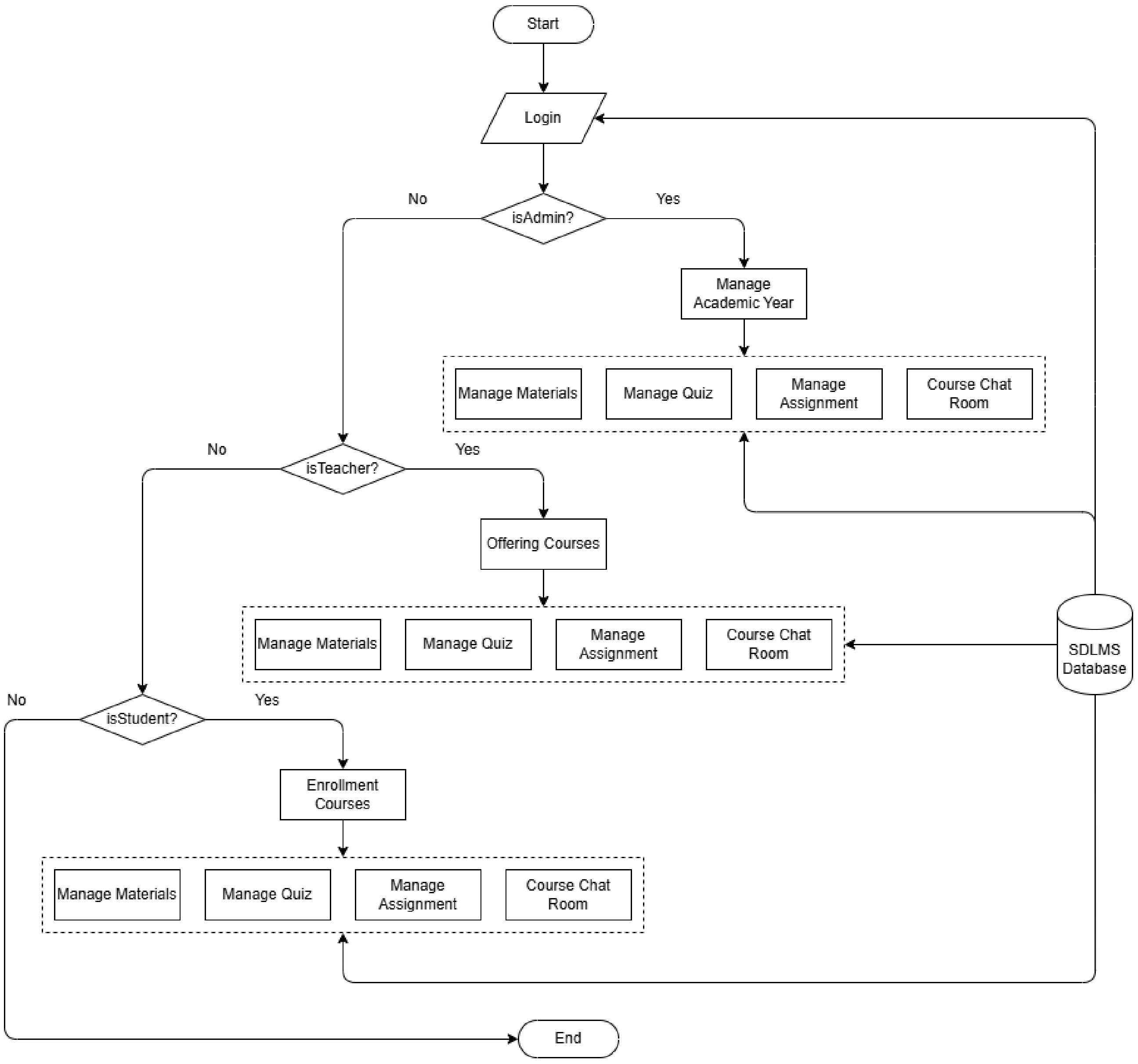


Figure 2.1 System Flow

2.2 Entity Relationship Diagram

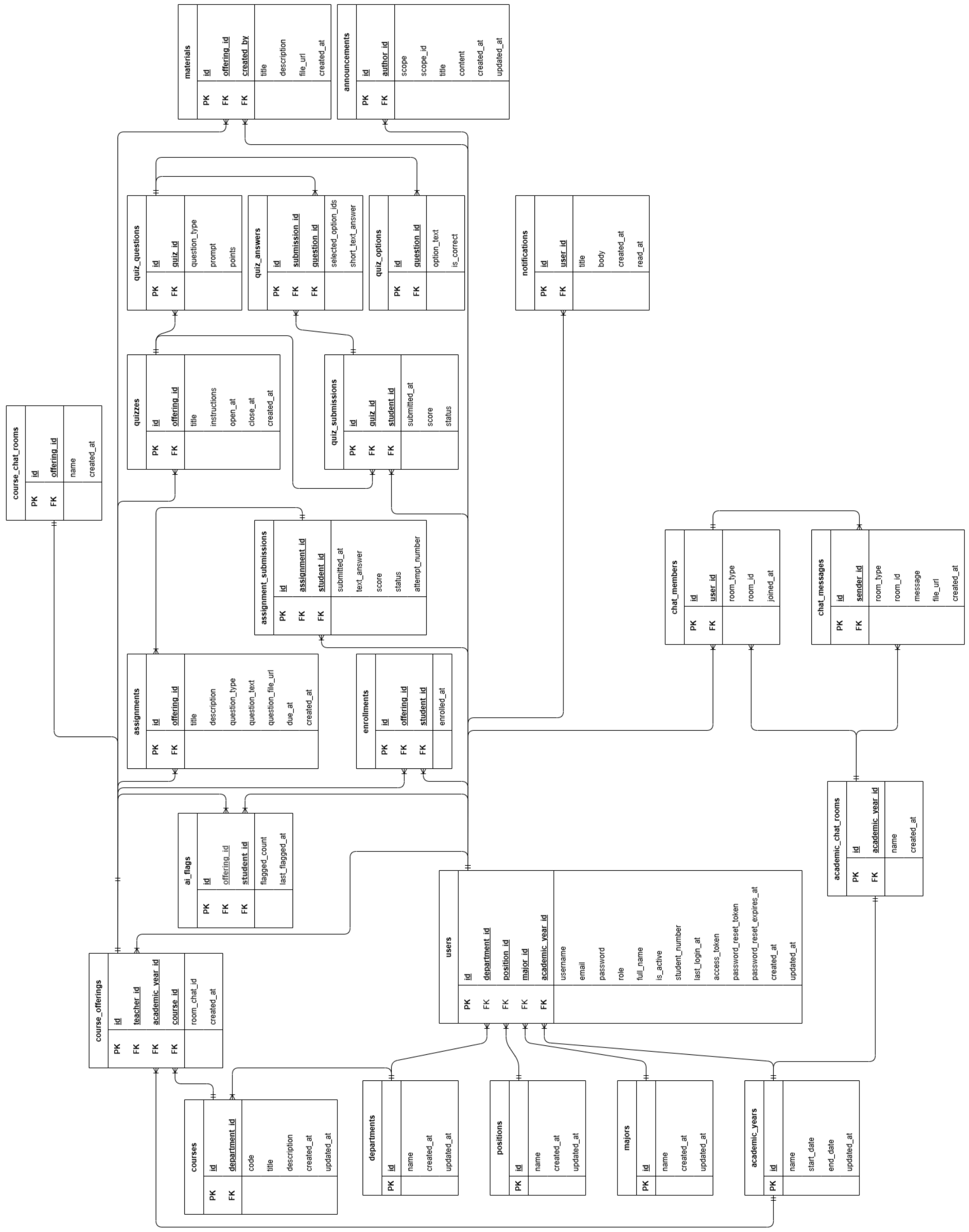


Figure 2.2 ER Diagram

2.3 Use Case Diagram

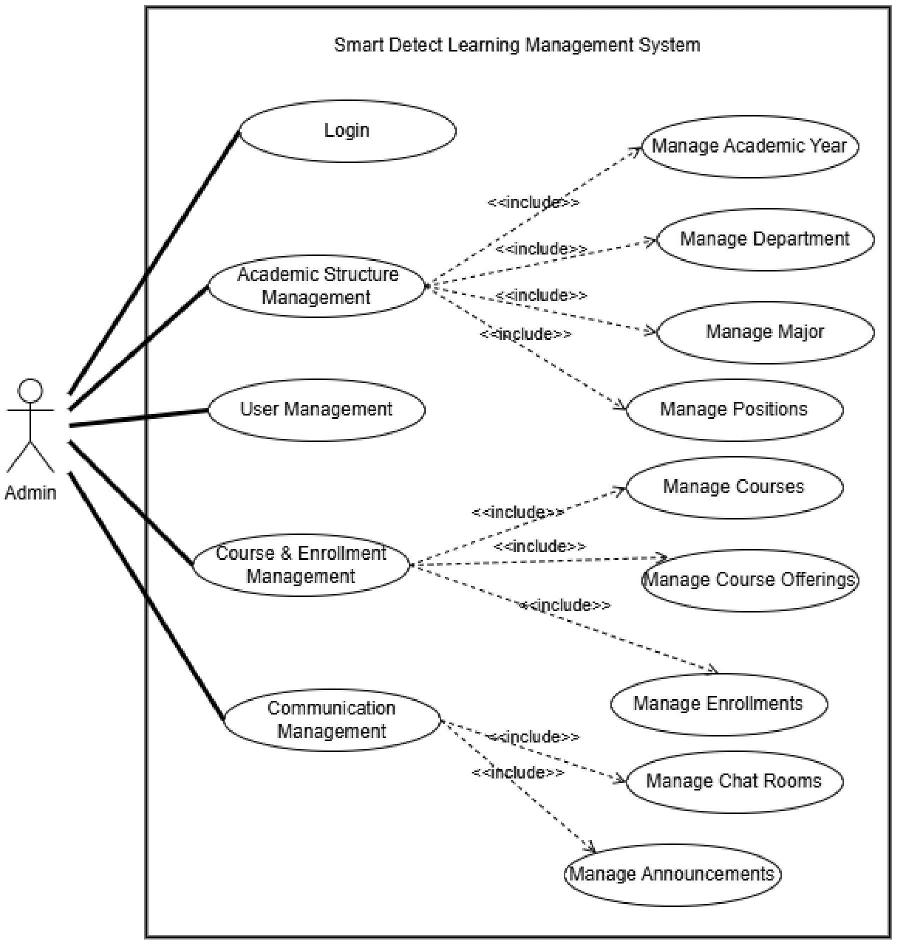


Figure 2.3 Use Case (Admin)

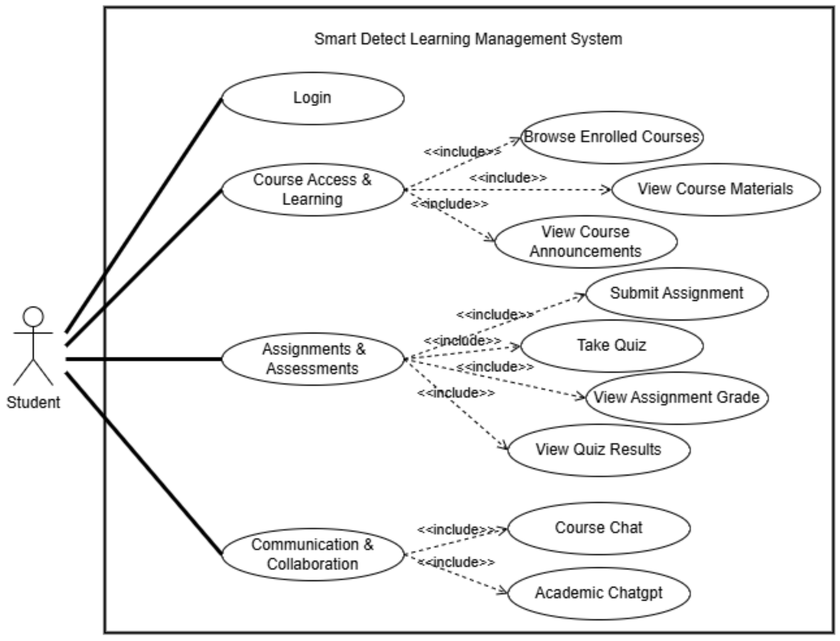


Figure 2.4 Use Case (Student)

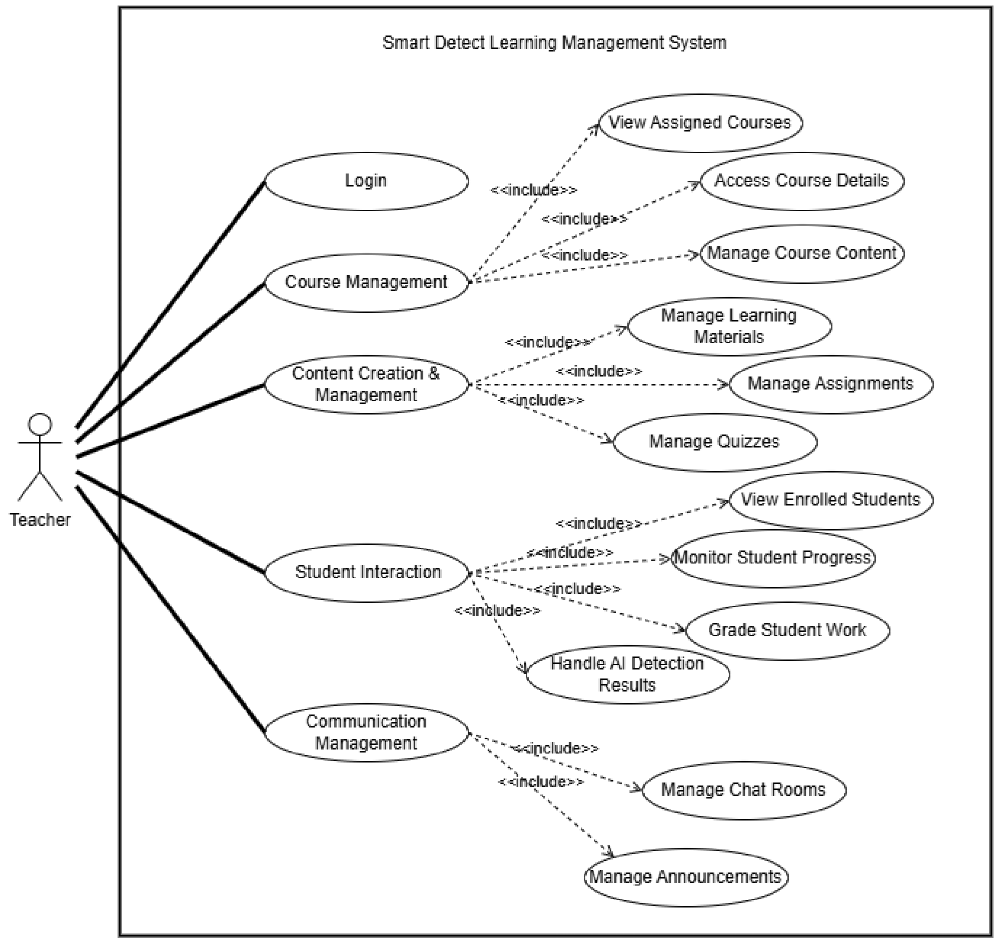


Figure 2.5 Use Case (Teacher)

2.4 Class Diagram

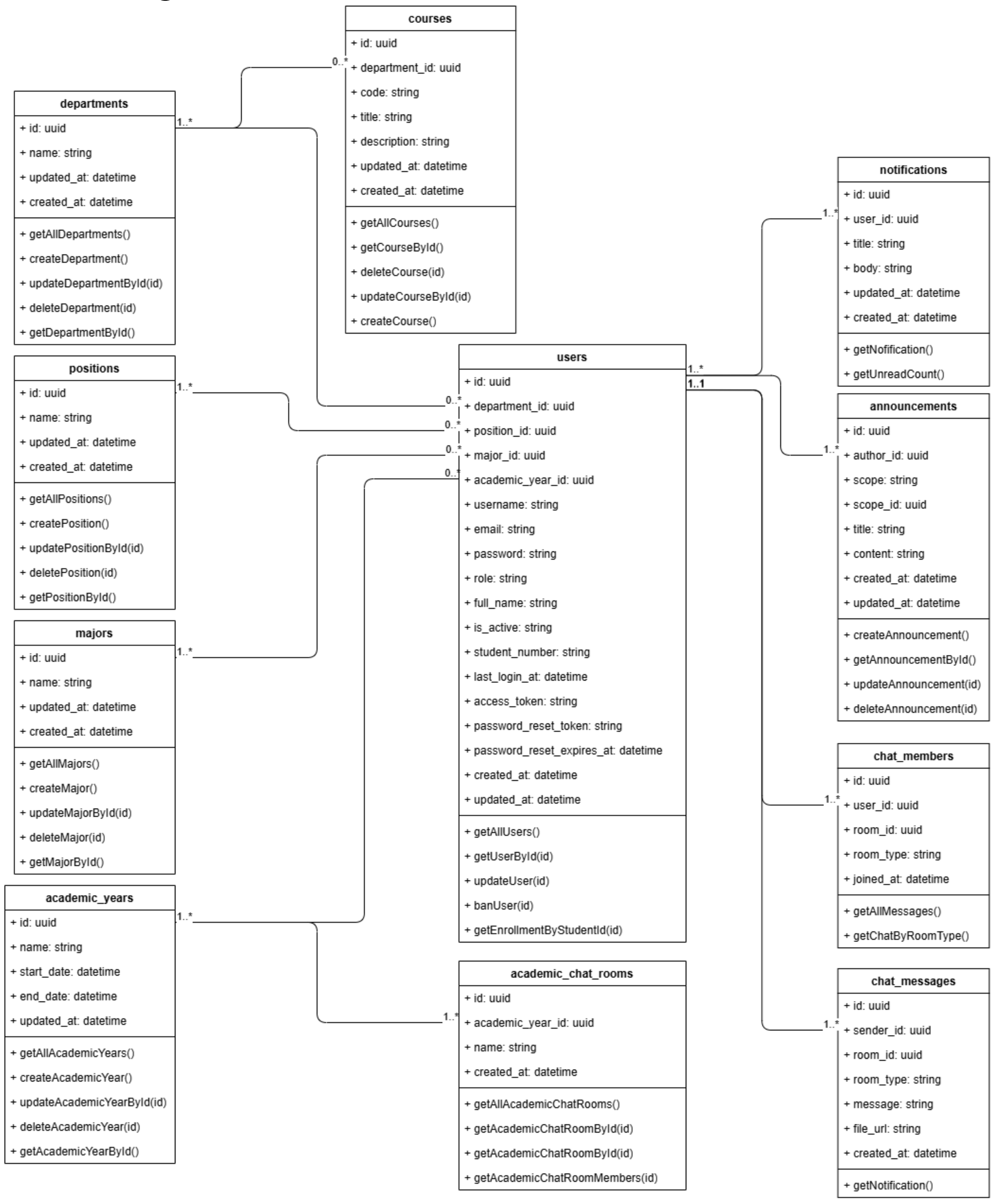


Figure 2.6 Class Diagram (Part 1)

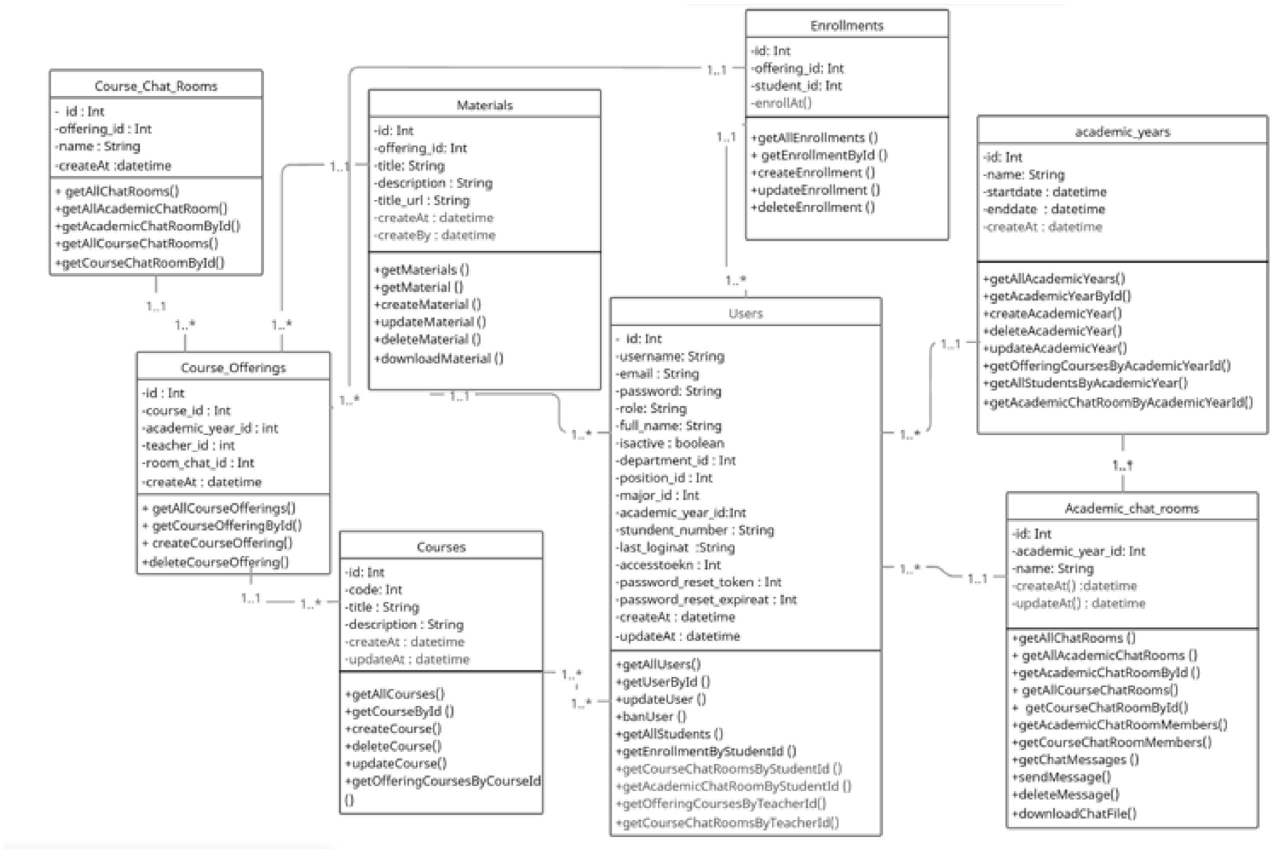


Figure 2.7 Class Diagram (Part 2)

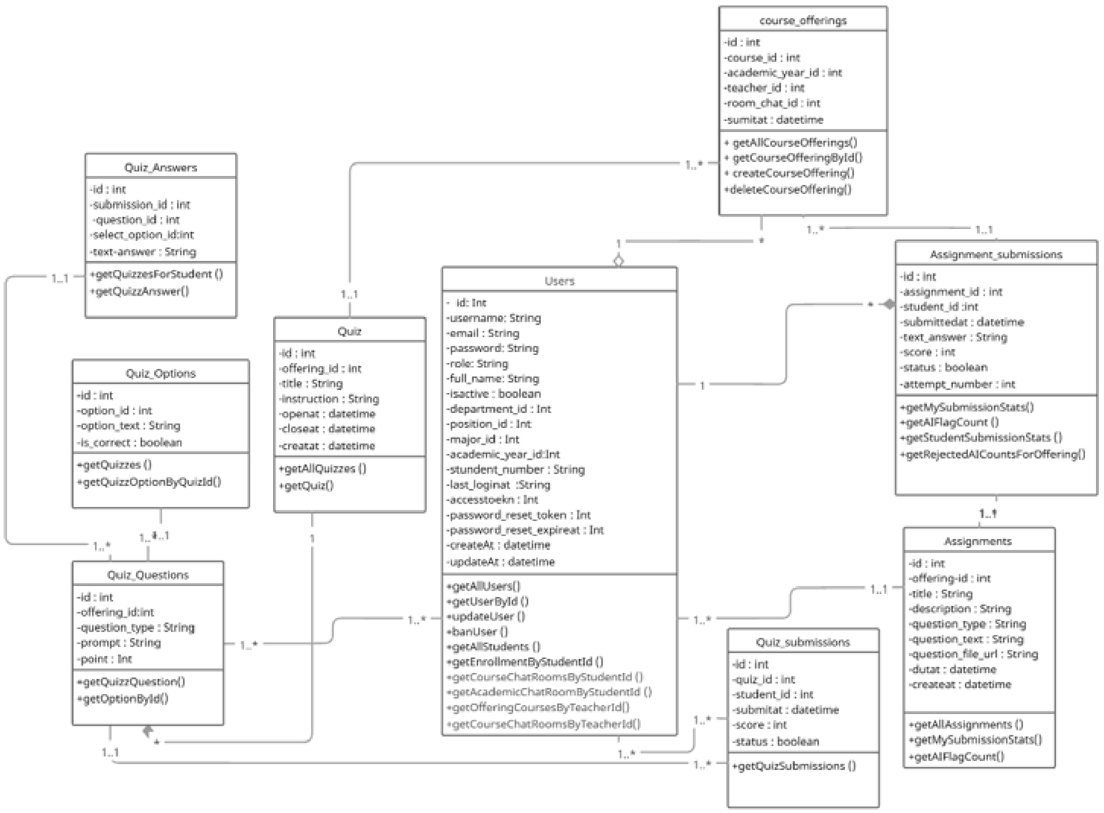


Figure 2.8 Class Diagram (Part 3)

2.5 Sequence Diagram

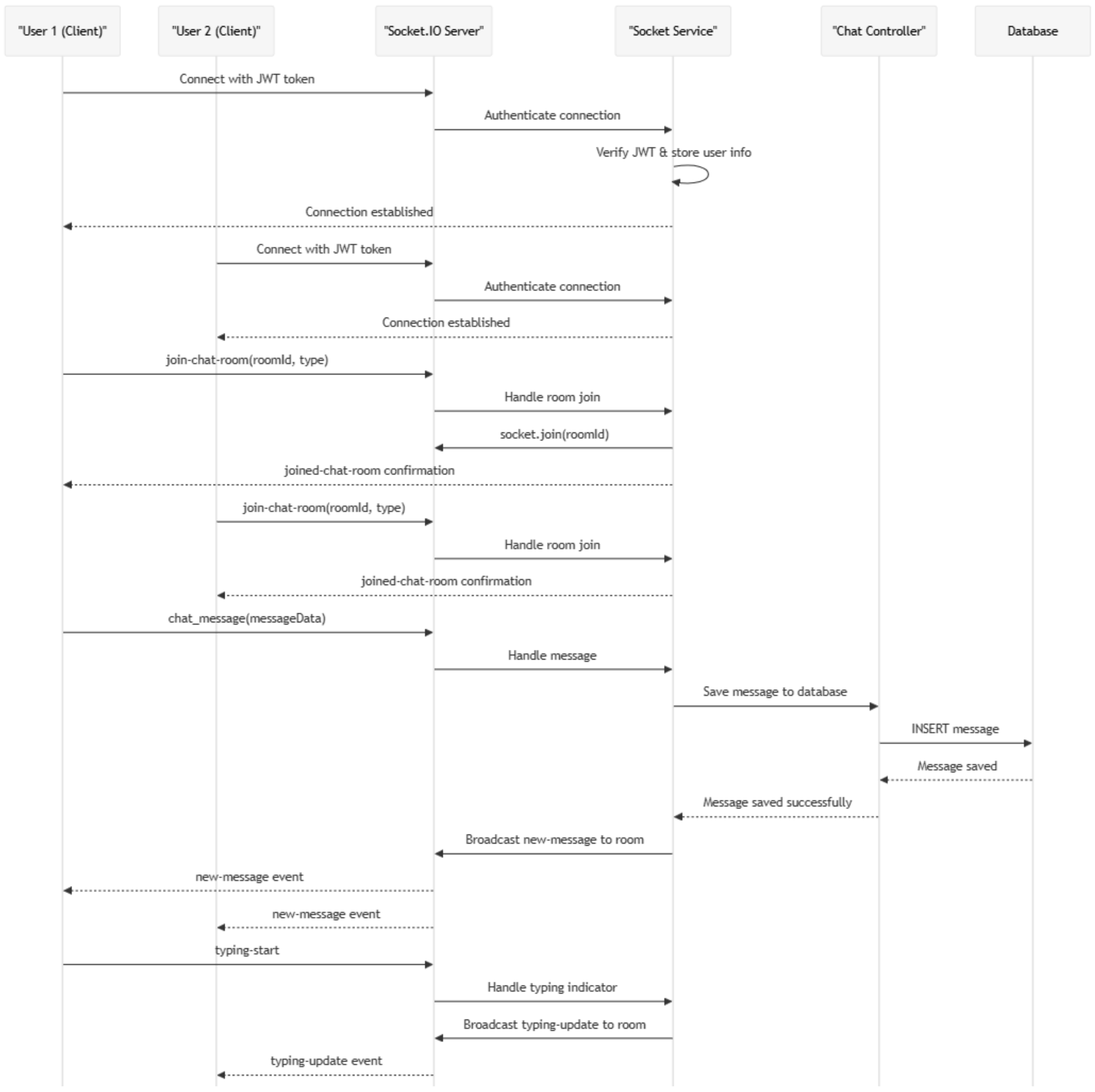


Figure 2.9 Sequence Diagram (Chatting)

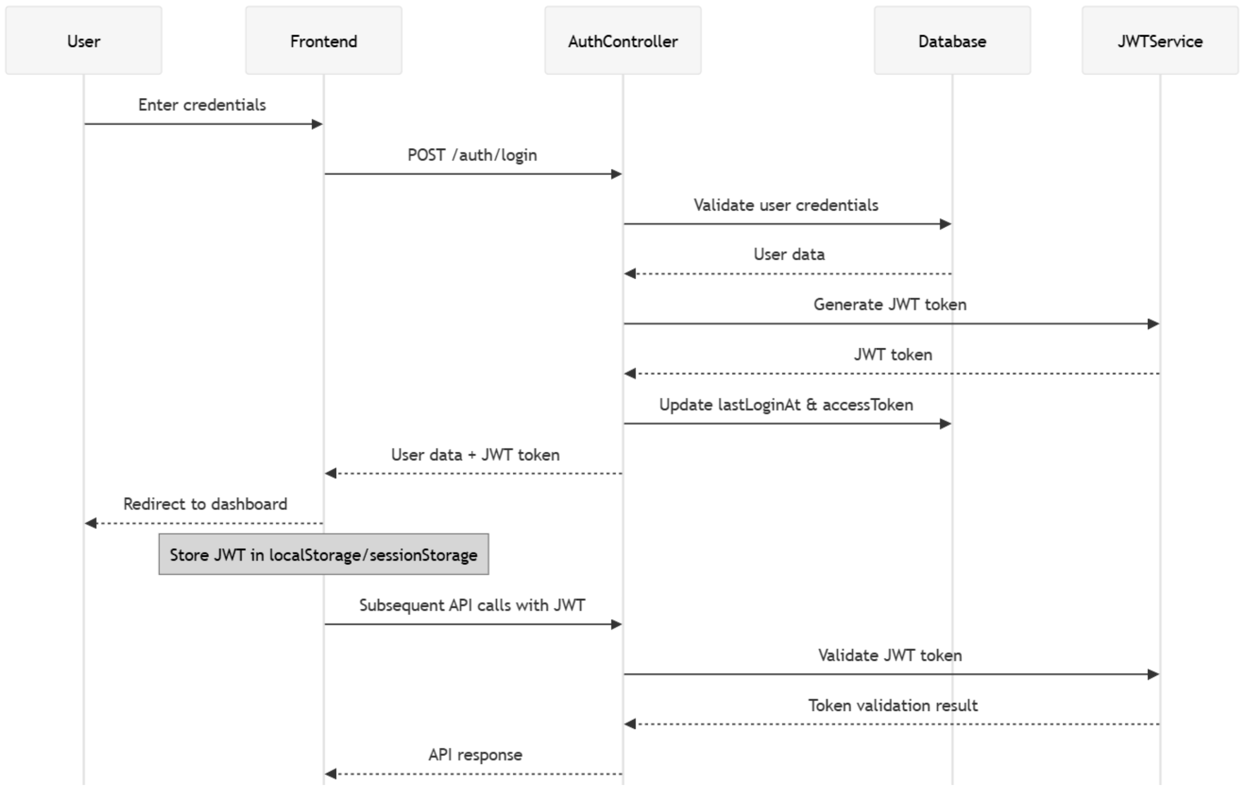


Figure 2.10 Sequence Diagram (Login)

CHAPTER 3

THEORY BACKGROUND

3.1 Database Architecture

The Smart Detect Learning Management System implements a comprehensive, multi-service database architecture utilizing PostgreSQL 17-alpine as the primary database management system, complemented by Redis 7-alpine for high-performance caching and pgAdmin 4 for administrative management. The entire database infrastructure is containerized using Docker Compose, providing automated backup systems, persistent data storage, and scalable deployment capabilities. This system employs a **shared-nothing architecture** because each service operates independently with its own dedicated resources – the PostgreSQL container maintains its own memory space, storage volumes, and processing capabilities without sharing these resources with other database instances, while Redis functions as a separate caching layer with isolated data storage, enabling horizontal scalability by adding more independent service instances as needed without resource contention or complex coordination mechanisms. This architecture ensures data integrity through ACID-compliant transactions, optimizes performance through intelligent caching strategies, and maintains operational reliability through health monitoring, automatic restarts, and comprehensive backup retention policies that support both development and production environments while facilitating easy scaling and maintenance operations.

3.1.1 Database Management System

The Smart Detect Learning Management System utilizes **PostgreSQL 17** as its primary database management system. PostgreSQL was chosen for its:

- **ACID Compliance**: Ensures data integrity and consistency
- **Advanced Features**: Support for JSON, arrays, and custom data types
- **Scalability**: Handles concurrent users and large datasets efficiently
- **Extensibility**: UUID support and custom enums for type safety

3.1.2 Database Schema Design

The database follows a **normalized relational model** with the following key characteristics:

Core Entities:

- **Users**: Multi-role system (Admin, Teacher, Student) with role-specific attributes
- **Academic Structure**: Departments, Positions, Majors, Academic Years
- **Course Management**: Courses, Course Offerings, Enrollments
- **Learning Content**: Materials, Assignments, Quizzes
- **Communication**: Chat Messages, Chat Rooms, Notifications
- **Assessment**: Quiz/Assignment Submissions, AI Flags

Data Types and Constraints:

- **UUID Primary Keys**: Ensures uniqueness across distributed systems
- **Enum Types**: Type-safe role and status management
- **Timestamp Tracking**: Created/updated timestamps for audit trails

- **Referential Integrity**: Foreign key constraints with cascade operations
- **Unique Constraints**: Prevents duplicate enrollments and submissions

3.1.3 Database Containerization

The database runs in **Docker containers** with:

- **PostgreSQL 17 Alpine**: Lightweight production database
- **pgAdmin 4**: Web-based administration interface
- **Automated Backups**: Daily, weekly, and monthly backup retention
- **Environment Configuration**: Secure credential management via .env files

3.1.4 Caching Layer

Redis 7 introduced as a high-speed cache for read-heavy operations:

- Aggregated academic and course chat rooms with member lists (120s TTL)
- Chat message pages (10s TTL) to reduce rapid repeat queries during scrolling
- Pattern-based invalidation after message create/delete
- Startup cache warming pre-populates hot aggregates
- Graceful degradation: DB path continues if Redis unreachable

3.2 Application Architecture

The Smart Detect Learning Management System employs a modern three-tier application architecture comprising a React.js frontend, Express.js backend API, and containerized database layer, implementing a microservices-oriented design that separates concerns while maintaining cohesive functionality. The architecture follows RESTful API principles with real-time WebSocket communication capabilities, role-based access control, and intelligent AI integration for academic integrity monitoring. This design ensures scalability, maintainability, and security through clear separation between presentation, business logic, and data persistence layers, while supporting cross-platform compatibility and future enhancement capabilities through modular component design and standardized communication protocols.

3.2.1 System Architecture Pattern

The system implements a **3-tier architecture**:

- Presentation Layer (Frontend):
    - React.js 18: Component-based UI framework
    - Tailwind CSS: Utility-first styling framework
    - Socket.IO Client: Real-time communication
    - Role-based Routing: Dynamic navigation based on user roles
- Business Logic Layer (Backend):
    - Node.js: JavaScript runtime environment
    - Express.js: Web application framework
    - TypeScript: Type-safe JavaScript development
    - Socket.IO Server: WebSocket communication server
- Data Access Layer:

    - Drizzle ORM: Type-safe database operations
    - PostgreSQL Driver: Direct database connectivity
    - Connection Pooling: Efficient database resource management

3.2.2 API Architecture

The backend follows RESTful API principles:

- **Resource-based URLs:** `/api/users`, `/api/courses`
- **HTTP Methods:** GET, POST, PUT, PATCH, DELETE
- **Status Codes:** Proper HTTP response codes
- **JSON Communication:** Standardized data exchange format

3.2.3 Real-time Communication

Socket.IO enables real-time features:

- **Chat Messaging:** Instant message delivery
- **Room Management:** Course and academic year chat rooms
- **File Sharing:** Real-time file upload notifications

3.2.4 Security Architecture

- **JWT Authentication:** Stateless token-based authentication
- **Role-based Access Control:** Granular permission system
- **Password Hashing:** bcrypt encryption for user passwords
- **CORS Configuration:** Cross-origin request security
- **Helmet.js:** Security headers and protection

3.2.5 Caching Architecture

Aspect	Approach
Key Names	Name spaced: `chatrooms:academic:withMembers`, `chatmessages:TYPE:ROOM:limit=..:offset=..`
TTL Strategy	120s (room aggregates), 10s (paged messages)
Invalidation	Pattern delete on message write/delete
Warming	Asynchronous warm on startup
Metrics	Hits, misses, sets, deletes, hitRatio
Fallback	Auto host fallback (service → 127.0.0.1 → localhost)

Table 3.1 Caching Architecture

3.2.6 Observability Enhancements

- `/api/cache/metrics` endpoint for runtime cache stats
- Optional verbose logging via `CACHE_LOG=1`
- Planned: Prometheus / OpenTelemetry integration

3.2.7 Reliability Additions

- Non-blocking cache warm-up
- Pattern invalidation to avoid stale aggregates
- Defensive Redis error handling with host fallback

3.3 Database Transaction Model

The Smart Detect Learning Management System implements a comprehensive database transaction model built on PostgreSQL's ACID (Atomicity, Consistency, Isolation, Durability) compliance to ensure data integrity across complex educational workflows involving user management, course enrollment, assessment submission, and real-time communication. The transaction model employs isolation levels ranging from Read Committed for standard operations to Serializable for critical academic integrity processes, with sophisticated error handling and rollback mechanisms that maintain system consistency during concurrent user activities such as simultaneous quiz submissions, bulk enrollment operations, and real-time chat message processing. This approach guarantees that educational data remains accurate and reliable while supporting high-concurrency scenarios typical in academic environments during peak usage periods.

3.3.1 ACID Properties Implementation

Atomicity:

- **Transaction Boundaries:** All related operations succeed or fail together
- **Assignment Submissions:** AI detection and grading as single transaction
- **User Registration:** Profile creation and entering chat room atomically

Consistency:

- **Foreign Key Constraints:** Maintain referential integrity
- **Check Constraints:** Validate enum values and business rules
- **Unique Constraints:** Prevent duplicate data entries

Isolation:

- **Connection Pooling:** Manages concurrent database connections
- **Transaction Isolation Levels:** Prevents dirty reads and phantom reads
- **Optimistic Locking:** Handles concurrent updates safely

Durability:

- **WAL (Write-Ahead Logging):** Ensures committed transactions persist
- **Automated Backups:** Daily, weekly, and monthly backup strategy
- **Point-in-time Recovery:** Restore capability for data protection

3.3.2 Transaction Patterns

Read Transactions:

```
-- User authentication and profile retrieval
SELECT * FROM users WHERE email = ? AND is_active = true;
```

Write Transactions:

```
-- Assignment submission with AI detection
BEGIN;
INSERT INTO assignment_submissions (…) VALUES (…);
UPDATE ai_flags SET flagged_count = flagged_count + 1 WHERE …;
INSERT INTO notifications (…) VALUES (…);
COMMIT;
```

Complex Transactions:

- **Course Enrollment:** Student enrollment + chat room membership
- **Quiz Submission:** Answer recording + score calculation + notification
- **File Upload:** File storage + database record + permission validation

3.3.3 Concurrency Control

- **Row-level Locking:** Prevents concurrent modification conflicts
- **Optimistic Concurrency:** Version-based conflict detection
- **Connection Pooling:** Manages database connection lifecycle
- **Deadlock Prevention:** Consistent lock ordering strategies

3.3.4 Future Consistency Enhancements (Planned)

- Version columns for high-contention tables (assignments/materials)
- Retry wrapper for serialization errors if isolation raised
- Optional advisory locks for complex batch operations

3.4 Tools and Technologies Used

The Smart Detect Learning Management System leverages a comprehensive technology stack comprising modern web development frameworks, enterprise-grade databases, containerization platforms, and specialized educational technology tools to deliver a robust, scalable learning management solution. The technology selection prioritizes performance, security, maintainability, and developer productivity while ensuring compatibility with educational institution requirements and future enhancement capabilities. This stack integrates cutting-edge frontend frameworks with reliable backend technologies, advanced database systems, and intelligent AI services to create a cohesive platform that supports diverse educational workflows from course management to real-time collaboration and academic integrity monitoring.

3.4.1 Backend Technologies

Technology	Version	Purpose
Node.js	22.12.0	JavaScript runtime environment
Express.js	5.1.0	Web application framework
TypeScript	5.9.2	Type-safe JavaScript development

Drizzle ORM	0.44.4	Type-safe database operations
Socket.IO	4.8.1	Real-time bidirectional communication
bcrypt	6.0.0	Password hashing and encryption
jsonwebtoken	9.0.2	JWT token generation and validation
multer	2.0.2	File upload handling middleware
helmet	8.1.0	Security headers and protection
cors	2.8.5	Cross-origin resource sharing

Table 3.2 Backend Technologies

3.4.2 Database Technologies

Technology	Version	Purpose
PostgreSQL	17-alpine	Primary database management system
pgAdmin	4	Web-based database administration
Postgres-backup-local	latest	Automated scheduled logical backups
Docker	Latest	Database containerization
Docker Compose	Latest	Multi-container orchestration
Redis	7-alpine	High-speed caching layer

Table 3.3 Database Technologies

3.4.3 Frontend Technologies

Technology	Version	Purpose
React.js (Vite)	7.1.2	Component-based UI framework
Tailwind CSS	4.1.12	Utility-first CSS framework
Socket.IO Client	4.8.1	Real-time client communication
React Router	7.8.1	Client-side routing and navigation

Table 3.4 Frontend Technologies

3.4.4 Development Tools

Tool	Purpose
Drizzle Kit	Database migration and schema management
Drizzle Studio	Visual database explorer
nodemon	Development server auto-restart
ts-node	TypeScript execution environment

Table 3.5 Development Tools

3.4.5 DevOps and Deployment

Technology	Purpose
Docker	Database migration and schema management
Docker Compose	Multi-service orchestration
Environment Variables	Configuration management
Automated Backups	Data protection and recovery
Cache Warming	Reduced cold-start latency
Cache Metrics	Runtime cache performance visibility

Table 3.6 DevOps and Deployment

3.4.6 Caching and Observability

Component	Purpose
ioredis	Redis client with robust reconnection
Cache Warmer	Pre-populates common aggregates
Metrics Collector	Tracks hit/miss/set/delete ratios
Pattern Invalidation	Maintains coherence after writes

Table 3.7 Caching and Observability

3.4.7 Backup and Recovery

Aspect	Details
Tool / Image	`prodrigestivill/postgres-backup-local` (container name: backup)
Schedule	Cron-like expression via `BACKUP_SCHEDULE` env (daily at 03:00 UTC in current config)
Retention	Environment-driven: days / weeks / months (rotational purge)
Format	Compressed SQL dumps (`.sql.gz`) stored under `database/backups/` (daily / weekly / monthly folders)
Restore Script	`database/scripts/restore_latest.sh` automates selecting most recent dump
Verification	Periodic manual test restore recommended (staging instance)
Recovery Point Objective (RPO)	Within last successful scheduled backup window
Recovery Time Objective (RTO)	Depends on dump size; typical small DB < a few minutes
Future Enhancements	Point-in-time recovery (WAL archiving), offsite replication, integrity hash checks

Table 3.8 Backup and Recovery

Restore (Conceptual Steps):
1. Stop application writers (optional if restoring to fresh instance).
2. Decompress selected archive: `gunzip < backup.sql.gz`.
3. Recreate / clean target database.
4. Apply dump: `psql -U <user> -d <db> -f backup.sql`.
5. Re-run migrations if schema drift expected.
6. Validate critical tables & counts.

3.5 Performance Evaluation

The Smart Detect Learning Management System undergoes comprehensive performance evaluation through multi-dimensional analysis encompassing database performance metrics, application response times, real-time communication efficiency, and system resource utilization under various load conditions. The evaluation framework employs both synthetic benchmarking and real-world usage pattern simulation to assess system behavior during typical educational scenarios such as course enrollment periods, assignment submission deadlines, quiz sessions, and peak chat activity. Performance measurement is conducted through integrated monitoring systems including Prometheus metrics collection, Redis cache analysis, PostgreSQL query performance tracking, and custom application-level instrumentation that provides detailed insights into system bottlenecks, optimization opportunities, and scalability characteristics essential for educational institution deployment planning.

3.5.1 Database Performance Metrics

Query Performance:

- **Index Optimization**: Strategic indexing on frequently queried columns
- **Query Execution Time**: Average response time < 100ms for simple queries
- **Connection Pooling**: Maintains 10-50 concurrent connections efficiently
- **Cache Hit Ratio**: PostgreSQL buffer cache efficiency > 95%

Scalability Metrics:

- **Concurrent Users:** Supports 100+ simultaneous active users
- **Database Size:** Efficiently handles databases up to 10GB+
- **Transaction Throughput:** Processes 1000+ transactions per minute
- **Backup Performance:**
    - Full backup completion within 5 minutes for 10GB database
    - Incremental backups complete in < 2 minutes
    - 70-80% compression ratio with gzip
    - 7 daily, 4 weekly, 6 monthly retention policy
    - Web-based restore via admin interface
    - One-click restore functionality

Metric Formulas

Metric	Formula	Notes
Avg Query Latency	sum(query_time_ms) / count(queries)	Collected per endpoint or SQL label
P95 / P99 Latency	latency value at 95th / 99th percentile	Requires histogram or raw samples
Throughput (TPS)	committed_tx / seconds_window	Use PG stat or app counter
Cache Hit Ratio (App)	hits / (hits + misses)	From cache metrics endpoint
Cache Hit Ratio (PG)	(blks_hit) / (blks_hit + blks_read)	From pg_stat_database
Error Rate	error_responses / total_responses	Exclude client 4xx if desired

Table 3.9 Metric Formulas

3.5.2 Application Performance

Response Time Analysis:

- **API Endpoints:** Average response time 50-200ms
- **File Uploads:** 10MB files upload within 30 seconds
- **Real-time Messaging:** Message delivery latency < 100ms
- **Page Load Times:** Initial page load < 2 seconds

Resource Utilization:

- **Memory Usage:** Backend process consumes 100-500MB RAM
- **CPU Utilization:** Normal operation uses 5-15% CPU
- **Network Bandwidth:** Optimized for low-bandwidth environments
- **Storage Efficiency:** Compressed backups reduce storage by 70%

3.5.3 Cache Performance

Metric	Description
hits / misses	Demand distribution and effectiveness
hitRatio	hits / (hits + misses) – primary efficiency KPI
sets	Volume of new cache materializations
deletes	Pattern or key invalidations
warmDuration	Startup hydration time (logged)

Table 3.10 Cache Performance

Expected: high reuse for room membership lists; lightweight ephemeral message page caching reduces DB spikes on rapid pagination.

3.5.4 Real-time Communication Performance

Socket.IO Metrics:

- **Connection Establishment:** < 500ms connection time
- **Message Broadcasting:** Delivers to 100+ clients within 200ms
- **Room Management:** Efficient chat room scaling
- **File Transfer:** Real-time file sharing with progress tracking

3.5.5 Security Performance

Authentication Metrics:

- **JWT Token Validation:** < 10ms validation time
- **Password Hashing:** bcrypt processing time 100-300ms
- **Session Management:** Stateless token approach reduces server load
- **Access Control:** Role-based permissions evaluated in < 5ms

3.5.6 Optimization Strategies

Database Optimization:

- **Indexing Strategy:** Composite indexes on frequently joined columns
- **Query Optimization:** Efficient JOIN operations and subqueries
- **Connection Management:** Pool size tuning based on load patterns
- **Backup Optimization:** Incremental backups for large datasets

Application Optimization:

- **Caching Strategy:** In-memory caching for frequently accessed data
- **Code Splitting:** Lazy loading for improved initial load times
- **Asset Optimization:** Compressed images and minified JavaScript
- **CDN Integration:** Static asset delivery optimization

3.5.7 Backup & Recovery Performance

Backup Strategy:

- **Automated Backups:** Daily at 03:00 UTC with retention policies
- **Compression:** 70% size reduction with gzip compression
- **Web UI Restore:** Admin dashboard backup restoration
- **Recovery Time:** Full restore completes in 10-15 minutes
- **Disaster Recovery:** RTO < 30 minutes, RPO < 24 hours

3.5.8 Monitoring and Analytics

Performance Monitoring:

- **Database Monitoring:** (Planned) EXPLAIN ANALYZE sampling & pg_stat monitoring
- **Application Metrics:** \`http_requests_total\`, \`http_request_duration_seconds\`

histogram (RPS, latency P50/P90/P95/P99, error rate)

- **Cache Metrics:** `cache_hits_total`, `cache_misses_total`, `cache_sets_total`, `cache_deletes_total`, `cache_hit_ratio`
- **Runtime Metrics:** `sdlms_process_resident_memory_bytes`, `sdlms_nodejs_eventloop_lag_*`, GC duration histogram
- **Real-time Usage:** (Planned) socket gauges & counters
- **Health Checks:** `/health` (JSON) + metrics endpoint for scrape health

Capacity Planning:

- **Growth Projections:** Track RPS & active users vs. resource curves
- **Resource Scaling:** Alert on sustained memory growth & P95 latency thresholds
- **Performance Benchmarks:** Capture baseline histograms pre-release
- **Bottleneck Identification:** Combine slow route label analysis with profiling / flame graphs

3.6 AI Text Detection

The **AI Text Detection API** is a Flask-based web application that detects whether a given text is AI-generated or human-written. It uses machine learning (Logistic Regression) and natural language processing to provide predictions via a REST API and a web interface.

3.6.1 Key Features

- **Text Classification**: Predicts if input text is AI-generated or human-written.
- **Model Training**: Retrains the model using uploaded CSV datasets.
- **File Upload**: Accepts CSV uploads for new training data.
- **Model Status & Info**: Endpoints to check model status and metadata.
- **Text Preprocessing**: Cleans and normalizes text (contraction expansion, punctuation removal, etc.).
- **Persistence**: Saves trained models, vectorizers, and label mappings.

3.6.2 Architecture

- **Backend**: Python 3, Flask
- **ML Libraries**: scikit-learn (LogisticRegression, TfidfVectorizer), pandas
- **Persistence**: Pickle (model artifacts), JSON (metadata)
- **File Structure**:
    - models: Model artifacts and metadata
    - uploads: Uploaded CSV datasets
    - templates: HTML templates

3.6.3 API Endpoints

- / : Home page (web interface)
- /model/status : Model status and metadata (JSON)
- /api/info : Project and model information (JSON)

- /upload-csv : GET (upload form), POST (upload CSV)
- /train : POST (train model on latest dataset)
- /predict : POST (predict label for input text)
- Error handlers: 404, 413, 500

3.6.4 Data Flow

1. **Upload**: User uploads a CSV file with text and label columns.
2. **Training**: Model is trained on uploaded data with preprocessing.
3. **Prediction**: New text is preprocessed and classified.
4. **Persistence**: Model, vectorizer, and label mapping are saved.

3.6.5 Preprocessing Steps

- Lowercasing
- Removal of newlines and single quotes
- Expansion of common contractions
- Removal of punctuation
- Whitespace normalization

3.6.6 Model Details

- **Vectorizer**: Tfidf Vectorizer (configurable n-grams, max features, stop words)
- **Classifier**: Logistic Regression (balanced class weights, liblinear solver)
- **Label Mapping**: Supports flexible label names (e.g., "human", "ai", "0", "1")

3.6.7 Error Handling

- Handles missing files, invalid CSVs, and file size limits
- Returns informative JSON error messages

3.6.8 Security & Limits

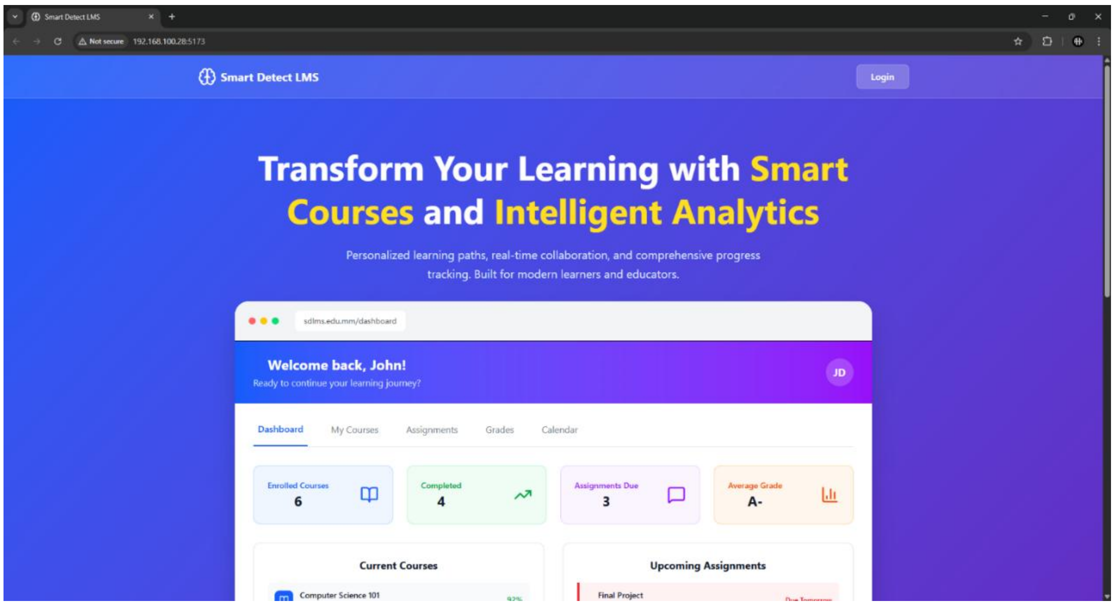
- File upload size is configurable (default 100 MB)
- Only CSV files are accepted for training

CHAPTER 4

IMPLEMENTATION

4.1 Home Page

This is a welcome landing page that highlights the advantages of the Smart Detect Learning Management System, including an interactive dashboard preview, smooth role-based authentication redirection, and a thorough platform description. Experience cutting-edge analytics, real-time collaboration tools, and intelligent course administration for administrators, teachers, and students.



Smart Detect LMS
Login
Transform Your Learning with Smart Courses and Intelligent Analytics
Personalized learning paths, real-time collaboration, and comprehensive progress tracking. Built for modern learners and educators.
Welcome back, John!
Ready to continue your learning journey?
Dashboard
My Courses
Assignments
Grades
Calendar
Enrolled Courses 6
Completed 4
Assignments Due 3
Average Grade A-
Current Courses
Upcoming Assignments

Figure 4.1 Home Page

4.2 Admin Pages

Admin can see system overview like user registration, chat room creation, quizzes and assignment upload for each month and system composition.

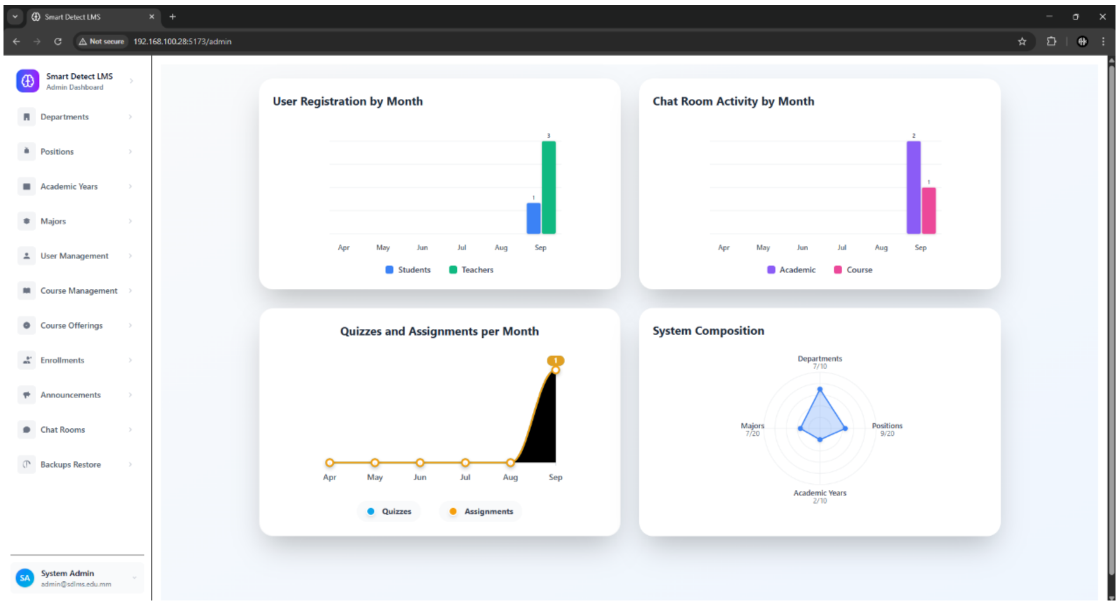
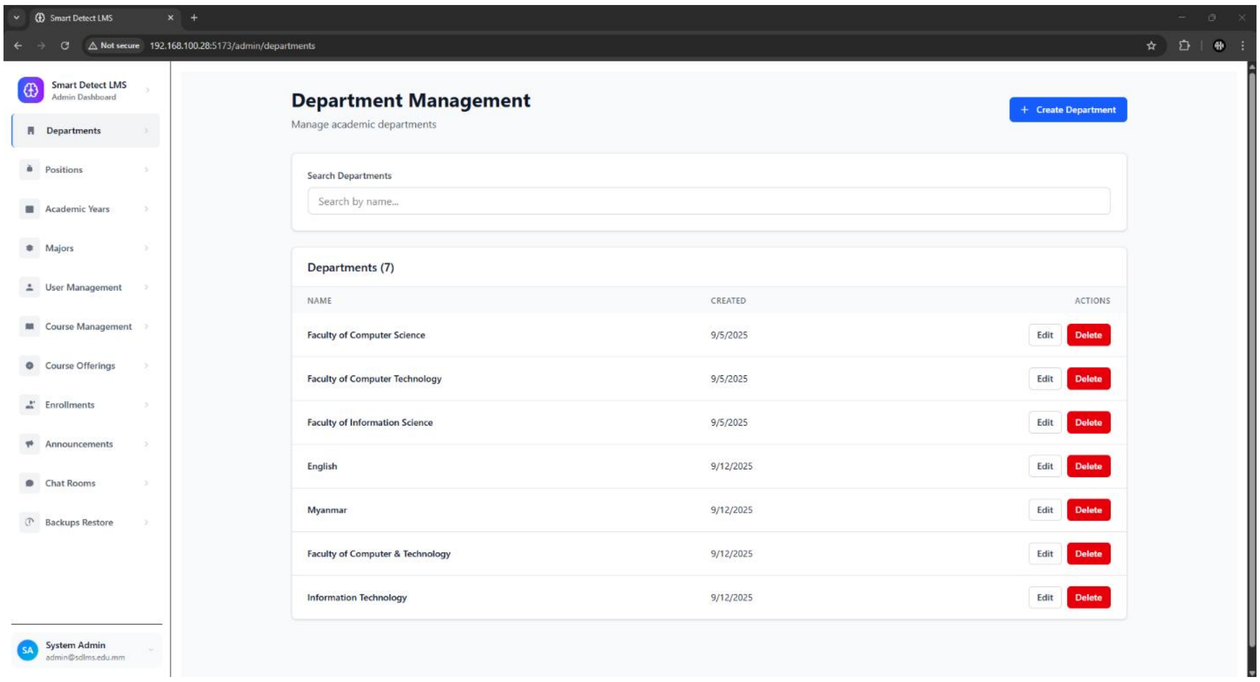


Figure 4.2 System Overview Page

On this page, the admin can manage the department. Create a new department and edit or delete existing ones.



Department Management

Manage academic departments

Search Departments

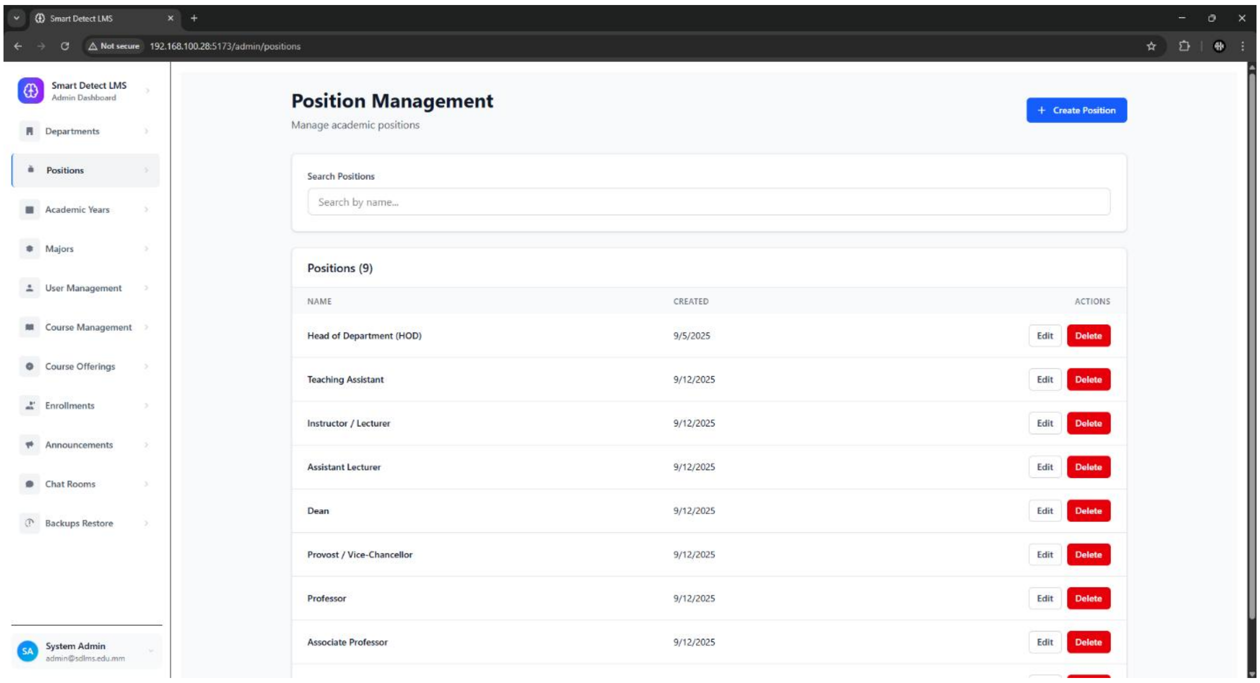
Search by name...

Departments (7)

NAME	CREATED	ACTIONS
Faculty of Computer Science	9/5/2025	Edit Delete
Faculty of Computer Technology	9/5/2025	Edit Delete
Faculty of Information Science	9/5/2025	Edit Delete
English	9/12/2025	Edit Delete
Myanmar	9/12/2025	Edit Delete
Faculty of Computer & Technology	9/12/2025	Edit Delete
Information Technology	9/12/2025	Edit Delete

Figure 4.3 Department Management Page

The bulk of this page is under the administrator's control. Change or eliminate your existing major and add new ones.



Position Management

Manage academic positions

Search Positions

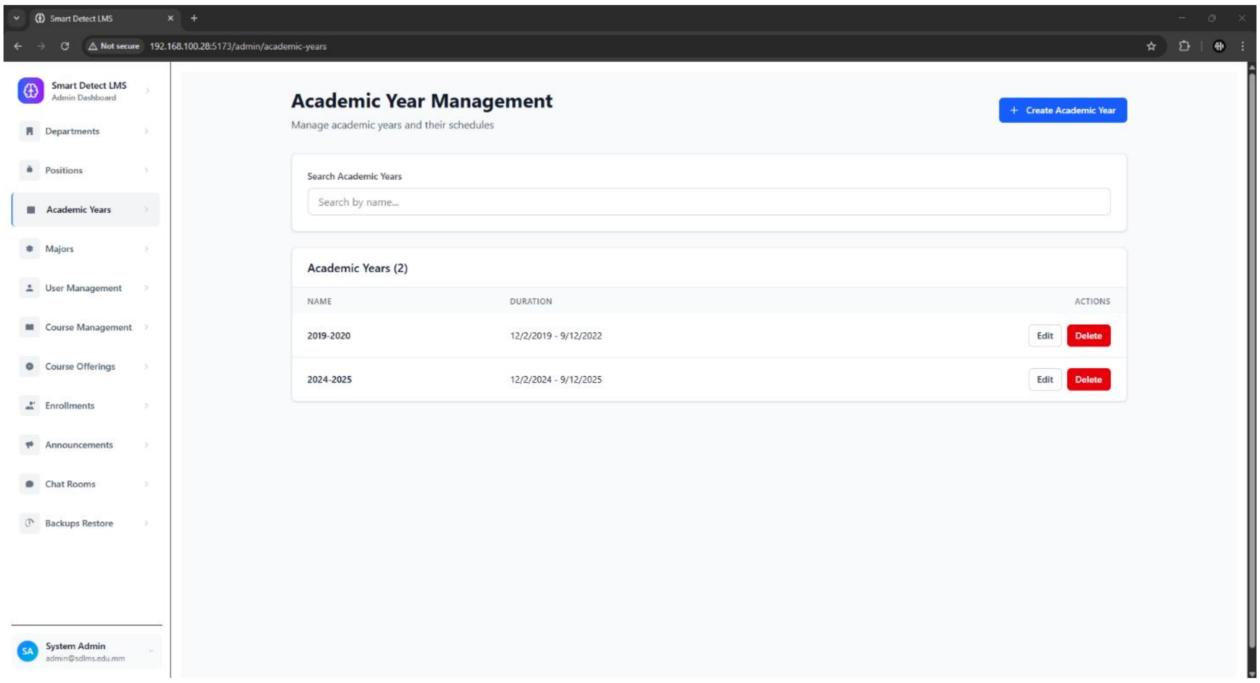
Search by name...

Positions (9)

NAME	CREATED	ACTIONS
Head of Department (HOD)	9/5/2025	Edit Delete
Teaching Assistant	9/12/2025	Edit Delete
Instructor / Lecturer	9/12/2025	Edit Delete
Assistant Lecturer	9/12/2025	Edit Delete
Dean	9/12/2025	Edit Delete
Provost / Vice-Chancellor	9/12/2025	Edit Delete
Professor	9/12/2025	Edit Delete
Associate Professor	9/12/2025	Edit Delete

Figure 4.4 Position Management Page

This is where the administrator may oversee the school year. Make a new school year and make changes or removals to previous ones. When new academic year create, new academic chat room is created.



Academic Year Management

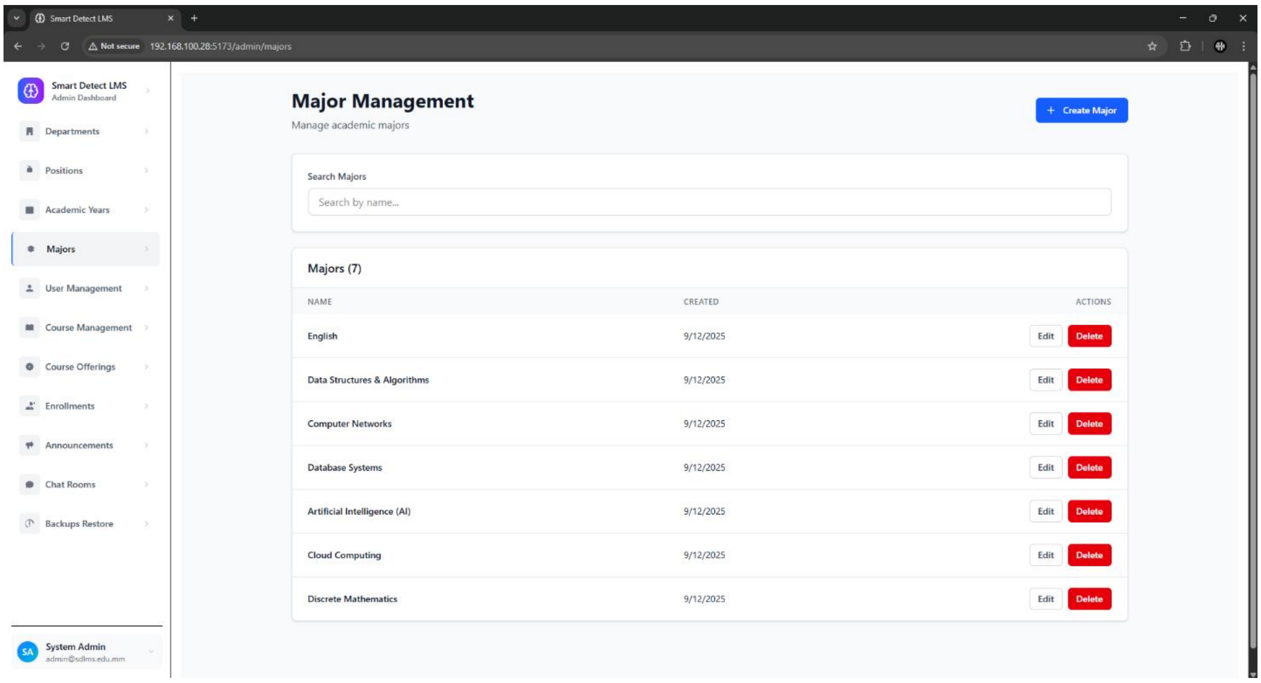
Manage academic years and their schedules

Academic Years (2)

NAME	DURATION	ACTIONS
2019-2020	12/2/2019 - 9/12/2022	Edit Delete
2024-2025	12/2/2024 - 9/12/2025	Edit Delete

Figure 4.5 Academic Year Management Page

The bulk of this page is under the administrator's control. Change or eliminate existing major and add new ones.



Major Management

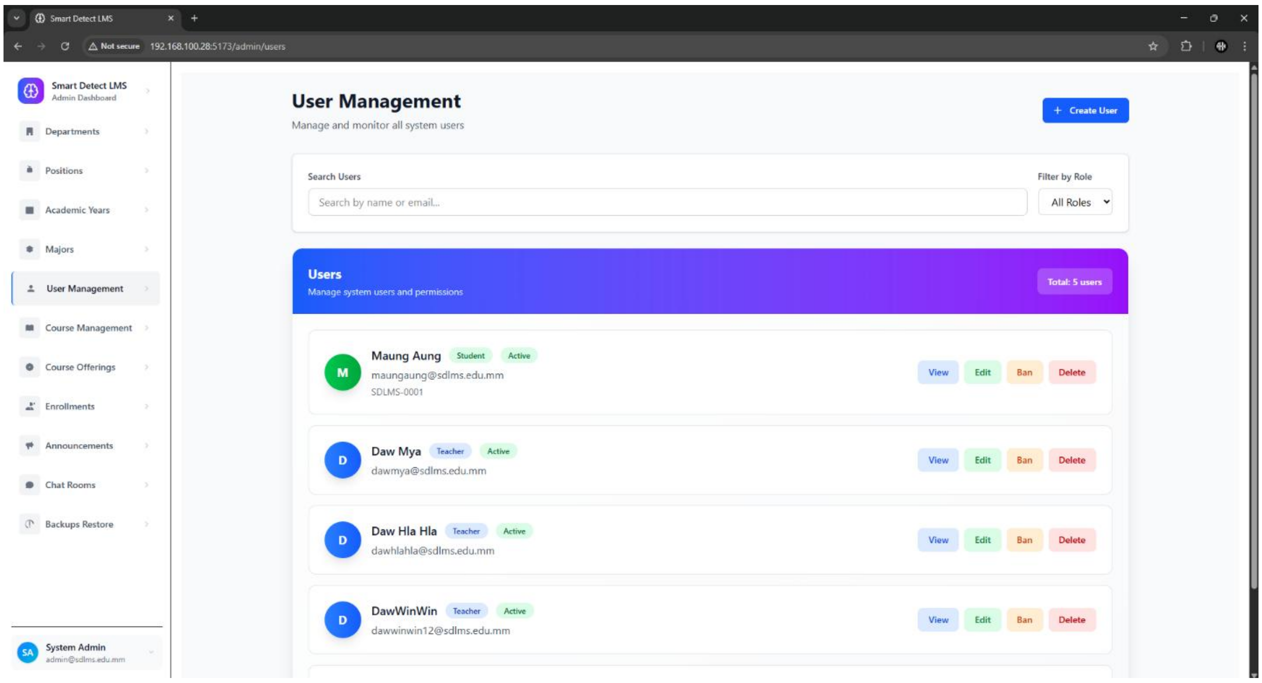
Manage academic majors

Majors (7)

NAME	CREATED	ACTIONS
English	9/12/2025	Edit Delete
Data Structures & Algorithms	9/12/2025	Edit Delete
Computer Networks	9/12/2025	Edit Delete
Database Systems	9/12/2025	Edit Delete
Artificial Intelligence (AI)	9/12/2025	Edit Delete
Cloud Computing	9/12/2025	Edit Delete
Discrete Mathematics	9/12/2025	Edit Delete

Figure 4.6 Major Management Page

On this page, admin can add new user with role (admin, teacher or student). And then edit, delete or ban. When admin create student account, check academic year and add to his academic chat room automatically.


Smart Detect LMS

Not secure 192.168.100.28:5173/admin/users

Smart Detect LMS
Admin Dashboard

- Departments
- Positions
- Academic Years
- Majors
- User Management
- Course Management
- Course Offerings
- Enrollments
- Announcements
- Chat Rooms
- Backups Restore

System Admin
admin@sdlms.edu.mm

User Management
Manage and monitor all system users

+ Create User

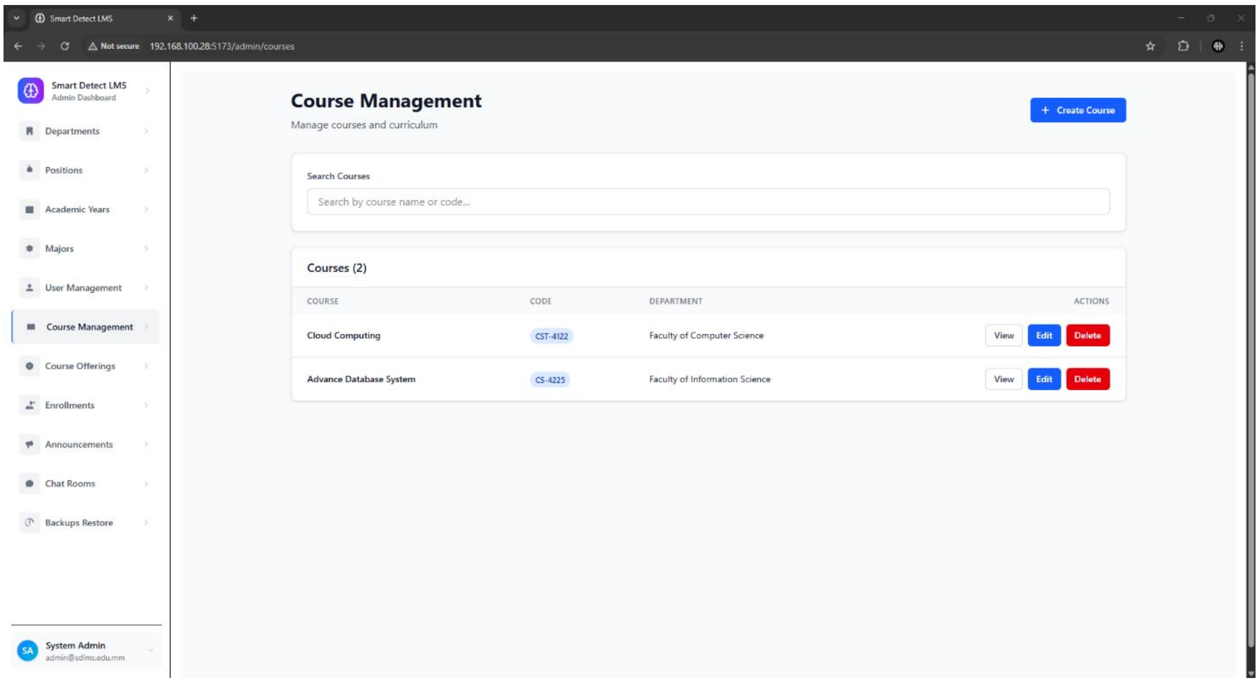
Search Users: Search by name or email...
Filter by Role: All Roles

Users
Manage system users and permissions
Total: 5 users

Maung Aung (Student, Active)	maungaung@sdlms.edu.mm, SDLMS-0001	View Edit Ban Delete
Daw Mya (Teacher, Active)	dawmya@sdlms.edu.mm	View Edit Ban Delete
Daw Hla Hla (Teacher, Active)	dawhlahla@sdlms.edu.mm	View Edit Ban Delete
DawWinWin (Teacher, Active)	dawwinwin12@sdlms.edu.mm	View Edit Ban Delete

Figure 4.7 User Management Page

On this page, the admin can manage the course. Create a new course and edit or delete existing ones.


Smart Detect LMS

Not secure 192.168.100.28:5173/admin/courses

Smart Detect LMS
Admin Dashboard

- Departments
- Positions
- Academic Years
- Majors
- User Management
- Course Management
- Course Offerings
- Enrollments
- Announcements
- Chat Rooms
- Backups Restore

System Admin
admin@sdlms.edu.mm

Course Management
Manage courses and curriculum

+ Create Course

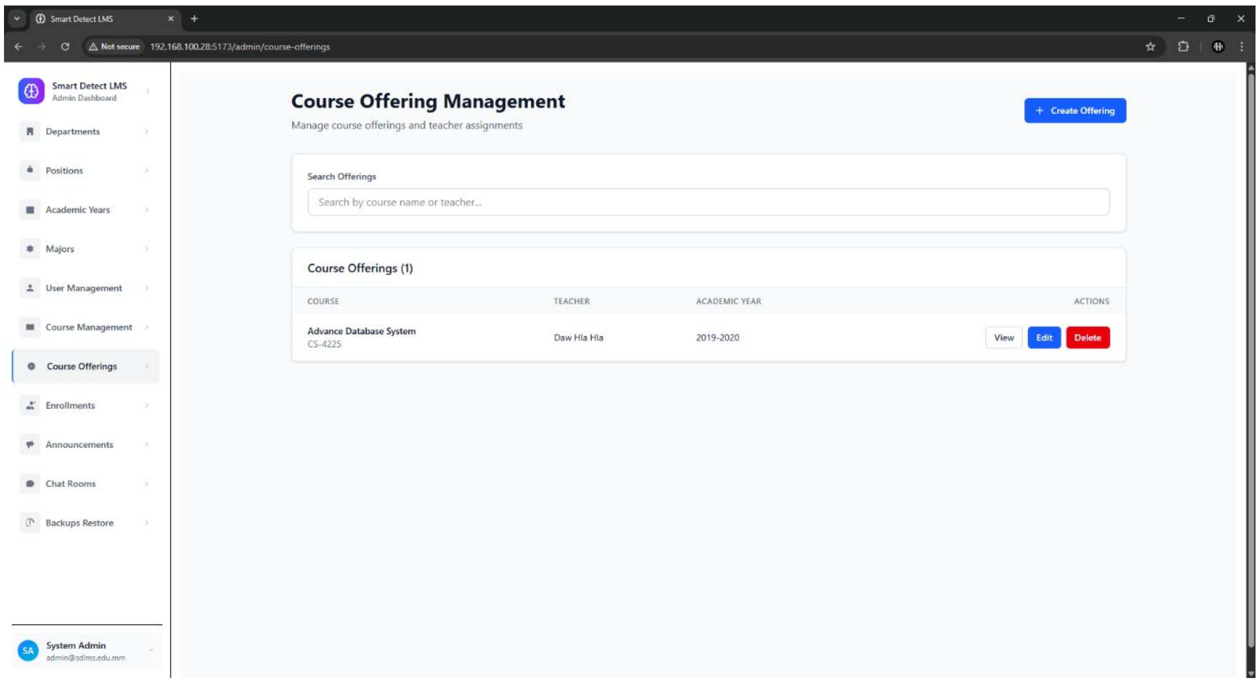
Search Courses: Search by course name or code...

Courses (2)

COURSE	CODE	DEPARTMENT	ACTIONS
Cloud Computing	CST-4122	Faculty of Computer Science	View Edit Delete
Advance Database System	CS-4225	Faculty of Information Science	View Edit Delete

Figure 4.8 Course Management Page

Admin can assign teacher for each course. And then reassign or delete offering course. When admin create course offering, new chat room for course offering is created, assigned teacher is added to chat room automatically.



Course Offering Management

Manage course offerings and teacher assignments

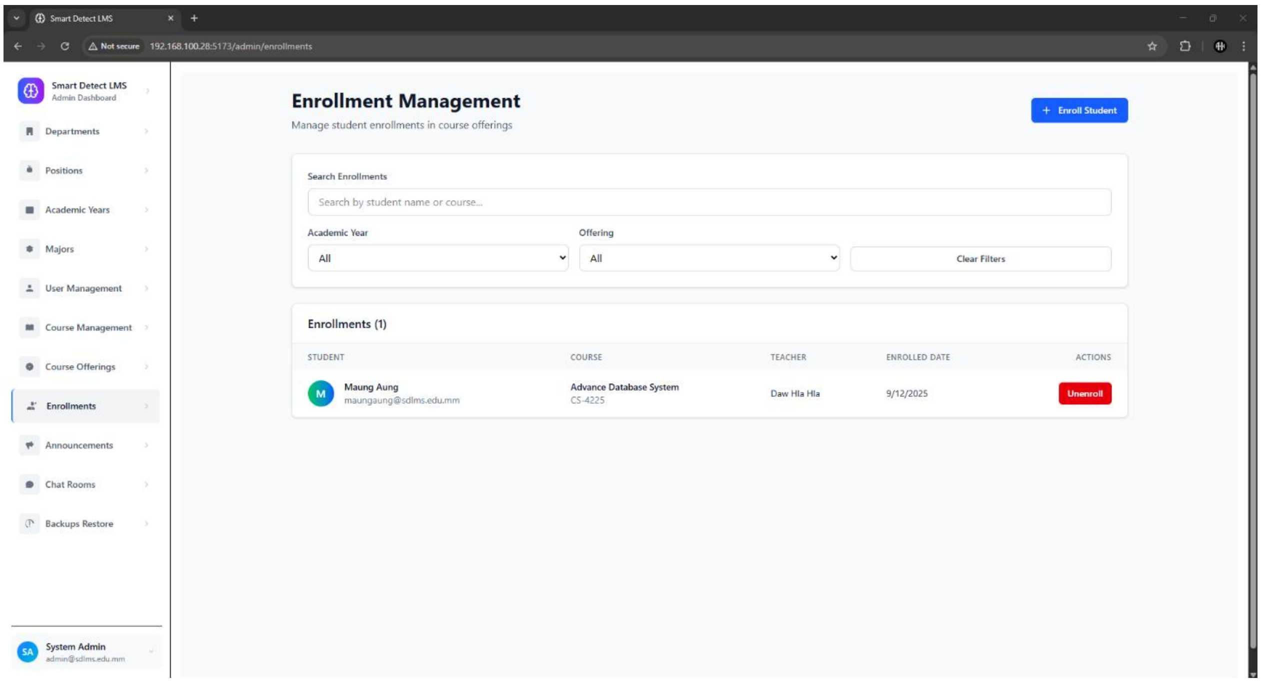
Search Offerings: Search by course name or teacher...

Course Offerings (1)

COURSE	TEACHER	ACADEMIC YEAR	ACTIONS
Advance Database System CS-4225	Daw Hla Hla	2019-2020	View Edit Delete

Figure 4.9 Course Offering Management Page

In this page, admin enroll student to the course and added to offering course’s chat room.



Enrollment Management

Manage student enrollments in course offerings

Search Enrollments: Search by student name or course...

Academic Year: All

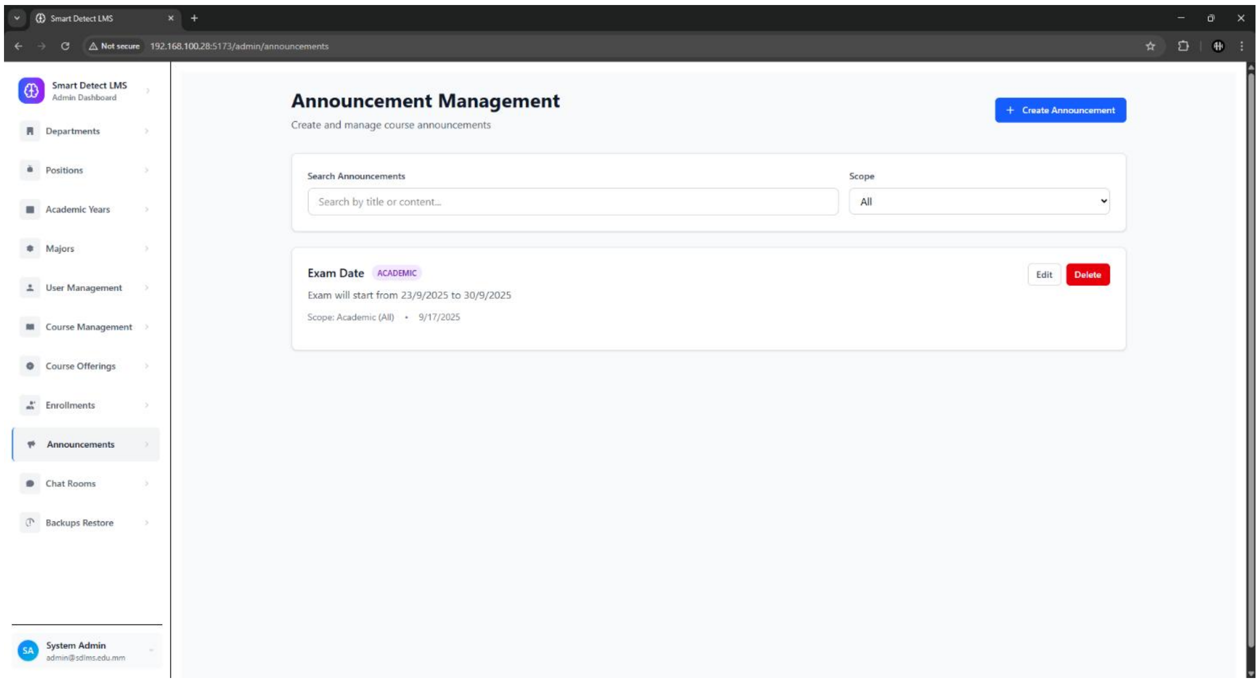
Offering: All

Enrollments (1)

STUDENT	COURSE	TEACHER	ENROLLED DATE	ACTIONS
Maung Aung maungaung@sdlms.edu.mm	Advance Database System CS-4225	Daw Hla Hla	9/12/2025	Unenroll

Figure 4.10 Enrollment Management Page

Admin can announce from this page.



Announcement Management

Create and manage course announcements

Search Announcements: Search by title or content...

Scope: All

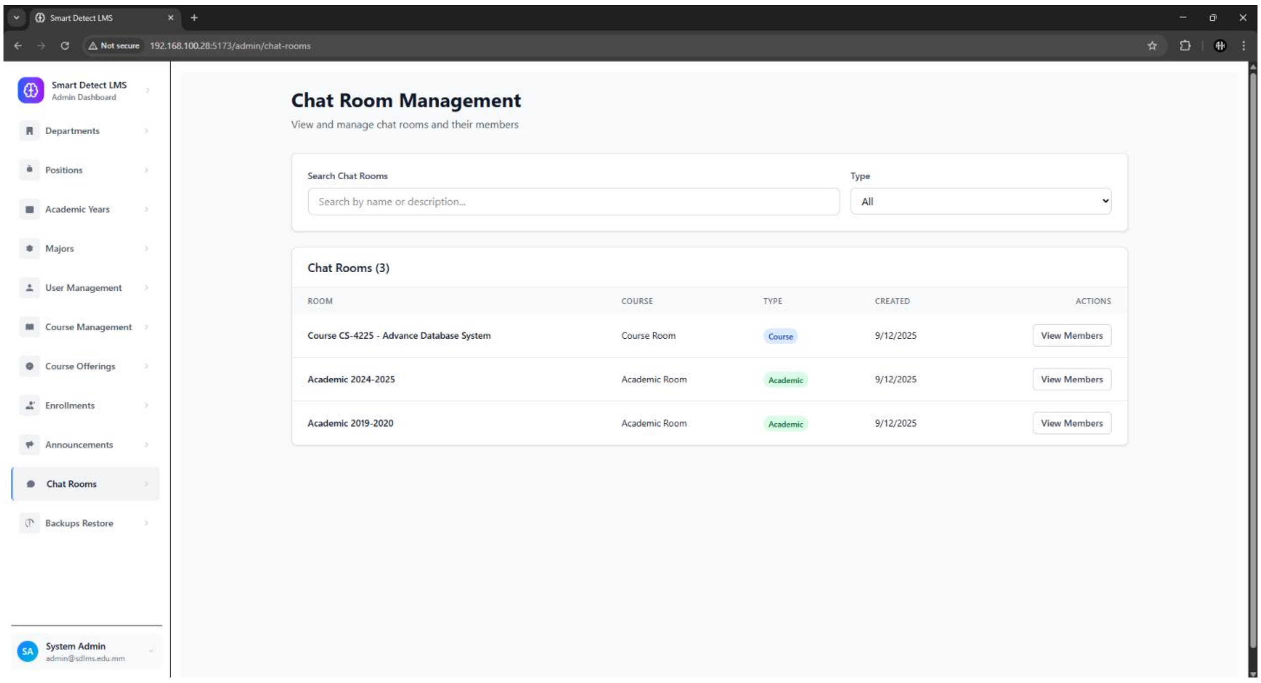
Exam Date ACADEMIC

Exam will start from 23/9/2025 to 30/9/2025

Scope: Academic (All) • 9/17/2025

4.11 Announcement Management Page

Admin can check user list from this page.



Chat Room Management

View and manage chat rooms and their members

Search Chat Rooms: Search by name or description...

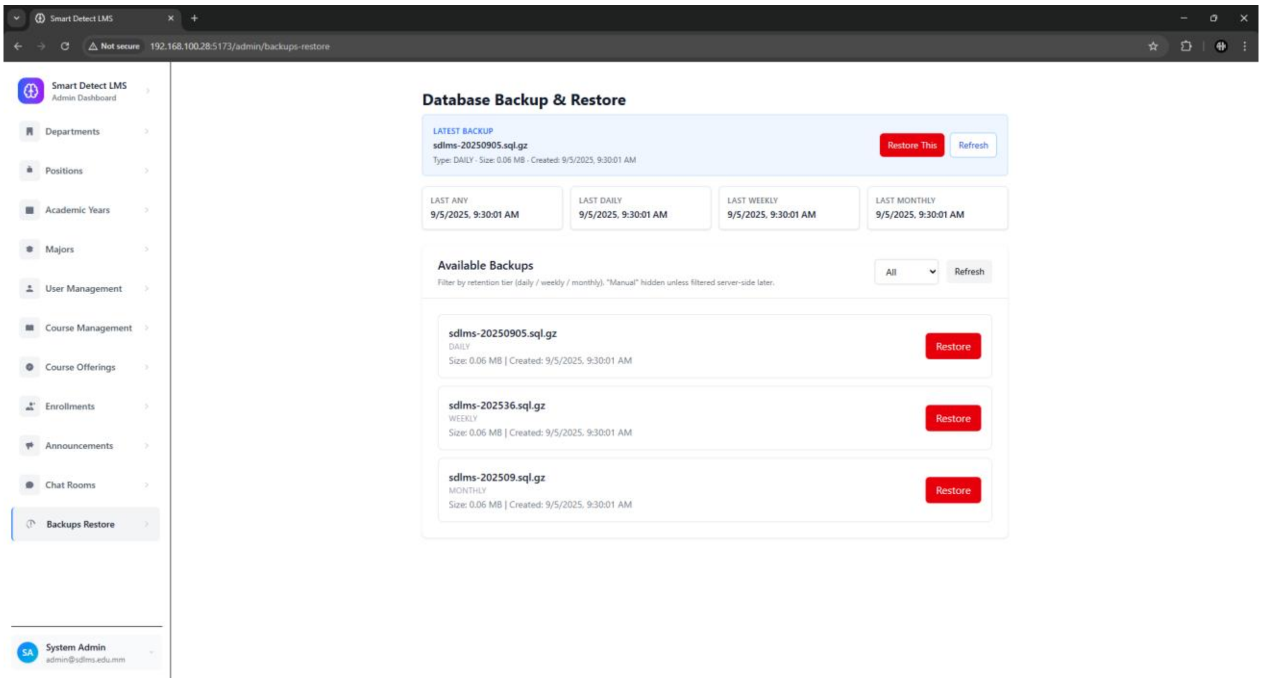
Type: All

Chat Rooms (3)

ROOM	COURSE	TYPE	CREATED	ACTIONS
Course CS-4225 - Advance Database System	Course Room	Course	9/12/2025	View Members
Academic 2024-2025	Academic Room	Academic	9/12/2025	View Members
Academic 2019-2020	Academic Room	Academic	9/12/2025	View Members

4.12 Chat Room Management Page

When admin need to restore data, can get backup from this page.


Database Backup & Restore

LATEST BACKUP
sdlms-20250905.sql.gz
Type: DAILY · Size: 0.06 MB · Created: 9/5/2025, 9:30:01 AM

LAST ANY	LAST DAILY	LAST WEEKLY	LAST MONTHLY
9/5/2025, 9:30:01 AM	9/5/2025, 9:30:01 AM	9/5/2025, 9:30:01 AM	9/5/2025, 9:30:01 AM

Available Backups
Filter by retention tier (daily / weekly / monthly). "Manual" hidden unless filtered server-side later.

sdlms-20250905.sql.gz
DAILY
Size: 0.06 MB | Created: 9/5/2025, 9:30:01 AM

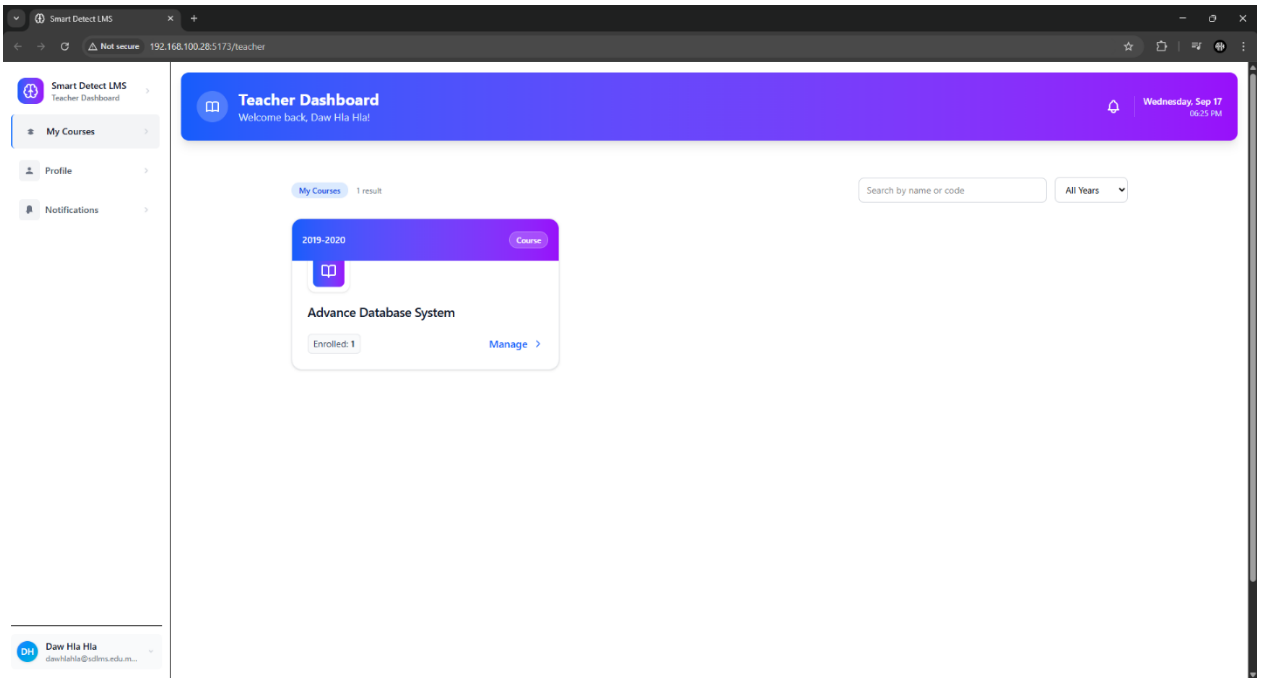
sdlms-202536.sql.gz
WEEKLY
Size: 0.06 MB | Created: 9/5/2025, 9:30:01 AM

sdlms-202509.sql.gz
MONTHLY
Size: 0.06 MB | Created: 9/5/2025, 9:30:01 AM

Figure 4.13 Database Backup & Restore Page

4.3 Teacher Pages

Teacher's all assigned courses are shown on this page.


Teacher Dashboard
Welcome back, Daw Hla Hla!

Wednesday, Sep 17
06:25 PM

My Courses 1 result

2019-2020 Course
Advance Database System
Enrolled: 1

Figure 4.14 Teacher's Home Page

This page shows assigned course's overview.

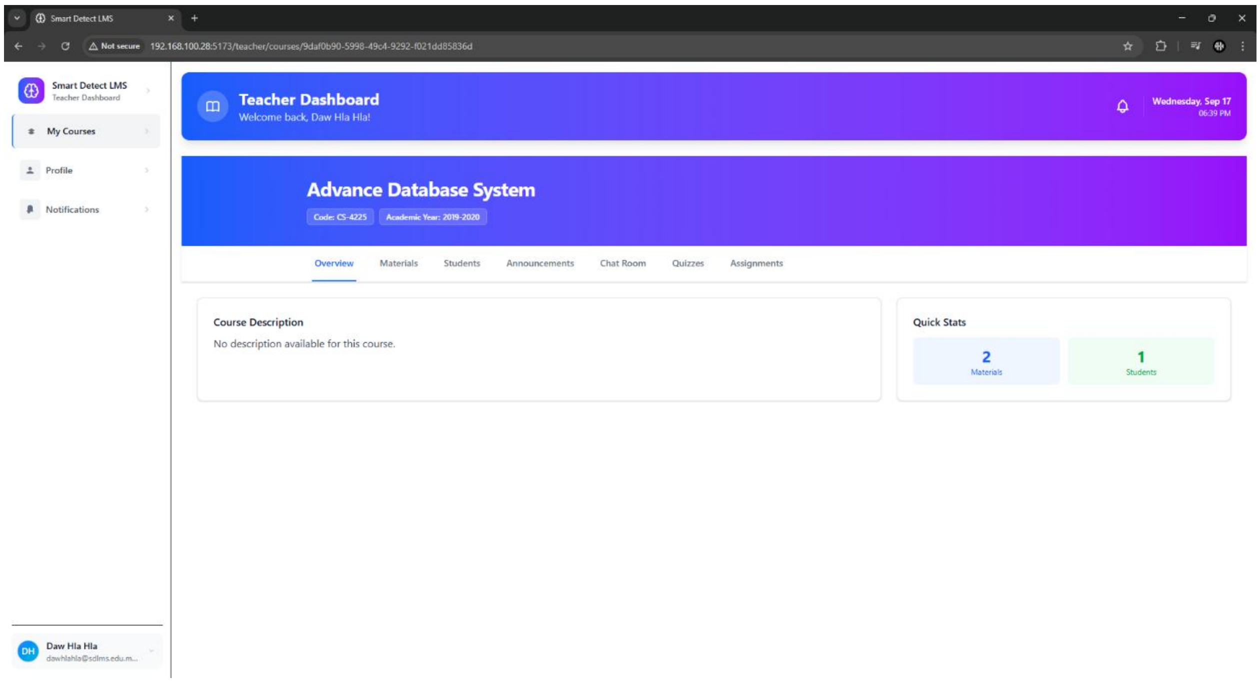


Figure 4.15 Course Overview Page

On this page, teacher can upload course materials with three types (Images, Videos and Documents). And can delete the materials.

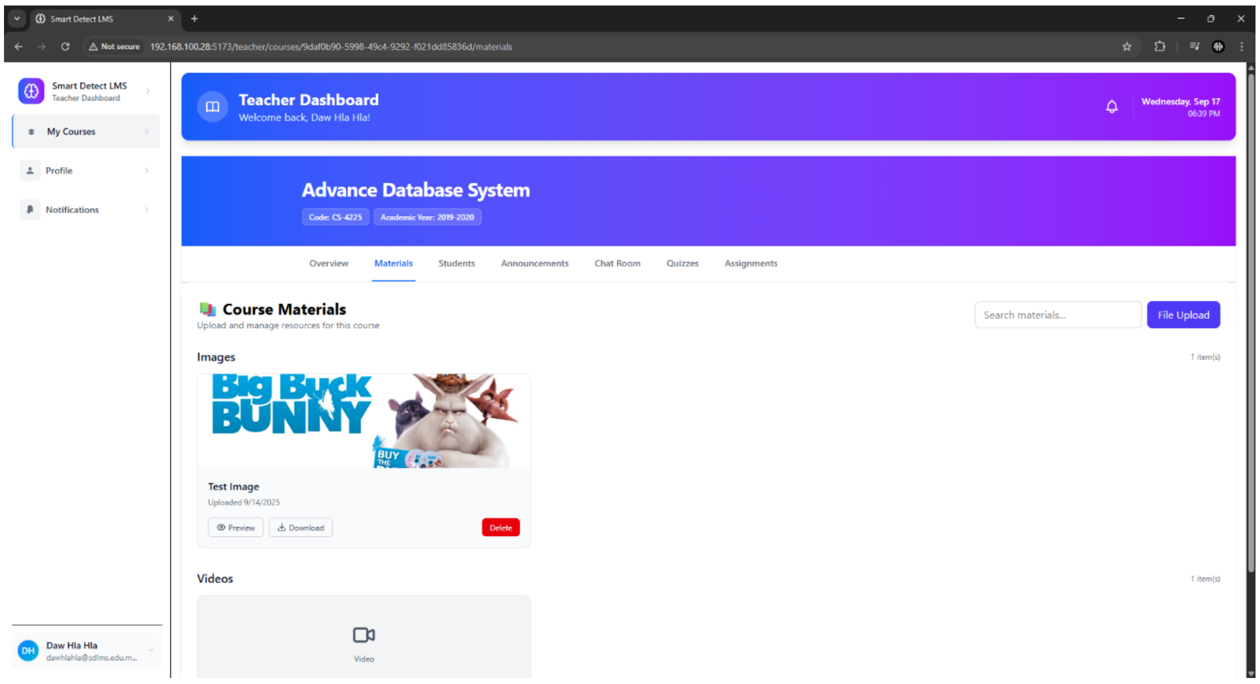
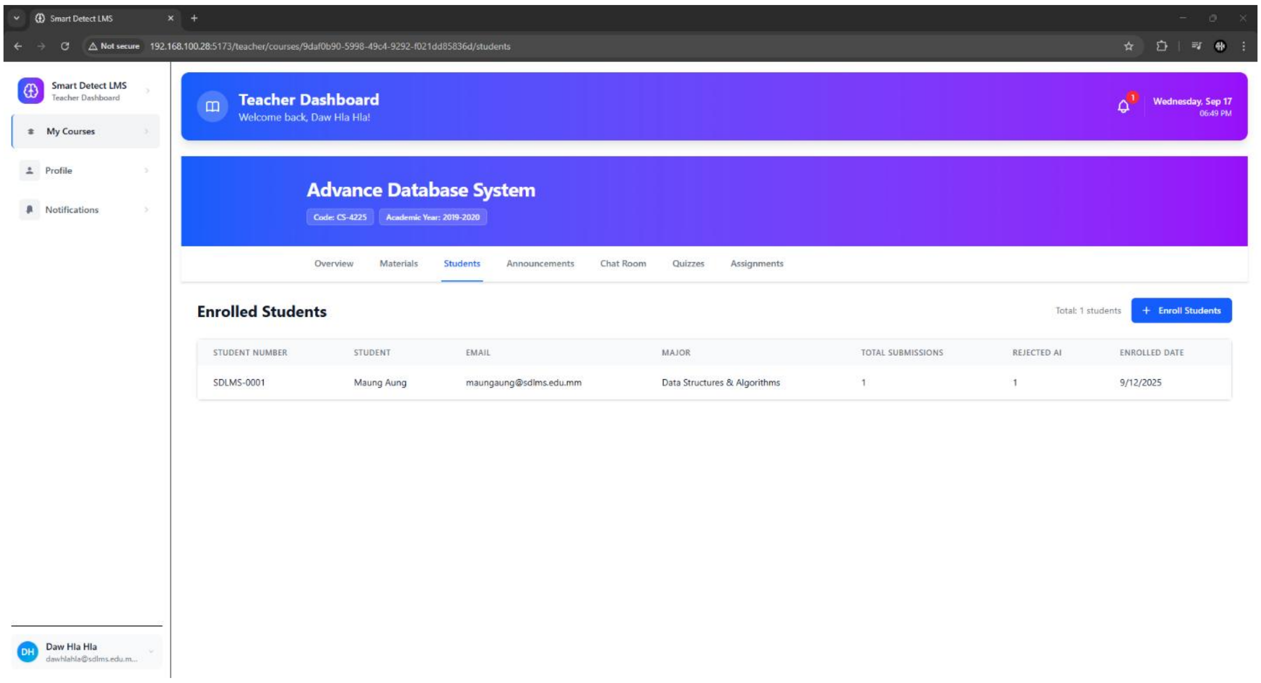


Figure 4.16 Course Materials Page

Teacher can check enrolled students' submission and ai used count on this page.



Enrolled Students

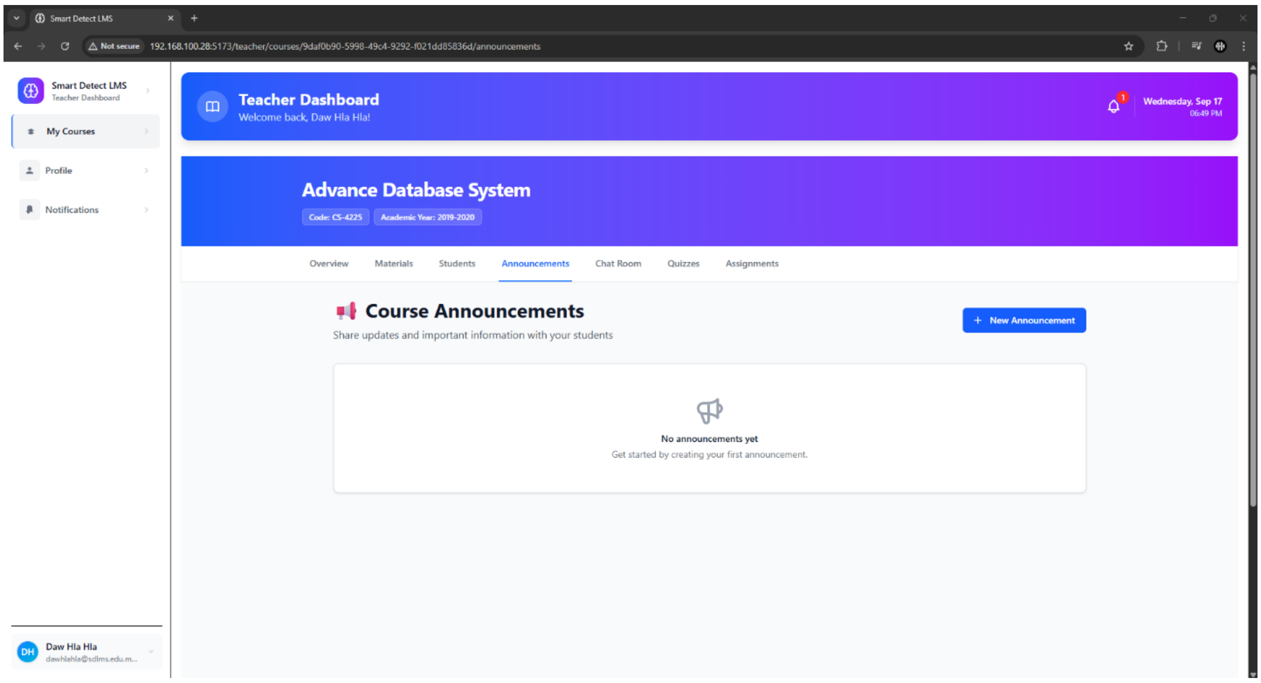
Total: 1 students

Enroll Students

STUDENT NUMBER	STUDENT	EMAIL	MAJOR	TOTAL SUBMISSIONS	REJECTED AI	ENROLLED DATE
SDLMS-0001	Maung Aung	maungaung@sdlms.edu.mm	Data Structures & Algorithms	1	1	9/12/2025

Figure 4.17 Enrolled Students Page

Teacher can announce deals with course from this page.



Course Announcements

Share updates and important information with your students

No announcements yet

Get started by creating your first announcement.

Figure 4.18 Course Announcements Page

This is group chat for offering course, where teacher and enrolled students can send both text and images.

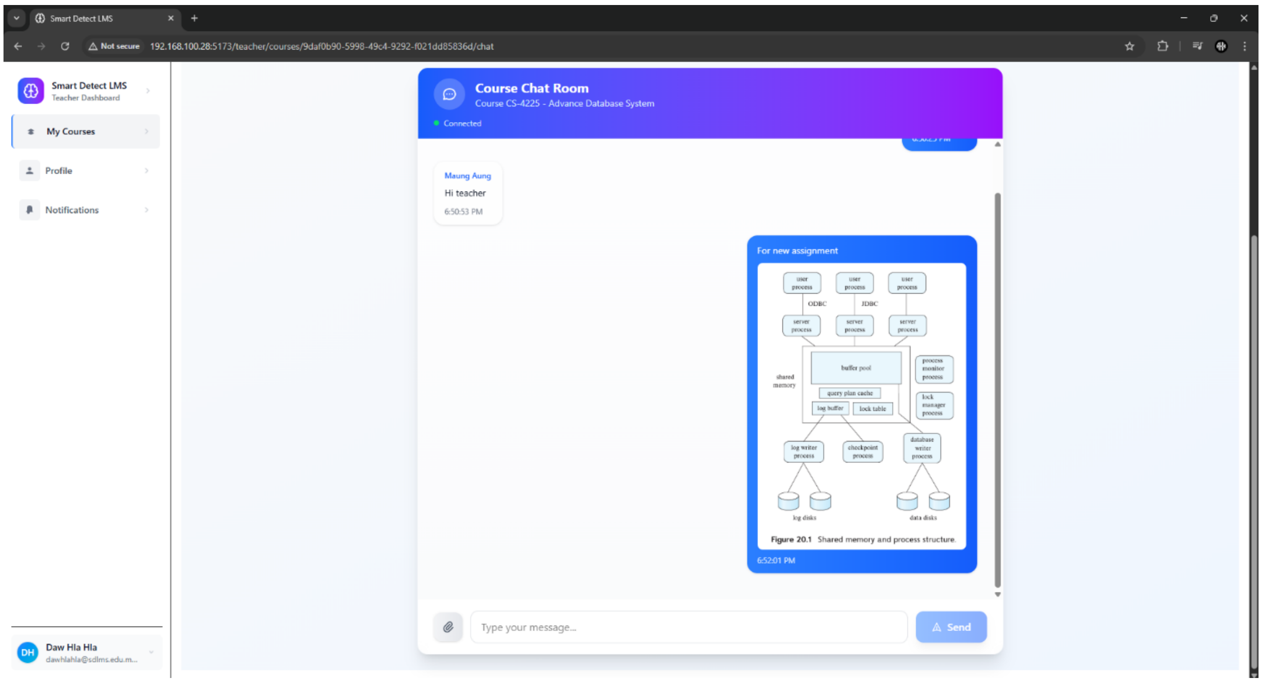


Figure 4.19 Course Chat Room Page

Teacher can upload quiz and check student submission with grade.

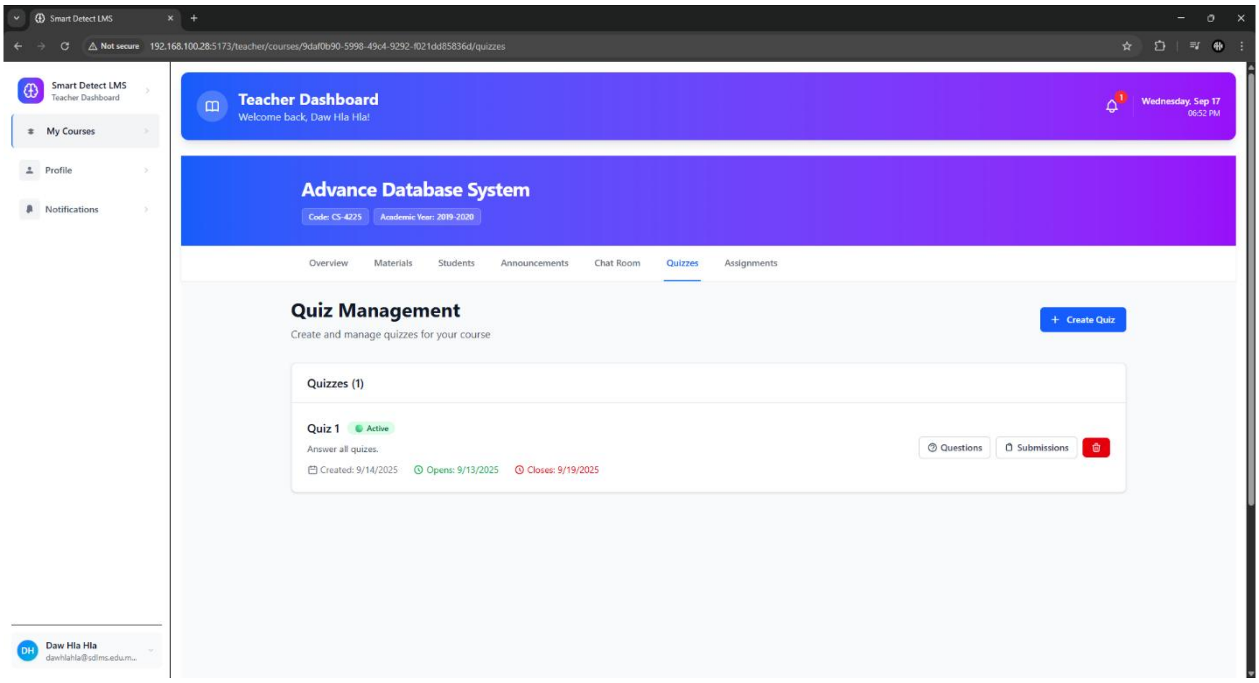
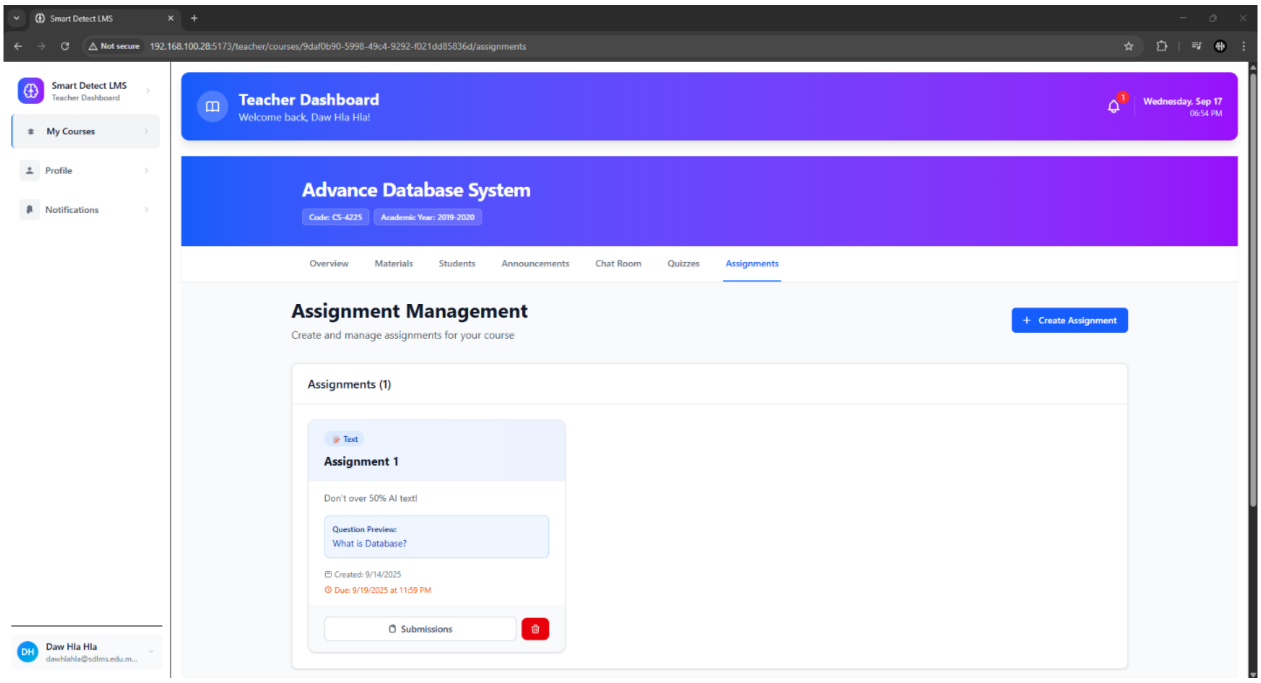


Figure 4.20 Quiz Management Page

Teacher can upload assignment and check student submission with grade. When student's answer is ai generated, teacher can


Smart Detect LMS

192.168.100.28:5173/teacher/courses/9daf0b90-5998-49c4-9292-f021dd85836d/assignments

Smart Detect LMS
Teacher Dashboard

My Courses
Profile
Notifications

Daw Hla Hla
dawhlahla@sdlms.edu.m...

Teacher Dashboard
Welcome back, Daw Hla Hla!

Wednesday, Sep 17
06:54 PM

Advance Database System
Code: CS-4225 Academic Year: 2019-2020

Overview Materials Students Announcements Chat Room Quizzes Assignments

Assignment Management
Create and manage assignments for your course

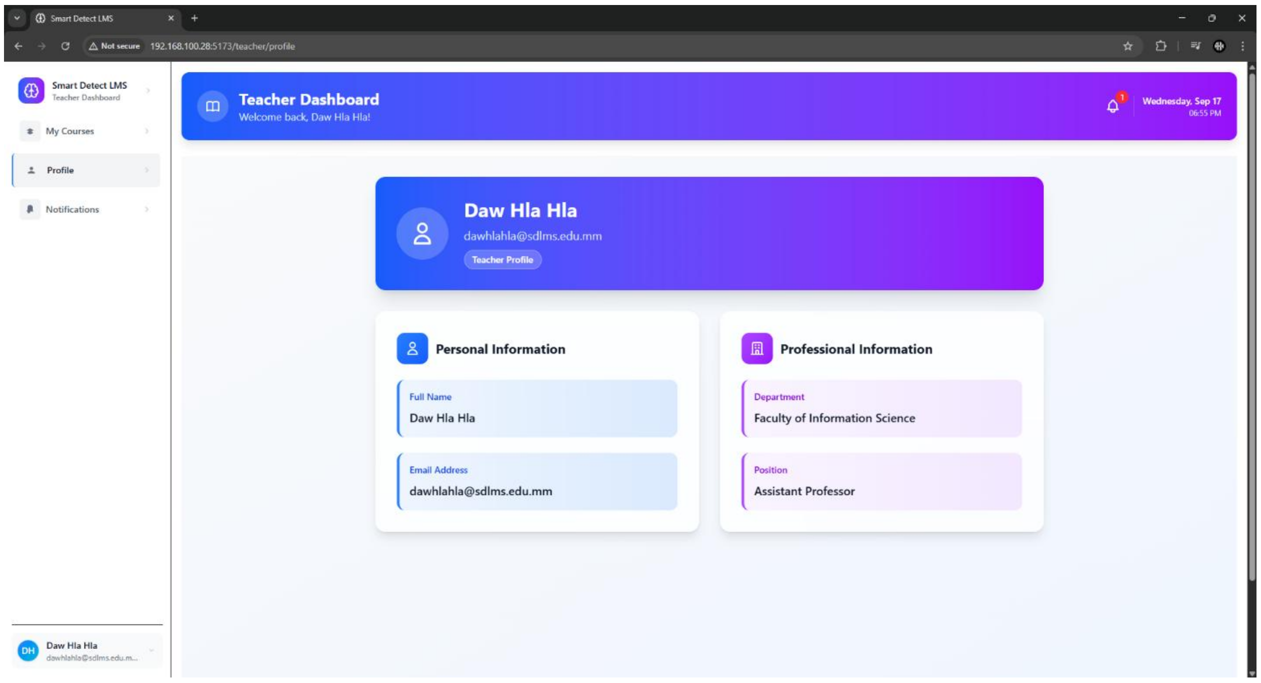
+ Create Assignment

Assignments (1)

Text
Assignment 1
Don't over 50% AI text!
Question Preview:
What is Database?
Created: 9/14/2025
Due: 9/19/2025 at 11:59 PM
Submissions

Figure 4.21 Assignment Management Page

This is only to check profile. If want to edit data, need to connect with admin.


Smart Detect LMS

192.168.100.28:5173/teacher/profile

Smart Detect LMS
Teacher Dashboard

My Courses
Profile
Notifications

Daw Hla Hla
dawhlahla@sdlms.edu.m...

Teacher Dashboard
Welcome back, Daw Hla Hla!

Wednesday, Sep 17
06:55 PM

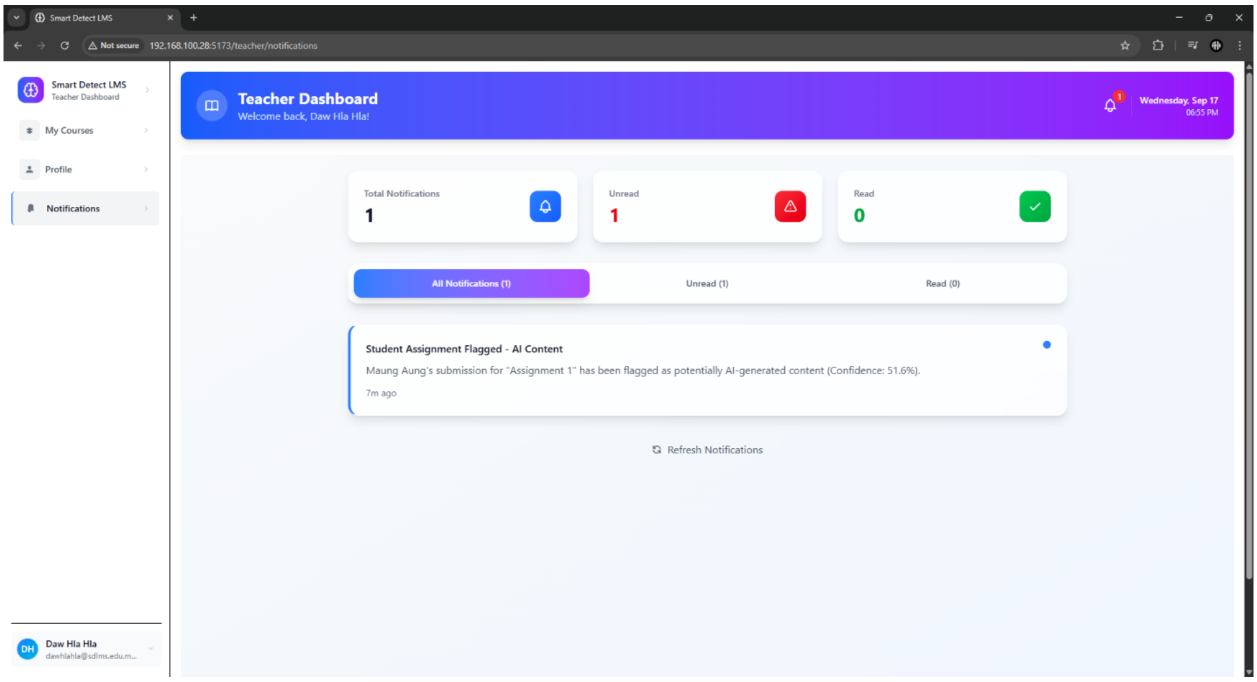
Daw Hla Hla
dawhlahla@sdlms.edu.mm
Teacher Profile

Personal Information
Full Name
Daw Hla Hla
Email Address
dawhlahla@sdlms.edu.mm

Professional Information
Department
Faculty of Information Science
Position
Assistant Professor

Figure 4.22 Profile Page

All notifications can be seen from this page. When an enrolled student submits AI-generated text in an assignment, the system will send a notification to both the teacher and the student.



Student Assignment Flagged - AI Content

Maung Aung's submission for "Assignment 1" has been flagged as potentially AI-generated content (Confidence: 51.6%).

7m ago

Figure 4.23 Notifications Page

4.4 Student Pages

Student's all enrolled courses are shown on this page.

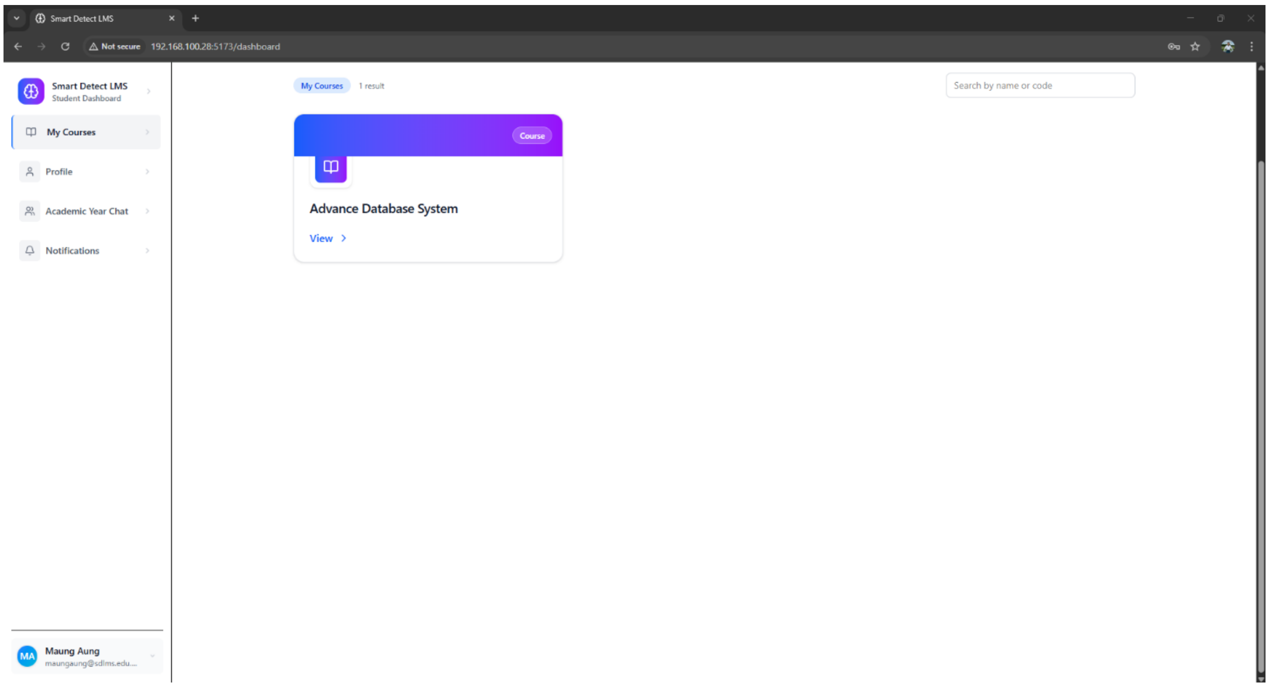
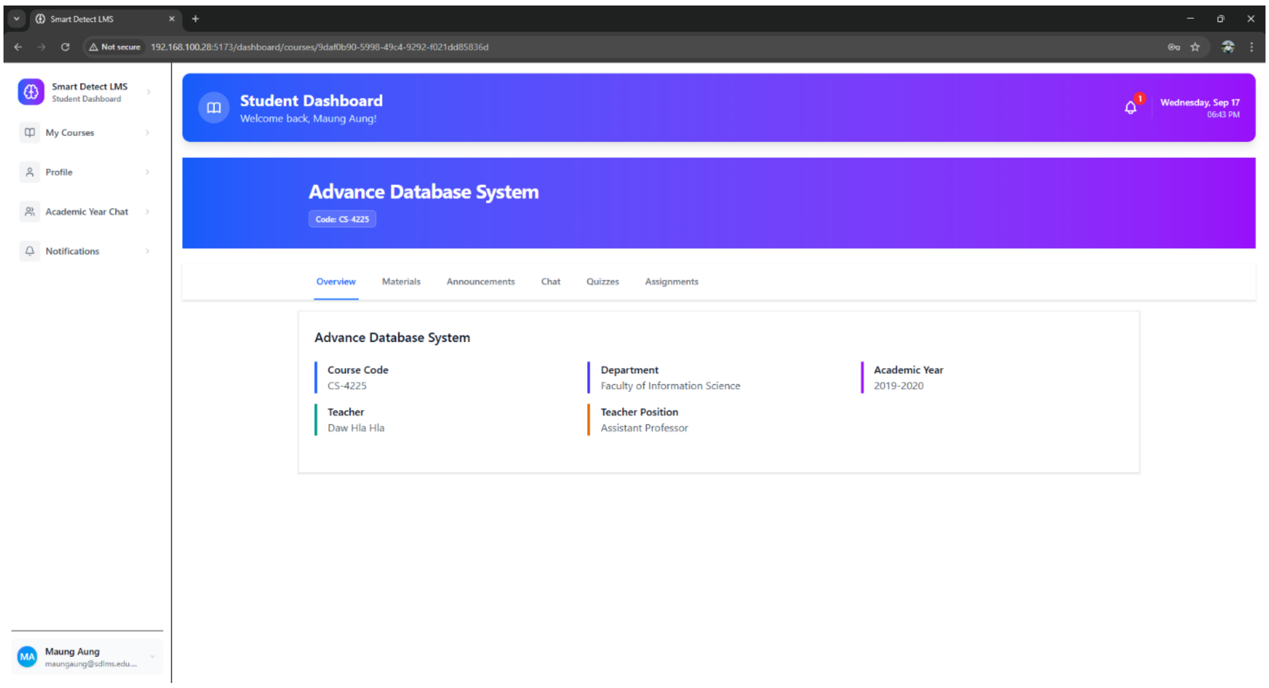


Figure 4.24 Student's Home Page

This page shows enrolled course’s overview.



Advance Database System

Course Code CS-4225	Department Faculty of Information Science	Academic Year 2019-2020
Teacher Daw Hla Hla	Teacher Position Assistant Professor	

Figure 4.25 Course Overview Page

Enrolled students can view and download materials, that are uploaded by teacher.

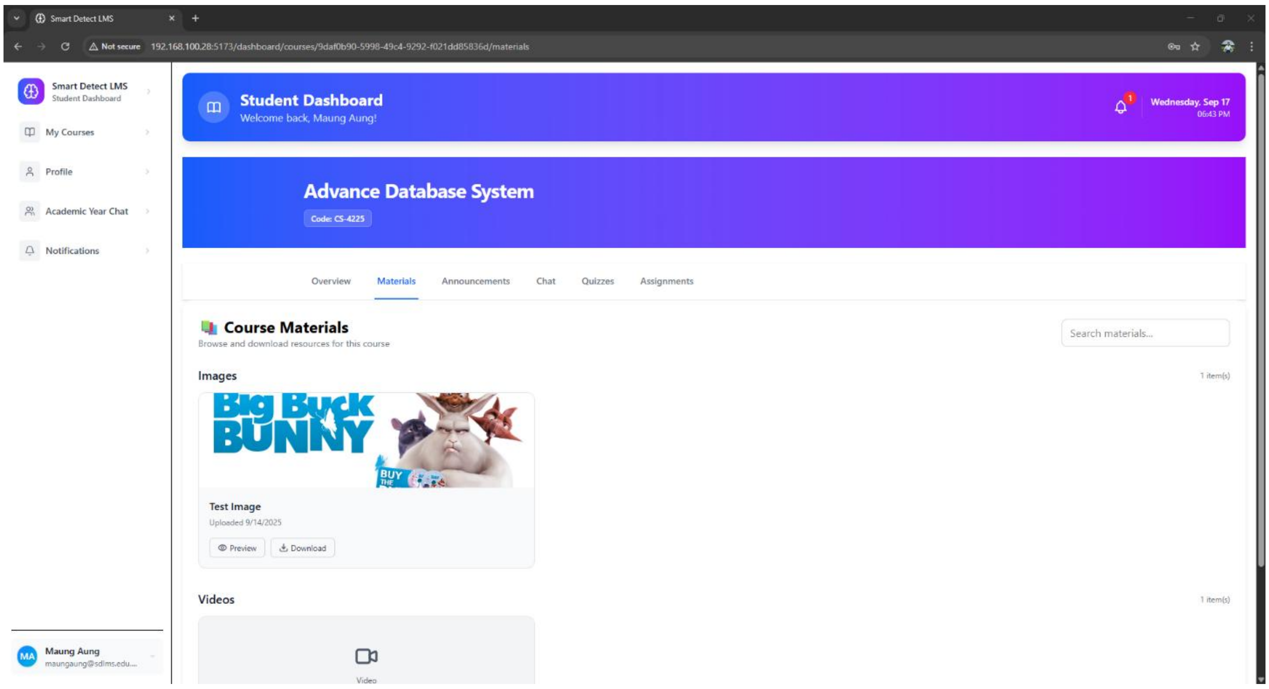
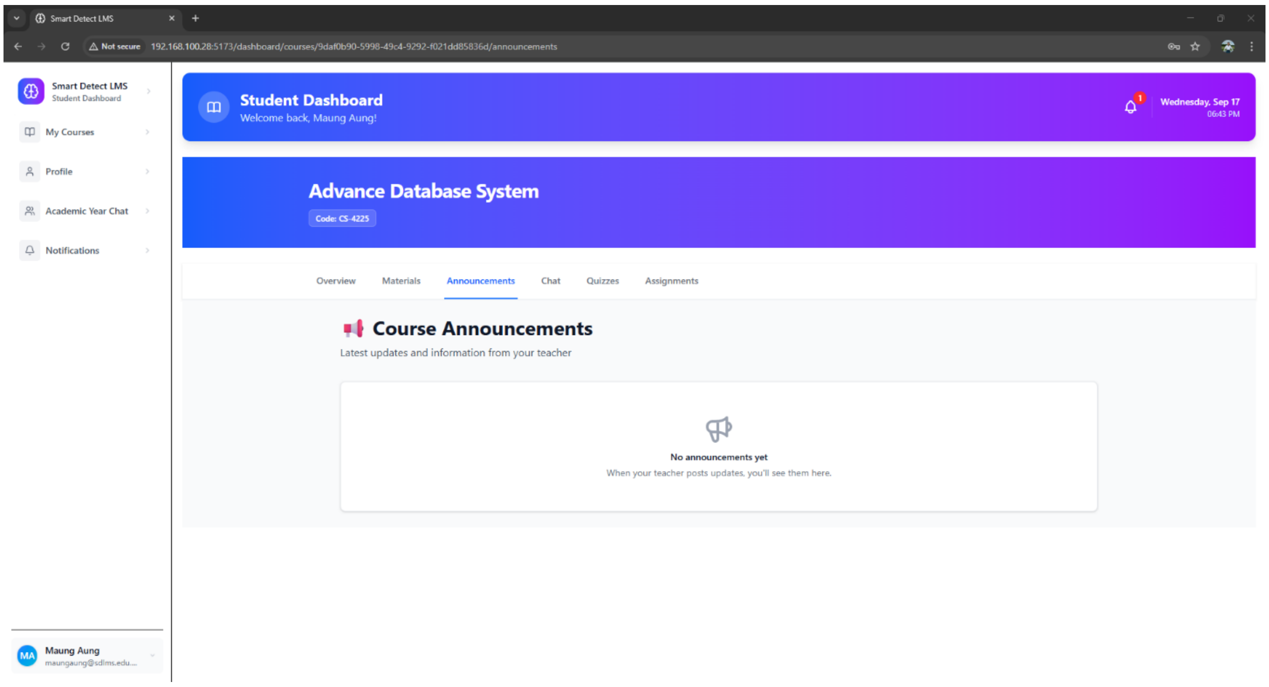


Figure 4.26 Course Materials Page

Student can read course announcements from teacher.



Smart Detect LMS

Not secure 192.168.100.28:5173/dashboard/courses/9daf0b90-5998-49c4-9292-f021dd85836d/announcements

Smart Detect LMS
Student Dashboard

- My Courses
- Profile
- Academic Year Chat
- Notifications

Student Dashboard
Welcome back, Maung Aung!

Wednesday, Sep 17
06:43 PM

Advance Database System
Code: CS-4225

Overview | Materials | Announcements | Chat | Quizzes | Assignments

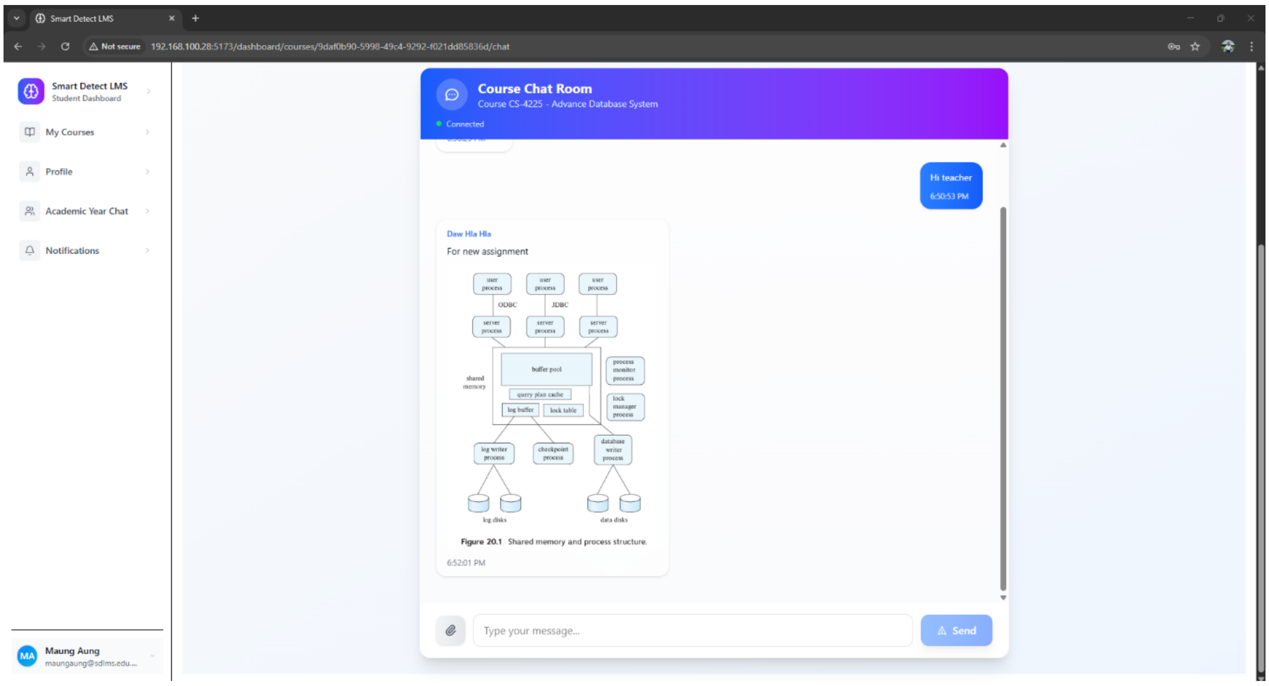
Course Announcements
Latest updates and information from your teacher

No announcements yet
When your teacher posts updates, you'll see them here.

MA Maung Aung
maungaung@sdlms.edu...

Figure 4.27 Course Announcements Page

Students can chat with not only teacher but also classmates in this area.



Smart Detect LMS

Not secure 192.168.100.28:5173/dashboard/courses/9daf0b90-5998-49c4-9292-f021dd85836d/chat

Smart Detect LMS
Student Dashboard

- My Courses
- Profile
- Academic Year Chat
- Notifications

Course Chat Room
Course CS-4225 - Advance Database System
Connected

Hi teacher
6:50:53 PM

Daw Hla Hla
For new assignment

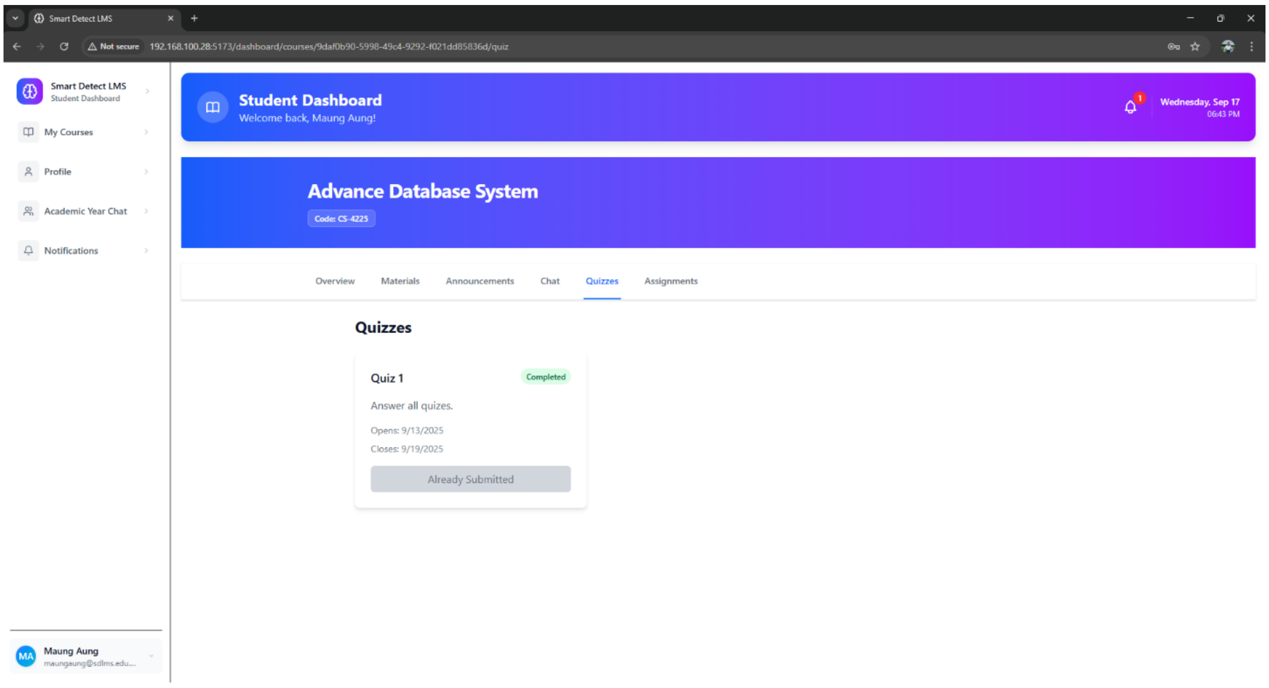
6:52:01 PM

Type your message...
Send

MA Maung Aung
maungaung@sdlms.edu...

Figure 4.28 Course Chat Room Page

Students can answer quizzes.



Quizzes

Quiz 1 Completed

Answer all quizes.

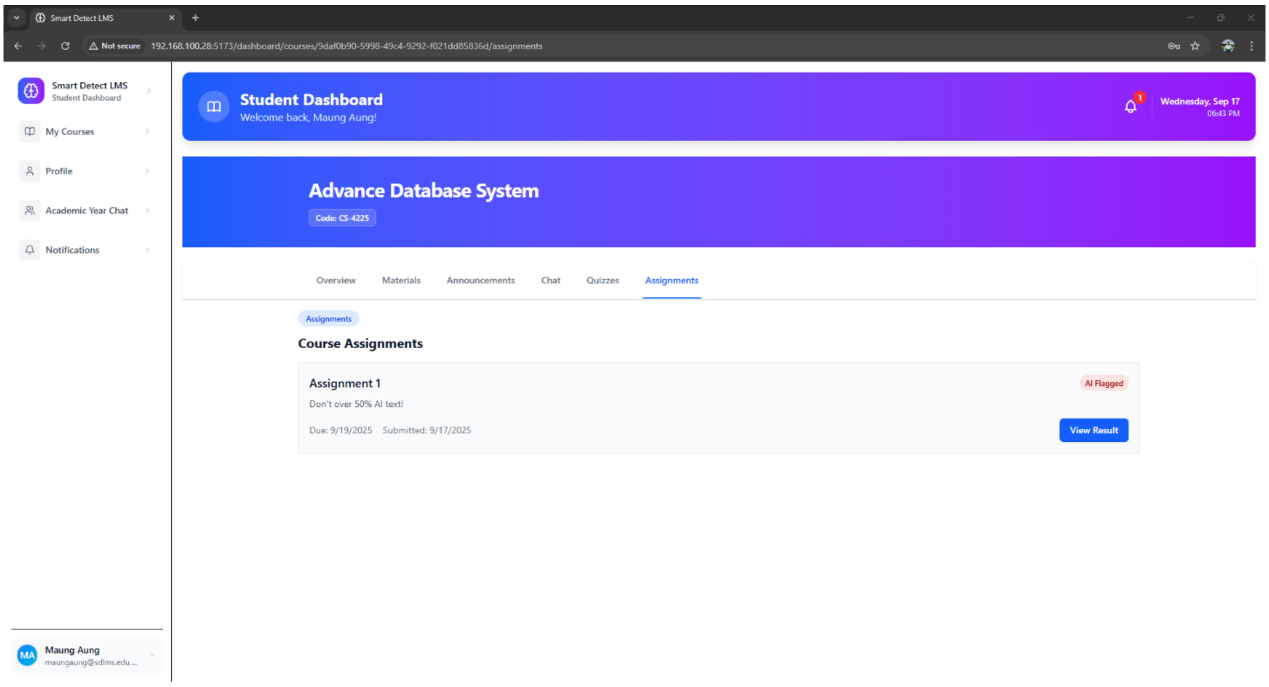
Opens: 9/13/2025

Closes: 9/19/2025

Already Submitted

Figure 4.29 Quizzes Page

Students can submit assignment. When over 50% ai generated, assignment will be rejected automatically from system. If teacher allows, student can answer again.



Assignments

Course Assignments

Assignment 1 AI Flagged

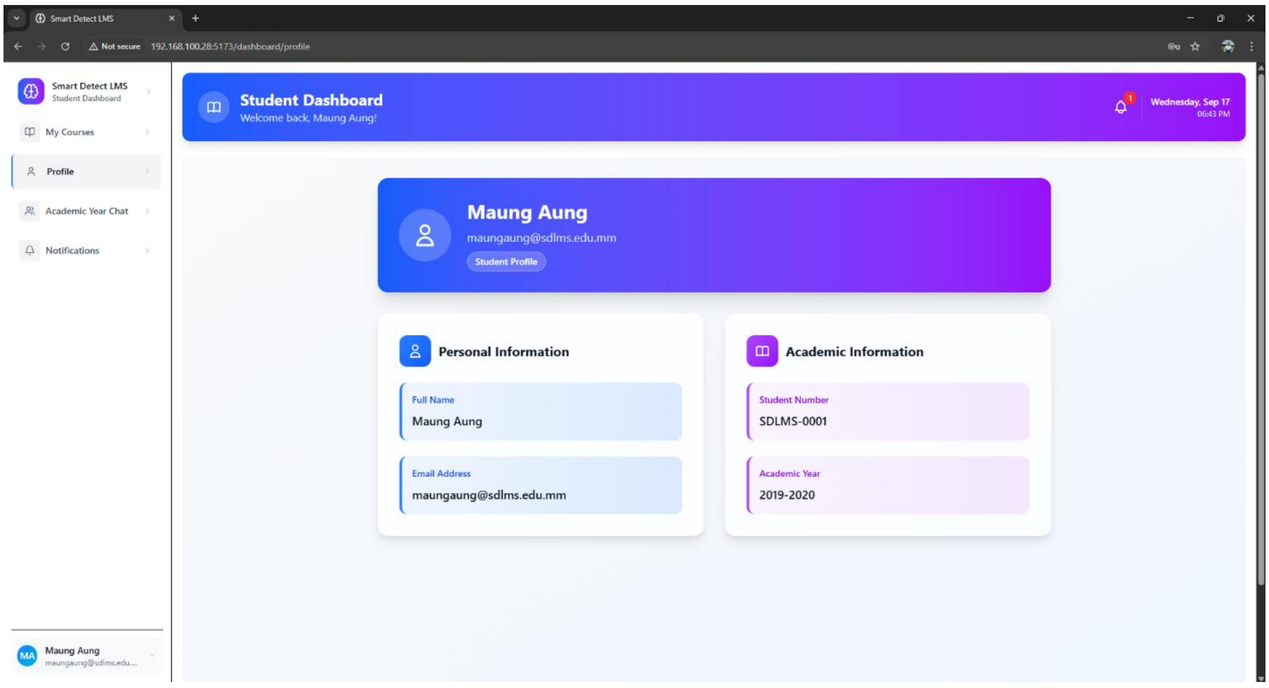
Don't over 50% AI text!

Due: 9/19/2025 Submitted: 9/17/2025

View Result

Figure 4.30 Assignments Page

This is only to check profile. If want to edit data, need to connect with admin.



Student Dashboard
Welcome back, Maung Aung!

Maung Aung
maungaung@sdlms.edu.mm

Personal Information

Full Name: Maung Aung

Email Address: maungaung@sdlms.edu.mm

Academic Information

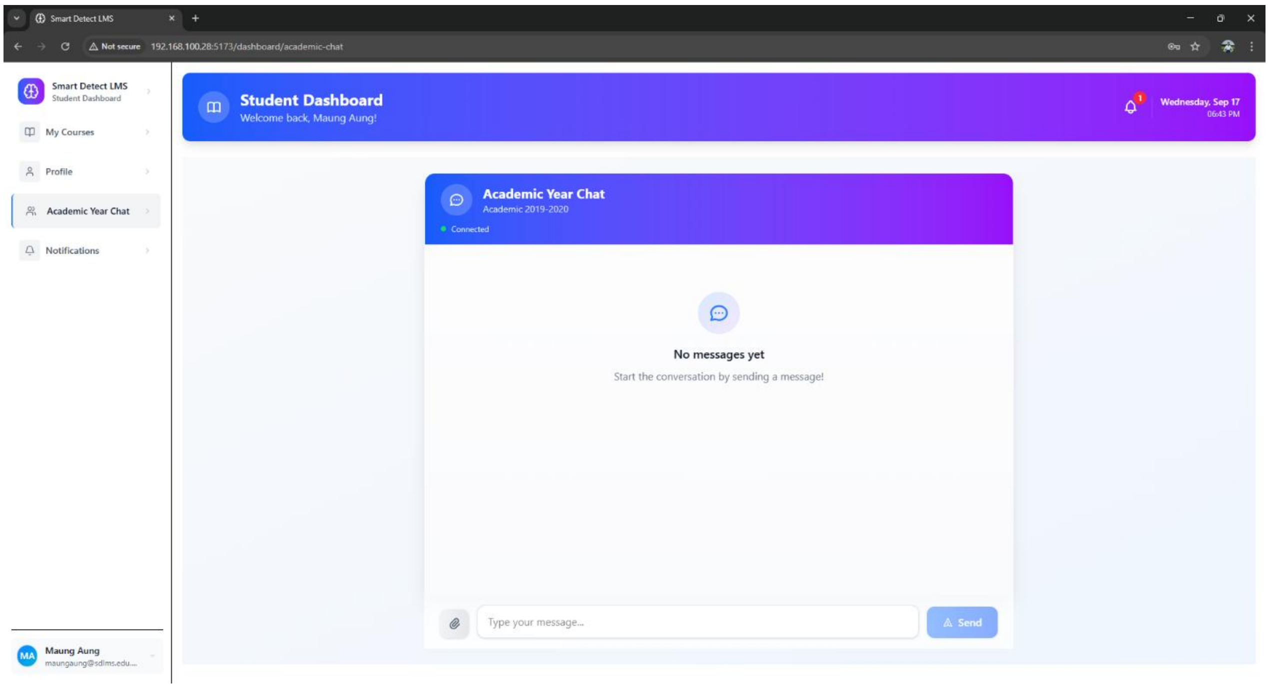
Student Number: SDLMS-0001

Academic Year: 2019-2020

Maung Aung
maungaung@sdlms.edu...

Figure 4.31 Profile Page

Students can chat with classmates in this area. They can send both text and images.



Student Dashboard
Welcome back, Maung Aung!

Academic Year Chat
Academic 2019-2020

No messages yet
Start the conversation by sending a message!

Maung Aung
maungaung@sdlms.edu...

Figure 4.32 Academic Chat Room Page

CHAPTER 5
CONCLUSION

The Smart Detect Learning Management System (SD-LMS) represents a successful implementation of modern educational technology, delivering a comprehensive full-stack web application that addresses critical needs in academic environments. Built with React frontend, Express.js backend, and PostgreSQL database, the system demonstrates technical excellence through its scalable Docker-based architecture, real-time Socket.IO communication, and intelligent AI content detection capabilities. The platform effectively serves three distinct user roles - administrators who manage users and courses, teachers who create and assess content, and students who engage with learning materials - each with tailored interfaces that enhance their specific workflows. Key innovations include automated AI-powered plagiarism detection that maintains academic integrity while providing transparent feedback, real-time chat functionality for collaborative learning, comprehensive course and assignment management tools, and robust performance monitoring through Redis caching and Prometheus metrics. The system's role-based access control, automated backup systems, and intuitive user interfaces reflect deep understanding of educational institution requirements. SDLMS successfully validates how thoughtful software engineering can transform educational experiences, providing a solid foundation for future enhancements while maintaining focus on usability, security, and scalability in addressing real-world academic challenges.

REFERENCES

[1] https://lms.ucsmub.edu.mm
[2] https://quillbot.com/ai-content-detector

PROJECT’S GITHUB REPO

[1] Smart Detect Learning Management System
https://github.com/aung-khantkyaw/smart-detect-learning-management-system
[2] AI Text Detection
https://github.com/aung-khantkyaw/ai-text-detection