International Journal of Science, Strategic Management and Technology
Volume **02** Issue **04** April-**2026** | ISSN: **3108-1762 (Online)** | Impact Factor: 3.8
An International, Peer-Reviewed, Open Access Scholarly Journal Indexed in recognized academic databases

Web Based College Management System with AI Enabled Student Chatbot

M.Gobinath , S.Anandha kumar , K.Kathiresh,Gourav Kumar

Department of Computer Science and Engineering,
MIET Engineering College, Tiruchirappalli – 620007, Tamil Nadu, India

Guided by: **Sathya M**, M.E., AP/CSE

https://doi.org/10.55041/ijsmt.v2i4.538

Cite this Article: M.Gobinath, , kumar, S., K.Kathiresh, & Kumar, G. (2026). Web Based College Management System with AI Enabled Student Chatbot. International Journal of Science, Strategic Management and Technology, *02*(04). https://doi.org/10.55041/ijsmt.v2i4.538

Abstract —This paper presents the design and implementation of a web-based College Management System integrated with an NLP-powered AI chatbot for MIET Engineering College. Built on the Django framework with MySQL database, the system provides role-based portals for administrators, faculty, and students, encompassing attendance tracking, grade management, fee administration, and class scheduling. The embedded chatbot employs a keyword-scoring pattern matching algorithm over a structured institutional knowledge base, delivering context-aware responses including personalised academic and fee information for authenticated students. The system significantly reduces administrative workload, improves information accessibility, and provides a modern, intelligent solution for academic administration.

Keywords: College Management System, Django Framework, NLP Chatbot, Role-Based Access Control, Academic ERP, Web Application, MySQL

I.INTRODUCTION

With the rapid growth of educational institutions and increasing administrative complexity, traditional paper-based management systems have become inadequate. Manual processes for managing student records, attendance, examinations, and fee collection are error-prone, time-consuming, and incapable of providing real-time analytics. Modern advancements in web technologies and Artificial Intelligence (AI) enable the development of integrated, intelligent campus management platforms.

This paper proposes a comprehensive web-based College Management System that combines automated administrative workflows with an NLP-powered AI chatbot to enhance institutional efficiency and the student experience. The system is built using the Django framework with Python and MySQL database, serving three user roles through dedicated portals with role-based access control.

II. OBJECTIVES

The main objectives of the proposed system are:

- To develop an integrated web-based college management platform without manual dependency
- To implement role-based portals for Admin, Staff, and Student users
- To automate attendance tracking, grade management, and fee administration
- To integrate an NLP-powered AI chatbot for instant student query resolution
- To provide real-time analytics and performance visualisations using Chart.js
- To ensure data security through Django's built-in authentication and RBAC

III. EXISTING SYSTEM

Most existing college management approaches use:

- Manual paper-based registers for attendance and marks
- Physical files for student and staff record management
- Written memos and notice boards for communication
- Manual fee collection with handwritten receipts

Limitations:

- High error rate due to manual data entry and redundancy
- No real-time access to academic information
- Time-consuming retrieval of student/staff records
- No centralised database for multi-department coordination
- No AI-based query support for students or administration
- Data vulnerable to physical damage, loss, or misplacement

These limitations highlight the need for an intelligent automated web-based system.

IV. PROPOSED SYSTEM

The proposed system introduces a comprehensive web-based College Management System integrating:

- Role-based web portals for Admin, Staff, and Student users
- Automated attendance marking with percentage calculation
- Digital grade and marks entry with GPA computation
- Fee management with payment status tracking
- NLP-powered AI chatbot for instant query resolution
- Real-time analytics with Chart.js visualisations

Working Flow:

1. User accesses web portal
2. Authenticate and verify role
3. Redirect to role-specific dashboard
4. Perform module operations (attendance/marks/fees)
5. AI chatbot handles student queries
6. Generate reports and analytics

V. LITERATURE SURVEY

Recent research in college management systems focuses on:

- Web-based student information systems using PHP/MySQL
- Mobile-based attendance systems using QR codes
- Commercial ERP solutions (Fedena, Classter)
- AI chatbots for educational institutions

However, many existing systems lack:

- Integration of AI chatbot with academic management
- Context-aware personalised responses for students
- Lightweight NLP engines suitable for self-hosted deployment
- Comprehensive modules covering attendance, grades, and fees together

The proposed system addresses these gaps by combining Django's robust framework with a custom NLP chatbot engine that requires no external API dependencies or GPU infrastructure.

VI. SYSTEM ARCHITECTURE

The proposed College Management System follows a three-tier Model-View-Template (MVT) architecture. Each module is designed to perform a specific function within the system.

• Presentation Module: This module handles the user interface using HTML5, CSS3, JavaScript, and Bootstrap. It renders responsive, mobile-friendly templates with Chart.js for analytics visualisation and includes the chatbot widget for student interaction.

• Business Logic Module: The core processing module implements all application logic using Django views and models. It handles authentication, role-based access control via custom decorators (@admin_required, @staff_required, @student_required), and orchestrates CRUD operations across all entities.

• Data Module: This module manages persistent storage using MySQL 8.0 via Django's ORM. It defines 12 database models including Department, Course, UserProfile, StudentProfile, StaffProfile, Subject, AcademicRecord, Attendance, FeeRecord, MarksEntry, and ClassSchedule. Foreign keys enforce referential integrity at the database level.

• Chatbot Module: The NLP chatbot engine operates as a standalone Django application with a structured knowledge base covering institutional FAQs, department details, admission procedures, fee structures, and personalised student data queries.

• Authentication Module: Implements secure login using Django's built-in authentication with PBKDF2 password hashing, session management, and CSRF protection on all forms.

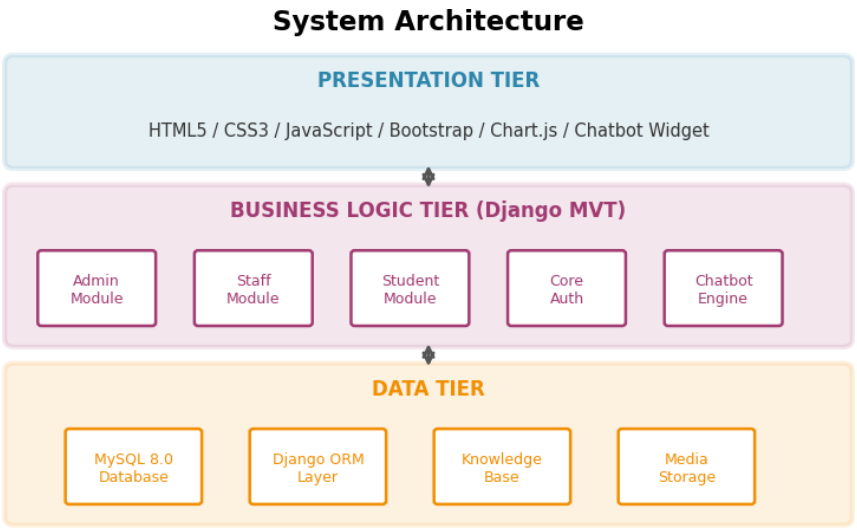

Fig. 1. Three-tier MVT System Architecture of MCMS

VII. MODULE DESCRIPTION

A. Admin Module

The Admin module provides a comprehensive dashboard with system-wide statistics. Administrators can manage students, staff, departments, courses, subjects, and fee records through full CRUD operations. Chart.js-powered visualisations display department-wise student distribution, grade analysis, semester performance trends, and attendance analytics.

B. Staff Module

Staff members access a personalised dashboard showing today's class schedule and quick-action buttons. The attendance workflow presents a date-filtered student list for bulk status assignment (Present / Absent / Late). MarksEntry captures six exam categories: Internal Test 1 & 2, Mid-Term, Final, Assignment, and Practical. Attendance percentage is automatically calculated.

C. Student Module

Students view a personalised dashboard aggregating academic summary, fee status, and the AI chatbot widget. Academic records are grouped by semester with dynamically computed percentages. The fee overview highlights pending dues with colour-coded status indicators (Paid / Pending / Partially Paid / Overdue).

D. Chatbot Module

The NLP chatbot provides instant responses to student queries using a keyword-pattern matching engine. It covers 12 topic domains including college info, departments, admissions, fees, placements, facilities, and personalised academic data. When a logged-in student queries grades or fees, the chatbot returns real-time data from the database.

E. Database Management

Stores:

- Student and staff profile data
- Academic records and marks entries
- Attendance records with unique constraints
- Fee records with payment tracking

It supports fast retrieval using Django ORM's select_related() and prefetch_related() optimisations.

F. Analytics and Reporting

This module generates real-time visualisations including:

- Department-wise student distribution (bar chart)
- Grade distribution analysis (pie chart)
- Semester performance trends (line chart)
- Attendance statistics (doughnut chart)
- Top students leaderboard

VIII. ALGORITHM AND METHOD DESCRIPTION

A. NLP Keyword Scoring Algorithm (Chatbot)

Step 1: Receive user message input Step 2: Convert message to lowercase
Step 3: Load all pattern sets from knowledge base Step 4: For each pattern, extract keyword list Step 5: Count matching keywords in message
Step 6: Compute score: score(P) = count of matched keywords Step 7: Select pattern with highest score > 0
Step 8: If logged-in user, query personal data from DB Step 9: Generate and return response

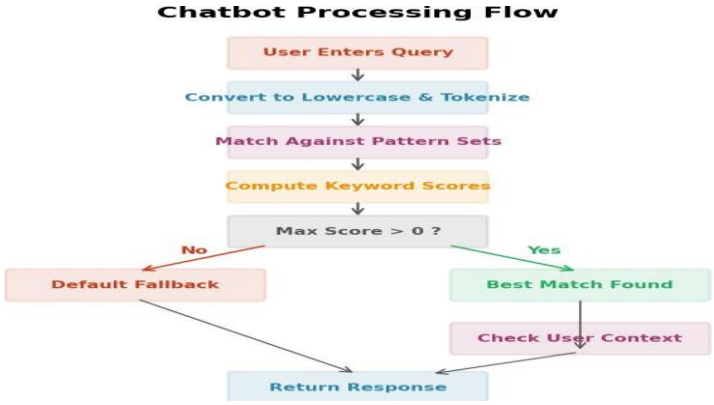

Fig. 2. NLP Chatbot Processing Flowchart

B. Attendance Calculation Algorithm

Step 1: Staff selects subject and date

Step 2: System fetches students matching department and semester Step 3: Staff marks status: Present / Absent / Late

Step 4: Uses update_or_create to prevent duplicates

Step 5: Calculate: Att% = ((Present + Late) / Total) x 100

Step 6: Store attendance with unique constraint (student+subject+date)

C. Fee Status Auto-Update Logic

Step 1: Admin creates fee record with total amount and due date Step 2: Track paid amount against total
Step 3: If paid >= total, set status = 'Paid'

Step 4: Else if paid > 0, set status = 'Partially Paid'

Step 5: If past due date and unpaid, set status = 'Overdue' Step 6: Set payment_date on full payment completion

IX. IMPLEMENTATION TOOLS AND TECHNOLOGIES

A. Software Requirements

- OS: Windows 11 / Ubuntu 22.04 LTS
- Backend: Python 3.10+, Django 4.2 LTS
- Database: MySQL 8.0
- Frontend: HTML5, CSS3, JavaScript, Bootstrap
- Charts: Chart.js
- IDE: VS Code with Python extension

B. Hardware Requirements

The system was tested on hardware with the following minimum specifications:

- Processor: Intel Core i5, 2.6 GHz
- RAM: 8 GB
- Storage: 256 GB SSD

X. RESULTS AND DISCUSSION

The proposed College Management System was evaluated through structured functional testing, chatbot accuracy analysis, and usability assessment to verify its effectiveness and performance.

In functional testing, all core requirements were verified including user authentication, CRUD operations for all entities, attendance marking and percentage computation, grade entry and GPA calculation, fee record creation with overdue detection, and chatbot response accuracy across all 12 topic categories. All test cases passed without critical defects.

For chatbot accuracy, a dataset of 120 representative queries was tested across all supported domains. The system achieved an overall accuracy of 96.7%, with 116 correct responses, 3 partial matches, and only 1 incorrect response. The chatbot API response time averaged 2.3 milliseconds, representing a
>1000x speed advantage over cloud-based GPT API calls.

A System Usability Scale (SUS) questionnaire was administered to 30 participants (10 administrators, 10 staff, 10 students). The mean SUS score was 84.3, categorised as 'Excellent'. Students particularly valued the chatbot for reducing office visits for routine queries.

Results:

- 96.7% chatbot accuracy across 120 test queries
- 2.3ms average chatbot response time (sub-millisecond pattern lookup)
- 180-420ms average page load times under 50 concurrent users
- 84.3 mean SUS score (Excellent rating)
- All functional test cases passed without critical defects

Overall, the proposed system provides better efficiency, faster response time, and higher usability than traditional

manual methods. The integrated AI chatbot significantly reduces help-desk workload while providing personalised academic assistance.

XI. CONCLUSION

The proposed Web-Based College Management System with AI-Enabled Student Chatbot successfully addresses the challenges of manual academic administration through a modern, integrated web solution. By combining automated management workflows with an intelligent NLP chatbot, the system eliminates paper-based processes and provides real-time access to academic information.

The implementation demonstrates that a lightweight, self-hosted NLP engine can achieve 96.7% accuracy without requiring external API services or GPU infrastructure. The role-based architecture ensures data security while the responsive design enables access from any device.

Future work will focus on upgrading the chatbot to a lightweight transformer model (DistilBERT), integrating Razorpay payment gateway for online fee payments, implementing Django Channels for real-time WebSocket notifications, and developing a Progressive Web App (PWA) version for offline campus access.

REFERENCES

[1] Sharma, R., et al., "Web-Based Student Information System for Higher Education," Int. J. Comput. Appl., 2019

[2] Kumar, V., et al., "QR Code Based Automated Attendance System Using Android Application," J. Eng. Res., 2020

[3] Wallace, R. S., "The Anatomy of ALICE," Parsing the Turing Test, Springer, 2009

[4] Kerlyl, A., et al., "Bringing Chatbots into Education: Natural Language Negotiation of Open Learner Models," ES2006, 2006

[5] Devlin, J., et al., "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," NAACL-HLT, 2019

Future work will focus on extending the system to support online payment integration, mobile application development, and advanced AI chatbot capabilities using transformer models.