ISSN: 2581-3404 (Online) **IF: 5.86 (SJIF)**
IJIRTM, Volume-10, Special Issue-2, April-2026

ChronoCampus – AI Powered College Resource Optimization System

Ashika[1], Gungun[2], Payal Kharb[3], Shally Nagpal[4]
Department of Computer Science & Engineering (AI & DS), Panipat Institute of Engineering and Technology, Panipat, India [1,2,3,4]
ashika922005@gmail.com[1], gungunsachdevahere@gmail.com[2], kharbpayal@gmail.com[3], shally.ngpl@gmail.com[4]

Abstract. *Managing academic activities efficiently involves proper scheduling, using resources and controlling access. Conventional ways of generating timetables are lengthy and prone to conflicts and the current systems tend to be un-scaled and rigid. In this paper, the author suggests a full-stack Smart Classroom Management System, ChronoCampus, which is based on artificial intelligence and role-based design to automate academic processes. It is a system that generates complete weekly schedules with a single inference using a Large Language Model (Google Gemini) using institutional information, including courses, faculty, and room availability, and meeting specific constraints. An access control system with four levels of Role-Based Access Control (RBAC) is used to secure access, which is supported by the authentication of the JSON Web Token (JWT) and authorization through the middleware. Moreover, an AI-enabled chatbot responds to academic inquiries in real-time, making it easier to use. The system is scalable and responsive and built with React, Node.js, and MongoDB. The experimental results prove the timetable generation is rapid with a high degree of constraint accuracy, which can be used effectively in the modern management of academic institutions.*

Keywords: Educational resource management, role-based access, academic scheduling, constraint based optimization, Genetic Algorithm.

Introduction

Learning institutions have a major problem with organization of time schedules, teacher synchronization and allocation of resources, which usually comprise of manual or constraint systems that cause inefficiencies and conflicts particularly in multi-departmental settings. The problem of timetable generation is a complicated combinatorial problem that has constraints in terms of faculty availability, course requirements, and room capacity, and, thus, the classical rule-based method is very time-consuming and prone to errors.

Recent developments in Large Language Models (LLMs) have made possible data-driven and adaptive scheduling by producing viable timetables in a single inference step, which is more complex and efficient. Moreover, the safe data management necessitates a robust Role-Based Access Control (RBAC), which is frequently in most systems. This paper suggests the creation of a Smart Classroom Management System, ChronoCampus, combining the use of AI-generated timetable generation with four-level RBAC structure. It is an LLM-based system that generates schedules according to institutional constraints and a chatbot

powered by AI to answer academic questions in real time. It is a modern, full-stack based academic management solution that offers a secure, efficient and scalable solution. The rest of the paper is divided into related work, methodology, implementation, results, and future scope.

2. Literature review

Table 1: Literature Review Table

Paper Title	Author(s)	Year	Findings
AI-Based Academic Timetable Generation Using NLP Models	Sharma et al.	2023	Utilizes NLP-based models to generate schedules from structured constraints; improves flexibility but limited in handling complex multi-department scenarios.
Large Language Models for Constraint-Based Scheduling	Lee et al.	2024	Demonstrates LLM capability to generate feasible schedules in a single inference; reduces computational overhead but lacks real-world deployment validation.
Intelligent Scheduling Systems Using Machine Learning	Kumar et al.	2022	Applies ML techniques for scheduling optimization; requires historical datasets and lacks adaptability to dynamic constraints.
Automated Academic Scheduling Using Rule-Based AI	Singh et al.	2021	Uses rule-based systems for timetable generation; ensures constraint satisfaction but lacks scalability and adaptability.
Role-Based Access Control in Web-Based Academic Systems	Verma et al.	2023	Implements RBAC for secure access; however, lacks fine-grained hierarchical control and integration with intelligent systems.
Conversational AI for Academic Assistance Systems	Patel et al.	2024	Introduces AI chatbots for academic queries using real-time data; improves usability but not integrated with scheduling modules.

3. Proposed System Architecture

ChronoCampus, the suggested system, is based on a modular full-stack model that encompasses an AI-based timetable generator and a secure approach of access control and real-time data processing. The architecture consists of four main layers: frontend, backend, AI scheduling engine, and database, which assure scalability, flexibility, and effective communication between the components.

A. System Overview

As an automated and simplified academic schedule, ChronoCampus is a full stack Smart Classroom Management System that intends to simplify and automate the academic timetable and offer safe and regulated entry. The system consists of three basic components, namely: AI-schedule generation system, Role-Based Access Control (RBAC) system, and AI-driven academic chatbot. All these are integrated into one web based system where academic activities such as the scheduling, resources allocation and access of information can occur efficiently. The schedule generator, an AI-powered process, involves a Large Language Model generating the best schedules based on institutional constraints, massively reducing the number of hand-made schedules and conflict of interests in scheduling. The RBAC system gives successive control in relation to access authority, such that different user classes such as administrators, faculty and students will obtain a warranted and secure access to system functions. In addition, the AI chatbot enhances the interaction practice with the user since it provides real-time and contextual response to the relevant academic question to rely on the live database. The design of the system takes into account scalability and flexibility and can accommodate the needs of institutions with diverse needs and growing datasets. It is easy to use and is smartly automated to minimize its administration and adds efficiency in its operations as well as to the existing academic management processes.

B. Core Components

The architecture consists of four main layers which include front end and backend to facilitate component and process scalability, flexibility and effective communication; AI scheduling engine, database.

Frontend Layer:

Users (administrators, teachers and students) can access role-based dashboards provided by the frontend layer which was developed with React. It also enables users to manage academic data, instruct timetables to generate schedules, and view schedules, and to dynamically reconfigure the interface with permissions based on their roles.

Backend Layer:

The application logic, API handling, and secure communication are handled by the backend layer that is implemented with the help of Node.js and Express. It authenticates by the use of JSON Web Tokens (JWT) and provides authorization by middleware to have controlled access to the system functionalities as it processes the request associated with timetables.

AI Timetable Generator:

The heart of the system is the AI timetable generator that is a product of the Gemini 2.5 Flash Large Language Model. It takes structured institutional data (such as courses, faculty, room availability) and derives entire weekly schedules in a single inference. The model will guarantee the satisfaction of constraints, including the prevention of scheduling conflicts, the maximization of resource allocation, without using conventional optimization algorithms.

Database Layer:

All institutional data, created timetables, and information about users are stored in the database layer that is constructed on the MongoDB Atlas. It facilitates effective data access and real- time updates, which allow smooth communication between AI module and user interfaces.

Access Control and Interaction Modules:

Role-Based Access Control (RBAC) system is integrated into the system that establishes the hierarchical roles, including Super Admin, Admin, Teacher and Student, to guarantee secure and limited access to

system resources. Also, an AI-based chatbot can improve usability by offering real-time answers to academic questions based on contextual information in the database.

Data Flow Overview:

Altogether, the architecture makes the data flow seamless, with administrative inputs being processed by the backend and used by the AI engine to generate timetables, being stored in the database and being provided to users via role-based interfaces.

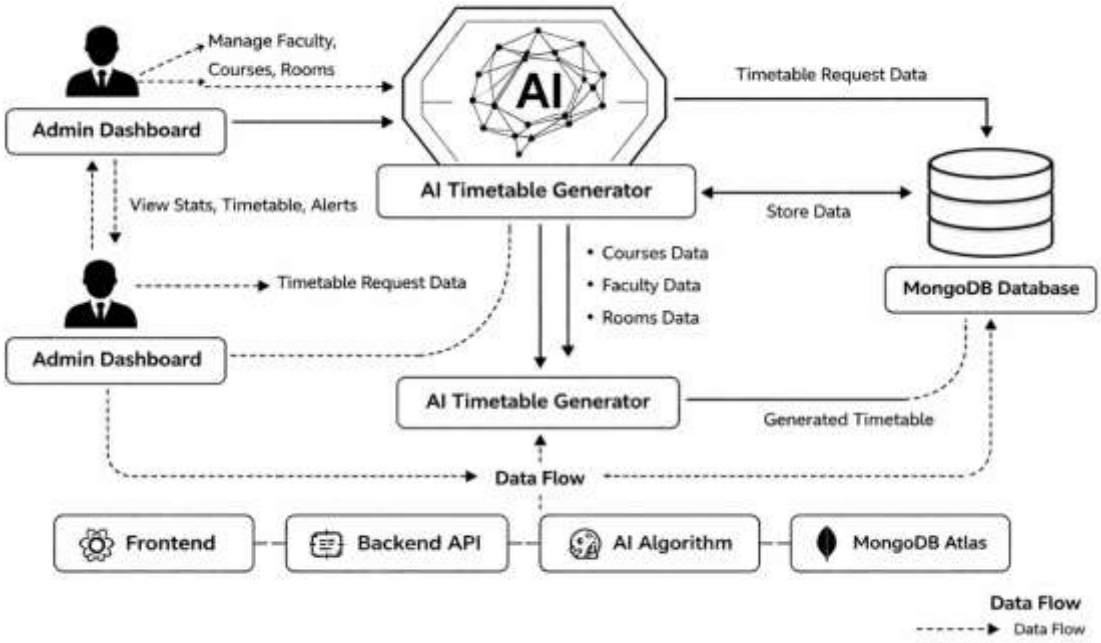

Figure 1: System Architecture

4. **Methodology**

The system, the ChronoCampus has been built on an existing full-stack architecture that is scalable and has a performance. This is created with the Express.js node.js backend to which the backend is coded to make a RESTful API and MongoDB Atlas as the database and models of data using Mongoose. JWT is used to authenticate and password hashing is done using bcrypt. The system also includes Google Gemini API that enables creation of timetables in the form of AI and chatbot and other such utilities as CORS and environment settings in order to guarantee safe and efficient functioning.

Among the most prominent aspects of the system, the integration of the Google Gemini API capable of providing not only the generation of a timetable with the help of AI but also chatbot is worth mentioning. The system facilitates the processing of institutional data (structured form) by a Large Language Model (Gemini 2.5 Flash) and creates the optimization of schedules with the aid of institutional data. The overall timetable generation process follows a system work process, whereby the inputs are collected, the timetable is generated, evaluated and refined, and high quality outputs are ensured.

A. Data Collection

The process starts with the gathering of institutional data, which is the basis of generating timetables. This incorporates elaborate details of faculty members (their specialization and availability), course arrangement, room and laboratory capacities, their predetermined time schedule, and institutional time-table restrictions. These restrictions can be such as a compulsory break, limitation of sessions, and departmental demands. The data collected is verified and organized and then forwarded to the scheduling module to ensure consistency and completeness.

B. Preliminary Schedule Generation

The initial calendar is generated along the lines of the input generated with the assistance of the Large Language Model (Gemini 2.5 Flash). Unlike the traditional method in which the optimization techniques are employed to arrive at a viable schedule, the proposed system offers a viable schedule after just one inference step. The model considers the limitations in structure and creates a schedule so as to fulfill all the requirements, including proper placement of faculty, placement of rooms and placement of time slots. This greatly saves time in creating a schedule relative to traditional approaches.

C. Evaluation of Schedule

The planned schedule is consequently checked against set up performance pointers to ensure that it is quality and deliverable. This analysis includes checking any kind of conflict in schedules, which it could be in form of duplicated faculty schedules, or reservations of rooms. Also, the system evaluates the way resources are used to make sure that the available rooms and time slots are used efficiently. The faculty workload is also analyzed to ensure that there is no overloading of some people and that there is also fairness in this. This is achieved to ensure that the timetable is in harmony with policies and job demands of the institution.

D. Fitness Check

Once the evaluation process has been completed, a fitness validation process is implemented in an attempt to define whether or not the generated timetable falls within all of the required constraints and quality standards or not. Checked constraints include hard (e.g., no overlapping schedules) and soft (e.g., balanced workload) constraints. Should the schedule meet these criteria it is said to be acceptable and is passed to the last phase. The step is a workflow decision point whereby it ensures the high quality schedules pass through.

E. Optimization Loop

The system establishes an optimization process in case the schedule does not meet the desired fitness requirements. The proposed system does not rely on genetic algorithms in selection, crossover and mutation as it is the case with the traditional methods instead, they utilise the iterative refinement with the assistance of the LLM. The system reinvents improved variations of the timetable by reshaping constraints of inputs, or re-feedbacking the model. This approach exploits the flexibility of LLMs to settle on a more optimal solution having low degree of computational complexity and low development cost.

F. Output Generation

Once the optimal or near-optimal schedule is achieved, this schedule is generated and inserted into the database of MongoDB. The result of the system is the structuring, consistency and preparedness of the timetable to be implemented. The stored schedule is then opened to the users through role based dashboards with administrators, faculty and students accessing the relevant schedules depending on their access.

G. Process Termination

When the timetable has been successfully validated and stored, the process ends. At this stage a conflict free and optimized schedule is given by the system which is now intended to be applied in the real world. The entire workflow is effective, stable and most importantly scalable and as such the system can be used in dynamic academic environment.

System Integration:

The front and back end are integrated by use of RESTful API which makes sure that there is smooth communication between various parts of the system. The frontend is based on the axios to process the HTTP requests, and JWT tokens are automatically attached to the request header to preserve authentication. The system is created to effectively perform asynchronous tasks that make the data refreshing and API services work in real-time. The seamless interaction carried out by this integration allows user interfaces, back-end services, and external AI APIs, which leads to the creation of a unified and receptive application.

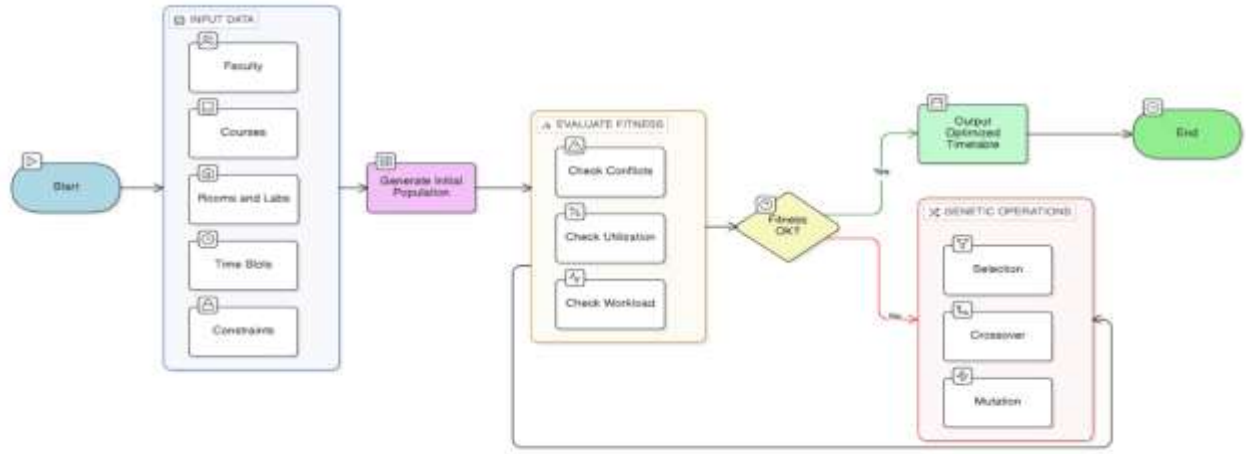

Figure 2: Methodology Flowchart

5. Results and Discussion

The simulated multi-department academic setting has proven to be highly effective with the ChronoCampus system exhibiting remarkable gains in timetable generation efficiency and the general usability of the system. The AI-based module had the capability of creating entire weekly schedules in just a few seconds, saving hours of time as compared to manual systems. It met important scheduling requirements like faculty specialization, resource allocation and completeness of a session, with very accurate results with low retries. Role-Based Access Control (RBAC) mechanism guaranteed the security and limited access to all system capabilities, whereas the dynamic user interface improved security and user-friendliness. Also, the AI chatbot responded to queries in real-time and with the correct information, depending on live database information, providing better user interaction and access to information. In general, the system was a consistent, effective and easy-to-use way of scheduling and managing academic activities in the present day. The system was also very stable in the management of the various levels of complexity of input with no impact on performance. Inherent dashboard and visualization also augmented

the decision making and monitoring by the administrators. Overall, the system was a coherent, efficient and convenient method of planning and coordinating academic affairs in the current day.

Performance Analysis

ChronoCampus system was experimented in a simulated, multi-department, academic scenario to quickly test the provisional schedule generation effectiveness, constraint fulfillment and the reactivity of the system. The AI-based schedule generator outperformed the state-of-the-art ones a lot, generating complete weekly schedules within a 4 to 10 seconds with a single inference of the Gemini API. The system was very accurate in meeting constraints like faculty assignment, room-utilization and session-completeness. Small disagreements were also sometimes witnessed because of the probabilistic nature of LLMs but were successfully addressed by regeneration. Moreover, the RBAC system provided secure and limited access to the whole user roles, enhancing system reliability.

Table 2: Performance Comparison between Proposed and Traditional System

Variable	Traditional Method	Proposed AI System
Processing Time	High (hours)	Low (seconds)
Manual Effort	High	Minimal
Accuracy	Moderate	High
Scalability	Limited	High
Real-Time Capability	Not Supported	Supported

Practical Advantages

The system has a number of useful applications in the real-life academic settings. It enables time on demand schedule creation and revision, which assists in saving much labor carried out by the management. The incorporation of RBAC ensures the secure user access control, and the AI chatbot can enhance the interaction with users by providing answers to the live information in real-time. The combined system eliminates the use of different applications making it more efficient and usable.

Table 3: Practical Advantages of Proposed System

Aspect	Description
Real-Time Processing	Allows real-time timetable generation and updates. Less Administrative Burden & manual scheduling.
Secure Access Control	RBAC role-based data protection.
AI Integration	Responds in real-time through chatbot.
System Integration	Brings together several academic functions to a platform.

Limitations

There are a few disadvantages of the system besides its benefits. The adaptation to third-party AI services results in the possibility of API access and network delay issues. Moreover, the probabilistic style of LLMs also implies that the initial outputs may contain minor contradictory pieces, and should be re-generated as a result. With a single attempt, lack of an iterative optimization can limit guaranteed conflict-free results.

Table 4: Limitations of Proposed System

Limitation	Interpretation
Network Latency	Performance is related to internet connection.
Probabilistic Output	May can cause small conflicts at first.
AI Dependency	Depends on outside Gemini API access.

6. Conclusion and Future Work

This paper presents a proposal of a Smart Classroom Management System called ChronoCampus, which can be used to address challenges of schedules and administrative control within engineering schools. It is a combination of a timetable generator basing on AI, and an ordered Role-Based Access Control (RBAC) system can provide one and scalable system of academic activity. The Gemini API enables it to generate entire weekly schedules with a single inference, with significantly less time and effort invested compared to the traditional methods, and high constraint compliance achieved at a low regeneration cost using a Large Language Model. The RBAC system offers role-based and secure access with the assistive of backend authentication and frontend dynamic rendering to enhance security and usability. Additionally, a smart chatbot will provide the ability to communicate with the academic content on-the-fly and intuitively. ChronoCampus is created on the basis of an existing full-stack structure, demonstrating a strong level of realistic feasibility, improving the performance of schedule, and reducing the administrative burden that makes it a viable step towards the future academic administration system.

ChronoCampus is a strong academic management foundation with the room to be improved further to build its scalability and functionality. The project could be developed further by implementing automated emails notifying about the schedule changes and announcements and a mobile application to facilitate the user experience and accessibility of this service. The timetable generation process should be optimized with the assistance of a hybrid approach that presupposes the implementation of both the generation method based on the LLM and the constraint validation method to decrease the number of conflicts and improve consistency. Moreover, it can be the support of multi-institution that will allow the centralization in the management of a diversity of colleges or departments. The enhancements of AI chatbot by introducing the conversational memory and introducing analytics can help supply additional data of the workload and resources of the faculty. Such a functionality as a calendar, real-time feeds, and complicated dashboards could also enhance the overall functioning of the system and its usability.

References:
[1] E. K. Burke and S. Petrovic, "Recent research directions in automated timetabling, European Journal of Operational Research, vol. 140, no. 2, pp. 266–280, 2002.
[2] A. Wren, Scheduling, timetabling and rostering - a special relationship? in Proc. PATAT, Springer, 1996, pp. 46–75.
[3] T. M. Muller, UniTime: Ten years of experience, in Proc. PATAT, 2012.
[4] H. Babaei, J. Karimpour, and A. Hadidi, A survey of approaches to university course timetabling problem, Computers & Industrial Engineering, vol. 86, pp. 43–59, 2015.
[5] R. S. Sandhu, E. J. Coyne, H. L. Feinstein, C. E. Youman, Role-based access control models, IEEE Computer, vol. 29, no. 2, pp. 38–47, 1996.
[6] N. Ferraiolo et al., Proposed NIST standard for role-based access control, ACM Transactions on Information and System Security, vol. 4, no. 3, pp. 224–274, 2001.
[7] Google LLC, Gemini API Documentation, 2024. [Online]. Available: https://ai.google.dev.
[8] MongoDB Inc., "MongoDB Atlas Documentation, 2024. [Online]. Available: https://www.mongodb.com/docs/atlas.
[9] React Documentation, React - The Library for Web and Native User Interfaces, 2024. [Online].
[10] G. Bradski, OpenCV Library, Dr. Dobb Journal of Software Tools, 2000.