iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 2026 **Issue:** Conference **Month of publication:** May 2026

DOI: https://doi.org/10.22214/ijraset.2026.83201

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue V May 2026- Available at www.ijraset.com
National Conference on Integrated Approaches to Sustainable Innovation 2026 (IASI 2026)

An Intelligent Automated Timetable Generation System Using Optimization Techniques

Soumyodip Thanadar[1], Arpan Maity[2], Tannu Jha[3], Abriti Hazra[4], Dhrubajyoti Ghosh[5]
Department of Computer Science & Engineering, OmDayal Group of Institutions, Uluberia, Howrah

Abstract- With the rapid growth of institutions and organizations, there is a strong need for an efficient, reliable, and user-friendly automatic scheduling system. Existing tools like FET and ASIMUT provide timetable generation but are often difficult to use and lack proper integration with other systems. To address these issues, we developed an Automated Timetable Management System (ATMS). The system is designed to generate and manage schedules efficiently while supporting real-time updates and minimizing conflicts. It can be applied not only in educational institutions but also in offices, hospitals, and event management scenarios. The system supports multiple platforms and provides features such as application-based access, chatbot assistance, and email notifications. It also ensures security through password encryption and OTP verification. Compared to existing solutions, ATMS offers better efficiency, scalability, and ease of use. Overall, the system provides a practical and flexible approach to automated scheduling, making timetable management simpler and more effective.

Keywords - Cloud Accessibility, Full Calendar Library, SQLite Database, SMTP Integration, Google Gemini API

I. INTRODUCTION

Timetable management systems have evolved from manual scheduling to automated solutions. Manual methods required significant time and effort and were prone to errors such as overlapping schedules and improper resource allocation. Automated systems help reduce these issues and improve efficiency. Existing tools like FET and ASIMUT provide timetable generation, but they often lack flexibility, are not very user-friendly, and do not integrate well with systems such as Learning Management Systems (LMS) and Enterprise Resource Planning (ERP). This creates challenges when real-time updates and smooth system interaction are required .To address these limitations, we developed an Automated Timetable Management System (ATMS). The system is built using Python, Tkinter, and SQLite Studio, providing a scalable framework for managing schedules of faculties, students, and subjects. ATMS supports real-time timetable modification, allowing quick resolution of scheduling conflicts. It can be used not only in educational institutions but also in offices, hospitals, and event management. The system improves accessibility through a user-friendly interface, chatbot support for quick assistance, and email notifications using SMTP. It also ensures security through password encryption and One-Time Password (OTP) verification. Additionally, the system uses technologies such as Artificial Intelligence, Machine Learning, Natural Language Processing, and data analytics to analyze and resolve scheduling issues effectively. Future improvements include mobile application support, personalized timetable generation, cloud integration, and enhanced notification features. Overall, the system provides better efficiency, accuracy, and usability compared to traditional methods and existing solutions.

II. LITERATURE SURVEY

There has been a portion of research into making the timetables modern automatically because it is such a multifaceted problem, which actually difficult to solve in many certain scenarios. To create such schedules that run effortlessly and do not have any issue or conflicts, always people have tried all sorts of methods using several technologies. For example, they have used techniques based on some specific rules, which algorithms that copycat some natural evolution, and even different types of artificial intelligence. Musa, U. B., & Oyelakin, A. M. in their research paper “A Survey of Approaches for Designing Course Timetable Scheduling Systems in Tertiary Institutions” [1], presented a proper comprehensive review of many different scheduling methods used in advanced the education system. The authors anatomized styles like constraint satisfaction, graph colouring, and meta- heuristics. Their study decided that cold-blooded methods, especially those combining the constraint propagation with the original hunt, performing better in handling large no. of scale datasets.

They also refocused out that utmost being systems warrant real- time rigidity and proper integration with institutional platforms, which limits their practical usability.

Xin Gu., Muralee Krish, Sweta Thakur & Shaleeza Sohail, in "From Integer Programming to Machine Learning: A Technical Review on Solving University Timetabling Problems" [2], Bandied the transition from traditional optimization ways to ultramodern AI- grounded approaches. This particular paper especially highlights how the Integer Linear Programming (ILP), though accurate, but also suffers from some of the scalability issues. In discrepancy, machine literacy ways, especially deep underpinning literacy, give briskly and more flexible results. The authors verified that the similar models can pointedly reduce the calculation time while maintaining the result quality, making them suitable for the dynamic scheduling environments. S. R. Khokale, Akshay Jadhav, Rupali Chavan, Parag Iwanate & Sakshi Wani, through their work "A Survey Paper on Timetable Generator Using Artificial Intelligence Methods" [3], concentrated on AI- driven scheduling ways. They discovered properly the use of artificial neural networks, inheritable algorithms, and fuzzy sense systems. Their findings suggest that the inheritable algorithms are mainly operative due to their capability to handle the multiple constraints contemporaneously and produce near the optimal results through the iterative elaboration. Hosny, M. I., in "A Survey of Genetic Algorithms for the University Timetabling Problem" [4], handed an in- depth analysis of inheritable algorithm- grounded results. This particular paper clarifies how the chromosomes are used to signify the schedules and how inheritable drivers like mutation and crossover helps in discovering the result space issue. The study concludes that inheritable algorithms are largely flexible and can be acclimated to colourful institutional conditions, making them one of the most extensively espoused approaches Colorni, A., Dorigo, M., & Maniezzo, V., in their work "Metaheuristics for High School Timetabling" [5], presented an evolutionary calculation – grounded approach for working complex timetabling problems. The authors employed some metaheuristic methods, mainly inspired by some inheritable algorithms, to efficiently explore the large result spaces and handle multiple scheduling constraints. Their research also recognized that the comparable ways are largely effective in reducing the conflicts and generating the doable calendars within sensible computational time. This paper stressed the sturdiness and inflexibility of metaheuristic styles when functional to the real-world scheduling scripts, and making them proper for the practical academic timetabling arrangements classifications. Kohshori, M. S., & Abadeh, M. S. in "Hybrid Genetic Algorithms for University Course Timetabling" [6], classified a mongrel method that combines the inheritable algorithms with the original hunt directions. This system specially improves the both confluence speed and result quality. Their results showed that hybridization, helps overcome the limitations of standard inheritable algorithms, which especially in complex scheduling scripts. Nugraha, T. A., & Karisma, T. P., Nur. H., in "University Course Timetabling with Genetic Algorithm: A Case Study" [7], applied some practical Inheritable algorithms to a real-world university scheduling issue. This study elaborated that the automated scheduling systems outstrip the homemade styles in terms of the conflict reduction and resource application. It also emphasized the significance of considering practical constraints similar as faculty vacuity and room allocation. Beligiannis, G. N., C Moschopoulos & S D Likothanassis, in "A Genetic Algorithm Approach to School Timetabling" [8], deeply focused on the applying evolutionary calculation direction to the main academy position scheduling. Their method bettered rigidity in constraint running and produced more balanced calendars. This specific study verified that adaptive algorithms can be effectively manage more real world scheduling complications system issues. Velin Kralev, Radoslava Kraleva and Borislav Yuruko., in "An Event Grouping Based Algorithm for University Timetabling" [9], proposed an indispensable optimization fashion grounded on grouping affiliated events. This system reduces the main computational complexity and improves the appropriate scalability, making it more effective for some large datasets. The authors show case that organizing events into the clusters simplifies the arrangement process without compromising the main resulted quality. Finally, Cornei, L. M., and Breabăn, M. E., in "Enhancing Genetic Algorithms with Graph Neural Networks for Solving the University Course Timetabling Problem" [10], introduced a rare ultra-modern mongrel method joining the inheritable algorithms with the Graph Neural Networks. Their model leverages the literacy capability of neural networks to better the decision- making within the inheritable algorithm frame. These results designate the substantial advancements in the prosecution time and resulted quality, and even pressing a promising way for the unborn exploration.

III. METHODOLOGY

The key purposes of the Automatic Timetable Management System are as shadows:

- To proposal and instrument an AI-driven computerized development machine accomplished of engendering conflict-free, enhanced schedules for academic organizations by negligible physical involvement.

- To progress an instantaneous struggle recognition and determination apparatus that robotically recognizes and determinations scheduling clatters concerning faculty, rooms, and time slots.
- To integrate an ordinary linguistic chatbot boundary motorized by the Google Gemini API, empowering users to request, alter, and collect timetable suggestion informally.
- To deliver multi-platform availability over a desktop GUI (Tkinter), a network boundary (Full Calendar), and automatic SMTP-based electronic mail announcements, confirming investors remain well-versed transversely campaigns.
- To certify vigorous data safety over OTP-based two-factor authentication and encrypted key storing, defending penetrating organized evidences.
- To permit continuous amalgamation through institutional organizations such as ERP and LMS platforms, confirming that development information is harmonized by larger administrative roadmaps.
- To authenticate the projected organization done qualified estimation in contradiction of prevailing tackles (FET, ASIMUT), representative enhancements in development efficiency, serviceability, and scalability.

The development of the Automatic Timetable Management System (ATMS) surveys a structured, multi-phase procedure merging requirements engineering, constraint-based development, AI incorporation, and iterative testing. The worldwide procedure is systematized into the succeeding phases:

A. Requirements Analysis

The requirements of Investor were congregated from hypothetical superintendents, faculty affiliates, and scholars through organized meetings and investigations. Real-world necessities (e.g., computerized space task, struggle recognition) and nonfunctional requirements (e.g., security, accessibility, performance) were expectable. Predominant apparatuses such as FET and ASIMUT were analyzed to distinguish breaches in serviceability, elasticity, and consolidation.

B. System Design

The system architecture was premeditated by means of a layered tactic including a Presentation Layer (Tkinter-based GUI and cross-platform mobile app), a Professional Judgement Layer (scheduling engine, conflict resolver, and AI chatbot module), and a Data Layer (SQLite database for persistent storage). A slab diagram and data flow diagram (DFD) stayed perverse to archetypal component infrastructures and facts movement across the structure.

C. Development System and AI Amalgamation

The core progress locomotive leases a constraint-satisfaction slant enlarged with a materialistic experimental to allocate subjects, faculty, and classrooms to time slots while nourishing hard restraints (e.g., no faculty double-booking, room capacity limits) and enhancing lenient constraints (e.g., preferred time windows, workload balance). Conflict recognition is implemented as a instantaneous validation module that commencements reflexive reorganization upon constraint destruction. The Google Gemini API is combined as a familiar AI layer, permitting a chatbot boundary over which users can request timetables, report struggles, and request alterations in natural linguistic.

D. Implementation

The system was accomplished in Python using Tkinter for the desktop GUI and SQLite for data persistence.

SMTP-based email notifications were cohesive to alert investors of calendar changes. OTP-based authentication and AESencrypted key payload were incorporated to verify facts preserve. The Full Calendar library was used to render collaborating agenda views on the network interface. Cloud approachability was proficient over deployment on a cloud-hosted server, enabling remote access from numerous device categories.

E. Testing and Estimation

Unit testing, integration testing, and user acceptance testing (UAT) were conducted using representative academic datasets. Performance metrics together with timetable generation time, conflict resolve rate, and user fulfillment notches were placid and associated against physical development and prevailing tools.

Active modification circumstances (faculty nonappearance, room unattainability) persisted simulated to validate immediate malleability. Feedback from UAT conferences was used to polish the boundary and arrangement lucidity preceding to concluding deployment.

1) *Existing systems :* Numerous computerized scheduling apparatuses remain presently in practice transversely enlightening organizations wide-reaching. The two most broadly referenced organizations remain FET (Free Timetabling Software) and ASIMUT (Advanced Scheduling and Information Management for Universities and Teaching institutions). FET is a free, open-source development claim that practices an erudite constraint-based procedure to create schedules for institutes and campuses. It chains a extensive variety of rigid and lenient constrictions then is accomplished of management significant development difficulties. FET activates principally as a desktop submission and depend on physical facts access, distributing calendars to inert setups such as HTML and CSV. ASIMUT is a profitable development and supply administration stage extensively used in accomplishment sculptures seminaries and conservatoires. It offers web-based admittance, room booking, and calendar views. However, its customization options are incomplete for all-purpose speculative institutions, and it carries substantial certifying charges that limit implementation in resource-constrained surroundings.

2) *Advantages :* The advantages are:

- FET is spontaneously accessible and open-source, creation it available to organizations with inadequate resources.
- Both apparatuses sustenance a comprehensive assortment of solid and soft development restrictions, enabling cohort of valid schedules for compound recognized assemblies.
- ASIMUT proposals a web-based boundary with room reservation and chart views, providing a degree of distant convenience.
- FET has an active developer communal and ropes numerous disseminate arrangements (HTML, CSV, XML), facilitating downstream data use.

3) *Disadvantages:* The disadvantages are:

- Mutually FET and ASIMUT lack instantaneous self-motivated malleability; any variations to faculty accessibility of scope projects typically need physical re-scheduling and renaissance.
- Neither system delivers native incorporation with recognized ERP or LMS platforms, necessitating manual data import/export and producing data organization trials.
- The user interfaces of both tools are complex and not user-friendly for non-technical administrators, resulting in steep knowledge bends and condensed implementation rates.
- There is no AI-based chatbot or conversational interface, restraining the capability of faculty and students to request or cooperate with the timetable organization intuitively.
- Security features such as OTP-based validation and encoded recommendation packing are absentminded, levitation worries about illegal data entree in interacted distributions.
- ASIMUT conveys substantial permitting and contribution budgets, version it unreachable to slighter or resource-constrained organizations in emergent areas.

IV. PROPOSED SYSTEMS

The future Automatic Timetable Management System (ATMS) is an inclusive, AI-integrated development stage planned to overwhelmed the boundaries of prevailing apparatuses by merging smart computerization, instantaneous malleability, vigorous safety, and multi-platform availability into a solitary amalgamated resolution. At its essential, ATMS pays a constraint-satisfaction development machine that automatically assigns subjects, faculty, and classrooms to period slots though implementing both solid restrictions (e.g., no room double-booking, no faculty overlap) and lenient restrictions (e.g., preferred teaching hours, workload distribution). After energetic variations happen—such as faculty nonattendance or classroom unattainability—the organization repeatedly perceives the resultant struggle and initiations a rearrangement development without necessitating full rejuvenation of the schedule, thereby ensuring continuousness and diminishing disturbance.

A key individual feature is the incorporation of the Google Gemini API as an entrenched AI chatbot. This familiar interface allows superintendents, faculty, and scholars to cooperate with the scheme in ordinary verbal—querying schedules, requesting alterations, and getting context-aware answers—without directing complex menus. This significantly lowers the fence to use and progresses the global user involvement.

Multi-platform availability is accomplished over a desktop application created with Tkinter, a web interface rendered using the Full Calendar library, and SMTP-based computerized email notifications that alert investors whenever timetable changes are finished. All schedule data is stored in an SQLite database, providing a insubstantial yet consistent persistence sheet appropriate for both local and cloud-hosted deployments. Security is required through OTP-based two-factor authentication at login and AES-encrypted storage of user credentials, defending institutional data in contradiction of unauthorized admittance. The system is also designed with ERP and LMS integration hooks, enabling synchronization with existing institutional information systems and positioning ATMS as a scalable enterprise-grade scheduling solution.

1) Software Systems used as Tkinter, Python, Vs code. Tkinter is a standard Python library for creating graphical user interfaces (GUIs). It provides a framework for building windows, dialogs, buttons, menus, and other GUI elements in Python applications. Tkinter is based on the Tk GUI toolkit, which originated as part of the Tcl scripting language but has been ported to many other languages, including Python. Python is a widely-used, high-level programming language known for its readability and simplicity. It is used for a variety of purposes, including web development, data analysis, artificial intelligence, scientific computing, and more. Tkinter is one of the libraries available in Python for creating GUI applications. VS Code (Visual Studio Code) is a popular open-source code editor developed by Microsoft. It supports various programming languages and features like syntax highlighting, code completion, debugging, and version control integration. VS Code is highly extensible through its vast ecosystem of extensions, which can enhance its functionality for specific programming tasks or languages.

2) Hardware Systems used as Laptop or PC, Windows 7 or higher, I3 processor system or higher, 8 GB RAM or higher and 100 GB ROM or higher.

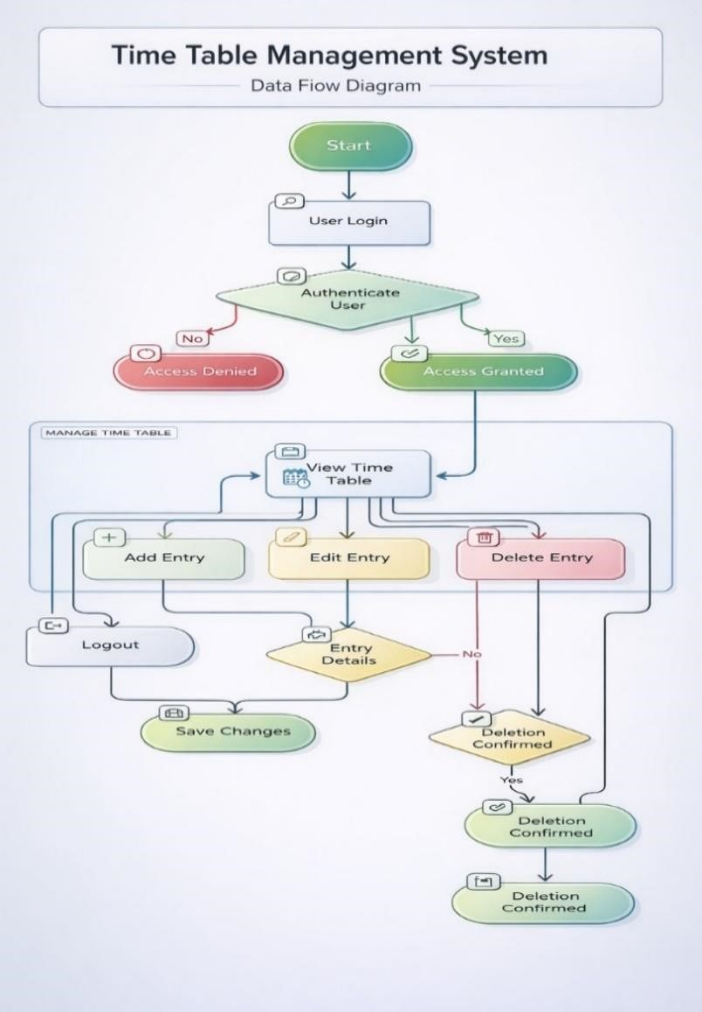


Fig.1 Block Diagram Representation

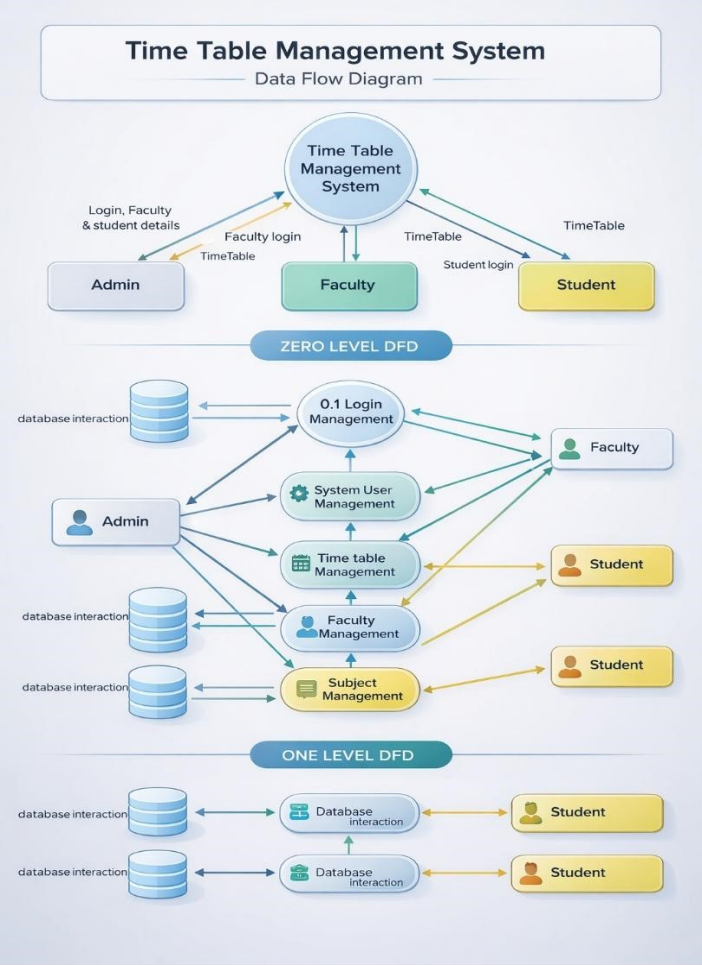


Fig.2 Data Flow Diagram Representation

V. DEMONSTRATION

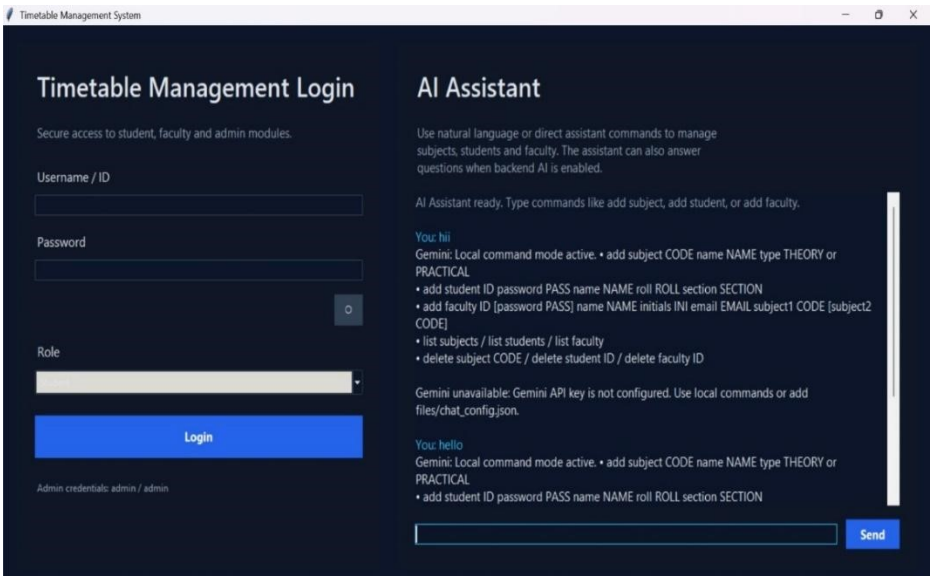


Fig. 3 Login Page of Time Table Management

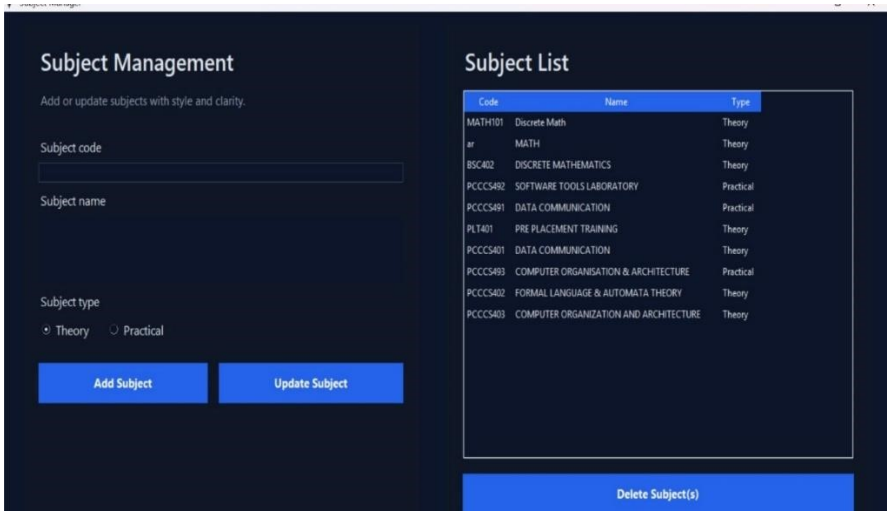


Fig. 4 Time Table Management Dashboard

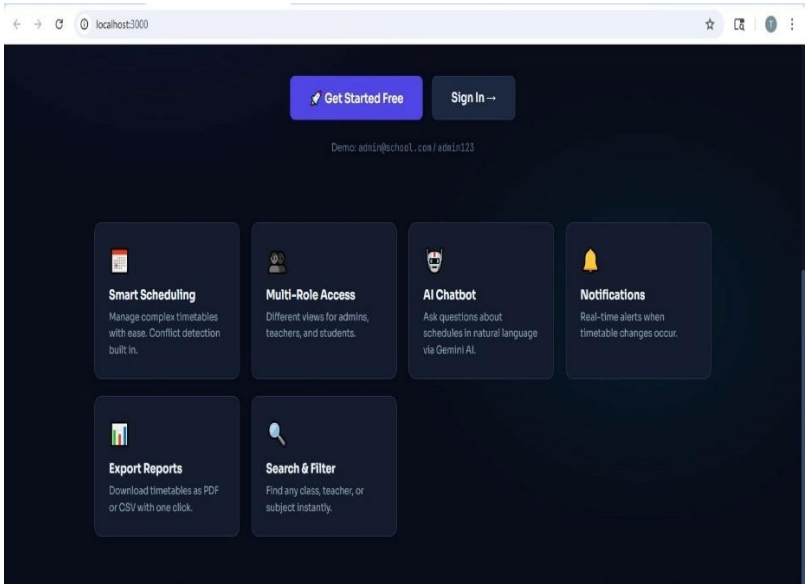


Fig 5. Internal Dashboard

VI. RESULT ANALYSIS

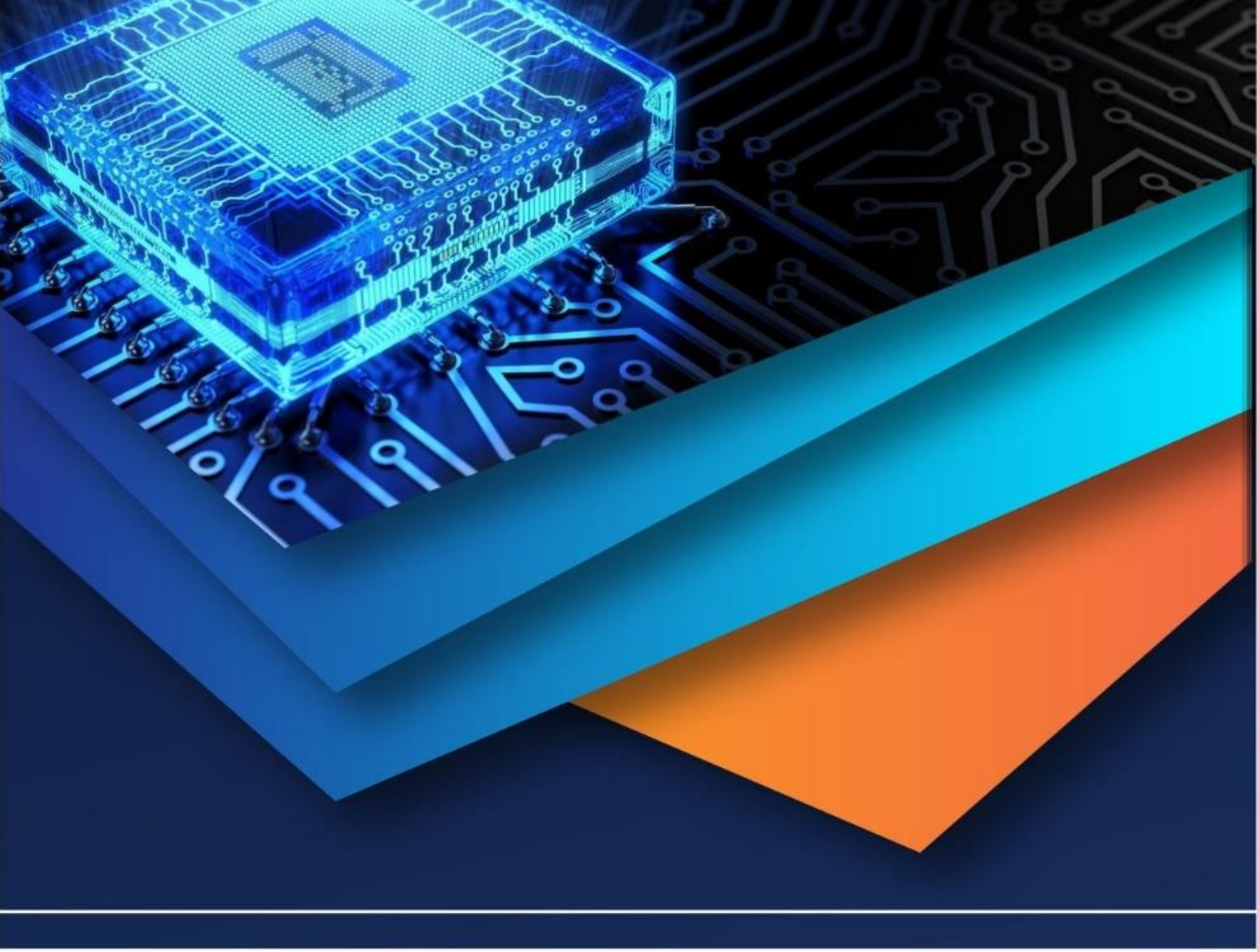
The proposed Automatic Timetable Management System (ATMS) was validated using real data in academia that involved several courses, lecturers, and class requirements. The objective of the evaluation was to analyse the proper system efficiency, conflict management, and flexibility. The result shows that the ATMS produces optimized timetables in pointedly less time compared to manual scheduling and existing tools. Common conflicts such as overlapping the faculty schedules and classroom unavailability were automatically detected and classroom unavailability were automatically detected and resolved with minimal user interference. When dynamic changes were introduced such as faculty absence or room reassignment. These demonstrations that the system is well prepared to handle with the real-time variations. The user experience has been improved significantly with the implementation of the interface on multiple platforms such as the web, mobile, and automated notifications. Additionally, integration with the systems of the organization as well as authentication processes further increased the dependability of the system. Altogether, the outcome verifies that the ATMS is a more efficient method compared to traditional ways.

VII. FUTURE PROSPECTIVE

Machine learning algorithms could be implemented in future designs to increase accuracy in predictions. There is possible for developing the system for the main distribution on cloud platforms and integrating it with the biometric classifications system. AI-based workload balancing and analytics dashboards can also be added.

REFERENCES

[1] Musa, U. B., and Oyelakin, A. M. "A Survey of Approaches for Designing Course Timetable Scheduling Systems in Tertiary Institutions." Journal of Systems Engineering and Information Technology, vol. 8, no. 3, 2024, pp. 112–129.
[2] Xin Gu., Muralee Krish, Sweta Thakur and Shaleeza Sohail "From Integer Programming to Machine Learning: A Technical Review on Solving University Timetabling Problems." Computation, vol. 13, no. 1, 2024, pp. 10–32.
[3] Mrs. S. R. Khokale, Akshay Jadhav , Rupali Chavan, Parag Iwanate and Sakshi Wani "A Survey Paper on Timetable Generator Using Artificial Intelligence Methods." International Research Journal on Advanced Engineering Hub, vol. 8, no. 3, 2023, pp. 77–85.
[4] Hosny, M. I. "A Survey of Genetic Algorithms for the University Timetabling Problem." ResearchGate, 2011.
[5] Colorni, A., Dorigo, M., and Maniezzo, V. "Metaheuristics for High School Timetabling." Proceedings of the IEEE International Conference on Evolutionary Computation, 1996, pp. 1–6.
[6] Kohshori, M. S., and Abadeh, M. S. "Hybrid Genetic Algorithms for University Course Timetabling." ResearchGate, 2012.
[7] Nugraha, T. A., Karisma, T. P., Nur. H., "University Course Timetabling with Genetic Algorithm: A Case Study." Journal of Engineering Technology, 2017. [8] Beligiannis, G. N., C Moschopoulos and S D Likothanassis, "A Genetic Algorithm Approach to School Timetabling." Journal of the Operational Research Society, 2009.
[8] Velin Kralev, Radoslava Kraleva and Borislav Yuruko, "An Event Grouping Based Algorithm for University Timetabling." arXiv preprint, 2016. [10] Cornei, L. M., and Breabăn, M. E. "Enhancing Genetic Algorithms with Graph Neural Networks: A Timetabling Case Study" arXiv preprint, 2026.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY