

Automated Time-Table Management System

Dr. Smitha Padshetty¹, Abdul Jaleel Khan Anas², Abdul Rahman Syed³, Harsh Kulkarni⁴

¹Professor, Department of Computer Science, PDA College of Engineering, Kalaburagi, India

^{2,3,4}Student, Department of Computer Science, PDA College of Engineering, Kalaburagi, India.

ABSTRACT

This project presents a web-based Automatic Timetable Generation System aimed at simplifying and digitalizing academic scheduling in higher-education institutions. The system offers a unified platform for students, faculty members, and administrators to efficiently create, manage, and visualize class and examination timetables. Administrators can dynamically generate weekly schedules by selecting parameters such as subject, faculty, semester, section, classroom, and class type (theory or laboratory). Built-in constraint validation ensures conflict-free scheduling by checking faculty availability, avoiding semester and section overlaps, and allocating continuous time slots for laboratory sessions. The system also prevents scheduling during breaks and lunch hours, thereby improving the accuracy and practicality of the generated timetables. Students can securely access personalized timetables based on their semester and section, while faculty members can view individual teaching schedules through a dedicated staff interface. In addition, the system includes an examination timetable module and a real-time notification feature that allows administrators to broadcast academic announcements instantly to all users. Developed using HTML, CSS, and JavaScript, the application employs browser-based storage for lightweight data persistence and quick access without server dependency. Overall, the proposed system reduces manual workload, minimizes scheduling errors, enhances transparency, and provides a reliable digital solution for modern academic timetable management.

Keywords: Timetable, Web Application, HTML, CSS, JavaScript, Academic Scheduling.

1.INTRODUCTION

Timetable preparation is a critical and time-consuming administrative task in educational institutions. Traditional manual methods of timetable generation often lead to scheduling conflicts, inefficient utilization of resources, and frequent revisions due to changes in faculty availability, class types, or academic requirements. As institutions grow in size and complexity, managing class schedules, laboratory sessions, examinations, and faculty workloads becomes increasingly challenging. To address these issues, an Automatic Timetable Management System provides a structured and technology-driven solution that simplifies the scheduling process. By automating timetable creation, the system ensures accurate allocation of subjects, faculty members, classrooms, and time slots while adhering to predefined academic constraints such as breaks, lunch hours, laboratory durations, and semester-section combinations.

The proposed system offers a web-based platform that supports multiple user roles including administrators, faculty, and students. Administrators can efficiently design and manage weekly class timetables, examination schedules, and institutional notifications. Faculty members can view their personalized teaching schedules, while students can access their class timetables based on semester and section selection. With features such as conflict detection, lab session validation, role-based access, and real-time updates, the system significantly reduces human effort and scheduling errors. By providing a centralized, user-friendly, and automated approach, the Automatic Timetable Generation System enhances academic planning efficiency and supports the smooth functioning of educational institutions.

1.1 OBJECTIVES

- To design and develop a web-based automatic timetable generation system that efficiently manages academic scheduling activities in higher-education institutions.
- To provide the ability to add, edit, and delete courses, faculty members, classrooms, and laboratory details while preventing conflicts such as double-booked teachers or rooms.
- To ensure conflict-free timetable generation by validating faculty availability, semester-section overlaps, laboratory continuity, and institutional break periods.
- To enable students and faculty members to easily view and access personalized timetables through a

user-friendly web interface for quick and convenient schedule lookup.

- To support real-time updates and modifications to timetables, ensuring that any changes are immediately reflected and communicated to all users.
- To provide a cost-effective and paperless solution that reduces dependency on printed timetables and manual notice boards.
- To integrate examination timetable management and academic notifications within a single centralized platform for improved communication.
- To enhance transparency, consistency, and coordination between administrators, faculty, and students through role-based access control.

2.LITERATURE SURVEY

[1] Gu et al. (2025): Gu et al. carried out a comprehensive review of integer programming (IP) and machine learning (ML) techniques applied to university timetabling problems. Their study analyzed how classical IP methods provide exact solutions but often fail to scale efficiently with the growing size and complexity of modern universities. To overcome this, they discussed the emergence of hybrid optimization models that integrate machine learning to guide or reduce the search space for integer programming. Such models allow the system to adapt dynamically to institutional data, making timetabling more efficient and personalized. They also emphasized the importance of data-driven decision-making in academic scheduling, where historical data can be leveraged to predict demand, detect bottlenecks, and improve resource allocation. This work highlights how combining formal optimization with AI opens new pathways for intelligent timetable generators that are both accurate and adaptable.

[2] Musa and Oyelakin (2023): Musa and Oyelakin presented a detailed survey covering a range of methods including rule-based systems, heuristics, and case-based reasoning for course timetable generation. Their study compared these techniques in terms of efficiency, flexibility, and ease of implementation across higher education institutions. They noted that rule-based systems are easy to design but rigid, while heuristic approaches offer faster solutions but often sacrifice optimality. Case-based reasoning, on the other hand, leverages past solutions to guide new schedules but depends heavily on the availability of high-quality datasets. A key contribution of their survey was the discussion of trade-offs between adaptability and computational cost, which institutions must balance depending on their size and complexity. For your project, their insights show that heuristics could provide a practical baseline solution, while case-based reasoning could improve adaptability over time.

[3] Dayani et al. (2023): Dayani and colleagues proposed a heuristic-based timetable generator driven by priority rules. Their system focused on resolving real-time conflicts between teachers and classrooms, which are common challenges in academic scheduling. Unlike purely algorithmic approaches, their heuristic model allowed administrators to assign weights or priorities (e.g., senior faculty preferences, core subjects, or limited lab availability). The model dynamically adjusted timetables when conflicts arose, making it highly practical for day-to-day operations. One of the major strengths of their system was its lightweight computational requirements, making it feasible for smaller institutions with limited infrastructure. This work demonstrates the effectiveness of heuristics as a flexible, fast, and user-friendly solution for timetable generation, especially when integrated with conflict resolution features.

[4] Patil et al. (2024): Patil et al. introduced a hybrid algorithmic model that combined genetic algorithms (GA) with machine learning to generate conflict-free timetables. The genetic algorithm provided a robust framework for searching large solution spaces, while machine learning helped in predicting constraints and guiding the evolutionary process. Their model was particularly effective in adapting to dynamic academic scenarios, such as last-minute faculty replacements or course adjustments. They tested their system in a university environment and found significant improvements in reducing clashes and balancing faculty workload compared to traditional scheduling methods. Moreover, the adaptive nature of the hybrid approach meant that the system could continuously learn and improve over time. This study is highly relevant to your project as it shows how combining optimization with learning-based techniques can achieve both scalability and flexibility.

[5] Abhishek M. B. et al. (2024): Abhishek and his team developed a smart timetable scheduler that went beyond scheduling to integrate with IoT-enabled classroom management systems. Their approach used genetic algorithms and particle swarm optimization (PSO) for timetable creation while incorporating devices like Raspberry Pi to automate tasks such as facial recognition-based attendance. The system also sent automated notifications to students and faculty based on timetable changes, ensuring real-time synchronization. This integration demonstrated how timetable generation can become part of a smart campus ecosystem. The combination of optimization techniques with IoT not only ensured efficient scheduling but also improved accountability and communication within institutions. Their work illustrates how timetable generation can evolve into a multi-functional system that addresses both academic and administrative needs.

3.METHODOLOGY

The proposed Automatic Timetable Generation System is implemented using a structured and rule-based methodology to ensure efficient scheduling, conflict avoidance, and ease of use. The system is designed as a web-based application and follows a modular architecture consisting of data acquisition, constraint processing, timetable generation, storage, and visualization modules.

A. System Design and Architecture

The system adopts a client-side web architecture developed using HTML, CSS, and JavaScript. Timetable data, examination schedules, and notifications are stored in structured JSON format using browser-based persistent storage. Role-based access control is implemented to differentiate between administrator, faculty, and student users.

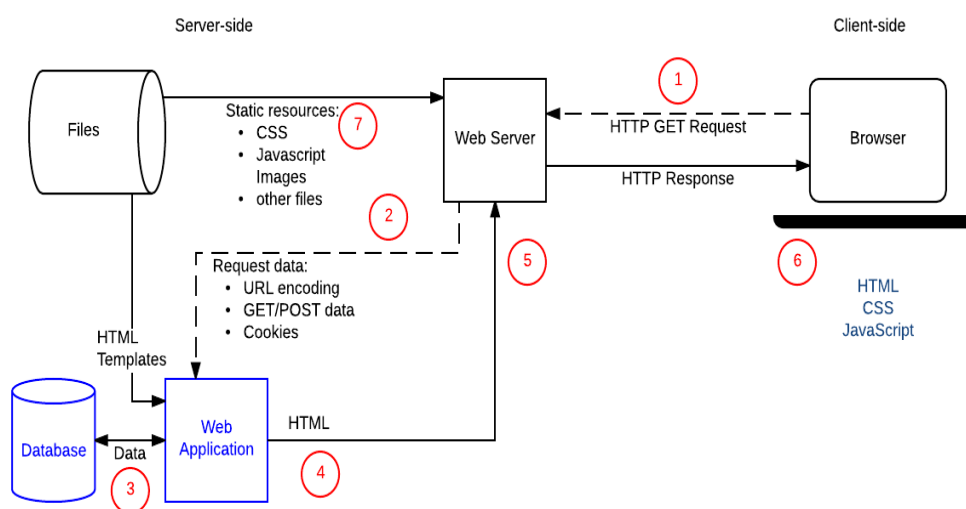


Figure 1: System Architecture

Figure 1 is a system architecture which shows the user requests from Student, Faculty, and Administrator interfaces which are processed by the JavaScript Application Logic, where role-based access control and timetable constraints are applied. Validated data is stored and retrieved in browser-based persistent storage (localStorage/sessionStorage) using JSON format, and the processed output is dynamically rendered back to the respective user interfaces.

B. Input Data Collection

The administrator provides all essential scheduling inputs through an interactive interface, including:

Academic day and time slot

Subject and course details

Faculty information

Semester and section

Classroom or laboratory details

Class type (Theory or Laboratory)

This structured input ensures completeness and uniformity of timetable data.

C. Constraint Modeling

The system incorporates a set of academic and operational constraints to ensure feasibility of the generated timetable:

No overlapping classes for the same semester and section

Faculty cannot be assigned to multiple classes at the same time

Laboratory sessions must occupy two consecutive time slots

Scheduling during break and lunch periods is prohibited. These constraints are modeled using conditional validation rules executed during timetable generation.

D. Conflict Detection and Resolution

Before finalizing any timetable entry, the system performs real-time conflict detection. If a conflict is identified, such as duplicate slot allocation or invalid lab scheduling, the system rejects the entry and provides immediate feedback to the administrator. This approach ensures that only valid and conflict-free schedules are stored.

E. Timetable Generation Process

Once validated, the timetable entries are dynamically generated and mapped into a weekly grid format. The generation process follows a deterministic, rule-based approach rather than random allocation, ensuring predictability and reliability in scheduling.

F. Data Storage and Retrieval

All timetable-related data is stored in persistent client-side storage in JSON format. Efficient retrieval mechanisms allow dynamic rendering of:

Student-specific timetables based on semester and section

Faculty-specific personal timetables

Administrative views for monitoring and updates

G. Examination Timetable and Notification Integration

In addition to class scheduling, the system supports examination timetable generation and a centralized notification module. Administrators can publish academic announcements, which are instantly accessible to all users, enhancing communication efficiency.

H. Performance and Usability Considerations

The system emphasizes low computational overhead, rapid data access, and a user-friendly interface. Visual differentiation of class slots, breaks, and laboratory sessions improves clarity and usability for end users.

4.RESULTS

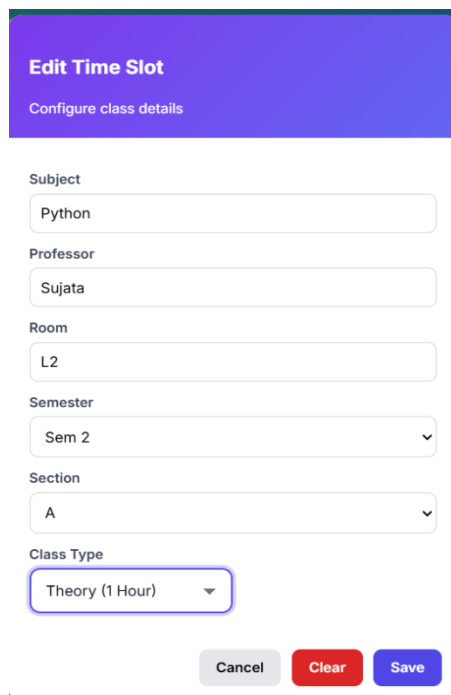


Figure 2: Edit Time Table slot

Figure 2 shows the Edit Time Table Slot interface is a modal dialog used by the administrator to configure or modify a specific timetable slot. It provides a structured form to ensure accurate and conflict-free scheduling.

Field-wise Explanation:

Subject: Specifies the course or subject name to be scheduled (e.g., Python).

Professor: Indicates the faculty member assigned to teach the subject.

Room: Defines the classroom or laboratory where the session will be conducted.

Semester: Selects the academic semester for which the class is scheduled.

Section: Identifies the class section (e.g., A, B, C).

Class Type:

Theory (1 Hour): Allocates a single time slot.

Lab / Practical (2 Hours): Automatically requires two consecutive time slots and validates continuity.

Action Buttons:

Cancel: Closes the dialog without saving changes.

Clear: Resets all input fields in the form.

Save: Validates inputs, checks for scheduling conflicts, and saves the entry to persistent storage.

Functional Significance:

This dialog enforces academic constraints (such as lab duration and break handling), reduces human error, and ensures that timetable entries are consistent across student, faculty, and administrative views. It plays a central role in enabling reliable and automated timetable generation.

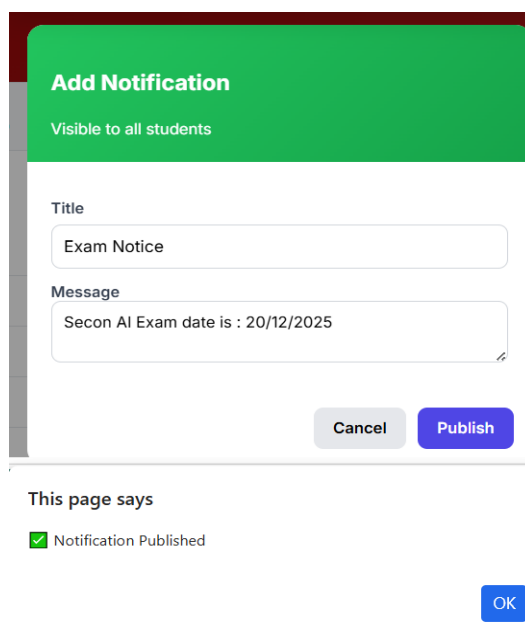


Figure 3: Add Notification

Figure 3 shows the Add Notification module allows the administrator to publish important academic announcements that are instantly visible to all students. Through this interface, the administrator enters a concise

title to summarize the notification and a detailed message to convey essential information such as examination dates, schedule changes, or institutional notices. Upon clicking the Publish button, the notification is validated, stored in persistent browser-based storage, and immediately displayed in the student notification panel. A confirmation message is shown to indicate successful publication. This module ensures efficient, real-time communication within the system, eliminating the need for external notice boards and enhancing the overall effectiveness of academic information dissemination.

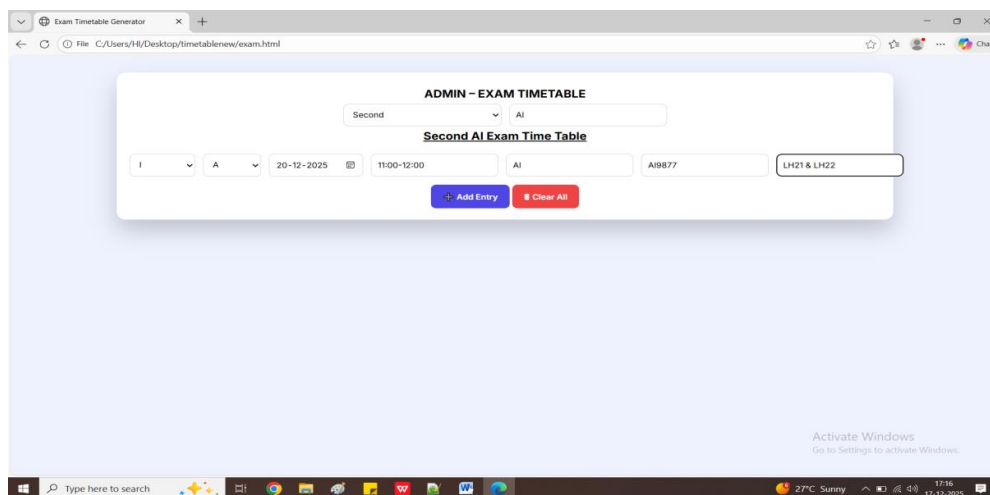


Figure 4: Add Exam Timetable

Figure 4 shows the Admin – Exam Timetable interface is designed to allow administrators to efficiently create and manage examination schedules. Through this panel, the administrator can define the exam type and exam name, which are dynamically reflected as the exam timetable heading. This ensures clear identification of the examination being scheduled, such as internal assessments or unit tests.

The form enables structured entry of examination details including semester, section, examination date, time slot, subject name, subject code, and examination room allocation. Multiple room entries can be specified for parallel seating arrangements. Action buttons such as Add Entry allow the administrator to save individual exam records, while the Clear All option enables removal of existing exam data when required.

All entered examination details are stored in a structured format and are later retrieved for display in the student-facing exam timetable view. This module centralizes exam scheduling, reduces manual errors, and ensures consistent dissemination of examination information across the system.

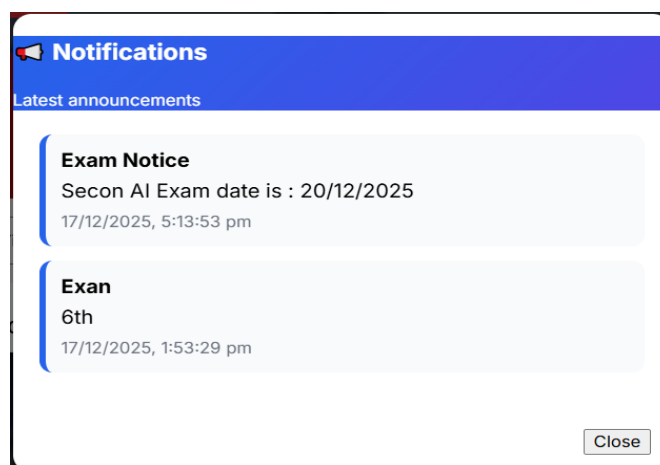


Figure 5: View Notification

Figure 5 shows the **Notifications** panel displays the latest academic announcements published by the

administrator in a clear and organized manner. Each notification is presented with a **title**, a brief **message describing the announcement**, and a **timestamp** indicating when it was issued. This allows students to quickly identify important updates such as examination schedules, notices, or urgent instructions. The notifications are listed in reverse chronological order so that the most recent announcements appear first. A **Close** button is provided to exit the panel after viewing. This feature ensures timely and effective communication between the administration and students, improving awareness and reducing reliance on physical notice boards.

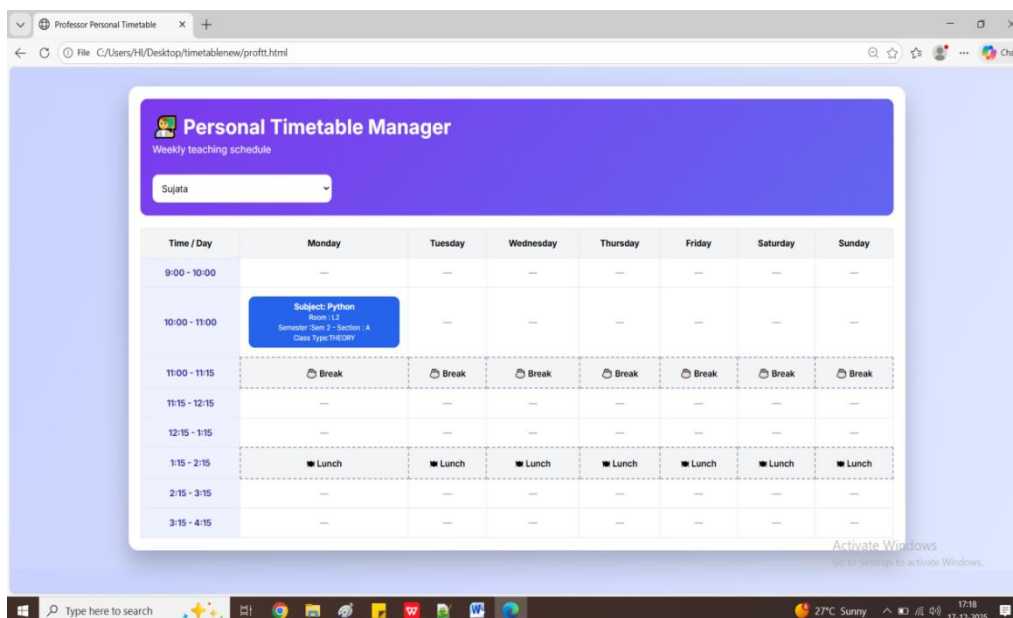


Figure 6: Staff personal time table

Figure 6 illustrates the Personal Timetable Manager interface which provides a personal time table for the faculty members with a clear and personalized view of their weekly teaching schedule. After selecting a faculty name, the system dynamically generates a time-day grid that displays all assigned classes for the selected professor. Each timetable entry includes essential details such as the subject name, classroom or laboratory, semester and section, and class type, ensuring that faculty can easily understand their teaching responsibilities. Time slots are arranged vertically, while days of the week are displayed horizontally, enabling quick identification of free periods, breaks, and occupied slots. Special slots such as breaks and lunch hours are visually distinguished from regular teaching periods to enhance readability. If no class is scheduled in a particular slot, it is marked as empty, helping faculty plan their activities efficiently. This module improves schedule transparency, reduces dependency on manual coordination, and supports effective academic planning for teaching staff.

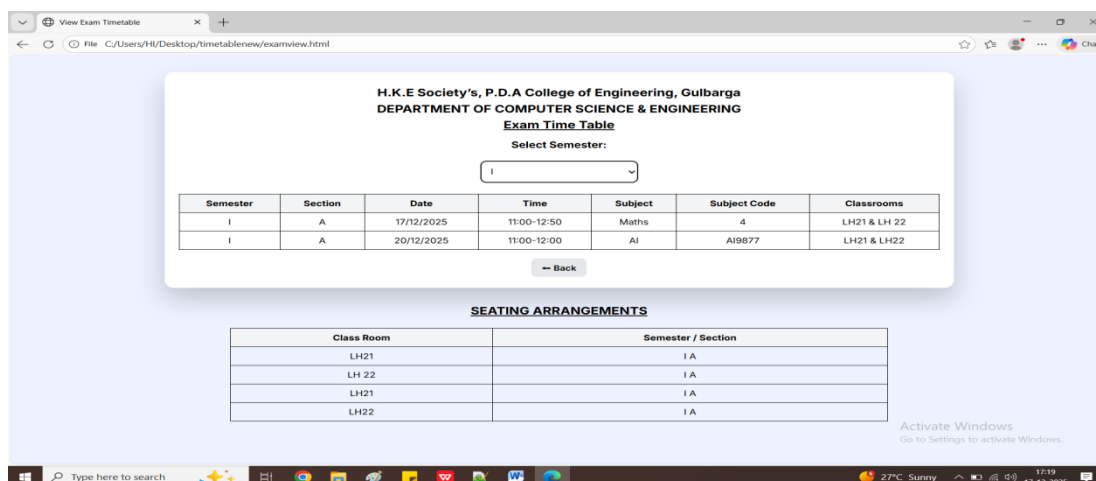


Figure 7: View Exam Time Table

Figure 7 shows the Exam Timetable and Seating Arrangement page that presents a consolidated and structured view of examination details for students and staff. At the top, institutional and department information is clearly displayed along with the exam title, ensuring proper identification of the examination session. The interface allows users to select the semester and section, after which the corresponding exam timetable is dynamically displayed in tabular form. This table includes essential details such as exam date, time, subject name, subject code, and allotted examination rooms, providing students with complete and accurate examination information in a single view.

Above the timetable, a dedicated Seating Arrangements section is provided, which maps classrooms or lecture halls to specific semester–section groups. This helps students quickly identify their assigned exam rooms, reducing confusion on the day of the examination. The clear tabular layout, separation of timetable and seating information, and centralized presentation improve readability and usability. Overall, this module enhances examination management by ensuring transparency, minimizing errors, and facilitating smooth coordination between students, faculty, and administrators.

5.CONCLUSION

In conclusion, The Automatic Timetable Generator provides an efficient and reliable solution for one of the most challenging tasks in educational institutions creating conflict-free academic schedules. By automating the process, it minimizes manual effort, reduces human errors, and ensures optimal allocation of faculty, classrooms, and courses. The system addresses both hard and soft constraints, making sure that timetables are practical, balanced, and fair for students and teachers alike. This project demonstrates how the integration of optimization techniques and software automation can significantly improve institutional management and resource utilization. In the future, the Automatic Timetable Generator can be upgraded by using advanced technologies like artificial intelligence and machine learning. This will help the system not only create timetables but also learn from past data, understand user preferences, and make better decisions. For example, it could automatically suggest the best schedule for faculty and students while reducing conflicts and improving efficiency.

REFERENCES

- [1] Gu, J., Chen, Y., & Zhao, L. (2025). A comprehensive review of integer programming and machine learning techniques for university timetabling. **Journal of Scheduling and Optimization**, 2025.
- [2] Musa, H., & Oyelakin, A. (2023). Survey on heuristic, rule-based, and case-based reasoning approaches for course timetable generation. *International Journal of Computer Applications in Education*, 2023.
- [3] Dayani, M., Rahman, A., & Singh, P. (2023). Heuristic-based automatic timetable generator for resolving teacher-classroom conflicts. *Proceedings of the International Conference on Smart Computing and Systems*, 2023.
- [4] Patil, R., Deshmukh, A., & Kulkarni, V. (2024) Hybrid genetic algorithm and machine learning model for dynamic timetable generation. *International Journal of Artificial Intelligence in Education*, 2024.
- [5] Abhishek, M. B., Kumar, S., & Thomas, J. (2024) Smart academic timetable scheduler integrating GA-PSO and IoT-enabled components. *IEEE Conference on Emerging Trends in Computing*, 2024.
- [6] Yadav, P., & Sharma, D. (2023) Automatic timetable generator using genetic algorithms for conflict-free scheduling. *International Journal of Computer Science and Applications*, 2023.
- [7] Davison, L., & Brown, T. (2024) Multi-objective integer programming for hybrid learning-based university timetable problems. *European Journal of Operational Research*, 2024.
- [8] Bonetta, G., et al. (2023) Sequence-to-sequence reinforcement learning framework for class and job-shop scheduling. *Neural Computing and Applications*, 2023.

[9] Tassel, M., & Cohen, A. (2023). Reinforcement learning with constraint programming for efficient timetable scheduling. *Journal of Applied Artificial Intelligence*, 2023.

[10] Khadivi, H., Zhang, Y., & Lee, C. (2023). Actor-critic and PPO algorithms in deep reinforcement learning for academic scheduling. *IEEE Transactions on Neural Networks and Learning Systems*, 2023.

[11] Schiewe, P., & Hoffman, J. (2023). TimPassLib: An open-source modular library for integrated timetable generation and routing. *SoftwareX*, 2023.

[12] Ansarilari, M., & Farouk, R. (2023). Scheduling framework integrating timetable coordination with resource allocation for large-scale education systems. *International Conference on Smart Systems Engineering*, 2023.

[13] Missaoui, A., & Khalid, Z. (2023). A systematic review of metaheuristic algorithms for efficient academic scheduling. *Journal of Intelligent Systems*, 2023.