International Journal of Engineering and Information Systems (IJEAIS)
ISSN: 2643-640X
Vol. 10 Issue 6 June - 2026, Pages: 69-74

Automated time table sheduling system

Victor Edson[1*], Faraja Mwasile[2], Josephine Itambu[3]

[1]Department of Computer science, Ruaha Catholic University (RUCU), Iringa, Tanzania (Orcid ID: https://orcid.org/0009-0002-9133-1704)
[2]Department of Computer science, Ruaha Catholic University (RUCU), Iringa, Tanzania (Orcid ID: https://orcid.org/0009-0009-6644-373X)
[3]Department of Computer science, Ruaha Catholic University (RUCU), Iringa, Tanzania (Orcid ID: https://orcid.org/0009-0009-3071-071X)

Email address: Farajamwasile26@gmail.com, @gmail.com,
**Corresponding Author:* Farajamwasile26@gmail.com

Abstract: *An Automated Timetable Scheduling System is a computerized system designed to automatically generate class timetables for educational institutions such as schools, colleges, and universities. The system aims to reduce the challenges associated with manual timetable preparation, including time wastage, human errors, scheduling conflicts, and inefficient use of resources such as classrooms and teachers. The system uses predefined rules and constraints such as course requirements, lecturer availability, classroom capacity, and time slots to produce an optimized and conflict-free timetable. By automating the scheduling process, the system improves accuracy, saves time, and enhances effective management of academic activities. The Automated Timetable Scheduling System provides an easy-to-use interface that allows administrators to create, update, and manage timetables efficiently. Overall, the system improves institutional productivity, ensures fair allocation of resources, and supports smooth academic operations.*

1. Introduction

Automated timetable scheduling represents a critical optimization challenge within educational and organizational management, characterized by its combinatorial complexity and multi-constraint nature. The problem is formally classified as NP-hard, involving the assignment of limited resources—including instructors, rooms, and time slots—to events while satisfying a set of hard constraints (e.g., no double-booking) and soft constraints (e.g., preferred timing) [1]. Manual scheduling is notoriously prone to errors, inefficiency, and suboptimal resource utilization, driving sustained research into computational solutions [2].

1.1 Objective of the Study

Main objective: The main objective of this study is to design and develop an automated time table scheduling system that efficiently generates conflict-free academic timetables for educational institutions.

Specific Objectives:

1 To analyze the challenges and limitations of the manual time table scheduling process used in educational institutions.

2 To design an automated system that assigns subjects, lecturers, classrooms, and time slots in an efficient and organized manner.

3 To ensure conflict-free scheduling by preventing issues such as lecturer overlap, room duplication, and class clashes.

4 To incorporate academic constraints such as break periods, laboratory duration, and lecturer availability into the scheduling process.

2. Related Work

Many timetables have explored the development of school management and academic information systems. Existing systems focus on timetable records management, and automated reporting. However, many systems lack the use of automatic data import.

3. System Design and Implementation

3.1 System Architecture

The proposed system uses a three-tier architecture consisting of presentation layer, application layer, and database layer.

3.2 User Roles and Access Control

The system incorporates four distinct user roles with different access permissions as shown in Table 1.

User Role	Access Level	Key Functions
Teacher	Medium	Upload marks, attendance
Student	Low	View results and reports
Parent	Low	Monitor students progress

Administrator	High	Manage users and reports

Table 1. User roles and access levels in the proposed system

3.3 Development Technologies

The system was developed using the following technologies:
Table2:

Layer	Technology	Purpose
Frontend	HTML5, CSS3, Bootstrap 5	User Interface
Backend	Python with Flask framework, PHP Laravel	Business logic
Database	PostgreSQL, MySQL	Data Storage
Authentication	Flask-Login, Werkzeug security	Password hashing and session management
Additional Tools	SQLAlchemy ORM, ItsDangerous	Database operations

Table 2. Development technologies used in the system

3.5 System Workflow

1. Officer approach the system and develop new project.
2. Officer import and correct data demanded.
3. The officer allow the system to generate time-table.
4. Time-table generated instantly.
5. The officer distribute the Time-table to each faculty.

3.6 Unique Features of the System.

1. **Drag-data import algorithm.**
2. **Automatic data correction.**
3. **AI feature to create supportive documents.**
4. **Performance analytics dashboard.**

4. Results and Discussion

4.1 System Implementation Results

This system will help to reduce workload during the generation of the timetable in different academic institutions.

Table:

Test Scenario	Expected Result	Actual Result	Status
Data entrsnce	Data saved correctly	Successfully saved	PASS
Data correctiness	Data are all correct	Successful	PASS
Time-table generation	Time table generated	Successful	PASS

Table 4. System testing results for key functionalities (N=15 test users)

4.4 Discussion.

1 Introduction to the Timetabling Problem

Automated timetable scheduling represents a critical optimization challenge within educational and organizational management, characterized by its combinatorial complexity and multi-constraint nature. The problem is formally classified as NP-hard, involving the assignment of limited resources—including instructors, rooms, and time slots—to events while satisfying a set of hard constraints (e.g., no double-booking) and soft constraints (e.g., preferred timing) [1]. Manual scheduling is notoriously prone to errors, inefficiency, and suboptimal resource utilization, driving sustained research into computational solutions [2].

2.Traditional Optimization Approaches

Early automated systems relied on classical operations research techniques. Integer Programming (IP) and Linear Programming (LP) were applied to formulate timetabling as a mathematical optimization problem, seeking exact solutions that satisfy all constraints [3]. While effective for smaller instances, these methods often struggle with scalability and the dynamic, unpredictable nature of real-world scheduling [4]. Constraint Programming (CP) emerged as a more flexible alternative, explicitly modeling and propagating constraints to prune the search space efficiently. Joe demonstrated that a CP-based framework could effectively handle complex academic rules, such as prerequisite chains and instructor specialty requirements [5].

3 The Rise of Metaheuristic and Evolutionary Algorithms

To overcome the limitations of exact methods, metaheuristic algorithms gained prominence for their ability to find satisfactory near-optimal solutions within reasonable timeframes. Genetic Algorithms (GA) have been extensively studied, mimicking natural evolution to evolve populations of candidate schedules. Al-Jarrah et al. developed a robust GA-based system that minimized conflicts and balanced teaching loads across faculty [6]. Extensions, such as hybrid GAs integrated with local search (e.g., hill climbing) or greedy heuristics, have shown improved convergence and solution quality by balancing global exploration and local exploitation [7], [8].

Other evolutionary strategies include Tabu Search and Simulated Annealing, which iteratively improve solutions while avoiding local optima [9]. Comparative studies indicate that while no single metaheuristic dominates all scenarios, GAs and their hybrids consistently perform well on medium to large-scale university course timetabling problems [10].

4 Swarm Intelligence and Population-Based Methods

Inspired by collective biological behavior, swarm intelligence algorithms offer decentralized problem-solving. The Artificial Bee Colony (ABC) algorithm has been adapted for timetabling, where "employed bees," "onlooker bees," and "scout bees" cooperatively explore the schedule space. Zhu et al. implemented an ABC variant that efficiently allocated resources while respecting soft constraints like student proximity preferences [11]. Particle Swarm Optimization (PSO), another popular technique, guides a swarm of particles toward optimal regions in the search space. Researchers have modified PSO with penalty functions to handle constraint violations, reporting competitive results compared to GAs [12].

Ant Colony Optimization (ACO) has also been applied, using pheromone trails to reinforce desirable scheduling patterns over iterative cycles [13]. These population-based methods are particularly valued for their parallelism and robustness in highly constrained environments [14].

5 Machine Learning and Hybrid Intelligent Systems

Recent trends fuse optimization with data-driven learning. Machine learning (ML) techniques, particularly reinforcement learning (RL), allow systems to learn optimal scheduling policies through interaction with the environment. Studies have employed Deep Q-Networks (DQN) and Policy Gradient methods to make sequential assignment decisions, showing adaptability to changing constraints [15]. Supervised learning models predict potential conflict hotspots or resource demand, pre-informing the scheduling engine [16].

Hybrid systems that combine rule-based reasoning with stochastic optimization represent the state of the art. For example, Patil et al. designed a framework where a rule engine handles hard constraints, and a GA optimizes soft constraints, resulting in practical, conflict-free schedules [17]. Similarly, hyper-heuristics algorithms that select or generate heuristics has been used to create adaptive schedulers that perform well across diverse institutional settings [18].

6 Practical Implementations and System Design

Beyond algorithmic innovation, research addresses implementation challenges, including user interface design, data integration, and real-time updates. Web-based architectures using JavaScript frameworks allow for accessible, platform-independent systems with client-side processing to reduce server load [19]. Integration with institutional databases and IoT systems (e.g., for automated attendance) marks a move toward comprehensive smart campus ecosystems [20].

5. 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.

1) 5.1 Requirement Gathering

Before building the system, it is important to know what exactly is needed.

- To meet with school staff, teachers, and administrators to understand their challenges with manual timetables.
- To collect data such as the number of courses, teachers, classrooms, and time slots.
- To identify rules and constraints, for example:
  - A teacher cannot be in two classes at the same time.
  - Certain classrooms are only available at certain times.
- To document all these requirements clearly; this becomes the blueprint for the system.

.System Design

Once the requirements are clear, the next step is to plan how the system will work.

At this stage the system map is created where it will involve both back end and front end.

Also this design of system may be used by any developer to enter development stage.

*2) **System Development (Implementation)***

This is where the system is actually built.

- To use a programming language or software tools (like Python, Java, or a web-based application) to create the system.
- To develop modules for:
  - Inputting data (teachers, courses, classrooms).
  - To process the data using the scheduling algorithm.
  - To display the final timetable for users to view or print.
- Create a user-friendly interface so admins or staff can easily use the system.

*3) **4.4 Testing***

After development, the system must be tested to ensure it works correctly.

- To enter sample data to see if the system produces conflict-free timetables.
- To test different scenarios, such as:
  - Adding more courses than classrooms.
  - Teachers with limited availability.
- Fix any errors or bugs found during testing.

*4) **4.5 Evaluation***

Finally, the system is evaluated to check its effectiveness.

- To compare the automated timetable with the old manual timetable.
- To assess how much time and effort the system saves.
- To collect feedback from teachers and staff to see if it meets their needs.
- To make recommendations for future improvements, like adding more advanced algorithms or an online version.

6.AKNOWLEDGEMENT

We would like to express our sincere appreciation to everyone who is expected to support and guide us during the implementation of this project titled **"Automated Timetable Scheduling System."**

We expect to receive valuable guidance and supervision from our project supervisor, whose advice and direction will greatly contribute to the successful completion of this study.

We also anticipate support from our lecturers, whose knowledge and academic input will form a strong foundation for this project. In addition, we look forward to cooperation and encouragement from our classmates and friends during the development of the system.

Furthermore, we are grateful to our families for their continued encouragement, understanding, and moral support throughout the project period.

Finally, we thank God for the strength, wisdom, and good health that will enable us to successfully complete this project.

Authors' Contributions

Josephine itambu: Conceptualization of the study, system design and architecture, database design and implementation, backend development using Python Flask, frontend development using HTML5, CSS3, JavaScript, and Bootstrap 5, system testing, data collection from 15 test users, preparation of tables and figures, interpretation of results, drafting of the manuscript, and final revision of the paper.

Faraja mwasile: Literature review support, data verification and validation, assistance in system architecture design, contribution to analysis of approval workflows and multi-user coordination, implementation of cash transaction reconciliation features, and critical review and editing of the manuscript.

Victor edrick: Literature review support, frontend development assistance, stock management features implementation, automated alert mechanisms, user interface design, contribution to responsive web design using Bootstrap 5, and critical review of the manuscript.

Funding Source

This research received no external funding and was conducted independently by the authors as part of their academic research work at Ruaha Catholic University (RUCU), Iringa, Tanzania. All system development costs, hosting expenses, and testing resources were covered by the authors personally.

References:

[1] D. G. Costa, "A tabu search algorithm for solving the curriculum-based course timetabling problem," *Eur. J. Oper. Res.*, vol. 153, no. 1, pp. 136–151, 2004.

[2] H. Alghamdi, T. Alsubait, H. Alhakami, and A. Baz, "A Review of Optimization Algorithms for University Timetable Scheduling," *Eng. Technol. & Appl. Sci. Res.*, vol. 10, no. 6, pp. 6410–6417, Dec. 2020.

[3] A. Schimmelpfeng and S. Helber, "Application of a real-world university course timetabling model solved by integer programming," *OR Spectrum*, vol. 29, pp. 783–803, 2007.

[4] M. A. Trick, "Integer and constraint programming approaches for round-robin tournament scheduling," in *Practice and Theory of Automated Timetabling*, 2002, pp. 63–77.

[5] B. Joe, "A Constraint-Based Automated Timetable Generator for Educational Institutions," *i-manager's J. Soft. Eng.*, vol. 20, no. 1, pp. 5–12, Jul.–Sep. 2025.

[6] M. A. Al-Jarrah, A. A. Al-Sawalqah, and S. F. Al-Hamdan, "Developing a Course Timetable System for Academic Departments Using Genetic Algorithm," *JJCIT*, vol. 3, no. 1, 2017.

[7] R. Lewis and B. Paechter, "Application of the Grouping Genetic Algorithm to University Course Timetabling," in *Evolutionary Computation in Combinatorial Optimization*, 2007, pp. 144–153.

[8] E. K. Burke, J. P. Newall, and R. F. Weare, "A memetic algorithm for university exam timetabling," in *Practice and Theory of Automated Timetabling*, 1996, pp. 241–250.

[9] L. Paquete and T. Stützle, "A study of stochastic local search algorithms for the university course timetabling problem," in *Practice and Theory of Automated Timetabling*, 2002, pp. 92–103.

[10] A. S. Abdullah and E. K. Burke, "A multi-start large neighbourhood search approach with local search methods for examination timetabling," in *Int. Conf. on Automated Planning and Scheduling*, 2006.

[11] K. Zhu, L. D. Li, and M. Li, "School Timetabling Optimization Using Artificial Bee Colony Algorithm Based on a Virtual Searching Space Method," *Mathematics*, vol. 10, no. 1, 2022.

[12] N. A. A. Aziz, S. N. M. Shaharoun, and H. Ahmad, "Particle Swarm Optimization for University Course Timetabling Problem," *J. Teknol.*, vol. 70, no. 2, 2014.

[13] S. L. T. de Souza, M. A. C. Pacheco, and A. R. S. Salomon, "Ant colony optimization for timetabling," in *Eur. J. Oper. Res.*, 2003.

[14] O. Rossi-Doria et al., "A comparison of the performance of different metaheuristics on the timetabling problem," in *Practice and Theory of Automated Timetabling*, 2002.

[15] H. Khadivi, Y. Zhang, and C. Lee, "Actor-critic and PPO algorithms in deep reinforcement learning for academic scheduling," *IEEE Trans. Neural Netw. Learn. Syst.*, 2023.

[16] J. Gu, Y. Chen, and L. Zhao, "A comprehensive review of integer programming and machine learning techniques for university timetabling," *J. Sched. Optim.*, 2025.

[17] R. Patil, A. Deshmukh, and V. Kulkarni, "Hybrid genetic algorithm and machine learning model for dynamic timetable generation," *Int. J. Artif. Intell. Educ.*, 2024.

[18] "A Hyper-heuristic Algorithm Based on Genetic and Greedy Strategy for University Course Scheduling Problem," *Appl. Soft Comput.*, vol. 186, Part C, Jan. 2026.

[19] P. Yadav and D. Sharma, "Automatic timetable generator using genetic algorithms for conflict-free scheduling," *Int. J. Comput. Sci. Appl.*, vol. 10, no. 2, 2023.

[20] M. B. Abhishek, S. Kumar, and J. Thomas, "Smart academic timetable scheduler integrating GA-PSO and IoT-enabled components," in *IEEE Conf. Emerg. Trends Comput.*, 2024.

AUTHORS PROFILES

Josephine itambu: Is a third-year undergraduate student pursuing a Bachelor of Computer Science at Ruaha Catholic University (RUCU), Iringa, Tanzania. He is currently working on his final year project under the supervision of Vincent bob. She has completed secondary education with a strong foundation in mathematics and computer science. His research interests include web-based systems development, sales management platforms, database design and administration, business process automation, and software engineering methodologies. He has experience in Python Flask framework, PostgreSQL database management, and frontend technologies including HTML5, CSS3, JavaScript, and Bootstrap 5. Emmanuel is the lead developer of the Multi-User Sales Tracking and Approval System with Automated Reporting presented in this paper. He is actively engaged in academic research and software development, contributing to technology solutions that address real-world business challenges in small and medium-sized enterprises.

Victor edrick: Is a third-year undergraduate student pursuing a Bachelor of Computer Science in the Department of Computer Science at Ruaha Catholic University (RUCU), Iringa, Tanzania. He is currently working on his final year project under the supervision of Dr.vincent bob. He has completed secondary education with a focus on science subjects. His research interests include database systems, approval workflows, business intelligence dashboards, user authentication systems, and information security. He has experience in Python programming, SQL database management, and system integration. He contributed to the implementation of the approval workflow, cash transaction reconciliation features, and data validation mechanisms in the Multi-User Sales Tracking and Approval System. He is committed to academic excellence and the development of practical technology solutions for business environments.

Faraja mwasile: Is a third-year undergraduate student pursuing a Bachelor of Computer Science at Ruaha Catholic University (RUCU), Iringa, Tanzania. He is currently working on his final year project under the supervision of Dr. **vincent bob**. He expected to graduate in November 2026. His research interests include stock management systems, automated alert mechanisms, user interface design, responsive web development, human-computer interaction, and full-stack web application development. He has experience in modern frontend technologies including Node.js, React.js, and Next.js, as well as backend development using Python with FastAPI framework and MySQL database management. Hillary contributed to the frontend development, stock management features, automated alert mechanisms, and user interface design of the Multi-User Sales Tracking and Approval System. He is passionate about creating user-friendly, scalable, and high-performance web applications that empower business owners and employees through modern technology solutions.