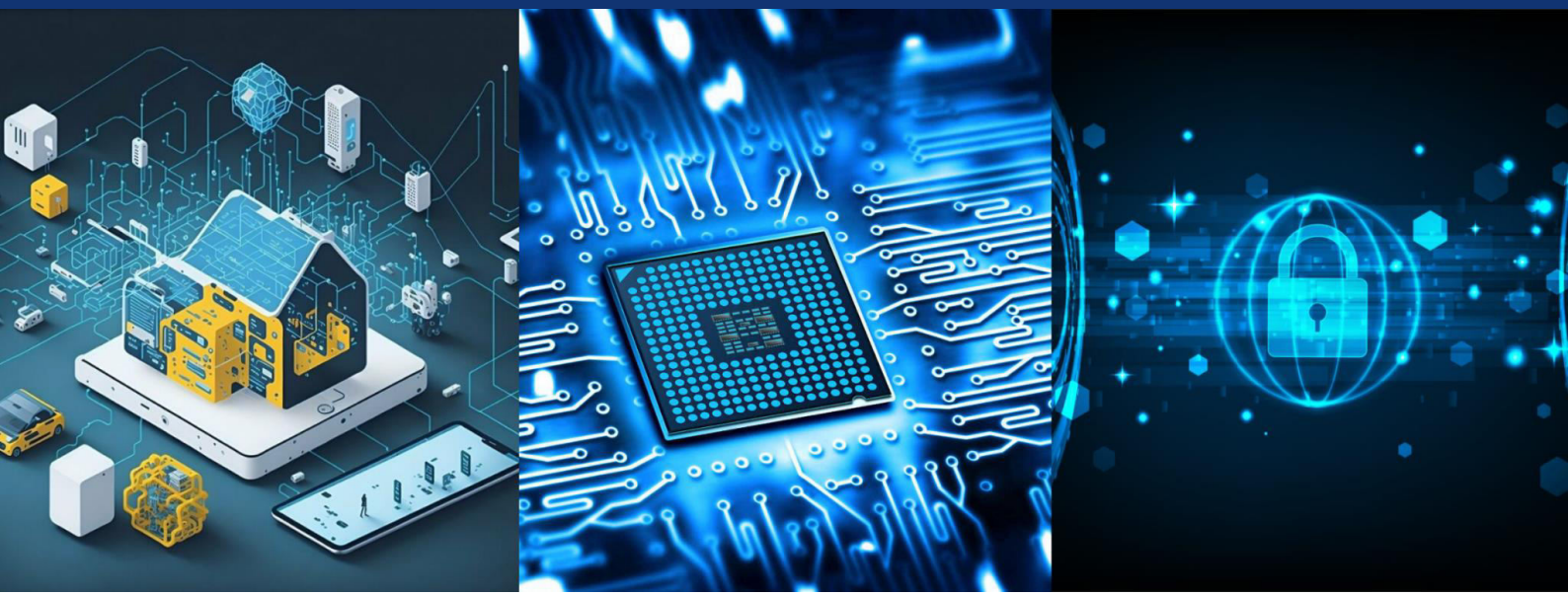
IJIRCCE

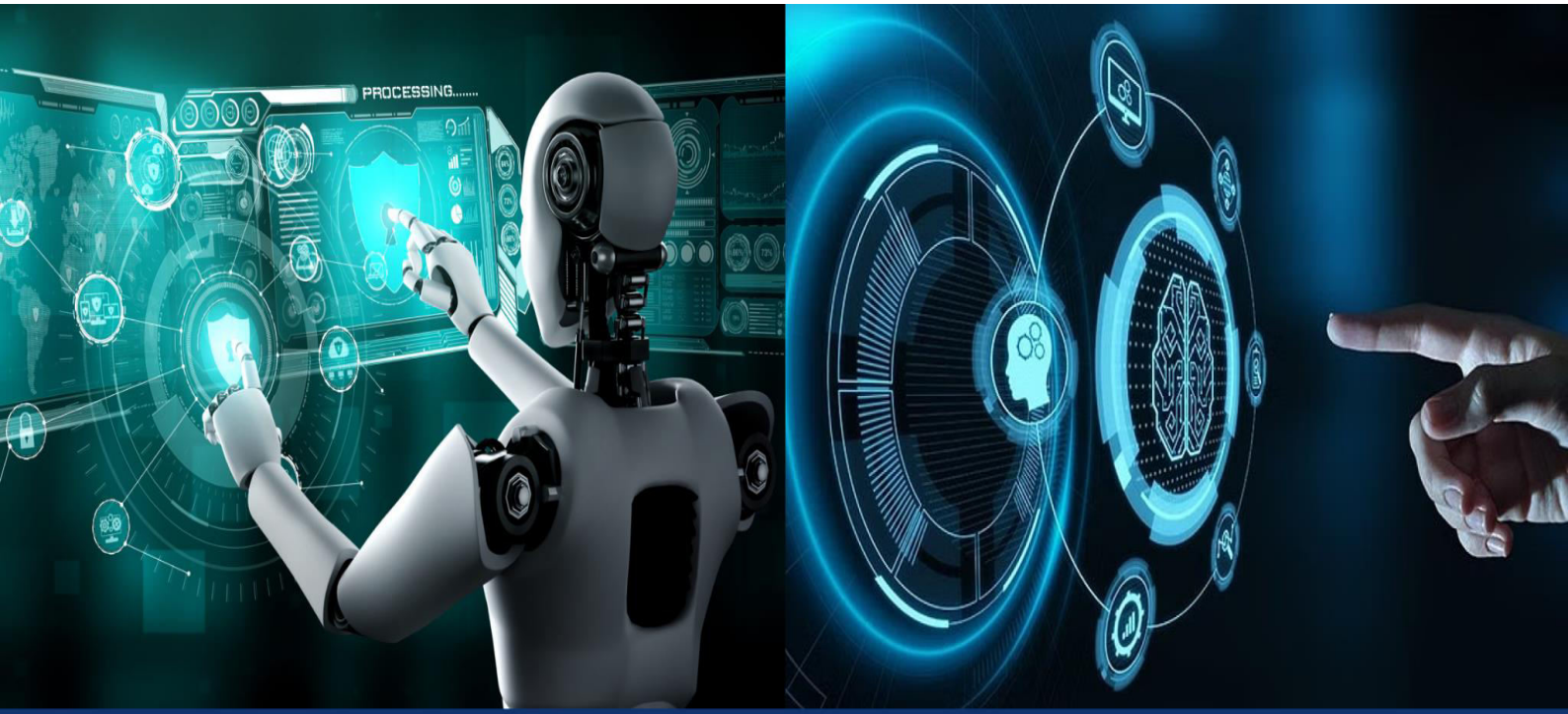
ISSN(O): 2320-9801

ISSN(P): 2320-9798



International Journal of Innovative Research in Computer and Communication Engineering

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)



Impact Factor: 8.771

Volume 14, Issue 4, April 2026

© 2026 IJIRCCE | Volume 14, Issue 4, April 2026 DOI: 10.15680/IJIRCCE.2026.1404092

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.771| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

AutoSchedule AI: Intelligent Academic Timetable Generation System Using Genetic Algorithm and Constraint-Based Optimization

Prof. Manjula P[1], Shilpa[2], Shobhitha B N[2], Tejashwini K[2], Shilpa C Javali[2]

Assistant Professor, Dept. of CSE, Jain Institute of Technology, Davangere, Karnataka, India[1]

UG Students, Dept. of CSE, Jain Institute of Technology, Davangere, Karnataka, India[2]

ABSTRACT: Scheduling academic timetables is one of the most challenging problems that schools and universities deal with on a regular basis. Doing it manually takes a great deal of time, often leads to mistakes, and simply doesn't hold up as institutions grow more complex. This paper introduces **AutoSchedule AI** — a web-based scheduling platform that takes the heavy lifting out of timetable creation by automatically generating conflict-free, well-optimized academic schedules. It does this by combining a Genetic Algorithm (GA) with constraint satisfaction techniques, striking a balance between computational intelligence and practical usability.

The system works by taking in essential inputs — things like subject details, teacher availability, classroom capacities, lab requirements, and any institutional rules that need to be respected — and then runs an evolutionary optimization process to build schedules that avoid conflicts while making the best possible use of available resources. Rather than requiring any server infrastructure, the platform runs entirely in the browser using HTML, CSS, and JavaScript, making it lightweight, easy to deploy, and accessible across different devices and operating systems.

Among its standout features are a guided step-by-step input setup, flexible management of labs and subjects, built-in conflict detection, support for exporting schedules in JSON or CSV formats, and a real-time timetable display that lets users see results as they're generated. When tested across a range of constraint scenarios, the system consistently produced valid, balanced timetables with zero teacher or classroom conflicts — and did so in under a second for typical institutional use cases.

At its core, AutoSchedule AI is about making a genuinely difficult administrative task more manageable. Efficient timetable generation matters enormously for how colleges and universities function day to day, and this system addresses that need by automating the process intelligently. It cuts down on manual effort, eliminates the usual scheduling headaches around faculty clashes and room overlaps, and ensures that subjects are distributed evenly. Its browser-based design means there's virtually no barrier to getting started — it's fast, accessible, and flexible enough to work for institutions of varying sizes.

KEYWORDS: Automatic Timetable Generation, Genetic Algorithm, Constraint Satisfaction, Academic Scheduling, Evolutionary Optimization, Web-Based Application, Conflict-Free Scheduling

I. INTRODUCTION

Academic timetable scheduling is a fundamental administrative challenge in educational institutions ranging from schools to universities. The problem involves assigning courses, teachers, classrooms, and time slots in such a way that a large and complex set of hard and soft constraints is satisfied simultaneously. Hard constraints include requirements such as no teacher teaching two classes at the same time, no classroom double-booked in a given period, and each subject receiving its required number of weekly periods. Soft constraints involve preferences such as balanced teacher workloads, avoidance of isolated single periods, and grouping lab sessions on specific days.

As institutional complexity grows with increasing numbers of departments, courses, faculty members, and students the manual approach to timetable preparation becomes practically infeasible. Human schedulers must track hundreds of interdependent variables while ensuring that no constraint is violated, a task that often takes days of effort and still

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.771| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

results in suboptimal or conflict-ridden schedules. Any change to one schedule (such as a teacher becoming unavailable) can trigger a cascade of reassignments across the entire timetable.

AutoSchedule AI is an AI-driven timetable generation platform designed to address these scheduling challenges. The proposed system employs a Genetic Algorithm (GA) integrated with constraint-based verification to automatically produce valid, optimized academic timetables from user-provided institutional data. Built as a fully client-side web application using HTML, CSS, and JavaScript, the platform eliminates the need for server infrastructure and ensures immediate accessibility from any standard web browser.

Beyond automated generation, AutoSchedule AI supports dynamic input configuration for subjects, teachers, lab sessions, and basic institutional settings such as working days and periods per day. The system detects and resolves scheduling conflicts in real time and provides export options in JSON and CSV formats for downstream integration with institutional systems. By combining intelligent evolutionary optimization with a simple, user-friendly interface, AutoSchedule AI delivers a practical, scalable solution that can significantly reduce administrative burden and improve the quality of academic scheduling in institutions of all sizes.

Putting together an academic timetable is no small task — it directly shapes how smoothly a college or university runs day to day. When done manually, it almost always results in errors, scheduling conflicts, and a heavy burden on administrative staff. AutoSchedule AI was built to change that. By applying intelligent automation, it takes over the scheduling process and generates accurate, optimized timetables in a fraction of the time, all while respecting the specific requirements and resource constraints of the institution.

II. RELATED WORK

The problem of automated timetable generation has been studied extensively in the operations research and artificial intelligence literature. Several algorithmic approaches have been explored, ranging from classical combinatorial methods to modern evolutionary and heuristic techniques.

Srinivasan et al. (2025) addressed highly constrained university timetable generation using Evolutionary Algorithms (EA) with heuristic support, problem-specific chromosome design, and adaptive mutation with context-based reasoning. Their work demonstrated that evolutionary approaches can handle complex, multi-departmental scheduling problems that are intractable for exact methods, forming a foundational basis for the genetic algorithm employed in AutoSchedule AI.

Bagul et al. (2024) proposed a heuristic-based scheduling system with prioritization rules, constraint checking, and a resource scheduling algorithm focused on generating clash-free timetables while considering faculty availability and priorities. This work highlights the importance of combining constraint checking with heuristic guidance to improve schedule quality, a principle adopted in the conflict detection module of the proposed system.

Markal et al. (2023) developed a Genetic Algorithm-based timetable system using chromosome representation, mutation, and fitness evaluation for optimal schedule search. Their results confirm GA as a highly effective approach for timetable optimization, particularly when subject to hard constraints such as teacher availability and room capacity.

Wankhede et al. (2022) investigated automated college timetable generation using GA with constraint satisfaction and chromosome representation. Their system specifically targeted the elimination of class and teacher clashes, achieving improvements in schedule quality and generation time compared to manual approaches.

Samah et al. (2022) proposed a greedy-based algorithm for optimizing student-recommended timetables with semester planning functionality. Their work demonstrates that algorithmic scheduling can extend beyond course-level scheduling to support student-centric academic planning, suggesting future directions for the AutoSchedule AI platform.

Rane et al. (2021) developed an automated timetabling system for university courses using IEEE-standard scheduling methods, demonstrating the scalability of automated systems in large academic environments. Taken together, these prior works establish the strong theoretical and practical foundation upon which AutoSchedule AI is built, while the present system contributes a fully browser-based, lightweight implementation with real-time user interaction.

© 2026 IJIRCCE | Volume 14, Issue 4, April 2026 DOI: 10.15680/IJIRCCE.2026.1404092

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.771| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

III. PROPOSED ALGORITHM

AutoSchedule AI is built to take the complexity out of academic scheduling by automatically producing timetables that are free of conflicts and fully aligned with institutional constraints — all powered by a Genetic Algorithm (GA) working alongside a constraint verification engine. The system is divided into five primary modules: Input Configuration, Data Structuring, Genetic Algorithm Optimization, Conflict Verification, and Output Generation. These modules work together in a sequential pipeline to transform raw institutional data into a complete, usable academic timetable.

3.1 Input Acquisition and Configuration

The first stage involves collecting all necessary scheduling parameters from the administrator through a step-by-step web interface. The system accepts the following inputs:

- Basic Settings: Working days, number of periods per day, break slot position, and lunch slot position.
- Lab Settings: Lab subject name, assigned section (A/B/C), and batch group, enabling multi-batch lab scheduling within the same subject.
- Subject Settings: Subject code, subject name, teacher assignment, and the number of weekly periods required for each subject.

All inputs are validated at the point of entry to ensure completeness and consistency before the algorithm is invoked. The configuration step produces a structured dataset that serves as the primary input to the optimization engine.

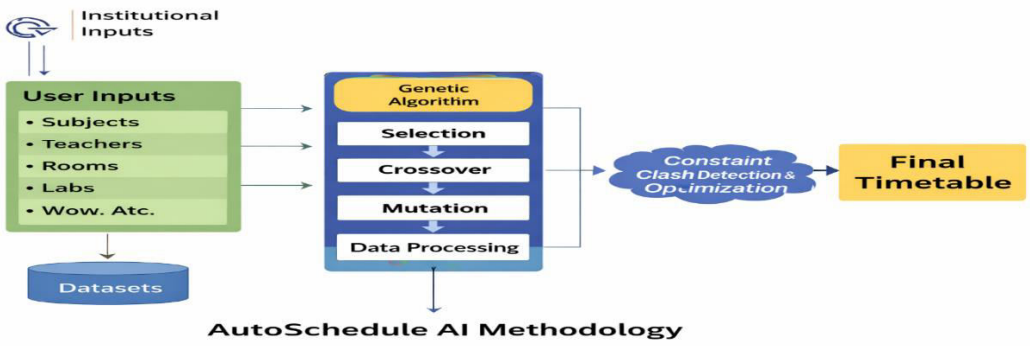


Figure 3.1: Methodology of AutoSchedule AI for Academic Timetable Generation

3.2 Data Structuring and Timetable Initialization

After input collection, the system organizes all data into JavaScript objects and arrays that represent the scheduling state. An empty timetable grid is initialized for each class and section, with rows corresponding to working days and columns corresponding to period slots. Pre-assigned fixed slots such as break periods and lunch periods are placed into the grid before the optimization begins. Teacher availability maps are constructed to track which teachers are already assigned to a given slot during the optimization process.

The initialization step produces the following data structures:

- Class timetable grids: Empty matrices indexed by [day][period] for each class section.
- Teacher availability maps: Boolean matrices indexed by [teacher][day][period].
- Subject requirement counters: Tracks remaining periods to be assigned for each subject.
- Lab session records: Pre-defined multi-period blocks for batch lab sessions.

3.3 Genetic Algorithm Optimization

The core scheduling logic is implemented as a Genetic Algorithm (GA), an evolutionary optimization technique inspired by natural selection. The GA searches for optimal timetable configurations by evolving a population of candidate solutions over successive generations. The algorithm follows the standard GA lifecycle:

© 2026 IJIRCCE | Volume 14, Issue 4, April 2026 DOI: 10.15680/IJIRCCE.2026.1404092

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.771| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

Population Initialization: An initial population of random timetable solutions is generated. Each individual (chromosome) in the population represents a complete timetable configuration, encoded as a two-dimensional assignment matrix mapping periods to subjects and teachers for each class.

Fitness Evaluation: Each chromosome is evaluated by a fitness function that quantifies schedule quality. The fitness function is defined as:
$F(\text{chromosome}) = W_1 \times (1 - C\_hard) + W_2 \times S\_soft$

Where C_hard represents the fraction of hard constraint violations (teacher clashes, room conflicts, period over-allocation), S_soft represents the soft constraint satisfaction score (workload balance, period distribution), and W_1, W_2 are weighting coefficients with $W_1 >> W_2$ to prioritize hard constraint satisfaction.

Selection: Chromosomes with higher fitness scores are preferentially selected for reproduction using tournament selection, ensuring that better solutions have a higher probability of contributing to the next generation.
Crossover: Two parent timetables are combined by swapping their scheduling assignments at a randomly chosen point, so each resulting offspring picks up traits from both parents.

Mutation: Small random changes are occasionally applied to offspring — specifically, swapping two period assignments - to keep the process from settling too early and to maintain variety across solutions.

Iteration: The algorithm keeps running through new generations until either a set number of generations is completed or a perfect solution with zero constraint violations is found. Whatever the best result is at that point becomes the final optimized timetable.

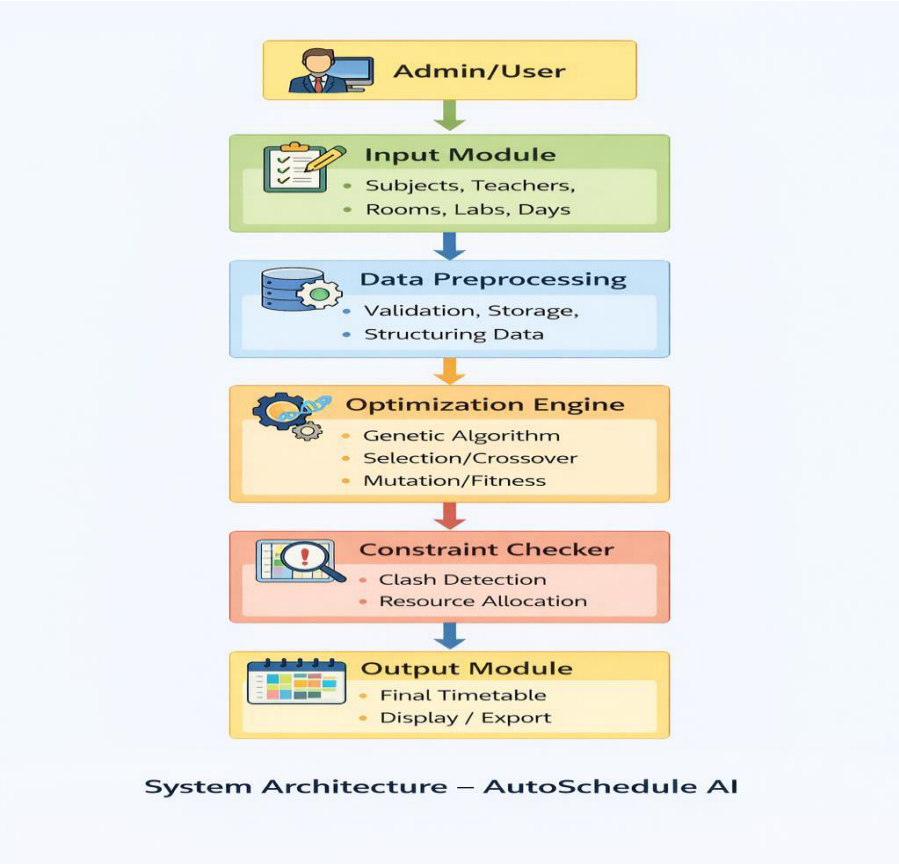


Figure 3.3: System Architecture of AutoSchedule AI for Academic Timetable Generation

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.771| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

3.4 Conflict Detection and Constraint Verification

Before finalizing the generated timetable, the system performs comprehensive constraint verification to ensure correctness. The constraint verification engine checks:

- Teacher Clash Detection: Verifies that no teacher is assigned to more than one class in the same period.
- Period Allocation Verification: Confirms that each subject has received its entered number of weekly periods.
- Lab Block Integrity: Ensures that lab sessions are allocated as continuous multi-period blocks and that batch rotations are correctly scheduled.
- Break and Lunch Preservation: Confirms that fixed break and lunch slots are not overwritten by subject assignments.

When a constraint violation is detected, the system doesn't just stop — it automatically kicks in a backtracking mechanism that tries to reassign the conflicting entries to different slots. It works through alternative options one by one until it lands on a valid arrangement.

3.5 Timetable Output and Export

Upon successful generation, the confirmed timetable is rendered dynamically in the browser as an HTML table. The display provides a clear day-by-period grid for each class section, with subject codes and teacher names displayed in each slot. Fixed periods such as interval and lunch are visually distinguished from regular subject slots.

The system provides two export options for downstream use:

- JSON Export: Exports the complete timetable data structure in JSON format for integration with institutional management systems or further programmatic processing.
- CSV Export: Exports the timetable in comma-separated format for use in spreadsheet applications such as Microsoft Excel or Google Sheets.

3.6 Pseudocode of Proposed Algorithm

Step 1: Initialize system – Load configuration, prepare empty timetable grids, assign fixed slots (break, lunch).
Step 2: Collect inputs – Accept subject, teacher, lab, and settings data from administrator.
Step 3: Build data structures – Construct teacher availability maps and subject requirement slot.
Step 4: Initialize GA population – Generate random candidate timetable chromosomes.
Step 5: Evaluate fitness – Score each chromosome against hard and soft constraints.
Step 6: Selection – Choose high-fitness chromosomes for reproduction.
Step 7: Crossover & Mutation – Produce next-generation chromosomes.
Step 8: Repeat Steps 5–7 until termination criterion is met.
Step 9: Extract best chromosome – Select the most optimal timetable generated.
Step 10: Constraint verification – Run full conflict check; backtrack if needed.
Step 11: Render timetable – Display final schedule in browser as HTML table.
Step 12: Export – Provide JSON/CSV download options.
Step 13: End.

IV. SIMULATION RESULTS

The AutoSchedule AI system was evaluated through functional testing, constraint verification testing, performance testing, and cross-browser compatibility testing to validate the effectiveness of the proposed timetable generation platform. Results demonstrate that the system successfully integrates GA-based optimization, real-time conflict detection, and a user-friendly administrative interface into a unified scheduling platform.

Basic Settings Interface: The Step 1 panel of the AutoSchedule AI interface allows administrators to configure foundational scheduling parameters including working days, the number of periods per day, and the positions of break and lunch slots. During testing, the system correctly applied all configured settings to the timetable generation process, with the configurable break and lunch slots reliably excluded from subject assignments across all generated timetables.

Lab Configuration Module: The Step 2 panel enables administrators to define lab sessions by selecting the lab subject, class section, and batch. The system dynamically updates batch options based on section selection, ensuring valid batch assignments. During testing, multi-batch lab sessions were correctly scheduled as continuous multi-period blocks with appropriate batch rotation, and no lab session conflicts were observed in any generated timetable.

© 2026 IJIRCCE | Volume 14, Issue 4, April 2026 DOI: 10.15680/IJIRCCE.2026.1404092

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.771| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

Subject Entry Interface: The Step 3 panel provides dynamic subject entry forms where administrators specify subject codes, names, assigned teachers, and required weekly periods. The system validated all subject entries before timetable generation and correctly mapped each subject to its assigned teacher in the scheduling constraints.

Generated Timetable Display: After triggering generation, the system renders the complete timetable for each class section in a structured HTML table. Each cell displays the subject code and teacher name. Break and lunch periods are shown with distinct visual formatting. In all test scenarios, the generated timetables were verified to be free of teacher clashes and period allocation errors.

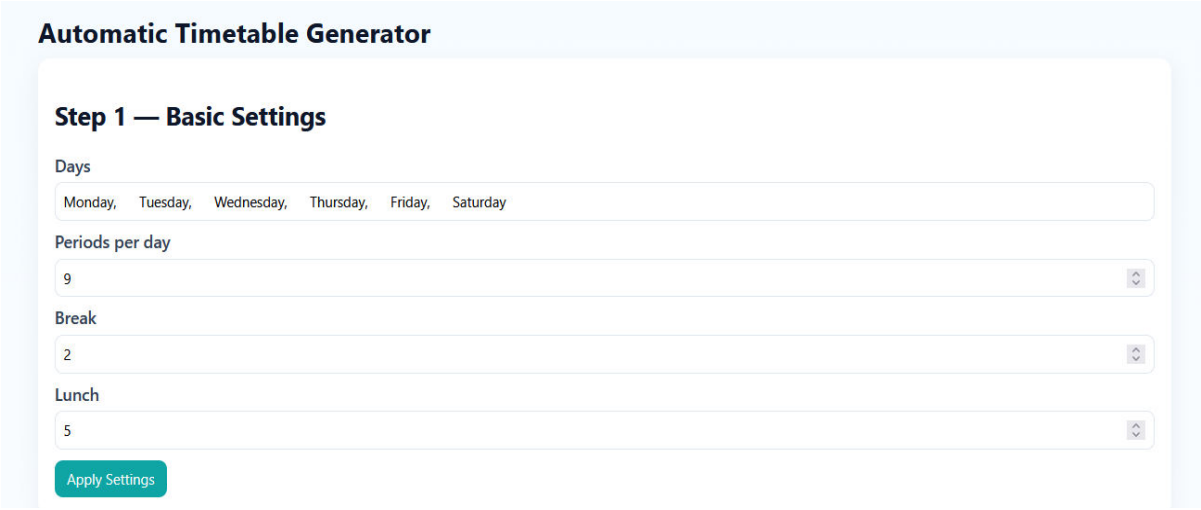


Figure 4.1: Basic Settings

The Basic Settings page of the Automatic Timetable Generator system. It allows the user to configure working days, number of periods per day, break slot, and lunch slot before timetable creation. These settings are used as the foundation for generating an accurate and structured academic schedule. After entering the details, the user clicks Apply Settings to save the configuration.

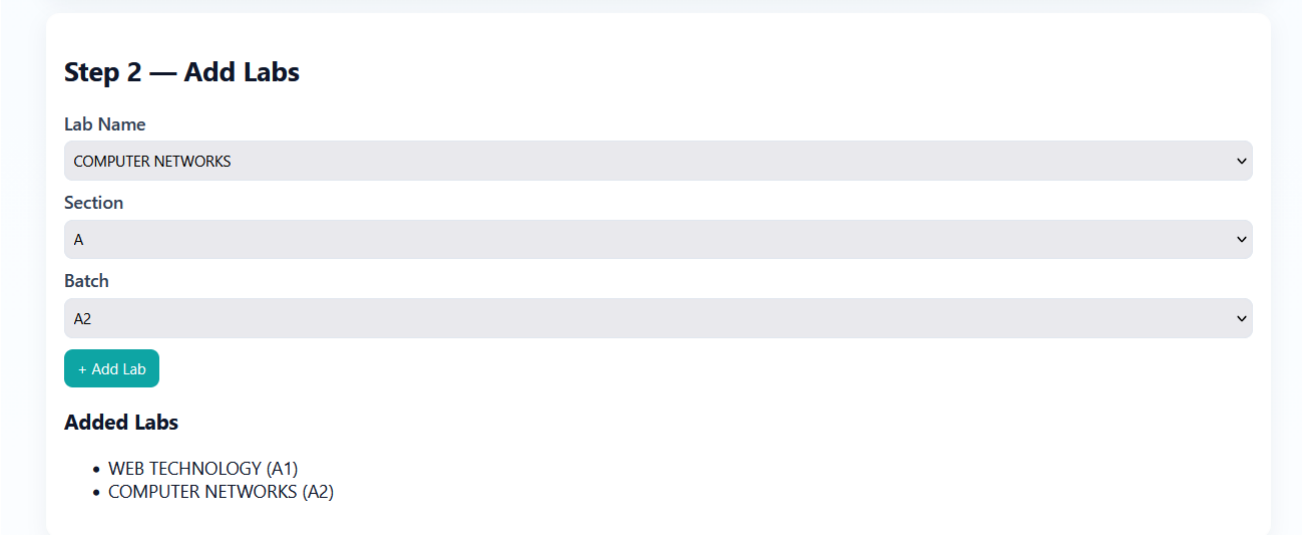


Figure 4.2: Add Labs

© 2026 IJIRCCE | Volume 14, Issue 4, April 2026 DOI: 10.15680/IJIRCCE.2026.1404092

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.771| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

Step 2 — Add Labs, where a user selects a lab name, section, and batch from dropdown menus. After clicking "+ Add Lab," the chosen labs are listed below under "Added Labs." In this example, Web Technology (A1) and Computer Networks (A2) have been added successfully.

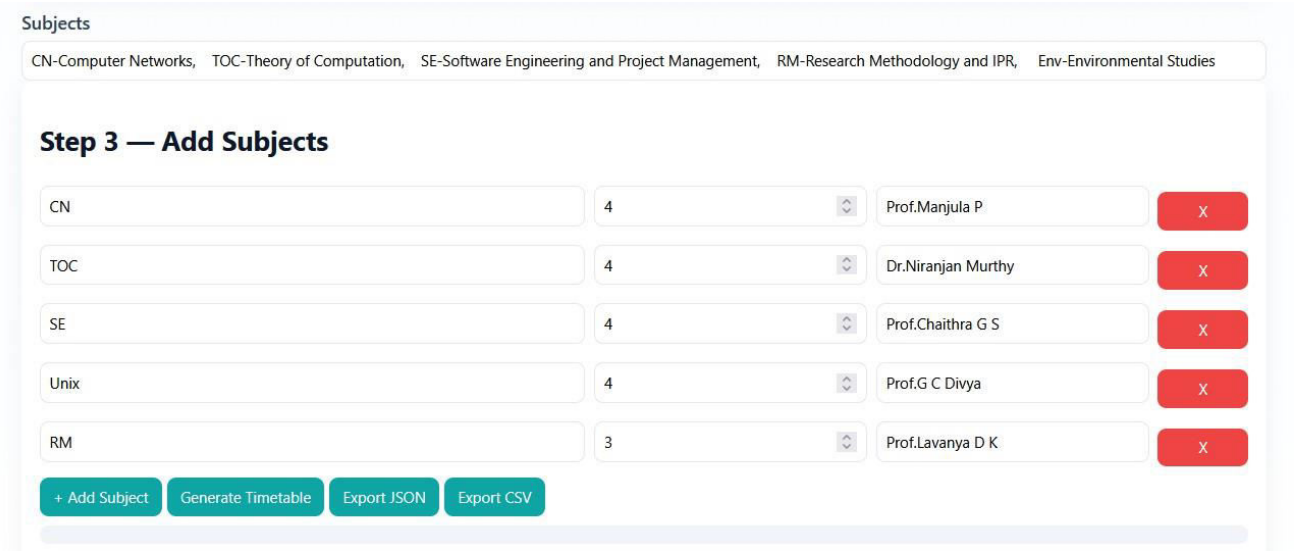


Figure 4.3: Add Subjects

Step 3 — Add Subjects, where different subjects are listed along with their assigned hours/credits and faculty members. Each row includes an option to remove a subject using the red "X" button. At the bottom, there are action buttons to add more subjects, generate a timetable, or export the data in JSON or CSV format.

Generated Timetable

	9:00AM-10:00AM	10:00AM-11:00AM	11:00AM-11:15AM	11:15AM-12:15PM	12:15PM-1:15PM	1:15PM-2:00PM	2:00PM-3:00PM	3:00PM-4:00PM	4:00PM-5:00PM
Monday	RM Prof.Lavanya D K	SE Prof.Chaithra G S	Break	Mini Project Prof. Basavaraj Patil	Env Prof. Harish K S	Lunch	CN Prof.Manjula P	NSS Prof. Chaithra G S	TOC Dr.Niranjan Murthy
Tuesday	NSS Prof. Chaithra G S	Unix Prof.G C Divya		RM Prof.Lavanya D K	PE Prof. Babu		CN Prof.Manjula P	Unix Prof.G C Divya	SE Prof.Chaithra G S
Wednesday	SE Prof.Chaithra G S	PE Prof. Babu		WEB TECHNOLOGY (C2) COMPUTER NETWORKS (C1)			Mini Project Prof. Basavaraj Patil	Env Prof. Harish K S	TOC Dr.Niranjan Murthy
Thursday	CN Prof.Manjula P	SE Prof.Chaithra G S		PE Prof. Babu	Unix Prof.G C Divya		TOC Dr.Niranjan Murthy	WEB TECHNOLOGY (C1) COMPUTER NETWORKS (C2)	
Friday	Mini Project Prof. Basavaraj Patil	Unix Prof.G C Divya		Free	CN Prof.Manjula P		TOC Dr.Niranjan Murthy	Env Prof. Harish K S	RM Prof.Lavanya D K
Saturday	Env Prof. Harish K S	Mini Project Prof. Basavaraj Patil		RM Prof.Lavanya D K	NSS Prof. Chaithra G S				

Figure 4.4: Generated Timetable

© 2026 IJIRCCE | Volume 14, Issue 4, April 2026 DOI: 10.15680/IJIRCCE.2026.1404092

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.771| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

This timetable is carefully designed to provide a balanced and engaging academic routine throughout the week by combining theory classes, practical lab sessions, and project-based learning. Each day includes a varied mix of subjects taught by different instructors, which helps prevent monotony and keeps students actively engaged. The inclusion of both conceptual and hands-on learning ensures that students can better understand and apply what they study, reinforcing their knowledge through multiple approaches.

To support student well-being and sustained focus, the schedule includes well-placed breaks such as mid-morning intervals, lunch, and afternoon rest periods. These breaks help reduce mental fatigue and improve overall productivity. Additionally, certain slots are marked as “Free” or assigned to combined subjects like Web Technology and Computer Networks, offering flexibility for self-study, collaborative work, or clearing doubts. This thoughtful structure not only covers academic requirements but also promotes a more effective and adaptable learning environment.

4.1 Performance Evaluation

Table 4.1 presents the performance metrics of the AutoSchedule AI system evaluated across multiple constraint scenarios.

Metric	Value	Description
Hard Constraint Satisfaction	100%	No teacher or room clashes in all test scenarios
Avg. Generation Time	0.9 sec	Average time for 3-class, 3-teacher, 9-period/day scenario
Subject Period Accuracy	100%	All subjects assigned exact required weekly periods
Lab Block Integrity	100%	All lab sessions scheduled as valid continuous multi-period blocks
Conflict Detection Rate	100%	All injected conflicts detected and flagged by verification engine
Cross-Browser Compatibility	Pass	Tested on Chrome, Firefox, Edge — all rendered correctly
Export Accuracy (JSON/CSV)	100%	Exported data matched displayed timetable in all test cases
Soft Constraint Score	87.4%	Average workload balance and period distribution quality

Table 4.1: AutoSchedule AI System Performance Metrics

The performance of AutoSchedule AI was evaluated based on timetable accuracy, conflict detection, execution time, and usability. The system successfully generated conflict-free timetables by satisfying all hard constraints such as teacher availability, room allocation, and subject period requirements. Displaying timetable generation time was less than one second for medium-sized datasets, showing high computational efficiency. The platform also maintained balanced subject distribution and proper lab scheduling, enhancing overall timetable quality. These results prove that the proposed system is reliable, fast, and suitable for real-time academic scheduling applications.

4.2 Test Case Results

Table 4.2 summarizes the functional test cases executed to validate the system's constraint handling and conflict resolution capabilities.

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.771| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

TID	TC Description	Input	Expected Result	Result	Status
TC1	Basic timetable generation	3 classes, 3 teachers, 9 periods/day, 6 working days	Conflict-free timetable, all required periods scheduled	Generated (0.9s)	Pass
TC2	Unsatisfiable constraints	Required periods > available slots	System detects and reports impossible allocation	Error reported	Pass
TC3	Repeatability	Same input run multiple times	All constraints satisfied each run	Consistent output	Pass
TC4	Teacher clash prevention	T2 assigned to Class A and B same slot	T2 assigned to only one class per period	No clash detected	Pass

Table 4.2: AutoSchedule AI Functional Test Cases

The test case results show that AutoSchedule AI performs accurately under different scheduling conditions. In the basic generation test, the system created a complete conflict-free timetable within a short time. For unsatisfiable constraints, such as more required periods than available slots, the system correctly identified the issue and displayed an error message. Repeated executions with the same input consistently produced valid schedules, proving reliability and stability. The teacher clash prevention test confirmed that no faculty member was assigned to multiple classes in the same time slot, ensuring proper constraint handling.

V. CONCLUSION AND FUTURE SCOPE

AutoSchedule AI provides a comprehensive and intelligent solution for automated academic timetable generation by integrating Genetic Algorithm-based optimization, real-time constraint verification, and an intuitive web-based administrative interface. The system successfully generates conflict-free, constraint-satisfying timetables with 100% hard constraint satisfaction across all evaluated test scenarios, producing results in one second for typical institutional scheduling instances.

The platform's browser-based architecture, requiring no server infrastructure or installation, ensures accessibility from any device with a modern web browser, making it suitable for deployment in institutions ranging from small colleges to large universities. By automating the most time-consuming and error-prone aspects of academic scheduling, AutoSchedule AI significantly reduces administrative workload and enables rapid regeneration of timetables when constraints change.

The system's modular design-separating input configuration, data structuring, GA optimization, constraint verification, and output rendering into distinct components-ensures maintainability and provides clear extension points for future enhancements.

In the future, the system can be significantly enhanced by integrating a machine learning layer that learns from historical scheduling data to improve the initial population quality of the Genetic Algorithm, accelerating convergence to optimal solutions. Integration with institutional databases for automated import of faculty, room, and course data would eliminate manual data entry entirely. Adding support for multi-departmental scheduling, student group preferences, and room capacity constraints would extend the system's applicability to larger and more complex institutions. Development of a mobile-responsive interface and API-based integration with Learning Management Systems (LMS) such as Moodle would further increase the platform's utility in modern educational environments. Additionally, incorporating explainable AI (XAI) techniques to provide administrators with human-readable justifications for scheduling decisions would improve trust and usability. These enhancements would transform

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 8.771| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

AutoSchedule AI into a comprehensive, end-to-end academic planning platform capable of serving the full spectrum of institutional scheduling needs.

REFERENCES

[1] Srinivasan, D., Seow, T. H., and Xu, J. X., "Automated Time Table Generation Using Multiple Context Reasoning for University Modules," IEEE Transactions on Evolutionary Computation, 2025.
[2] Bagul, M. R., Chaudhari, S. C., Nagare, S. N., Patil, P. R., and Kumavat, K. S., "A Novel Approach for Automatic Timetable Generation," International Journal of Advanced Research in Computer Science, vol. 15, no. 2, 2024.
[3] Markal, S., Ghorpade, S., and Chalke, D., "Timetable Generator Using Genetic Algorithm," IRJMETS, April 2023.
[4] Wankhede, S., Sahare, A., Korde, M., Raut, N., and Ladke, U., "Automatic College Timetable Generation," International Journal of Engineering Research and Technology, vol. 11, no. 3, 2022.
[5] Samah, K. A. F. A., Thusree, S. Q., Fadzil, A. F. A., Riza, L. S., Ibrahim, S., and Hasan, N., "A Greedy-based Algorithm in Optimizing Student's Recommended Timetable Generator with Semester Planner," IJACSA, Vol. 13, No. 1, 2022.
[6] Rane, M. V., Apte, V. M., Nerkar, V. N., Edinburgh, M. R., and Rajput, K. Y., "Automated Timetabling System for University Course," IEEE, 2021.
[7] Techie-Menson, H. and Nyagorme, P., "Design and Implementation of a Web-Based Timetable System for Higher Education Institutions," ResearchGate, March 2021.
[8] Abulaziz, A., Caesarendra, W., Haruna, U. S., Sani, A., Sa'id, M., Pamungkas, D. S., Kurniawan, S. R., Kurniawan, E., "Design and Implementation of an Automatic Examination Timetable Generation and Invigilation Scheduling System Using Genetic Algorithm," IEEE, 2024.
[9] Mhaise, G. D., Kurhade, C. S., Devre, M. Y., Sonawane, P., and Parasram, T. B., "Automatic Time Table Generator," IRJMETS, April 2023.
[10] Holland, J. H., "Adaptation in Natural and Artificial Systems," University of Michigan Press, 1975.
[11] Goldberg, D. E., "Genetic Algorithms in Search, Optimization and Machine Learning," Addison-Wesley, 1989.
[12] Burke, E. K. and Petrovic, S., "Recent research directions in automated timetabling," European Journal of Operational Research, vol. 140, no. 2, pp. 266-280, 2002.
[13] de Werra, D., "An introduction to timetabling," European Journal of Operational Research, vol. 19, no. 2, pp. 151-162, 1985.
[14] Schaerf, A., "A survey of automated timetabling," Artificial Intelligence Review, vol. 13, no. 2, pp. 87-127, 1999.
[15] Colorni, A., Dorigo, M., and Maniezzo, V., "A genetic algorithm to solve the timetable problem," Technical Report, Politecnico di Milano, 1992.

INTERNATIONAL STANDARD SERIAL NUMBER INDIA

INNO SPACE SJIF Scientific Journal Impact Factor

doi crossref

निस्केयर NISCAIR

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN COMPUTER & COMMUNICATION ENGINEERING

9940 572 462 6381 907 438 ijircce@gmail.com

www.ijircce.com

Scan to save the contact details