e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:07/Issue:03/March-2025 Impact Factor- 8.187 www.irjmets.com

DYNAMIC TIMETABLE GENERATOR USING GENETIC ALGORITHM

Kunjan Katore[*1], Punam Motewar[*2], Rasika Lahane[*3], Chaitali Nakhate[*4], Dr. Pankaj Bharane[*5]

[*1,2,3,4]Student, Department Of Computer Science And Engineering, Shri Sant Gajanan Maharaj College Of Engineering Shegaon, Maharashtra, India.

[*5]Professor, Department Of Computer Science And Engineering, Shri Sant Gajanan Maharaj College Of Engineering Shegaon, Maharashtra, India.

ABSTRACT

Efficient timetable generation is essential in educational institutions to manage faculty availability, resource allocation, and class scheduling. Traditional manual methods lead to scheduling conflicts, inefficiencies, and resource mismanagement, which become more challenging as institutions grow. This study proposes an automated timetable generation system using genetic algorithms, incorporating one-point crossover and mutation techniques to optimize scheduling iteratively. The system minimizes conflicts, ensures effective faculty and resource utilization, and adapts to constraints like workload and availability. The proposed approach enhances flexibility, adaptability, and scalability, making it suitable for institutions of various sizes while allowing real-time updates.

Keywords: Genetic Algorithm, Resource Scheduling, Timetable Automation, Optimization, Java Servlet.

I. INTRODUCTION

Creating and managing timetables for educational institutions is a complex and challenging task that requires careful planning, organization, and resource allocation. Educational institutions, ranging from schools to universities, need to schedule various courses, assign faculty members, allocate classrooms, and accommodate students efficiently. The process involves satisfying numerous constraints, such as avoiding scheduling conflicts, ensuring faculty availability, and optimizing room utilization [1][2]. The complexity increases with the size of the institution, as more variables and constraints must be considered. Traditional manual methods for timetable generation often fail to handle these intricacies effectively, leading to scheduling conflicts, inefficient use of resources, and increased administrative workload [3]. Thus, there is a growing need for an automated system that can generate timetables efficiently while satisfying all constraints [4].

One of the primary challenges in timetable generation is the satisfaction of multiple constraints. These constraints can be classified into hard constraints, which must be met to ensure feasibility, and soft constraints, which improve the quality of the timetable but are not strictly necessary [8]. Hard constraints include ensuring that no two classes are scheduled in the same room simultaneously, preventing instructors from being assigned to multiple classes at the same time, and adhering to predefined institutional policies [7-9]. Soft constraints, on the other hand, include preferences such as scheduling certain courses in preferred time slots or minimizing the number of consecutive classes for faculty and students [3]. Addressing these constraints manually is not only tedious but also prone to errors, making automation a necessity [8].

The traditional manual approach to timetable generation is not only time-consuming but also highly inefficient, particularly for large institutions with a vast number of courses and faculty members [9]. Administrative staff often spend weeks or even months designing and refining timetables, only to find that certain constraints have been violated, necessitating further revisions [10]. Moreover, last-minute changes, such as faculty unavailability or room reassignments, create additional challenges. A manual system lacks the flexibility to adapt to these changes efficiently, resulting in disruptions and inefficiencies [9]. Consequently, an automated timetable generation system is essential for improving accuracy, reducing errors, and ensuring that institutional resources are utilized optimally [10-12].

This project aims to develop an automated, efficient, and user-friendly timetable generation system tailored to the needs of a college. By leveraging advanced algorithms and computational techniques, the system can process large datasets, analyze constraints, and generate optimal schedules within a short time frame [13]. The system will integrate various constraints to ensure feasibility while optimizing the allocation of faculty, classrooms, and time slots [10]. Additionally, a user-friendly interface will allow administrators to input data, adjust preferences, and make necessary modifications with ease [7]. The goal is to create a seamless, efficient, and adaptable solution that enhances the scheduling process while minimizing administrative workload [5].

In conclusion, the development of an automated timetable generation system is essential for educational institutions seeking to streamline scheduling operations, reduce errors, and optimize resource allocation [4]. By addressing the limitations of traditional manual methods, this system will provide a robust and adaptable solution for handling complex scheduling requirements [1]. As institutions continue to expand and evolve, the need for efficient scheduling solutions will only grow, making this project a crucial step toward modernization and efficiency in academic administration [6].

II. LITERATURE SURVEY

"A Novel Genetic Algorithm-Based Timetable Generator for Optimized University Timetable Solution" by Ali Hasan Khan and Talha Imtiaz (2024), explores the use of a genetic algorithm (GA) for optimizing university timetable generation. The study highlights the efficiency of GA in reducing scheduling conflicts and improving resource utilization. However, it notes a key limitation: reliance on predefined constraints, making it less adaptable to real-time changes. While GA enhances scheduling under static conditions, further improvements are required for better flexibility in dynamic academic environments. [1]

"Automatic Time Table Generation Using Genetic Algorithm" by Mrs. G. Maneesha et.al (2021), focuses on automating timetable generation using genetic and heuristic algorithms. The system efficiently optimizes scheduling by considering subjects, teacher workload, and priorities. Despite reducing time and complexity, the research identifies a limitation in handling last-minute changes or exceptional cases, limiting its flexibility in real-time academic adjustments.[2]

"Design and Implementation of Time Table Generator" by Varsha S et.al (2022), discusses the development of a timetable generator aimed at improving scheduling transparency and reducing administrative workload. The study finds that automation enhances speed and efficiency but struggles with scalability and adaptability in larger institutions, making it less suitable for complex academic environments.[3]

"Automatic Timetable Generation System" by Rajshri Firke et.al (2023), presents a system using the Rapid Application Development (RAD) model to generate flexible timetables. The study highlights significant time savings but points out a limitation in the lack of advanced optimization techniques, restricting its effectiveness in managing complex scheduling conflicts in larger academic institutions.[4]

"Automated Timetable Generation Using Genetic Algorithm" by Shraddha Thakare et.al (2020), introduces a timetable generation system using genetic algorithms. The system effectively reduces scheduling conflicts but relies on basic genetic operators, limiting its efficiency in handling large-scale or complex scheduling problems. The paper suggests that hybrid algorithms or advanced mutation strategies could enhance performance in challenging environments.[5]

"Automated Timetable Generator" by Prof. Er. Shabina Sayed Ansari Ahmed et.al (2015), focuses on evolutionary algorithms to optimize timetable generation while managing hard and soft constraints. The study finds the system efficient in static scenarios but highlights its inability to accommodate real-time changes, making it less effective in dynamic academic settings where last-minute adjustments are common.[6]

"Smart Timetable System Using Machine Learning and Artificial Intelligence" by Suraj Nagtilak et.al (2023), proposes an AI and ML-based system for timetable optimization. The research demonstrates improved scheduling efficiency but notes a limitation in its dependence on historical data for training. This requirement makes the system less suitable for institutions with limited datasets, restricting its broader applicability.[7]

III. RESEARCH METHODOLOGY

3.1 Requirement Analysis

The first step involves gathering requirements from all relevant stakeholders, including administrators, teachers, and students, to identify their specific needs and constraints for the timetable generation system. Meetings, surveys, and interviews are conducted to understand scheduling priorities such as room availability, teacher workloads, and students' academic needs. These requirements are meticulously documented to ensure comprehensive coverage of system functionalities. The constraints, such as avoiding overlapping schedules for teachers and ensuring classrooms meet capacity and resource needs, are analyzed [4]. The process also involves defining how users interact with the system, specifying roles (e.g., admin, teacher, and student), and outlining the core functionalities, such as timetable generation, user authentication, and feedback collection [7].

3.2 Design Phase

3.2.1 Frontend Design: This phase is about designing the user interface so that it is intuitive and user-friendly. JSP, CSS, and JavaScript are used for structuring and styling the frontend. For dynamic content rendering, JSP (Java Server Pages) is used so that the users can interact with the system without any problem [6].

3.2.2 Backend Design: The backend is designed for efficiency and scalability. JSP, CSS, and JavaScript are used to structure and style the frontend of the system, while JSP allows dynamic content rendering for users [1]. The backend ensures proper data processing, interaction with the database, and enforcement of scheduling constraints.

3.2.3 Database Design: A database schema as an ER diagram is developed. The database schema contains tables for users, roles, courses, schedules, and feedback. Relations of entities are defined, such as relations between students and classes they attend or between teachers and their schedules [6]. Constraints on the uniqueness of course identifiers and the availability of time slots to prevent scheduling clashes are set [1][7].

3.3 Implementation

3.3.1 Frontend Development: Static web pages are developed using HTML and CSS to structure the interface aesthetically. JavaScript adds interactivity, such as validation of user inputs or dynamic schedule previews. JSP enables dynamic content rendering, where pages display real-time data fetched from the backend, including available courses, teacher schedules, and generated timetables [8].

3.3.2 Backend Development: All business logic is coded in Java, including user authentication/authorization features to ensure proper access for an administrator, teacher, or student. RESTful API endpoints or server-side logic process timetable constraints, run the genetic algorithm, and send responses to the frontend [11].

3.3.3 Database Development: A MySQL database is created based on the schema developed during planning. Data access layers are developed to ensure smooth interactions between the backend and the database without bottlenecks. CRUD (Create, Read, Update, Delete) operations manage data for users, courses, schedules, and feedback [6].

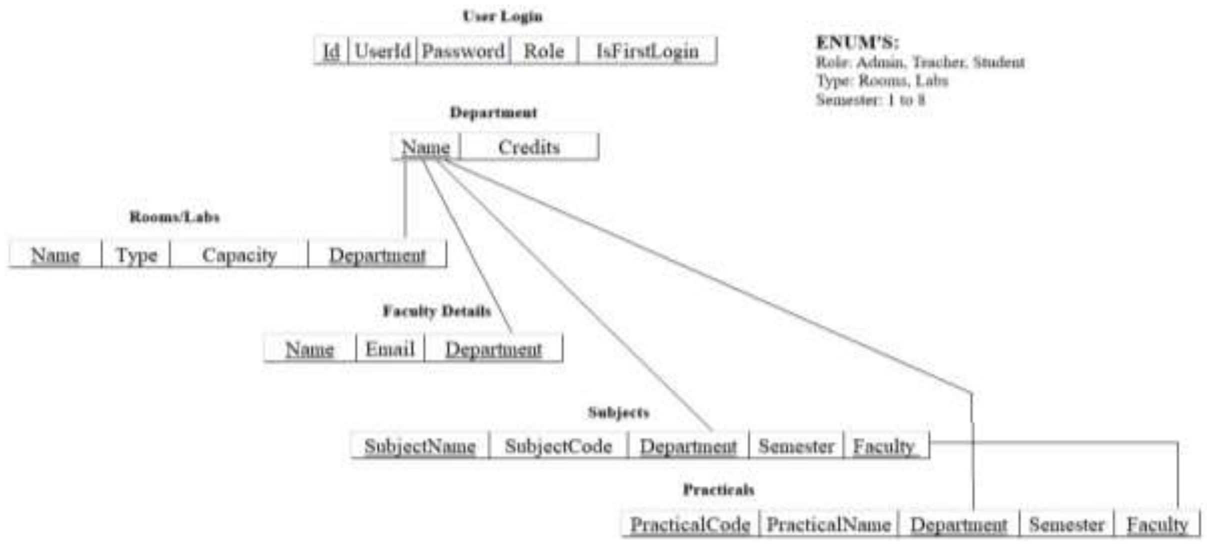

User Login
Id | UserId | Password | Role | IsFirstLogin
ENUM'S:
Role: Admin, Teacher, Student
Type: Rooms, Labs
Semester: 1 to 8
Department
Name | Credits
Rooms/Labs
Name | Type | Capacity | Department
Faculty Details
Name | Email | Department
Subjects
SubjectName | SubjectCode | Department | Semester | Faculty
Practicals
PracticalCode | PracticalName | Department | Semester | Faculty

Figure 3.1: Database Connection Flow

3.3.4 Timetable Generation: The genetic algorithm is implemented in Java for generating optimized timetables. Constraints, including room availability and teacher schedules, are processed by the algorithm to make it feasible. The algorithm evolves over generations with improvements in solutions by applying genetic operators like selection, crossover, and mutation [13]. The integration of this algorithm with the backend ensures the automated creation of conflict-free and optimized timetables [4].

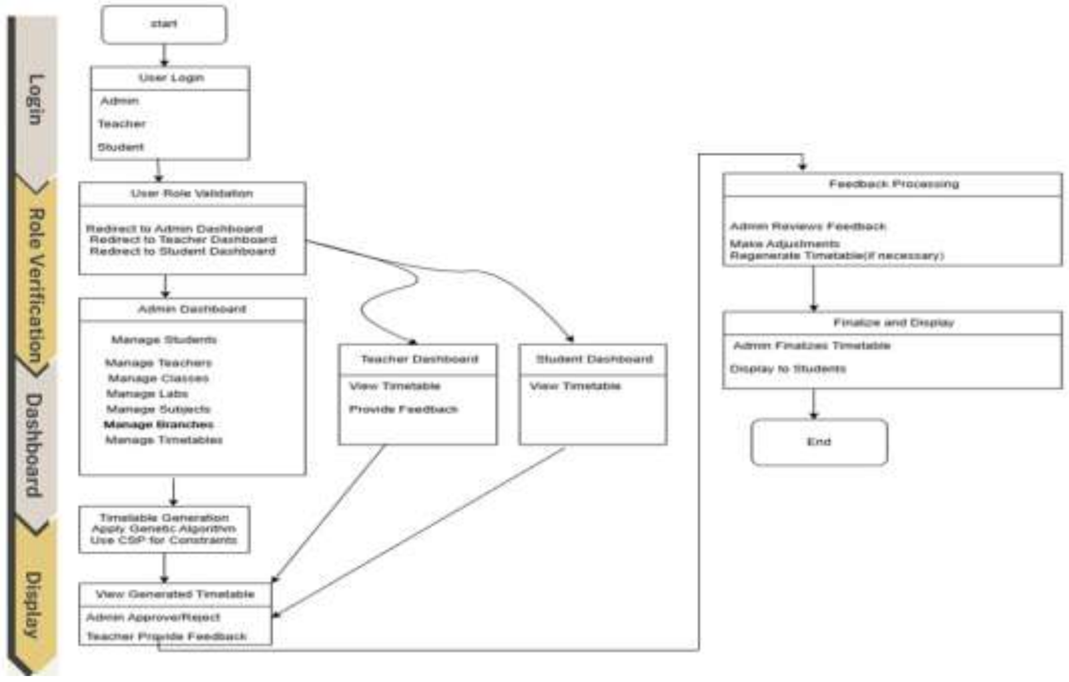


Figure 3.2: Flow of the Website

3.4 Testing

3.4.1 Unit Testing: Unit testing is conducted to identify bugs early in the backend logic, database queries, and frontend modules. It ensures that all individual functionalities, like generating schedules or authenticating users, work correctly [13].

3.4.2 Integration Testing: Testing the interaction between the frontend, backend, and database ensures smooth communication and proper data flow within the system. The process verifies that data inputted by users on the frontend is correctly validated, processed by the backend, and accurately stored or retrieved from the database [1]. This ensures data integrity, seamless functionality, and an optimal user experience.

3.4.3 System Testing: The complete system is tested to evaluate its functionality, performance, and usability. Test cases simulate real-world scenarios, such as generating timetables for different departments or handling simultaneous user interactions [5].

3.4.4 User Acceptance Testing (UAT): The system is tested using a group of end-users, including administrators, teachers, and students, to evaluate its ability to meet their expectations. The gathered feedback plays a crucial role in identifying usability issues and additional requirements that need to be addressed before deployment [3].

3.5 Deployment

3.5.1 Prepare Deployment Environment: The production server receives all necessary configurations, such as Java, MySQL, and a web server like Apache Tomcat. Security measures such as SSL certificates ensure that all user data are protected [6].

3.5.2 Application Deployment: The frontend, backend, and database components are deployed in the production environment, ensuring seamless integration and functionality. The application is made accessible to users through a secure URL, providing a safe and reliable access point [2]. Additionally, the server configuration is optimized based on traffic patterns to enhance performance, scalability, and responsiveness.

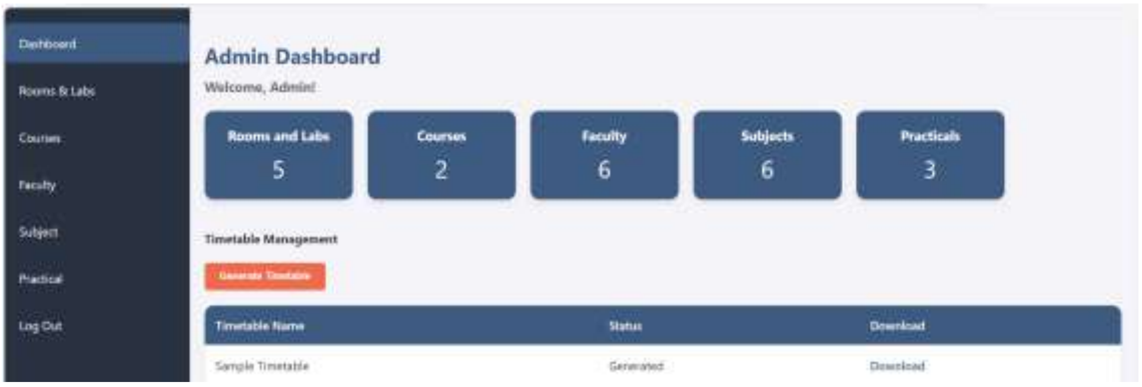
3.5.3 Final Testing: Prior to the production environment, a final round of testing is conducted to ensure all components work seamlessly. Performance tests assess the system's ability to withstand heavy loads, and security tests verify the protection of sensitive user data [6][5].

IV. RESULT AND DISCUSSION

1. Developed a dynamic web application using Java Servlets to automate the scheduling of classes, faculty, and rooms, streamlining the entire process and reducing manual efforts.

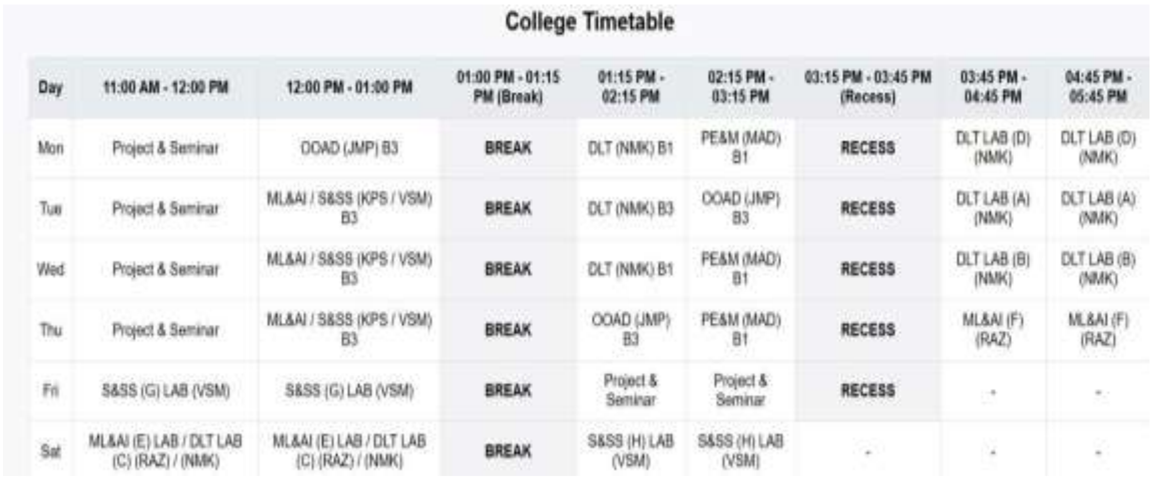
2. Implemented MySQL as the backend database to efficiently store and manage timetable data, ensuring real-time updates, reliable retrieval, and scalability for administrators, faculty, and students.
3. Integrated a secure login system with role-based access control, allowing administrators, faculty, and students to access and manage timetable information based on their specific privileges.
4. Designed an intelligent scheduling algorithm that accounts for constraints like faculty availability, room allocation, and course timings, ensuring conflict-free, optimized, and well-structured timetables.
5. Successfully implemented a genetic algorithm for timetable generation, incorporating selection, crossover, and mutation strategies to optimize resource allocation, resolve conflicts, and efficiently produce schedules meeting specified criteria.

The result of this research work is the timetable generated for the college/ university.



Timetable Name	Status	Download
Sample Timetable	Generated	Download

Figure 4.1: Admin Dashboard



College Timetable

Day	11:00 AM - 12:00 PM	12:00 PM - 01:00 PM	01:00 PM - 01:15 PM (Break)	01:15 PM - 02:15 PM	02:15 PM - 03:15 PM	03:15 PM - 03:45 PM (Recess)	03:45 PM - 04:45 PM	04:45 PM - 05:45 PM
Mon	Project & Seminar	OOAD (JMP) B3	BREAK	DLT (NMK) B1	PE&M (MAD) B1	RECESS	DLT LAB (D) (NMK)	DLT LAB (D) (NMK)
Tue	Project & Seminar	ML&AI / S&SS (KPS / VSM) B3	BREAK	DLT (NMK) B3	OOAD (JMP) B3	RECESS	DLT LAB (A) (NMK)	DLT LAB (A) (NMK)
Wed	Project & Seminar	ML&AI / S&SS (KPS / VSM) B3	BREAK	DLT (NMK) B1	PE&M (MAD) B1	RECESS	DLT LAB (B) (NMK)	DLT LAB (B) (NMK)
Thu	Project & Seminar	ML&AI / S&SS (KPS / VSM) B3	BREAK	OOAD (JMP) B3	PE&M (MAD) B1	RECESS	ML&AI (F) (RAZ)	ML&AI (F) (RAZ)
Fri	S&SS (G) LAB (VSM)	S&SS (G) LAB (VSM)	BREAK	Project & Seminar	Project & Seminar	RECESS	-	-
Sat	ML&AI (E) LAB / DLT LAB (C) (RAZ) / (NMK)	ML&AI (E) LAB / DLT LAB (C) (RAZ) / (NMK)	BREAK	S&SS (H) LAB (VSM)	S&SS (H) LAB (VSM)	-	-	-

Figure 4.2: College Timetable

V. FUTURE SCOPE

The future scope of this research in automated timetable generation using genetic algorithms and related optimization techniques is vast and promising. As educational institutions increasingly adopt technology for administrative tasks, there is a significant opportunity to refine and expand upon the current model. Future research could focus on integrating machine learning algorithms with genetic algorithms to create more adaptive and intelligent systems capable of learning from historical scheduling data. This hybrid approach would make the algorithm more responsive to dynamic and unforeseen changes, such as faculty absences or last-minute adjustments to course requirements. Furthermore, real-time feedback from stakeholders, including students, faculty, and administrators, could further enhance the responsiveness of the system. There is also potential to expand the scope of the system to handle more complex constraints, such as multi-campus scheduling, integration with external systems (e.g., student enrollment systems), and the ability to predict and resolve potential conflicts before they arise. Furthermore, the application of cloud computing and distributed systems could provide scalability, allowing the system to handle larger institutions or even be implemented at a global scale. This research would enhance flexibility, adaptability, and real-time processing capabilities to make timetable management in educational institutions more efficient, user-friendly, and robust.

VI. CONCLUSION

The automated timetable generation system based on genetic algorithms effectively tackles the challenges posed by the conventional manual scheduling system in academic institutions. Incorporating one-point crossover and mutation, the system improves scheduling progressively to minimize conflicts, optimize faculty and resource allocation, and responsiveness to institutional restrictions. The flexible nature of the approach enables it to be updatable in real time, thereby rendering it an easily scalable solution applicable to academic institutions of diverse sizes. Through the utilization of AI-powered optimization, this system increases scheduling effectiveness, decreases administrative burden, and enhances overall resource management, ultimately serving students, faculty, and administrators.

VII. REFERENCES

[1] A. H. Khan and T. Imtiaz, "A Novel Genetic Algorithm-Based Timetable Generator for Optimized University Timetable Solution," 2024 International Conference on Engineering & Computing Technologies (ICECT), pp. 1-6, 2024. DOI: 10.1109/ICECT61618.2024.10581296.

[2] G. Maneesha, T. Deepika, S. BhanuSri, N. RaviKumar, and P. Siva Nagamani, "Automatic Time Table Generation Using Genetic Algorithm," Journal of Emerging Technologies and Innovative Research (JETIR), vol. 8, no. 7, pp. 1-7, July 2021.

[3] V. S, D. S, B. G. L. S, S. V, V. S, and B. Gowda, "Design and Implementation of Time Table Generator," Journal of Emerging Technologies and Innovative Research (JETIR), vol. 9, no. 6, pp. 1-5, June 2022.

[4] R. Firke, P. Bhabad, O. Gangarde, A. Magar, and A. Tawlare, "Automatic Timetable Generation System," International Journal of Creative Research Thoughts (IJCRT), vol. 11, no. 10, pp. 1-6, Oct. 2023.

[5] S. Thakare, T. Nikam, and M. Patil, "Automated Timetable Generation Using Genetic Algorithm," International Journal of Engineering Research & Technology (IJERT), vol. 9, no. 7, pp. 1-5, July 2020.

[6] S. S. A. Ahmed, A. A. Ansari, and A. Z. Ansari, "Automated Timetable Generator," International Journal for Innovative Research in Science & Technology (IJIRST), vol. 1, no. 11, pp. 1-5, Apr. 2015.

[7] S. Nagtilak, N. Dongare, A. Salave, P. Dongare, G. Yeole, and P. Satarkar, "Smart Timetable System Using Machine Learning and Artificial Intelligence," International Journal of Advanced Research in Science, Communication and Technology (IJARSCT), vol. 3, no. 5, pp. 1-6, June 2023.

[8] O. K. Majeed, Z. ul Abideen, U. Arshad, R. H. Ali, A. Habib, andR. Mustafa, "Adaptivecloset: Reinforcement learning in personalized clothing recommendations," in 2023 18th International Conference on Emerging Technologies (ICET). IEEE, 2023, pp. 305–309.

[9] H. Ashfaq, U. Arshad, R. H. Ali, Z. ul Abideen, M. H. Shah, T. A. Khan, A. Z. Ijaz, N. Ali, and A. B. Siddique, "Mitigating crop losses: Ai-enabled disease detection in tomato plants," in 2023 International Conference on Frontiers of Information Technology (FIT). IEEE, 2023,pp. 190–195.

[10] M. H. Shah, M. A. Bakar, R. H. Ali, Z. U. Abideen, U. Arshad, A. Z. Ijaz, N. Ali, M. Imad, and S. Nabi, "Investigating novel machine learning based intrusion detection models for nsl-kdd data sets," in 2023 International Conference on IT and Industrial Technologies (ICIT). IEEE, 2023, pp. 1–6.

[11] A. Mashhood, Z. ul Abideen, U. Arshad, R. H. Ali, A. A. Khan, and B. Khan, "Innovative poverty estimation through machine learn- ing approaches," in 2023 18th International Conference on Emerging Technologies (ICET). IEEE, 2023, pp. 154–158.

[12] U. Arshad, H. H. Ellahie, Z. ul Abideen, and R. H. Ali, "An enhanced genetic algorithm framework for efficient solutions to capacitated vehicle routing problems," in 2023 18th International Conference on Emerging Technologies (ICET). IEEE, 2023, pp. 141–146.

[13] M. Iftikhar, R. H. Ali, M. Saleem, N. Shahzadi, U. Arshad, T. A. Khan, A. Z. Ijaz, N. Ali, and M. Imad, "Optimizing airline networks: A comparative analysis of graph-based techniques," in 2023 International Conference on IT and Industrial Technologies (ICIT). IEEE, 2023, pp. 1–6.