Automated Time Table Generator

Prof. S. R. Karjol[1], Mr. Ramesh Karjol[2], Miss. Phani Srushti[3], Mr. Shreyas Bewoor[4], Mr. Sunil Biradar[5], Mr. Raghavendra Hosagoudar[6]

[1]*Assistant Professor, Department of Computer Science and Engineering, B.V.V. Sangha's Basaveshwar Engineering College, Bagalkot*

[2]*Lecturer, Department of Mechanical, B.V.V. Sangha's Polytechnic (Autonomous), Bagalkot.*

[3,4,5,6]*Students, Department of Computer Science and Engineering, B.V.V. Sangha's Basaveshwar Engineering College, Bagalkot*

***Abstract*—This research explores the structured schedule that organizes subjects, faculty, rooms, and time slots for smooth academic functioning. Traditional manual timetable preparation is simple but slow, error-prone, and unable to handle conflicts such as overlapping classes, faculty shortages, or room limitations. It also becomes inefficient as institutions grow or new constraints arise. To overcome these limitations, the Automatic Timetable Generator System uses algorithms and automation to create optimized, clash-free schedules. It ensures proper allocation of subjects, faculty, classrooms, and time slots while allowing quick modifications. By reducing manual effort, preventing conflicts, and improving flexibility, the system offers a reliable, efficient, and scalable solution for modern educational environments.**

I. INTRODUCTION

An Automated Timetable Generator is a software system used to automatically generate and manage academic timetables for schools, colleges, and universities. Manual timetable preparation is time-consuming, difficult, and prone to errors like clashes and uneven workload distribution. This automated system uses algorithms to assign teachers, subjects, classrooms, and time slots without conflicts. It saves time, reduces manual work, improves accuracy, and makes scheduling flexible and efficient. The system can easily handle changes and ensures proper utilization ofacademicresources through optimized scheduling.

Purpose of the Project

a) To reduce the time, effort, and errors involved in manual timetable creation.
b) To automatically generate conflict-free, accurate, and optimized timetables.
c) To ensure fair and efficient allocation of teachers, subjects, rooms, and time slots.
d) To improve resource utilization and simplify academic management in institutions.
e) To provide a fast, flexible, and user-friendly solution that can adapt to timetable changes anytime.

II. AIM AND OBJECTIVES

The aim of the Smart Timetable Generator is to automate the process of creating class schedules.

To fulfill the aim of the project, the following are the objectives:

1. Automatic Generation: Generates a conflict-free timetable based on inputs.
2. Real-Time Modifications: Allows administrators to update schedules dynamically.
3. Conflict Detection: Highlights clash in faculty allocation, room usage, or class schedules.Optimized Allocation: Assigns resources efficiently to maximize utilization.
4. User-Friendly Interface: Easy for administrators and faculty to access and modify schedules.

III. LITERATUREOVERVIEW

Automated timetable generation has been an active area of research due to the increasing complexity of academic scheduling and the limitations of manual methods. Various approaches, including constraint-based modeling, heuristic techniques, and evolutionary algorithms, have been proposed to address scheduling conflicts and improve efficiency.

Dr. K. Santhi Sree *et al.* [1] presented a web-based automated timetable generator designed for educational institutions. The system employs constraint-based algorithms combined with graph-based conflict detection, heuristic optimization, and backtracking techniques to generate accurate and conflict-free timetables. The study demonstrates a significant reduction in timetable generation time—from several hours to a few minutes—while ensuring scalability and adaptability to real-world academic constraints such as faculty availability, room allocation, and subject distribution. The proposed approach proves effective for dynamic institutional environments.

In [2], Kudale *et al.* discussed the transition from conceptual design to practical implementation of timetable generation systems. The authors emphasized future enhancements such as predictive analytics, real-time updates, and improved UI/UX design to increase system usability. The study also highlighted the potential integration of timetable generators with Learning Management Systems (LMS) and Enterprise Resource Planning (ERP) platforms, enabling seamless synchronization of schedules, examinations, assignments, and attendance. Such integration supports holistic academic planning and efficient communication among students, faculty, and administrators, while also extending applicability to corporate and event scheduling domains.

Singhal *et al.* [3] proposed an automated timetabling solution based on a Genetic Algorithm (GA). The system effectively optimizes scheduling by minimizing conflicts and balancing workloads across faculty and student groups. Experimental results indicate a substantial reduction in manual effort and scheduling time, while improving accuracy and adaptability. The authors conclude that GA-based solutions provide a scalable and practical alternative to traditional scheduling methods for large institutions and organizations.

Firke *et al.* [4] developed a web-based automatic timetable generation system that considers both hard constraints (such as room availability and faculty clashes) and soft constraints (such as workload balance and preferences). The study focuses on flexibility and user-centric design, with future plans to enhance customization, scalability, and global adaptability. The proposed system aims to deliver a dynamic and efficient scheduling platform suitable for colleges and universities.

Utkarsh Kumar *et al.* [5] explored an automatic timetable generator using heuristic techniques. While the paper acknowledges advanced approaches such as Genetic Algorithms, Evolutionary Algorithms, and Constraint Programming, it emphasizes heuristic methods due to their faster execution and adaptability. The system generates conflict-free schedules by efficiently considering subjects, teachers, student groups, and available resources, making it suitable for institutions seeking rapid and automated scheduling solutions.

From the surveyed literature, it is evident that automated timetable generation systems significantly improve scheduling efficiency, accuracy, and scalability. However, challenges remain in achieving real-time adaptability, seamless system integration, and optimal handling of complex constraints. These gaps motivate the development of enhanced, intelligent, and fully integrated timetable generation systems.

IV. SOFTWARE REQUIREMENT SPECIFICATION

Scope of the System

The software provides automated scheduling for colleges, universities, and schools.

It handles:

a) Allocation of subjects, staff, classrooms, and time slots
b) Clash detection (teacher clash, room clash, subject overlap)
c) Modifications and regeneration of timetable
d) Storage and viewing of timetables
e) The system improves accuracy, saves time, and simplifies academic planning.

1. User Requirements

These requirements describe what the user expects the system to do, based on the challenges mentioned in your problem statement.

a) Easy Data Input:

Users must be able to enter faculty details, subject

names, department/semester data, classroom details, and time slots easily.

b) Generate Timetable Automatically:

Users should be able to generate a full timetable with a single click, without manually checking for conflicts.

c) Clash-Free Output:

The user expects that the system will avoid teacher overlaps, room conflicts, and subject clashes automatically.

d) Modify Timetable Easily:

Users should have the option to edit or regenerate the timetable when faculty availability or course requirements change.

e) View & Save Timetable:

The user should be able to view, download, or print the final timetable.

f) Simple and User-Friendly Interface:

Users should not need technical expertise to operate the system.

2. Software Requirements

- Operating System: Windows / Linux
- Backend Technology: Java / PHP (any chosen by you)
- Frontend: HTML, CSS, JavaScript (or GUI framework)
- Database: MySQL / SQLite
- Development Tools: VS Code.

3. Functional Requirements

These are the features the system must perform, directly addressing the problems stated.

a) Faculty Management Module:

- Add, update, delete faculty details.
- Define availability and working hours.

b) Subject & Class Management Module

- Enter subjects with weekly hours.
- Assign subjects to faculty oDefine student groups or semesters.

c) Classroom Allocation Module:

- Maintain list of available rooms.
- Ensure no two classes are assigned the same room at the same time.

d) Automatic Timetable Generation

- Algorithm generates conflict-free schedule.
- Ensures correct distribution of subject hours.

e) Clash Detection System

- Teacher not assigned to two classes at same time.
- No room used by two classes simultaneously.
- No class gets overlapping subjects.

f) Editing & Regeneration

- User can modify constraints.
- System regenerates updated timetable.

4. Non-Functional Requirements.

a) Performance Requirement

System should generate the timetable within a few seconds.

b) Usability Requirement

Interface must be simple, clean, and easy for administrative staff.

c) Reliability Requirement

Timetable must always be conflict-free and accurate.

d) Scalability Requirement

Should support multiple departments and semesters as the institution grows.

e) Security Requirement

Only authorized users should be able to edit or modify timetable data.

5. Domain Requirements

Domain-specific rules from the academic environment that the system must follow:

- A teacher cannot handle two classes at the same time.
- A classroom cannot be double-booked.
- Each subject must be scheduled according to required weekly hours.
- Faculty availability must be strictly followed.
- Timetable must follow institutional rules (breaks, working hours, slot durations).

V. METHODOLOGY

The methodology adopted for developing the Automated Time Table Generator is structured to address the issues highlighted in the problem statement—manual errors, scheduling conflicts, time consumption, and difficulty in making real-time updates. The development process follows a systematic and step-by-step approach to ensure that the final system is accurate, efficient, and aligned with institutional requirements.

1. Problem Analysis
The first step involves analyzing the existing manual timetable process and understanding its challenges such as overlapping classes, teacher unavailability, unequal workload distribution, and complexity in resource allocation. This analysis helps in identifying the exact featuresand constraints the automatedsystem must handle.

2. Requirement Gathering
Data such as faculty details, subjects, classroom availability, department structure, and time slots are collected. This step ensures that the system is designed based on real institutional constraints mentioned in the problem statement—faculty availability, subject hours, classroom limitations, and scheduling rules.

3. System Design & Modeling
In this phase, the structure of the system is created using UML diagrams, DFDs, and ER models. The system architecture is designed to handle conflict detection, data processing, and automatic scheduling. This design ensures resource optimization and real-time modification capability.

4. Algorithm Development
A scheduling algorithm (such as constraint-based algorithm or rule-based approach) is developed to generate conflict-free timetables.
The algorithm checks:
a) Teacher availability.
b) Room capacity and availability.
c) Subject hour requirements.
d) Avoiding clashes and overlaps.
e) Thisstep directly solves the problems highlighted in the manual method.

- Database Design

A well-structured database is created to store user name, password, email. Proper relational mapping ensures that the system retrieves and updates data efficiently, supporting scalability and easy modifications. Implementation
Based on the design, the front-end interface and back-end logic are developed. The interface allows users to input data easily, while the back-end automatically generates the timetable. All modules—faculty management, subject allocation, room scheduling, and timetable generation—are implemented.

- Testing & Error Checking

The system is tested to ensure:
a) No two classes overlap.
b) No teacher is double booked.
c) Rooms are allocated correctly.
d) Timetable meets institutional constraints.
e) This step ensures high accuracy and reliability of the automated system.

VI. PROPOSED MODEL

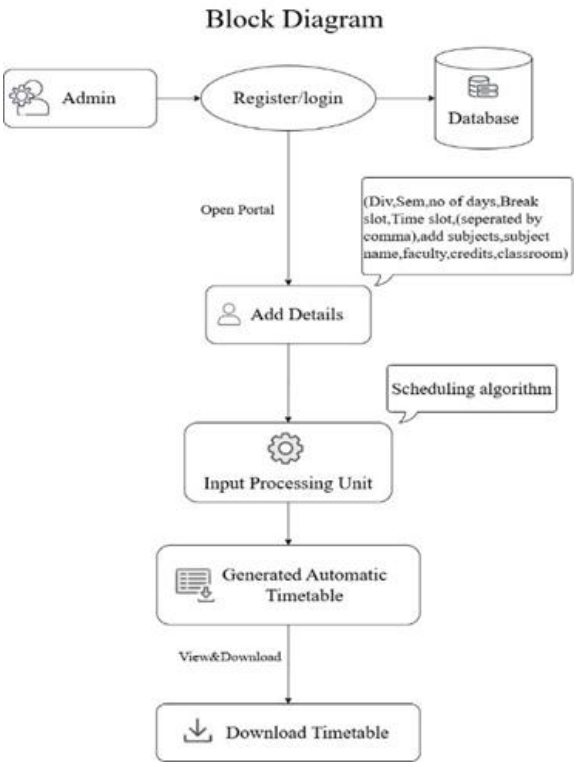

Fig.1

A. Data Flow Design
The Data Flow Design describes how information moves through the Automated Time Table Generator system. It explains the flow of data between the administrator, faculty, input module, scheduling unit, database, and final output. The design is based on the block diagram and follows the logical steps needed to generate a conflict-free timetable.

1) Admin Login & Portal Access
The process begins when the admin logs into the

portal. After authentication, the system grants access to the interface where the admin can manage academic data such as faculty information, subjects, classrooms, and time slots.

2) User Input Module

Once the portal is open, the admin (or authorized user) enters required scheduling details into the User Input moduleThis includes:

- Faculty availability.
- Subjects and weekly hours.
- Classroom capacity and availability.
- Semester/department details.
- This raw data forms the basis for timetable generation.

3) Input Processing Unit

The entered data is sent to the Input Processing Unit, where it is validated and converted into a structured format.

The system checks:

- Missing or incomplete data.
- Incorrect subject-hour mapping.
- Invalid faculty availability.
- After validation, the clean data is passed forward for scheduling.

4) Database Storage

All validated data is stored in the **Database** for future access.

The database holds:

- User name
- Password
- Email
- This ensures data consistency and supports real-time modifications.

5) Scheduling Algorithm

Based on the logic in our code timetable generator, the scheduling algorithm used is a

"Greedy Scheduling Algorithm (Diagonal Greedy Allocation) with Faculty-Conflict Avoidance Algorithm."

It ensures:

- Start from a random diagonal position.
- Try to place each hour (credit) one by one.
- If the slot is free + faculty not busy → allocate.
- Else → move to the next slot.
- Continue until all credits are placed.

6) Timetable Generation

The output of the scheduling algorithm is a Generated Automatic Timetable.

This timetable is created in a structured format, showing:

- Class timings.
- Faculty assignments.
- Room allocations.
- Subject distribution.

7) View & Download Timetable

Finally, the generated timetable is sent to the Download Timetable module.

Admin can:

- View the timetable
- Download it as PDF/Excel
- Make changes and regenerate if needed
- This completes the end-to-end data flow of the system.

B.User Interface Prototypes

The User Interface (UI) of the Automated Time Table Generator is designed to be simple, intuitive, and easy to navigate for administrators and faculty members. The purpose of the UI prototypes is to depict how users interact with the system visually and functionally. These prototypes help in understanding system flow, user actions, and screen organization before implementing the actual interface.

The following are the key UI screens used in the system:

1) Login Screen

This screen allows authorized users (Admin/Staff) to log into the system.

The prototype includes:

- Username/Email field
- Password field
- Login Button

Fig.2

The login page ensures that only authorized users can access and modify timetable-related data.

2) Dashboard / Home Screen

After logging in, the user is directed to the dashboard where they can navigate to different modules.

The dashboard typically displays:

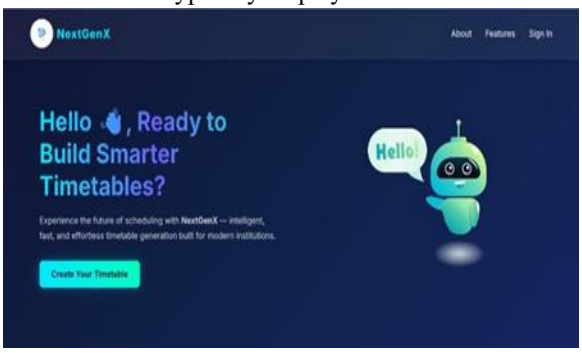

Fig.3

This screen serves as the control center of the system.

3) Faculty Input Form

This interface is used to enter:

- Add Faculty
- Add Subjects
- Semester
- Division
- Add Classrooms
- Generate Timetable
- View / Download Timetable
- Faculty name
- Available time slots
- Subjects handled
- Credits
- Classrooms

Fig.4

It ensures that faculty availability and teaching load are accurately captured, which is crucial for conflict-free scheduling.

4) Timetable Generation Screen

Once all data is entered, the user navigates to this screen to automatically generate the timetable.

The prototype contains:

- Button to generate timetable
- Progress indicator (optional)
- View results section
- The system runs the scheduling algorithm here and produces the final output.

5) Timetable Display Screen

After generation, the timetable is shown in a clear table format.

It displays:

- Days of the week
- Time slots
- Subjects
- Assigned faculty
- Classroom names

Fig.5

Users can visually verify the schedule for errors or conflicts.

6) Download Timetable Screen

This interface provides options to:

- Download timetable as PDF/Excel.
- Print timetable.
- Regenerate timetable if changes are needed.

Fig.6

It offers flexibility in sharing and storing the timetable.

C. Purpose of UI Prototypes

- Helps visualize system functionality before implementation.
- Ensures easy navigation and user-friendly interaction.
- Reduces design errors and improves clarity
- Assists developers and users in understanding expected behavior.

D. System Development / Implementation

The System Development and Implementation phase focuses on converting the system design into a fully functional Automated Time Table Generator. This stage involves developing each module, integrating the scheduling algorithm, and ensuring the system operates as intended. The implementation process follows the problem requirements of reducing manual effort, avoiding scheduling conflicts, and providing real-time timetable generation.

E. Module Development

1) Admin Login Module

This module verifies user credentials and ensures that only authorized users can access the system.

Password hashing and validation techniques are used for secure login.

2) Data Input Module

This module allows administrators to enter essential academic data such as:

- Faculty details and availability.
- Subjects and weekly hours.
- Classroom/lab availability.
- Department and semester details.
- All inputs are validated to prevent incorrect or incomplete data entry.

3) Input Processing Module

Once the data is entered, the system processes it by:

- Checking conflicts in availability.
- Structuring data in a usable format.
- Storing validated information into the database.
- This ensures clean and organized data for the scheduling algorithm.

4) Database Integration

A relational database (MySQL/SQLite) is implemented to store:

- Faculty information.
- Subject details.
- Classroom data.
- Timetable records.
- Foreign keys and relations are used to maintain integrity and support efficient data retrieval.

5) Scheduling Algorithm Implementation

This is the core functional component of the system.

The algorithm is designed to generate a conflict-free timetable based on constraints like:

- No teacher double-booking.
- No classroom overlap.
- Satisfying weekly hour requirements.
- Faculty availability slots.
- Separate slots for lab/theory classes.
- The algorithm assigns subjects, teachers, and rooms to suitable time slots while ensuring fairness and optimization.

6) Timetable Generation Module

After the scheduling algorithm runs, the system automatically produces the final timetable.

This module formats the output into clear table structures showing:

- Days
- Time slots
- Subjects
- Faculty
- Rooms

- The timetable is generated in real time and can be verified or edited.

7) View & Download Module This module allows users to:
- View the generated timetable on the screen
- Download it in PDF/Excel format
- Print the timetable
- Regenerate the schedule if changes occur
- It enhances usability and supports easy sharing of the final output.

8. System Integration

After developing each module individually, they are integrated to form a complete workflow:
- User logs in
- Inputs academic data
- Data gets processed and stored
- Scheduling algorithm runs
- Timetable is generated
- User views/downloads the output
- The system is tested after integration to ensure smooth communication between components.

F. Testing & Verification

During implementation, the system undergoes multiple types of testing:
- Unit Testing: Testing individual modules like data validation, constraint checking, or slot allocation to ensure each part works correctly.
- Integration Testing: Testing how combined modules—such as faculty allocation, subject mapping, and timetable generation—work together without conflicts.
- Validation Testing: Ensuring the generated timetable meets all institutional rules, constraints, and requirements set by the college.
- User Testing: Allowing administrators or faculty to use the system to verify that the timetable is correct, easy to use, and meets real-world needs.

G. Deployment

Once testing is complete, the system is deployed for use in the institution.

Deployment includes:
- Setting up database
- Providing access credentials
- Training staff in using the system

H. Maintenance

Post-deployment, the system may require updates such as:
- Adding new departments or subjects
- Modifying teacher availability
- Updating classroom details
- Regular maintenance ensures long-term performance and adaptability.

VII. CONCLUSION

The Automated Timetable Generator provides a smart, efficient, and reliable solution for managing academic schedules. By eliminating manual errors, saving time, and ensuring optimal allocation of resources, it improves the overall productivity of institutions. Its automation, flexibility, and ease of use make it a valuable tool for modern educational environments. With further enhancements and integration options, the system has strong potential to become a complete scheduling platform that supports smooth academic operations.

The automated approach also supports future scalability, allowing institutions to accommodate new courses, departments, and resources with minimal effort. Ultimately, the Automatic Timetable Generator System contributes to better academic planning, improved productivity, and a more organized learning environment for both students and staff.

REFERENCES

[1] K. Santhi Sree, “Automated Timetable Generator System,” International Journal of Engineering Technology Research & Management, vol. 6, June 2025.

[2] V. Kudale et al., “From Concept to Code: Implementing Timetable Generation,” Apr. 2025.

[3] S. Singhal, D. Agarwal, and Y. Bhardwaj, “Revolutionizing Scheduling: A Comprehensive Analysis of Automated Timetabling Solution,” May–June 2024.

[4] R. Firke et al., “Automatic Timetable Generation System,” Oct. 2023.

[5] U. Kumar et al., “Automatic Time Table Generator Using Heuristic Technique,” May

2022.

[6] Dr. Smitha Padshetty, Abdul Jaleel Khan Anas, Abdul Rahman Syed, and Harsh Kulkarni, "Automated Time-Table Management System," Journal of Scientific Research and Technology, vol. 3, no. 12, pp. 363–372, Dec. 2025.

[7] Prajakta Tanksali, Ila Dhond, Shivani Pednekar, VarshandaSingbal, and Shruti Sivaraman, "Automated Timetable Generation," International Journal of Scientific Research in Computer Science, Engineering and Information Technology (IJSRCSEIT), vol. 6, no. 3, pp. 389–396, May–June 2020.

[8] S. Meera, Nithish Kumar, and Sarath Kumar, "Machine Learning-Based Automatic Timetable Generation," Journal of Soft Computing and Computational Intelligence, vol. 1, no. 2, May–Aug. 2024.

[9] P. Uma, P. S. Sharvesh, M. Pradeep, P. Sathishkumar, and R. Senthilnathan, "Automatic Timetable Generation," International Journal for Research in Applied Science & Engineering Technology (IJRASET), Oct. 2023.

[10] Fahan N. A., "Automatic Timetable Generator," International Journal of Scientific Research in Engineering and Management (IJSREM), vol. 9, no. 5, pp. 1–9, May 2025.