Rizvi College of Engineering

Department of Computer Engineering

Project Synopsis Report

on

AI powered timetable scheduler and management

Submitted in partial fulfilment of the requirements

of the degree of

Bachelor of Engineering

by

Aimaan Khan (211P039)

Mariyum Siddique (211P023)

Siddharth Pallar (211P016)

Guide:

Prof. Shiburaj Pappu

University of Mumbai

2024 - 2025

Certificate

This is to certify that the project synopsis entitled “AI powered timetable scheduler and management” has been submitted by Aimaan Khan, Mariyum Siddique and Siddharth Pallar, under the guidance of Prof.Shiburaj Pappu in partial fulfillment of the requirement for the award of the Degree of Bachelor of Engineering in Computer Engineering from Rizvi College of Engineering, University of Mumbai.

Certified By

Prof. Shiburaj Pappu	Dr. Anupam Choudhary
Project Guide	Head of Department
Prof. _____	Prof. _____
Internal Examiner	External Examiner
Assoc. Prof. Shiburaj Pappu	Dr. Varsha Shah
Dean of Academic, RCOE	Principal, RCOE

Department of Computer Engineering
Rizvi College of Engineering,
Off Carter Road, Bandra(W), Mumbai-400050

Declaration

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

_____

Aimaan Khan 14

_____

Mariyum Siddique 54

_____

Siddharth Pallar 33

Date:

Abstract

In today's educational landscape, managing academic schedules has become a significant challenge due to the increasing number of students, expanding course offerings, and limited infrastructure. Manual creation of timetables is both time-consuming and prone to errors, leading to conflicts in scheduling and inefficient use of resources. To address these issues, this project aims to develop an AI-powered timetable scheduling system that automates the creation of academic schedules. The system will use advanced AI algorithms to optimize resource allocation, including teacher availability, room assignments, and lab scheduling, based on predefined constraints. Built on Node.js and MongoDB for a scalable and flexible backend, this system will not only generate conflict-free timetables but will also allow dynamic adjustments to accommodate last-minute changes. By eliminating manual intervention and leveraging AI, the project seeks to significantly reduce administrative workload while improving the accuracy and efficiency of academic scheduling in institutions of varying sizes.

Keywords: Automation, Conflict-Free Timetables, Resource Optimization, Scalability, Efficiency

Index

CHAPTER 1 INTRODUCTION

The **AI-Powered Timetable Scheduler and Management System** is being developed to address the growing complexities of academic scheduling in educational institutions[1]. As institutions expand with increasing student numbers, more diverse courses, and limited infrastructure, the task of manually creating and managing timetables becomes increasingly challenging. Traditional methods of scheduling often lead to errors such as overlapping classes, teacher unavailability, and room capacity constraints [2], which disrupt academic operations and lead to inefficient use of resources.

This project leverages artificial intelligence (AI) to automate the timetable generation process, ensuring a smarter, conflict-free schedule that adapts to real-time inputs. The AI component will be at the heart of this system, using algorithms that analyse a variety of constraints—such as teacher availability, subject prerequisites, room assignments, and lab requirements—to generate optimized schedules that minimize manual intervention. Unlike conventional rule-based systems, the AI algorithms will not only consider the existing constraints but also continuously learn from previous scheduling patterns and user feedback, allowing it to improve over time and provide more accurate scheduling solutions.

The system architecture is based on Node.js for backend development and MongoDB as the database to store and manage information about teachers, classes, rooms, and schedules. This combination ensures that the system is both scalable and capable of handling large datasets, making it adaptable to institutions of varying sizes. The web interface will allow administrators to input constraints and preferences, and the AI engine will process these inputs to generate a timetable that best suits the institution’s needs. The platform will also provide flexibility for real-time updates, such as changes in teacher availability or room assignments, which will be quickly recalculated to ensure minimal disruption.

The AI-powered system will offer advanced features such as automated conflict resolution, real-time schedule adjustments, and support for last-minute changes. This will significantly reduce the administrative burden on faculty and staff, enabling them to focus more on educational outcomes rather than logistical challenges. Furthermore, the system is designed to be user-friendly and intuitive, allowing non-technical users to interact with it easily. The long-term vision of the project is to integrate advanced machine learning techniques that can

predict future scheduling needs and optimize resource allocation based on historical data and institutional trends.

In summary, this project aims to revolutionize the way educational institutions manage their timetables by automating the entire scheduling process using AI. It will provide an efficient, flexible, and scalable solution that reduces errors, improves resource utilization, and adapts dynamically to changes in real-time, making it a vital tool for modern academic institutions.

CHAPTER 2 LITERATURE SURVEY

2.1 Paper 1: AI-Based Automatic Timetable Generator Using React [1]

Authors: Aviraj Latpate, Nihal Sayyad, Chaitanya Bargal, Adish Sawant, Prof. J.S Choudhari

- **Survey Existing System**: The paper discusses the development of an AI-based timetable generator using **ReactJS** for the frontend and **Firebase** for data storage. The existing system focuses on optimizing timetable generation using constraint satisfaction algorithms to account for teacher availability, classroom resources, and scheduling conflicts. The system allows administrators to dynamically manage timetables in real-time, offering a user-friendly interface for adjustments and customization.
- **Limitation of Existing System or Research Gap**: While the system efficiently generates timetables and resolves scheduling conflicts, the research lacks deep exploration into handling highly complex scheduling constraints, such as unique teacher or student preferences and infrastructural limitations for larger institutions. Additionally, the use of Firebase may limit the system's scalability for larger datasets.
- **Problem Statement and Objective**: The objective is to automate the timetable generation process, reducing manual labour and errors while optimizing resource allocation (classrooms, teachers). The problem is the frequent occurrence of scheduling conflicts and the lack of adaptability in existing systems to handle changes in real-time.
- **Scope**: This system is suitable for small to medium-sized institutions, offering real-time timetable generation and flexibility. It is scalable to a certain extent but might require database optimization or a move to other databases (e.g., MongoDB) to handle larger-scale institutions.

2.2 Paper 2: A Survey on Academic Timetable Scheduling Using AI and ML [2]

Authors: S.S Sonawane, Ram Belitkar, Aarya Jambhulkar, Raman Yadav

- **Survey Existing System**: This paper surveys existing academic timetable scheduling systems that utilize **Genetic Algorithms (GA)** and **AI/ML techniques**. The systems reviewed optimize timetable creation by processing constraints such as faculty availability, classroom resources, and course timings using AI and GA. These techniques significantly reduce the time spent on manual scheduling and improve overall efficiency.
- **Limitation of Existing System or Research Gap**: Although AI and GA improve the scheduling process, the systems reviewed still face challenges in handling extremely complex constraints, like varying room sizes or detailed student preferences. Moreover, while most systems focus on generating conflict-free timetables, they often lack strong interfaces for user engagement and adaptability when changes are required post-generation.
- **Problem Statement and Objective**: The problem is the manual effort required in existing scheduling processes and the frequent occurrence of conflicts between resources. The objective is to provide a scalable, conflict-free scheduling solution using **AI/ML** to automate the process, reducing human error and optimizing resource usage.
- **Scope**: This paper highlights the use of genetic algorithms and machine learning techniques in academic institutions of various sizes. It is particularly applicable in scenarios where scalability and efficiency are required, though the solution might need additional user interface enhancements for real-world usability.

2.3 Paper 3: Automated Timetable Generation Using Genetic Algorithm [3]

Authors: Shraddha Thakare, Tejal Nikam, Prof. Mamta Patil

- **Survey Existing System**: This paper presents a system to automate the generation of academic timetables using **Genetic Algorithms (GA)**. The GA framework is applied to optimize resource allocation and scheduling, encoding data like courses, rooms, and teacher availability into genetic sequences. The system uses evolutionary operators like crossover and mutation to iteratively find optimal timetable solutions.
- **Limitation of Existing System or Research Gap**: The system has a well-developed approach for generating conflict-free schedules, but it relies heavily on predefined

constraints, which may limit flexibility when last-minute changes are required. The research also does not address how to handle complex institutional requirements, like varying room capacities or multiple teacher preferences.

- **Problem Statement and Objective**: The problem is the high frequency of scheduling conflicts and the manual effort required in timetable generation. The objective is to develop a more efficient, conflict-free timetable generator using **Genetic Algorithms** to encode and optimize scheduling data.
- **Scope**: The system is suitable for mid-sized educational institutions, capable of generating conflict-free timetables efficiently. It can be adapted for more complex constraints, but future improvements are needed to make it flexible for larger institutions with diverse needs.

2.4 Paper 4: AI-Based Timetabling Algorithms: A Comparative Analysis [4]

Authors: Fernando S. Viray Jr., Melvin A. Ballera

- **Survey Existing System**: The paper conducts a comparative analysis of four AI-based algorithms—Tabu Search, Greedy Algorithm, Integer Linear Programming, and Bi-Partite Graph Approach—for solving timetabling problems. These algorithms are designed to optimize class scheduling in academic institutions by minimizing scheduling conflicts and adhering to institutional constraints. The existing system focuses on using local search and optimization techniques to create conflict-free schedules based on predefined hard constraints, such as room availability and teacher assignments, while aiming to satisfy as many soft constraints as possible.
- **Limitation of Existing System or Research Gap**: The current research highlights a limitation in the ability of the evaluated algorithms to fully satisfy soft constraints, such as teacher preferences and time-slot preferences for different subjects. Furthermore, the research does not delve deeply into handling the computational challenges that arise when the problem scales to larger institutions with complex constraints. Another limitation is the high computational time required, especially for algorithms like Integer Linear Programming, which can be inefficient when the number of constraints and variables increase.

- **Problem Statement and Objective**: The objective of this research is to evaluate the effectiveness of different AI-based timetabling algorithms in generating conflict-free schedules, while minimizing computational resources and satisfying both hard and soft constraints. The problem lies in finding the most efficient algorithm that can handle multiple constraints simultaneously, provide scalable solutions, and ensure high accuracy in timetable generation.
- **Scope**: The study is aimed at educational institutions that need to schedule courses, teachers, and classrooms across multiple departments. It focuses on comparing algorithms to find an optimal solution that satisfies most constraints while minimizing computational load. The scope is limited to algorithms that rely on local search and optimization techniques, with the potential for expanding to larger datasets by implementing advanced optimization methods or hybrid approaches.

CHAPTER 3 PROPOSED SYSTEM

3.1 Analysis/Framework/Algorithm

In this section, the analysis of the existing system is performed, and a hybrid AI framework is proposed to address the identified problems. The proposed system combines **Random Forest** and **Genetic Algorithm (GA)** to automate timetable generation. It assigns classrooms, labs, and teachers based on specific parameters like teacher availability, workload, and infrastructure capacity.

The Random Forest model classifies initial timetable configurations using historical data, learning successful scheduling patterns [1]. The Genetic Algorithm then optimizes these configurations by evolving them further, handling complex constraints dynamically [3]. This combination ensures iterative learning and adapts the generated timetables to meet evolving requirements, making the scheduling process more efficient and globally optimized.

The key components of this analysis are:

3.1.1. Problem Identification:

Manual timetable scheduling is inefficient and prone to conflicts, such as overlapping classes for teachers and classrooms [3]. There is no centralized system to manage and generate timetables in an automated manner, which often leads to errors and inconsistencies in scheduling. Additionally, human errors, such as misjudging room capacities or teacher availability, frequently result in last-minute adjustments that disrupt the timetable [5].

To address these issues, the proposed AI-based system combines **Random Forest** and **Genetic Algorithm** to automate and optimize the scheduling process. The **Random Forest** component learns from historical data, reducing human errors and improving initial timetable configurations. Meanwhile, the **Genetic Algorithm** dynamically adjusts the configurations to handle complex constraints, further minimizing disruptions and adapting to real-time changes.

3.1.2. Proposed Algorithm:

The system employs a hybrid AI-powered algorithm that combines **Random Forest** for initial classification and **Genetic Algorithm (GA)** for further optimization [1][3], ensuring no

overlapping or conflicting schedules. This approach addresses both necessary and optional scheduling criteria, making the process adaptable to changes and optimizing resource utilization [4].

Necessary Criteria:

- **No clash in classrooms**: Each room can only be assigned to one class at a time.
- **No clash in teacher availability**: A teacher should not be scheduled for more than one class simultaneously.
- **No clash in subject slots**: Subjects assigned to the same class should not overlap in their time slots.
- **Completion of contact hours**: The generated timetable ensures that the contact hours defined for each subject are met.

Optional Criteria (for better optimization):

- **Preferred time slots for teachers**: The system considers teacher preferences for lecture timings.
- **Minimizing gaps between lectures**: The AI aims to reduce gaps between consecutive lectures for both students and teachers.

Steps of the AI Algorithm:

1. **Input all available data** (teachers, subjects, classrooms, labs, workload).
2. **Set both necessary and optional constraints**:
    - The **Random Forest model** initially classifies timetable configurations based on these constraints, using past data to identify "good" configurations.
    - The **Genetic Algorithm** then refines these configurations, handling additional constraints like teacher preferences and minimizing gaps.
3. The AI engine starts timetable generation by iterating over subjects and assigning time slots based on teacher availability, room infrastructure, and other constraints.
4. The AI checks for conflicts after each iteration (teacher, class, room conflicts) and adjusts as necessary.
5. The system finalizes the timetable and displays it to the administrator. Any real-time changes (e.g., teacher absence or room unavailability) can be quickly resolved by rerunning the AI algorithm with updated inputs.

3.2 Details of Hardware & Software

Hardware:

- **Development Machine:** A high-performance personal computer or server with the following minimum specifications to ensure efficient development and testing of the AI model and backend systems:
    - **CPU:** Quad-Core Processor (i5 or higher), preferably with support for AI/ML tasks if needed (e.g., Intel i7, Ryzen 7, or higher).
    - **RAM:** 8 GB or higher to handle multiple simultaneous operations, including AI model training and database management.
    - **Storage:** 512 GB SSD or higher for faster data access and reduced latency during development and deployment.
    - **GPU (Optional):** A GPU equivalent NVIDIA RTX series to accelerate AI model training, especially for large datasets.

Software:

- **Backend Framework:**
    - **Node.js:** A JavaScript runtime used to build scalable backend solutions, offering asynchronous, event-driven capabilities crucial for handling multiple timetable generation requests.
    - **Express.js**: A web framework for Node.js that simplifies the development of RESTful APIs, making it easier to manage communication between the frontend, backend, and AI modules.
- **AI/ML Tools:**
    - **TensorFlow/PyTorch:** Libraries used for developing and training AI models. These frameworks are essential for the implementation of the AI-powered timetable generation algorithm, allowing for continuous learning and optimization.
    - **Scikit-learn:** A Python library for handling data pre-processing and applying machine learning algorithms.
- **Database:**

- **MongoDB:** A NoSQL database used to store flexible, schema-less data for teachers, subjects, classrooms, departments, and generated schedules. This allows for better scalability and adaptability in managing varying institutional data.

- **Frontend:**
  - **HTML/CSS:** For structuring and styling the user interface, ensuring that the system is user-friendly and responsive.
  - **React.js (Optional):** A modern JavaScript framework to enhance user interactions and improve the efficiency of rendering real-time updates on the web interface.
  - **EJS (Embedded JavaScript Templating):** For rendering dynamic pages and providing administrators with real-time scheduling information and conflict alerts.
- **Development Environment:**
  - **Visual Studio Code:** A powerful code editor used for both backend and frontend development, offering support for a variety of programming languages and integrated tools for efficient development.
  - **Jupyter Notebook:** For experimentation with AI model training and tuning, providing a flexible environment for running AI-related code.

This combination of hardware and software ensures that the system is capable of handling the complex requirements of AI-driven timetable generation while remaining scalable and flexible for future updates.

3.3 Design Details

The design of the system is divided into two major components: **System Architecture** and **Database Design**.

System Architecture

The system follows a **client-server architecture** to ensure scalability and responsiveness. The backend integrates a hybrid AI-powered model that combines **Random Forest** and **Genetic**

Algorithm to generate and optimize timetables, while the frontend provides administrators with an easy-to-use interface for inputting and managing scheduling data.

Client-Server Communication:

- The **client (web interface)** communicates with the server using HTTP requests. Users input constraints such as teacher availability, classroom details, and lab requirements.
- The **server** processes these inputs and uses the hybrid AI model to generate optimized timetable configurations:
- The **Random Forest** model handles initial classification based on historical data, identifying feasible timetables quickly.
- The **Genetic Algorithm** further optimizes these configurations by evolving the initial outputs to better fit complex constraints, such as teacher preferences and room capacities.
- The AI model is continuously trained and refined, improving scheduling efficiency based on user feedback and historical data.
- The **server interacts with the MongoDB database** to store and retrieve scheduling information, ensuring data consistency, conflict resolution, and real-time updates.

Key Modules

- **Teacher Module**:
    - Manages teacher records, including personal details, assigned subjects, and availability. The AI uses this data to avoid scheduling conflicts based on teacher workload and preferences.
    - The **Random Forest** component uses historical teacher data to predict potential conflicts, while the **Genetic Algorithm** refines allocations to achieve optimal workload distribution.
- **Class Module**:
    - Responsible for creating and managing classes, including their subject schedules. This module handles class-specific constraints, such as subject prerequisites or specific time slot requirements.

    - The AI considers these constraints during timetable generation, initially predicting feasible schedules using Random Forest and then optimizing the overall timetable structure with Genetic Algorithm adjustments.
- **Infrastructure Module**:
    - Manages available rooms, including classrooms and laboratories. The AI optimizes room allocation by considering factors such as room size and equipment availability for lab-based subjects.
    - The combination of Random Forest and Genetic Algorithm ensures efficient room utilization, dynamically adjusting to infrastructure constraints.
- **Timetable Module**:
    - Automates timetable generation by feeding all inputs (teacher, class, and infrastructure data) into the hybrid AI engine. This module resolves conflicts and dynamically adjusts schedules in case of last-minute changes (e.g., teacher unavailability).
    - The AI engine begins with Random Forest for initial timetable suggestions, then applies Genetic Algorithm-based adjustments to handle evolving constraints and real-time changes.

Database Design:

The database is designed to handle the flexibility required for dynamic timetable generation and storage. The key collections in **MongoDB** include:

- **Teachers Collection**:
    - Stores teacher data, including names, departments, subjects taught, workload, and availability. This collection is used to assign teachers to classes while avoiding scheduling conflicts.
- **Subjects Collection**:
    - Contains detailed information about the subjects offered in the institution, including the department, subject code, assigned teachers, and credit hours.
- **Infrastructure Collection**:
    - Holds data about available labs and classrooms, including room capacity and equipment details. The AI ensures that lab-based subjects are assigned to appropriate rooms with sufficient capacity and resources.
- **Timetables Collection**:

- Stores the generated timetables for each academic year and term, allowing administrators to review and modify schedules as needed. Each timetable is generated dynamically based on the latest data inputs from the modules.

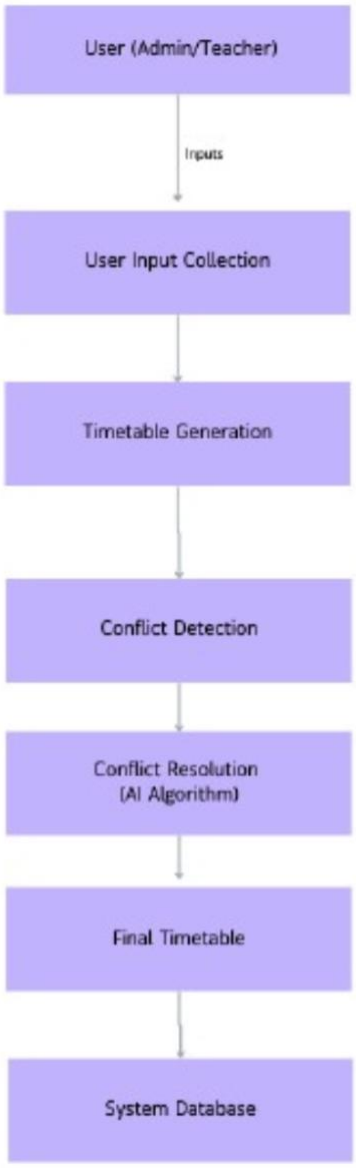

Fig 3. 1 Data Flow Diagram (DFD) for AI-Powered Timetable Generation

3.4 Methodology

The methodology adopted for solving the problem of timetable generation follows an **agile and iterative approach**, ensuring that the system evolves based on continuous feedback from users

and adapts to changing requirements. The integration of AI into the system allows it to optimize the process over time, making it smarter and more efficient with every iteration.

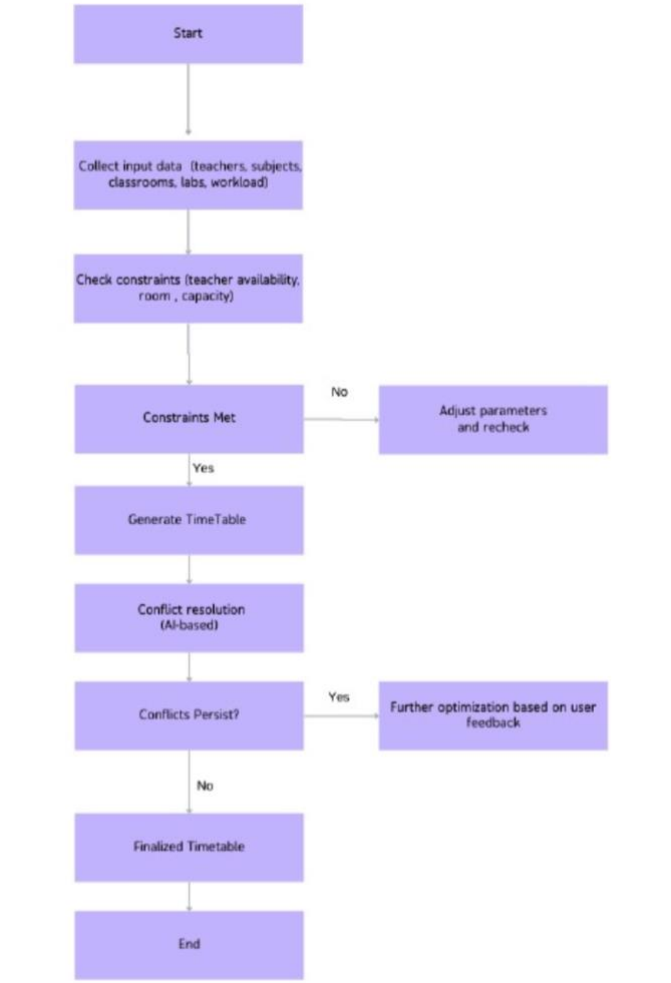

Fig 3. 2 Flowchart for Timetable Generation Methodology

3.4.1 Problem Definition

- The need for automating the timetable generation process was identified through consultations with end users, such as school or college administrators. Manual scheduling was found to be inefficient, time-consuming, and prone to errors, such as overlapping teacher schedules, double-booked rooms, and last-minute changes that were difficult to manage without causing disruption.
- The hybrid approach, combining Random Forest and Genetic Algorithm, was proposed to address these challenges. The Random Forest model focuses on learning historical patterns to generate feasible initial schedules, while the Genetic Algorithm refines these schedules, handling dynamic constraints and optimizing the overall timetable structure.

3.4.2 Requirements Gathering

- Through interviews and interactions with educational staff, the following key requirements were identified:
    - Preventing scheduling conflicts, such as no double-booking of teachers or classrooms.
    - Efficient management of infrastructure, such as classrooms and labs, based on capacity and resource availability.
    - Considering teacher preferences for lecture timings and minimizing the number of gaps between classes.
- The hybrid AI model is designed to fulfil these requirements:
    - The Random Forest ensures that initial schedules adhere to basic constraints based on past successful configurations.
    - The Genetic Algorithm fine-tunes these schedules to better meet teacher preferences and infrastructure utilization, achieving a more adaptive timetable.

3.4.3 System Design

- The system was modularized into multiple components, including the Teachers Module, Subjects Module, Classes Module, and the Timetable Module. This modular approach enables flexibility in development and ensures that each component can be improved or scaled independently.
- The database, structured with MongoDB, supports the flow of required data (teachers, classes, rooms, etc.) into the hybrid AI model, which processes and generates optimal timetables:
    - The Random Forest initially classifies timetable configurations based on historical patterns.
    - The Genetic Algorithm then optimizes these configurations, adjusting dynamically to meet complex requirements.
- The architecture supports real-time adjustments, allowing the system to adapt to changes (e.g., teacher unavailability) without significant disruption. The Genetic Algorithm handles this dynamic adaptability, ensuring a conflict-free schedule despite real-time changes.

3.4.4 AI Algorithm Implementation

- The timetable generation process is powered by the hybrid AI model, integrating Random Forest and Genetic Algorithm [1][3]:
    - The Random Forest model analyses historical scheduling data, focusing on constraint satisfaction to generate feasible initial timetables.
    - The Genetic Algorithm optimizes these initial timetables by evolving configurations to better fit user-defined constraints, such as room type, teacher availability, workload, and preferences.
    - This hybrid AI model continuously learns from user feedback and historical data, improving its scheduling approach over time.
- As the AI algorithm receives feedback from users, it adjusts its scheduling strategy to enhance efficiency and minimize manual intervention [7]. The Genetic Algorithm component allows for reinforcement, making the AI more adaptive to real-world challenges.

3.4.5. Testing:

- The system was rigorously tested using multiple test cases to cover a wide range of real-world scenarios. These test cases include:
    - Handling teacher absences and dynamically reassigning their classes to available time slots.
    - Managing room shortages and ensuring that classes are allocated based on size and resource requirements.
    - Detecting and resolving conflicting schedules for teachers, classrooms, and labs.
    - Ensuring that the AI-generated timetable adheres to all predefined constraints, including teacher preferences and institutional requirements.

3.4.6. Deployment and Maintenance:

- After passing testing, the system will be deployed on a cloud-based infrastructure, ensuring it can be accessed by administrators from anywhere via a web interface.

- **Ongoing Maintenance**: The AI model will be periodically retrained as more data is gathered from actual scheduling activities. The system will receive regular updates based on user feedback, and technological improvements will be made to optimize performance and scalability.
- Cloud deployment will allow for automatic scaling based on institutional size, ensuring the system can handle varying workloads without compromising performance.

Chapter 4: Implementation Plan and Status & Proposed

4.1 Implementation Plan

The implementation of the AI-powered timetable scheduler started in August 2024. The project follows an agile, iterative process, and the phases completed so far include backend development and the creation of a basic frontend. The remaining phases outline the future steps toward full project completion.

4.1.1 Phase 1: Information Gathering and Literature Survey (4th week of August – 2nd week of September 2024)

- **Objective:** Understand the scheduling needs of institutions, gather data, and review existing literature on timetable scheduling systems.
- **Key Tasks:**
    - Conduct interviews with educational staff to gather requirements and constraints.
    - Perform a detailed literature review to understand the existing solutions and gaps in timetable generation systems.
- **Status:** Completed.

4.1.2 Phase 2: Backend Development and Basic Frontend Setup (1st week of September – Mid-September 2024)

- **Objective:** Develop the backend and create a basic frontend for managing essential data such as classes, departments, teachers, subjects, and infrastructure.
- **Key Tasks:**
    - Set up MongoDB as the database for storing all relevant data.
    - Develop APIs using Node.js and Express.js for CRUD operations on data related to teachers, subjects, classrooms, etc.
    - Create a basic user interface using HTML/CSS to allow administrators to interact with the system (e.g., inputting data into the database).
    - Ensure proper relationships between the data entities.
- **Status:** Completed (both backend and basic frontend).

4.1.3 Phase 3: Timetable Generation Logic for a Single Class (Mid-September – Mid-October 2024)

- **Objective:** Implement timetable generation logic for a single class, avoiding infrastructure, subject, and teacher clashes.
- **Key Tasks:**
    - Implement core scheduling logic that factors in room availability, teacher workloads, and subject schedules.
    - Test the timetable generation to ensure conflict-free scheduling for single-class scenarios.
- **Status: Completed.**

4.1.4 Phase 4: AI Integration and Multi-Class Timetable Generation (Planned: October – November 2024)

- **Objective:** Extend the timetable generation logic to multiple classes and integrate the AI-powered conflict resolution model.
- **Key Tasks:**
    - Develop an AI algorithm that optimizes scheduling for multiple classes.
    - Train the AI model using real scheduling data and allow it to learn from past timetables.
    - Implement optional criteria (e.g., teacher preferences) to improve scheduling efficiency.
- **Expected Output:** A conflict-free timetable for multiple classes, optimized by AI.

4.1.5 Phase 5: Full Frontend Development (Planned: December 2024)

- **Objective:** Enhance the user interface and complete frontend development for full system functionality.
- **Key Tasks:**
    - Build dynamic forms to input and manage teacher, class, and infrastructure data.
    - Integrate the frontend with backend APIs and AI components.
    - Provide real-time visualization of generated timetables.
- **Expected Output:** A fully functional user interface for administrators and users.

4.1.6 Phase 6: Testing, Debugging, and System Optimization (Planned: January – February 2025)

- **Objective:** Conduct comprehensive testing of the AI-powered timetable system to ensure it performs well under multiple conditions.
- **Key Tasks:**
  - Test the AI-generated timetables for various scenarios (teacher absences, room shortages, etc.).
  - Debug issues and ensure that all constraints are respected in the generated timetables.
  - Optimize the AI algorithm for faster and more accurate scheduling.
- **Expected Output:** A bug-free, optimized system ready for deployment.

4.1.7 Phase 7: Deployment and Maintenance (Planned: March – April 2025)

- **Objective:** Deploy the system on a cloud platform and ensure that it is accessible and scalable for multiple users.
- **Key Tasks:**
  - Deploy the backend and frontend to a cloud platform (AWS or Azure).
  - Train administrators to use the system efficiently.
  - Begin collecting feedback for further iterations and improvements.
- **Expected Output:** A fully deployed, cloud-based timetable scheduling system.

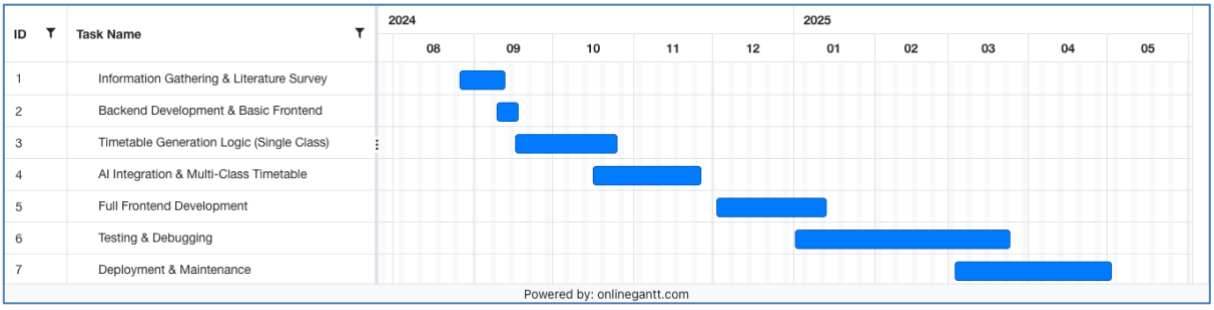

Fig 1 Gantt Chart

Conclusion

The Timetable Scheduler offers a robust solution to the challenges of manual scheduling in educational institutions. By automating the allocation of teachers, classrooms, and subjects, the system reduces the likelihood of conflicts and enhances operational efficiency. The use of Node.js and MongoDB ensures that the platform is both scalable and adaptable to the specific needs of various institutions, offering long-term flexibility and ease of maintenance. This project not only reduces the administrative workload but also improves the accuracy and effectiveness of timetable management, making it an essential tool for modern educational environments. Future enhancements, such as role-based access control and report generation, can further improve its functionality, ensuring a comprehensive solution for managing academic schedules.

References

[1] A. Latpate, N. Sayyad, C. Bargal, A. Sawant, and J. S. Choudhari, "AI-Based Automatic Timetable Generator Using React," *International Journal of Computer Applications*, 2022.

[2] M. Asif and A. Khalid, "Artificial Intelligence in Education: AI-Powered Timetable Scheduling System," *Journal of Educational Computing Research*, vol. 58, no. 3, 2021.

[3] Z. Habib and A. U. Haq, "Constraint-Based Timetable Scheduling Using Genetic Algorithms," in *Proc. Int. Conf. Computing*, 2020.

[4] K. A. Smith and D. L. Wilkinson, "Automated Scheduling Using AI: A Case Study in Timetabling," *IEEE Trans. Educ.*, vol. 52, no. 4, 2019.

[5] C. R. Knight and T. O'Donnell, "A Comparative Study of Scheduling Algorithms for Academic Timetable Generation," *Int. J. Scheduling*, vol. 15, 2020.

[6] M. Gendreau and J. Potvin, "Metaheuristics in Scheduling and Timetabling: A Survey," *Ann. Oper. Res.*, 2019.

[7] J. S. Zilberstein and A. Jacobs, "Machine Learning Approaches for Multi-Class Scheduling Problems," *J. Artif. Intell. Res.*, 2021.

[8] S. Sonawane, R. Patil, and M. Thakare, "AI in Timetable Scheduling: Implementation and Testing," *Proc. IEEE Conf. on EduTech*, 2020.

[9] R. Sharma and P. Sharma, "Automating Timetable Generation with AI Techniques," *Int. J. Adv. Res. Comput. Sci.*, vol. 12, 2021.

[10] F. S. Viray Jr. and M. A. Ballera, "AI-Based Timetabling Algorithms: A Comparative Analysis," *Proceedings of Researchfora 31st Int. Conf.*, 2018.