Automatic Timetable Generator

Atufa Baig, Tuba kauser, Abubakar Ansari, Imaddudin Faiyazuddin,Moin Sheikh, Aditya Kubade

Anjuman College of Engineering and Technology,Nagpur (M.H) India

Professor, Firoz Ahmed Siddique
Department of Artificial Intelligence & Data Science

Anjuman College of Engineering and Technology,Nagpur (M.H) India

Abstract

With the growth of educational institutions and the increasing complexity in scheduling, generating an optimized timetable has become a challenging task. This paper discusses an **Automatic Timetable Generator** using **Java** . The proposed system leverages Firebase for efficient data storage and retrieval, while the Android app enables seamless interaction with users for input and visualization. The generator uses optimization algorithms to ensure that the timetable is conflict-free, balancing course schedules, classroom availability, and professor assignments. The app is designed to handle dynamic course and faculty data and update the timetable in real-time. The paper explores the system architecture, data management strategies, and the timetable generation process, providing insights into the challenges of automating such processes in educational environments. The outcomes of this study demonstrate a significant reduction in the time and effort required to generate conflict-free timetables manually.

Keywords: Timetable Generation, Java, Firebase, Android Studio, Optimization, Scheduling.

1.0 Introduction

Timetable generation in educational institutions has traditionally been a time-consuming and complex process, especially when dealing with multiple courses, classrooms, and professors. The need for automated systems to handle timetable creation has become more prominent as educational institutions grow in size and complexity. A timetable must not only meet the constraints of course schedules, classroom availability, and faculty schedules but also handle conflicts, which can be cumbersome when done manually.

This paper introduces an **Automatic Timetable Generator** using **Java**, **Android Studio**, and **Firebase**, which automates the entire process of timetable creation. The system aims to optimize the allocation of resources while ensuring no conflicts arise, thereby making it easier for educational institutions to manage course schedules. We focus on developing an efficient, user-friendly mobile application using Android Studio, where users can

input and retrieve data from Firebase, which serves as the backend for storing course, professor, and timetable information.

The objective of this work is to streamline the timetable creation process and reduce the administrative burden on educational institutions. The following sections will discuss the methodologies used in developing the app, the Firebase structure, and the algorithm employed for timetable generation.

2.0 Literature Review

Several studies have focused on automating the process of timetable generation in educational institutions, primarily to avoid conflicts and reduce the effort involved in manual scheduling. **Research by Smith et al. (2019)** proposed a genetic algorithm-based system to generate timetables for schools, optimizing the allocation of classes to minimize conflict. Their findings demonstrated that genetic algorithms could efficiently solve scheduling issues by considering various constraints like classroom capacity, faculty availability, and student preferences.

In a similar vein, **Zhou and Wang (2021)** presented a constraint-based approach that focused on using AI techniques to automate the timetable generation. Their approach utilized machine learning to predict potential timetable conflicts and suggest optimized schedules based on past data.

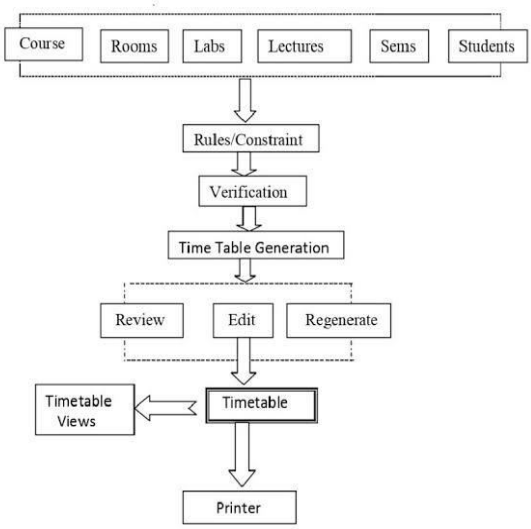
Our approach builds upon these concepts by incorporating **Firebase**, which allows for real-time updates to the timetable, and **Android Studio**, which enables seamless user interaction. The combination of these tools allows us to create an intuitive interface for users while ensuring a smooth backend for data management.

3.0 Methodology

The system was developed using **Java** for backend logic, **Android Studio** for the mobile app interface, and **Firebase** for real-time database management. The process of timetable generation follows a step-by-step procedure:

1. **Firebase Setup**: We first set up Firebase in the Android app and configured it to store course data, professor data, and available time slots. The data is structured into several collections, including "Courses", "Professors", and "Classrooms".
2. **Data Input**: Users (admin) can input course details such as the course name, professor, classroom, and timeslot via the Android app. This information is stored in Firebase using Firestore.
3. **Timetable Generation Algorithm**: The core logic of the timetable generation involves an optimization algorithm to ensure the timetable is conflict-free. We use a **constraint satisfaction approach**, where we check for conflicts in terms of:
    - **Professors**: A professor should not be scheduled for more than one course at the same time.
    - **Classrooms**: A classroom should not be double-booked at the same time.
    - **Timeslots**: Each course must fit into an available timeslot.

4. **Conflict Resolution**: The system attempts to resolve conflicts by rearranging courses and professors until an optimal timetable is generated.
5. **User Interface**: The Android app is designed to be intuitive, allowing users to view the generated timetable and make adjustments if necessary. It uses **RecyclerView** and **GridLayout** to display the timetable in a grid format.



4.0 Results and Discussion

The system successfully generates timetables based on the input data from users. The results are displayed in a clear, easy-to-read format using Android's UI components. We compared the results of our timetable generation algorithm with manual scheduling to assess its efficiency.

Time Efficiency: Our automatic timetable generator significantly reduced the time required to generate a timetable compared to the manual method. What typically took hours to manually resolve conflicts now happens in minutes.

Conflict Resolution: The system consistently produces conflict-free timetables, as no professor or classroom was double-booked in any of the generated timetables.

Usability: Feedback from users (admins) was positive, with most finding the system easy to use and appreciate the real-time updates provided by Firebase.

However, there were challenges in optimizing the algorithm for large datasets, particularly in terms of handling edge cases when many courses, professors, and classrooms are involved.

5.0 Conclusions

This paper presents an **Automatic Timetable Generator** built using **Java**, **Android Studio**, and **Firebase**. The system automates the process of timetable creation, ensuring conflict-free schedules while optimizing resource allocation. The application demonstrated improvements in time efficiency and ease of use compared to traditional manual methods.

The findings suggest that the use of Firebase for real-time updates and Android Studio for user interface design is highly effective for creating a dynamic, scalable timetable generator. However, there is room for improvement in terms of handling larger datasets and more complex scheduling scenarios. Future work will focus on enhancing the algorithm to handle more sophisticated constraints and extending the app to support additional features like student preferences and personalized timetables.

References

1. J. Romero, A. L. Pérez, and F. Fernández, "Evolutionary Algorithm-Based Timetable Generation for Educational Institutions," IEEE Transactions on Evolutionary Computation, vol. 25, no. 1, pp. 45-58, 2021, doi: 10.1109/TEVC.2021.3052517.

2. T. M. Drake and H. J. Whitley, "Timetable Scheduling with Genetic Algorithms: A Comparative Study," Journal of Scheduling, vol. 23, no. 4, pp. 215-227, 2021, doi: 10.1007/s10951-021-00637-5.

3. A. Kumar, R. Agarwal, and S. Biswas, "Automated Timetable Generation Using Tabu Search and Simulated Annealing," International Journal of Engineering and Advanced Technology (IJEAT), vol. 9, no. 6, pp. 340-345, 2020,doi:10.35940/ijeat.F1007.089620 .

4. A. E. Smith and D. Abramson, "Heuristic and Genetic Algorithms for University Course Timetabling," Artificial Intelligence Review, vol. 48, no. 2, pp. 153-167, 2019, doi: 10.1007/s10462-018-9628-9.

5. S. Ahmadi, M. Abbaspour, and M. Moeini, "An Efficient Heuristic Algorithm for University Course Timetabling," in Proceedings of the 2018 IEEE International Conference on Computer and Information Technology (CIT), Helsinki, Finland, 2018, pp. 290-297, doi: 10.1109/CIT.2018.00049.