Available online at www.worldscientificnews.com

World Scientific News

An International Scientific Journal

WSN 189 (2024) 288-310 EISSN 2392-2192

Hybrid technologies and genetic algorithms applied to school lecture timetable generation

Rotimi-Williams Bello*, Joyce Ayibane Godwill

Department of Mathematics and Computer Science, University of Africa, Toru-Orua, 561101 Sagbama, Bayelsa, Nigeria

*E-mail address: sirbrw@yahoo.com

ABSTRACT

This paper proposes an advanced lecture timetabling system utilizing hybrid technologies and genetic algorithms for optimal scheduling whereby resource allocation challenges and conflicts are addressed and minimized. Genetic algorithms offer a solution to complex scheduling problems by mimicking natural selection principles, thereby refining lecture timetables through selection, crossover, and mutation. The system formulates the timetabling problem as a genetic algorithm optimization task by integrating constraints into the fitness function, which include resource availability, activity dependencies, and user preferences; facilitating evaluation based on predefined objectives like conflict minimization and resource utilization maximization. The genetic algorithm employs a population-based search strategy, iteratively refining timetables over generations. The system demonstrates high solution quality, efficiency, and scalability when compared with traditional methods, thereby providing benefits to educational institutions and the like.

Keywords: Timetabling, Genetic algorithms, Optimization, Scheduling, Resource allocation

1. INTRODUCTION

The imperative to automate the manual process of creating lecture timetables within higher educational institutions remains an undeniable necessity. Despite witnessing the

(Received 31 December 2023; Accepted 22 January 2024; Date of Publication 24 January 2024)

automation of various administrative sectors across most institutions (Bello et al., 2019), the manual approach to lecture timetabling endures due to its intrinsic challenges. The intricacies of planning a lecture timetable impose a substantial administrative burden, demanding significant attention and effort from institutions due to its inherent constraint satisfaction nature. A lecture timetabling system involves the intricate orchestration of scheduling lecture times while optimizing available resources (Burke et al., 2007). This task ranks among the paramount responsibilities within any academic institution, albeit one that consumes a considerable amount of time in its periodic execution. Hence, meticulous allocation of lecture slots to align with the schedules of students, lecturers, and appropriate lecture halls is of utmost importance, guided by various constraints (Chowdhury et al., 2014).

The process of generating a lecture timetable is executed through a three-phase approach. The initial phase encompasses the collection of data from multiple departments, wherein each department contributes essential course information such as course title, course code, contact hours (comprising lectures and practical sessions), course units, and the designated lecturers. Subsequently, the second phase revolves around a comprehensive analysis of the amassed data, juxtaposed against the available physical spaces, such as lecture hall capacities. This phase acts as a critical bridge between the data provided by the departments and the physical resources within the institution. Lastly, the third phase involves the actual scheduling of lecture times. The culmination of the preceding phases results in three distinct outcomes: the omnibus/main control hall timetable, departmental input timetables, and the lecturers' timetable. As the lecture timetable crystallizes, it must harmoniously accommodate a set of stringent constraints, which include ensuring that no student is assigned conflicting lectures, guaranteeing that no lecturer is tasked with teaching multiple courses concurrently, and preventing the double-booking of lecture halls for simultaneous courses.

In response to these challenges, the proposed system emerges as a potent solution. The proposed solution is online-based system, and is primed to revolutionize the capturing, updating, and retrieval of lecture timetable-related information. The inherent nature of its web-based architecture empowers both lecturers and students with the ability to access vital lecture timetable information at their convenience. This accessibility transcends temporal and geographical barriers, providing real-time access to the ever-evolving landscape of lecture scheduling.

1. 1. The concept of lecture timetabling

The lecture timetabling system constitutes a complex orchestration of arranging lecture schedules and optimizing the allocation of available resources, aiming to harness their potential to the fullest. This intricate task, while being of paramount importance, is undeniably time-consuming, constituting a recurring responsibility within the realm of academic institutions. The process of crafting a lecture timetable necessitates a meticulous allocation of lectures to specific time slots that align with the constraints and preferences of students, lecturers, and available lecture halls. Recognizing its significance, the formulation of a lecture timetable mandates a judicious approach to ensure a seamless educational experience. The assignment of lectures to suitable time slots within the overarching schedule requires careful consideration of a multitude of factors, each carrying their own weight of constraints.

This intricate orchestration is often a manual endeavor, overseen by the standing committee of the academic board for lecture timetable in various academic institutions (Soyemi et al., 2017b). An inherent challenge in this process is the centralization of lecture timetabling,

wherein the scheduling transcends individual school or department boundaries. This central approach encompasses data collection from diverse departments within each school, serving as the bedrock for the ensuing preparation stages. This process unfolds in a three-fold manner, delineated into distinct phases that shape the final lecture timetable. The initial phase encompasses the crucial endeavor of data collection from various departments. Each department contributes an array of pertinent information, including course titles, course codes, contact hours encompassing lectures and practical sessions, course units, and the designated lecturers for each course.

Subsequently, the second phase entails an analytical dissection of the amassed data, juxtaposing it against the available lecture halls. This analytical endeavor forms the crux of the timetabling endeavor, aiming to align the manifold requirements with the available resources. The culmination of this process emerges in the form of the lecture time schedule, synthesized with three distinct outputs: the hall timetable, departmental input, and lecturers' timetable. These outputs coalesce to present a comprehensive and coherent representation of the academic landscape, effectively managing the allocation of resources while accommodating the multifaceted constraints and preferences of the academic community (Soyemi et al., 2017b). However, the orchestration of the lecture timetable transcends mere logistics. It necessitates a delicate balancing act, interwoven with a web of constraints that demand meticulous consideration. Chief among these constraints are imperatives such as ensuring that no student is scheduled to attend multiple lectures simultaneously, preempting a lecturer from teaching multiple courses simultaneously, and judiciously allocating lecture halls to specific courses without overlap.

In summation, the lecture timetabling process is an intricate endeavor that converges the demands of students, lecturers, and available resources into a harmonious schedule. Through a meticulous orchestration, encompassing data collection, analysis, and synthesis, academic institutions endeavor to craft a lecture timetable that optimally serves the educational ecosystem while upholding the constraints that define its intricacies.

1. 2. Manual timetabling generation

Timetable scheduling, as a complex computational challenge, falls under the category of NP-hard problems. NP-hard, denoting Non Polynomial-hard, underscores the inherent complexity of this task. It signifies that there isn't a specific algorithm that universally fits the bill for generating timetables. This is primarily due to the fact that timetable constraints exhibit substantial variation from one educational institute to another. The conventional approach to manual lecture timetabling demands a substantial investment of time, effort, and a cascade of paperwork. The manual process of crafting a lecture timetable is a multifaceted endeavor encompassing the strategic allocation of courses, lecturers, and resources like lecture halls into designated timeslots, all while abiding by a myriad of constraints.

This undertaking is often entrusted to committees who shoulder the responsibility of generating timetables manually. However, this process is replete with challenges that span from computational errors and scheduling conflicts in lecture halls to inadvertent omissions of courses, to name a few. In the educational realm, space management emerges as a critical and pressing concern, surpassing the urgency even in other sectors (Stephens, 2008). The effective management of space and facilities within educational institutions is a paramount endeavor that warrants meticulous attention. A study conducted by Chowdhury et al. (2014) echoes the sentiment that efficient space and facilities management are of utmost importance, especially

in institutes of higher learning. The intricate nature of space and facilities management within educational institutions underscores the imperative to handle this aspect with utmost efficiency and efficacy.

According to the insights presented by Burke et al. (2007) timetabling encapsulates the challenge of judiciously allocating diverse resources to scheduled meetings in a coherent manner. This multifaceted endeavor entails navigating through the interplay of various factors: the set of incidents denoted by E = {e1, e2, ..., en}, a corresponding set of designated times outlined as T = {t1, t2, ..., ts}, the assortment of venues symbolized by P = {p1, p2, ..., pm}, and the agents involved in orchestrating these incidents encapsulated in A = {a1, a2, ..., an}—for instance, lecturers. Formally defining the concept of a timetable involves a nuanced comprehension of these resources and time slots. It entails elements such as:

a. A collection of lecturers {t1, t2, ..., tn}
b. A compilation of courses {c1, c2, ..., cm}
c. An assemblage of lecture halls {r1, r2, ..., rq}
d. An array of time slots {p1, p2, ..., ps}

This approach strives to weave together a coherent arrangement of classes tailored to the schedules of each lecturer, while also ensuring that no lecturer is allocated to simultaneous classes. This methodology seeks to untangle several complexities, including accommodating courses with varying durations, handling joint courses, distributing courses across the week, and allocating appropriate lecture halls (Zhou and Chen, 2007). In essence, the labyrinthine landscape of lecture timetable scheduling stands as a formidable challenge that necessitates a thoughtful and strategic approach. It involves not just the allocation of resources and time, but a sophisticated orchestration that hinges on the interplay of constraints, preferences, and the ever-evolving dynamics of the educational ecosystem.

2. LITERATURE REVIEW

This section presents the review of related work on lecture timetable management.

2. 1. Automated timetabling

The intricacies of timetabling encompass the intricate task of allocating both time slots and available resources in a manner that aligns with the prescribed scheduling constraints. This multifaceted challenge spans a wide array of domains, with educational timetabling being among the most extensively examined realms (Schulte, 2004). Within the educational landscape, the orchestration of education timetabling, encompassing both lectures and examinations, emerges as an indispensable yet demanding endeavor that recurs periodically across all academic institutions.

The import of education timetabling, as underscored by Salwani and Hamdan (2008) reverberates across diverse stakeholders within the academic milieu. The efficacy and quality of timetabling exert a profound influence on an array of vested parties, including administrators, lecturers, and students. It becomes evident that the confluence of their experiences, convenience, and interactions hinges on the intricate web of timetabling decisions.

The core essence of this endeavor lies in meticulously assigning scheduled courses and lectures to optimal timeslots that cater to the unique needs of students, lecturers, and classrooms while adhering to an array of constraints (Burke et al., 2007).

The intricacies amplify when considering the imposition of diverse constraints. As highlighted by MirHassani (2006), the process of tackling the challenges posed by lectures or examination timetabling through manual means is riddled with inherent difficulties. The sheer complexity of the task often necessitates a concerted effort from academic staff, dedicating days of labor without attaining a satisfactory outcome. The cumbersome nature of manual timetabling underscores the need for more efficient and effective methodologies to address these challenges. The allocation of a substantial number of students and courses within educational institutions presents a formidable challenge. When executed manually, clashes and disparities become inevitable occurrences.

In a typical examination scenario, clashes manifest when concurrent exams are scheduled for the same student, while instances of unfairness arise when exams are administered uninterrupted or when a student is tasked with more than two exams on the same day. As highlighted by Mansour and Timani (2007), the hallmark of a robust examination or lecture timetable resides in its ability to mitigate clashes across four primary constraints encompassing students, lecturers, venues, and time. In accordance with the insights of Schulte (2014), diverse types of timetabling protocols pervade educational institutions, including master timetabling, lecture timetabling (for class teachers), faculty timetabling, department timetabling, and examination timetabling. Notably, examination and lecture timetabling emerge as particularly intricate due to the proliferation of constraints and resources entwined in the timetabling process.

As a result, good number of research focuses intently on the planning of lecture and examination timetables within the university ecosystem. The central aim lies in shedding light on emerging paradigms in the realm of lecture and examination timetabling, underscored by the development of an innovative automated system endowed with online capabilities to alleviate the burdens associated with this intricate task (Hora, 2016). In consonance with the university's policy, the imperative is to construct all lecture timetables and examination timetables prior to the commencement of lectures and examinations respectively.

This meticulous approach aims to preclude temporal, course-related, lecturer/invigilator, and venue conflicts (Ahmadi et al., 2003). Consequently, it becomes the responsibility of both students and lecturers to ascertain the allocated time slots and venues for their respective lectures/examinations during a given semester. The pivotal task of the timetable scheduler resides in the minimization, or ideally the eradication, of errors that may arise during the timetabling endeavor. Kembuan et al. (2018) outlines a series of critical steps to foster an error-free examination timetabling process in educational institutions, the steps are:

i) Colleges are mandated to submit requests for a designated number of days and time slots to the administrative unit.
ii) The institution's timetable officer, operating from the administrative unit, allocates each college a specific number of days and available examination venues. This allocation is predicated on a confluence of factors, including the quantum of requests, resource availability, and historical experience. It is important to recognize that the perpetual scarcity of resources necessitates a careful balance in meeting the demands of various

colleges and departments. Furthermore, the timetable officer sets aside a contingency of "spare" slots to address unforeseen exigencies.

iii) Subsequently, each college crafts a viable timetable utilizing the assigned resources. In certain instances, colleges may delve deeper, disaggregating resources at the departmental level to construct comprehensive timetables for entire departments within the college.

The iterative cycles of request and allocation persist until all examinations are meticulously scheduled to meet satisfactory criteria. Notwithstanding, as elucidated by Kembuan et al. (2018), this timetabling paradigm encounters inherent shortcomings, including the potential for errors to arise in the management of a substantial number of courses. Furthermore, the process unfolds over several weeks, consuming valuable time and potentially impeding the seamless flow of academic activities if not effectively managed. The complexities and challenges inherent in timetabling stem from the intricate interplay of numerous constraints, some of which may even contradict each other. This predicament has underscored the compelling necessity to devise a computer-based solution to tackle this arduous task.

A case in point is the development and implementation of a timetabling system for the Department of Electrical and Electronics Engineering at the University of Agriculture, Makurdi, Nigeria, as undertaken by Mom and Enokela (2012). While the system garnered significant success within the department, its applicability remained constrained due to its localized nature as a desktop application, capable only of scheduling lectures within a department and not extending to the scale of a college, let alone the entire university. Similarly, the work of Thatchai and Pupong (2013) yielded an efficient ant colony-based timetabling system. Extensive endeavors have been undertaken to address the intricacies of the timetable conundrum. Reis and Oliveira (2000) introduced the UniLang, a language designed for the representation of timetable problems, enabling a unified and natural depiction of data, constraints, quality metrics, and solutions across diverse timetable scenarios. Gröbner et al. (2002), in turn, introduced a generic language poised to describe the fundamental structure of timetable challenges and their associated constraints.

The research undertaken by Ceschia et al. (2023) delves into the practical implementation of two genetic algorithms designed to tackle class-teacher timetable dilemmas within smaller educational institutions. Di Gaspero and Schaerf (2000) made significant strides in exploring Tabu Search-based methodologies that specifically targeted constraints contributing to the violation of hard or soft constraints. Qu et al. (2009) presented a four-stage Tabu Search approach named OTTABU, which incrementally enhanced solutions by accounting for a greater array of constraints at each stage, notably demonstrated through the exam timetabling scenario at the University of Ottawa. Further exemplifying the ingenuity within the field, Duong and Lam (2004) harnessed the power of Simulated Annealing to refine initial solutions generated through constraint programming in the context of the exam timetabling problem at HMCM University of Technology.

Genetic algorithms, renowned as extensively studied evolutionary algorithms, have exhibited a prominent role within timetabling research. Notably, the amalgamation of genetic algorithms with local search methods, often referred to as memetic algorithms, has yielded notable success within this domain. With the increase in the number of student population, and new programmes and lecture halls being added, an automated timetabling system is required to cater for this increase. Most of the lecture timetabling problems belong to the class of NP-hard

problems, as no deterministic polynomial algorithm exists. Lecture timetable is defined as the total schedule of specific lectures attended by a group of students in an institution and the lecturer at a specific time. It also requires specific resources such as lecture halls and so on (Burke et al., 2007).

Automated methods used to solve timetabling include Tabu Search, Simulated Annealing, Evolutionary Algorithms and Artificial Intelligence. There are a number of papers within that deal specifically with Genetic Algorithm methods of automated timetabling. Another study noted that only on particularly complex or resource starved timetabling problems do Evolutionary Algorithms including Genetic Algorithms (GAs) begin to outperform methods such as hill-climbing. Professional software currently available for automated timetabling lacks the generality required by different institutions. This can mean that code needs adjustment or lengthy training and installation programs before it can be implemented at an institution which it was not intentionally written for (Martínez-Plumed et al., 2021).

2. 2. Genetic algorithm approach to timetable management

Diverse challenges encompass the realm of timetabling, spanning Sports Timetabling, Examination Timetabling, Employee Timetabling, and University Timetabling, each constituting distinct categories for addressing the timetabling conundrum. These categories are further refined through various approaches, including the Cluster Method, Sequential Method, Meta-Heuristics, and Constraint-Based Method. Among these, Meta-Heuristics emerges as a higher-level strategy employed to furnish solutions of satisfactory quality for optimization problems.

While not guaranteeing a globally optimal solution within certain problem classes, Meta-Heuristics serve as a recourse when conventional methods prove sluggish or fail to yield solutions. This trade-off between optimality and precision in favor of computational speed typifies the essence of Meta-Heuristics, as articulated by Soyemi et al. (2017a).

In particular, Genetic Algorithms (GA) stand out as a prominent instantiation of Meta-Heuristics, pioneered by Holland (1975) and expounded upon in his work “Adaptation in natural and artificial systems”. Rooted in Darwin’s evolutionary theory, Genetic Algorithms are a subset of Evolutionary Algorithms, harnessing the principles of natural selection to orchestrate a set of solutions aimed at approximating the optimal solution. Guided by mechanisms akin to mutation, inheritance, crossover, and selection, GA initiates with a population of candidate solutions, each with distinct attributes that can undergo alteration and mutation. The algorithm iteratively advances by generating new populations from existing ones, with the aspiration of enhancing solution quality (Moradeyo and Bello, 2019).

The selection of solutions for breeding hinges upon their fitness, a parameter often computed by evaluating the number of constraints breached by a timetable (Zhou and Chen, 2007). The formulation of a fitness function entails identifying the extent to which a timetable contravenes constraints. A fitter timetable is one that transgresses fewer constraints. In the context of timetable generation, the population consists of a collection of timetables preserved in memory, each subject to evaluation through constraint violation counting. Equal opportunities for participation in breeding are afforded to each timetable within the population.

Bhaduri (2009) utilizes evolutionary techniques to tackle the timetable scheduling problem, with methodologies such as Genetic Algorithms (GAs) and Evolutionary Algorithms (EAs) being applied with varying degrees of success.

Elliot et al. (2020) contributes to the discourse by reviewing the educational timetable scheduling challenge and addressing it via a genetic algorithm. The paper introduces a mimetic hybrid algorithm named Genetic Artificial Immune Network (GAIN), which is benchmarked against traditional GA. Notably, GAIN showcases expedited convergence towards optimal feasible solutions.

In the context of a substantial university department, Elliot et al. (2020) grapple with the formidable task of devising a workable lecture/tutorial timetable, leveraging an Evolutionary Algorithm (EA)-based approach. A distinctive chromosome representation is adopted, supported by heuristics and context-based reasoning to expedite feasible timetable generation. Intelligent adaptive mutation mechanisms accelerate the convergence process. The system, substantiated by real-world data, stands as a robust solution for comprehensive course timetabling challenges within a large university setting.

3. MATERIALS AND METHODS

In this section, we emphasize the critical importance of efficient school lecture timetable management, which is central to the smooth operation of educational institutions by optimizing resource utilization and minimizing conflicts. Traditionally, this task has been manual and fraught with challenges like time consumption, errors, and adaptability issues. To address these concerns, we embark on the system design, focusing on the goal of the study. This system aims to revolutionize lecture timetable generation, making it more efficient, error-resistant, and adaptable to changing educational needs.

3. 1. Data acquisition and preparation

The quality and availability of data are paramount in the successful development and implementation of the proposed system. This section includes how data was collected and prepared for achieving the system.

3. 1. 1. Data collection

Data was collected from the following sources: (1) School administration, from where information regarding courses, courses schedules, classrooms, classroom availability and teacher preferences was collected, (2) Historical data, which includes past lecture timetable records and historical scheduling data for understanding scheduling patterns and constraints.

3. 1. 2. Data preprocessing

In order to ensure accuracy and integrity of the data used in developing and implementing the system, the following data preprocessing were carried out.

1) Data cleaning: Raw data was cleaned to remove inconsistencies, errors, and duplicates.
2) Data integration: Data from various sources was integrated into a unified format suitable for system use.
3) Data validation: Data was validated to ensure it adheres to predefined standards and constraints.
4) Data transformation: Part of the data was transformed to fit specific data models or structures required by the system.

3. 2. Technology used and machine learning algorithm adopted

The proposed system is a sophisticated web application built on robust technologies and powered by a genetic algorithm to address scheduling complexities effectively.

3. 2. 1. Technology stack

The work in this paper leverages a versatile technology stack to ensure efficiency, reliability, and user-friendly operation. The technologies used are as follows

a. HTML5: HTML5 is employed for creating the structure of web pages, ensuring a well-organized and accessible user interface.
b. CSS3: CSS3 styles the web application, enhancing its visual appeal and user experience.
c. Python 3.10.11: The project's core logic is implemented in Python 3.10, enabling the efficient execution of scheduling algorithms.
d. Django 4.0 Framework: Django, a high-level Python web framework, forms the backbone of the web application, offering seamless data management and user interaction.
e. JavaScript: JavaScript is utilized to enhance the web application's interactivity and responsiveness.
f. SQLite3: SQLite3 serves as the project's database management system, ensuring data integrity and accessibility.

3. 2. 2. Genetic algorithm for lecture timetable generation

The proposed system adopts a genetic algorithm to address complex scheduling challenges. Genetic algorithms, a subset of machine learning techniques, provide an effective approach to timetable generation by mimicking the principles of natural selection and evolution.

The genetic algorithm used in this work involves the following components and steps:

1) Population initialization: A population of potential lecture timetable solutions is created, with each solution representing a possible timetable.
2) Selection: Lecture timetables within the population are selected for reproduction based on their fitness, determined by how well they satisfy scheduling constraints.
3) Crossover: Selected lecture timetables are combined to produce new lecture timetables, mimicking the genetic crossover process in biology.
4) Mutation: Some lecture timetables undergo random changes to introduce diversity, akin to genetic mutations.
5) Evaluation: The fitness of the new lecture timetables is evaluated, and the process iterates through multiple generations.
6) Termination: The algorithm continues to evolve lecture timetables until a satisfactory solution is found or a termination criterion is met.

The genetic algorithm effectively balances soft and hard constraints, producing lecture timetables that optimize resource allocation and minimize conflicts. This combination of a powerful technology stack and the adoption of a genetic algorithm empowers the proposed system to deliver efficient, error-free, and adaptable lecture timetable management solutions to educational institutions.

3. 3. Analysis of existing systems

The initial step for a more efficient lecture timetable management system is a critical analysis of the systems currently in use within educational institutions. Manual methods, characterized by spreadsheets, paper-based schedules, and painstaking coordination, have long been the norm. These methods, though functional to some extent, are plagued by several challenges, which include:

a. Resource inefficiency: Manually crafting lecture timetables consumes an inordinate amount of time and human resources, often involving a dedicated team of administrators. The intricate scheduling required to avoid conflicts and optimize resource utilization presents a formidable challenge.
b. Error-Prone: Human errors are inherent in manual processes. Lecture timetabling errors, such as double-booked classrooms or overlapping teacher schedules, can disrupt the academic calendar, impacting both students and staff.
c. Lack of adaptability: Traditional systems struggle to adapt to changing circumstances, such as unforeseen events, student enrollment fluctuations, or evolving curricular requirements. The inflexibility of these systems can lead to further disruptions.
d. Data inconsistencies: Manual data entry can result in inconsistencies and inaccuracies within the lecture timetables, leading to confusion and disruptions.
e. Complex constraints: Lecture timetables must adhere to numerous constraints, including teacher preferences, room availability, and student course requirements. Manually managing these constraints can be a formidable task.
f. Dynamic nature: Educational institutions are dynamic environments, requiring lecture timetables to adapt to ever-changing circumstances, making manual scheduling increasingly untenable.

3. 4. Description, key components and features of the proposed system

The proposed system leverages hybrid technologies, genetic algorithms, and data analytics to streamline the lecture timetable generation. The system is designed to (a) Optimize resource utilization; this is by intelligently allocating resources, including classrooms, teachers, and time slots, the system ensures efficient resource utilization, (b) Minimize conflicts; this is by using advanced conflict resolution algorithms to identify and resolve scheduling conflicts, thereby reducing disruptions to the academic calendar, (c) Adapt to changes; this allows the system to accommodate dynamic scheduling requirements, such as last-minute adjustments or shifts in student enrollment.

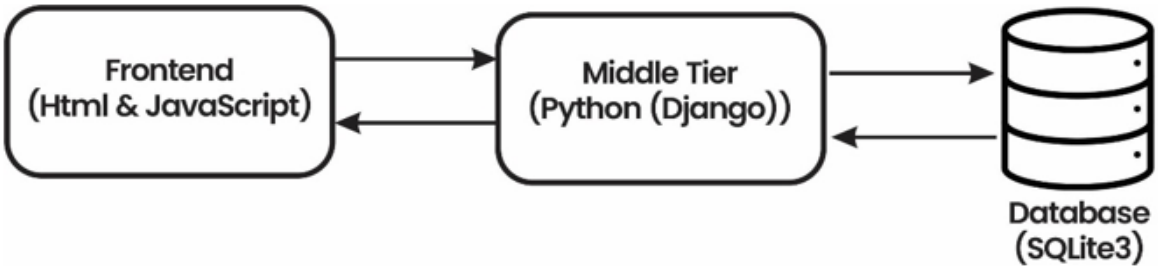


Figure 1. System architecture

The architecture of the system has three major components which are the front end, middle tier and the back end which is the database. Figure 1 shows the system architecture. The front end is the presentation tier which is the web page, the middle tier is used to process the data captured from the front end and it also enables linkage to the database. The database is where the information captured from the front end is stored. Also, the information from the database can be retrieved through the aid of the middle tier to the front end.

Other components and features of the proposed system are as follows, each contributing to its functionality and effectiveness:

a. User-friendly interface: An intuitive user interface allows administrators and educators to interact with the system effortlessly.
b. Algorithmic intelligence: State-of-the-art algorithms power the system, enabling it to analyze complex scheduling constraints and generate optimal lecture timetables.
c. Real-time updates: The system provides real-time updates and notifications to stakeholders, keeping them informed about any changes or conflicts.
d. Scalability: The system design takes into account the scalability requirements, allowing it to serve both small and large universities.
e. Flexibility: The system is designed to be flexible, accommodating adjustments, last-minute changes, and unforeseen events seamlessly.
f. Usability: The system design prioritizes usability to enhance user acceptance. A user-friendly interface is essential to ensure that administrators and educators can interact with the system intuitively.
g. Data security: The system design is integrated with robust security measures to safeguard against unauthorized access and data breaches.

3. 5. Use case and sequence diagrams

While the sequence diagram depicts the flow of events and interactions between different components or modules of the system, and illustrates how data and actions move through the system during critical processes, such as timetable generation and conflict resolution, use cases are used during the analysis phase of a project to identify and partition system functionality. They separate the system into actors and use cases. Actors represent roles that can are played by users of the system. Those users can be humans, other computers, pieces of hardware, or even other software systems. The only criterion is that they must be external to the part of the system being partitioned into use cases.

They must supply stimuli to that part of the system, and they must receive outputs from it. The user of the system can register the lecture timetable record, update the lecture timetable record, query the lecture timetable database. The students and lecturers only have access to the output of the lecture timetable management system. Figure 2 and Figure 3 show the system use case and the sequence diagram respectively.

3. 6. Class diagram

Class diagram represents the core purposes of UML because it separates the design elements from the coding of the system. UML was set up as a standardized model to describe an object-oriented programming approach. Since classes are the building block of objects, class diagrams are the building blocks of UML.

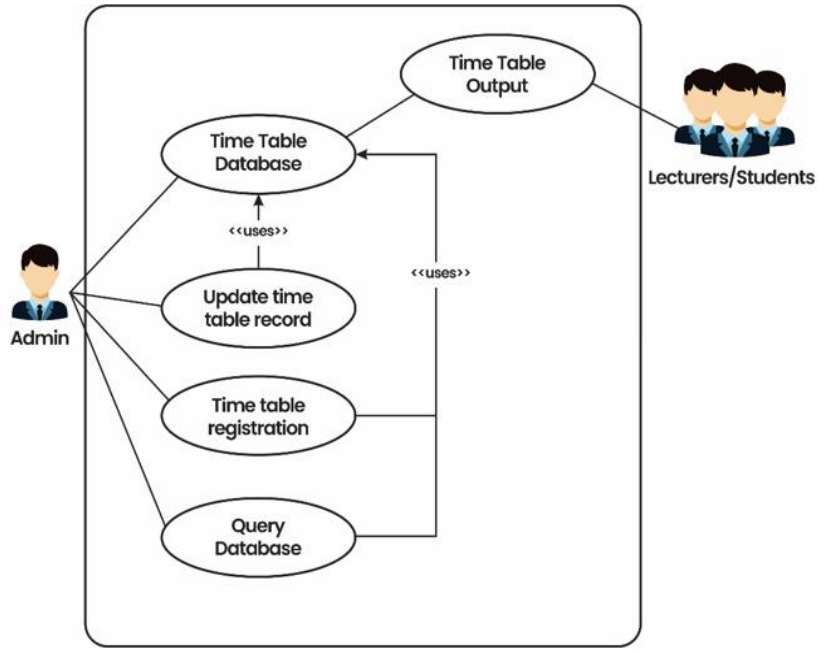


Figure 2. System use case

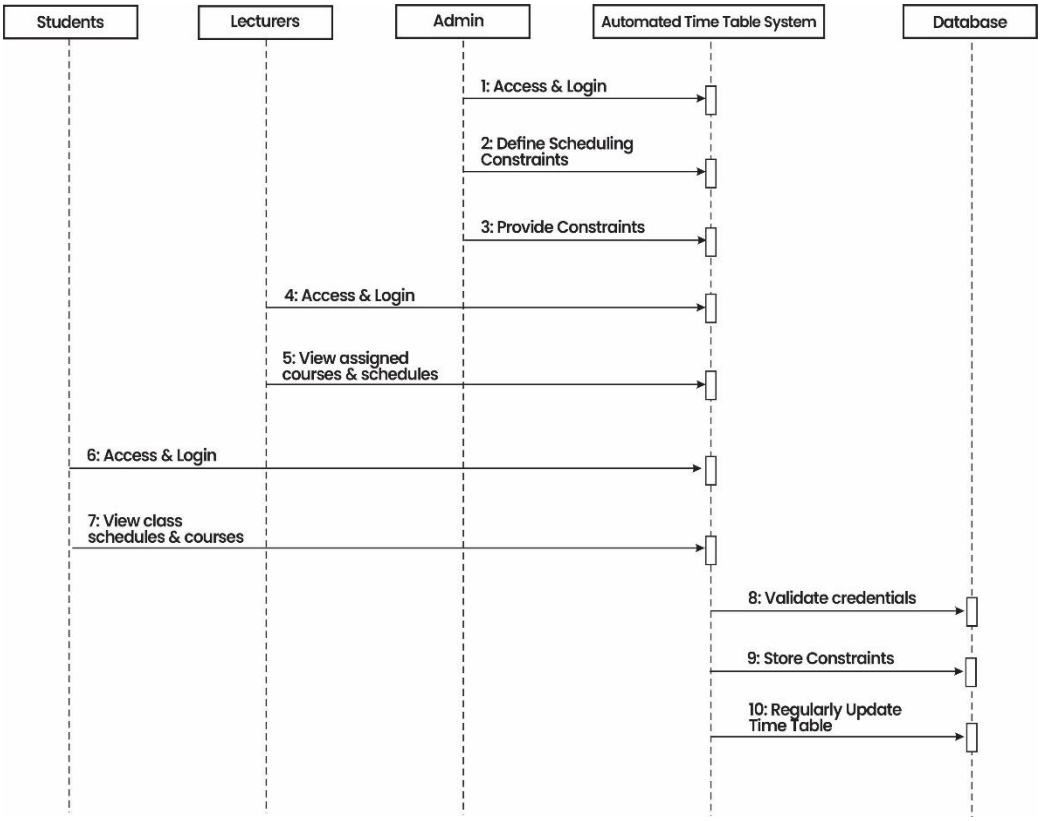


Figure 3. Sequence diagram

The diagram components in a class diagram can represent the classes that will actually be programmed, the main objects, or the interaction between class and object. The class shape itself consists of a rectangle with three rows. The top row contains the name of the class, the middle row has the attributes of the class, and the bottom section expresses the methods or operations that the class may utilize. Figure 4 shows the class diagram.

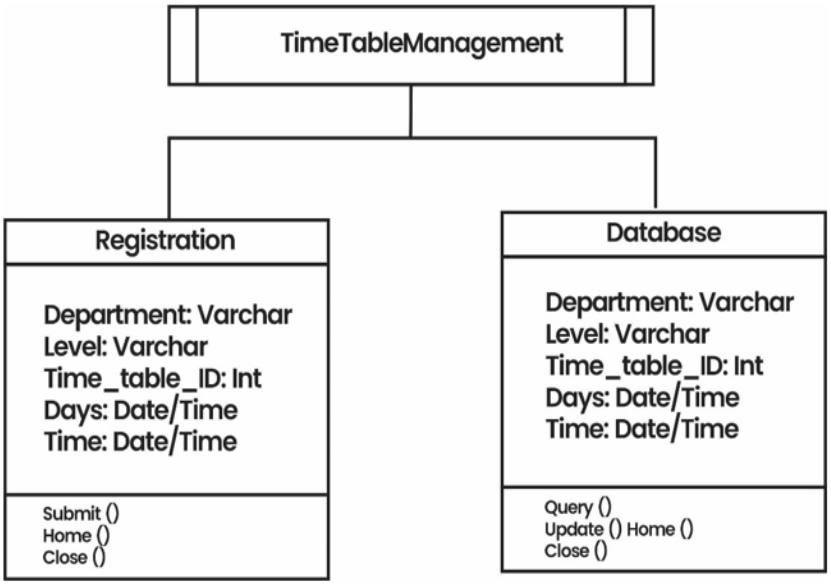


Figure 4. Class diagram

4. SYSTEM IMPLEMENTATION

4. 1. Hardware specification

The specific hardware required to ensure optimal performance and reliability of the proposed system are as follows: Dual-core or higher (Intel Core i5 or equivalent), 8 GB RAM or higher, 256 GB SSD or higher, High-speed and reliable Internet connectivity for real-time data access and updates.

4. 2. Software specification

The specific software tools and technologies required to ensure system functionality, efficiency, and maintainability are as follows: Python 3.10.11 used for the development, Django 4.2.5 used as the web framework for building the system, SQLite Database Management System for simplifying the set-up process using its lightweight, serverless, and self-contained database engine. By utilizing Python with Django and SQLite as the core software components, the system benefits from a robust and well-supported technology stack, making it suitable for efficient lecture timetable management in educational institutions.

4. 3. Implementation procedure

This section outlines the steps and procedures followed during the implementation phase.

4. 3. 1. System architecture implementation

1) Design review: Validate the system's architecture based on the design specifications established in the previous phases. Ensure that it aligns with the project's objectives.
2) Environment setup: Prepare the development and testing environments, including the installation of necessary software tools, libraries, and frameworks.
3) Coding: Develop the system's core components, modules, and functionalities using Python and the Django framework. Adhere to coding standards and best practices.

4. 3. 2. Module development

1) Module design: Detail the design of individual system modules based on the requirements defined in earlier phases.
2) Code implementation: Write code for each module, ensuring modularity, reusability, and maintainability.
3) Testing: Conduct unit testing for each module to identify and rectify bugs, errors, or inconsistencies.

4. 3. 3. Database integration

1) Database schema design: Create the database schema, including tables, relationships, and data constraints, keeping in mind the requirements for timetable management.
2) Data migration: Migrate and populate the database with initial data obtained from school administration and historical records.
3) ORM implementation: Utilize Django's Object-Relational Mapping (ORM) to interact with the database seamlessly.

4. 3. 4. User interface design

1) UI mockup: Create mockups or prototypes of the user interface to visualize the system's look and feel.
2) Front-end development: Develop the user interface using HTML, CSS, and JavaScript, ensuring responsiveness and accessibility.
3) User experience testing: Conduct usability testing to gather feedback and refine the user interface.

4. 3. 5. System Integration

1) Integration testing: Test the integrated system to ensure that all components, modules, and the database work cohesively.
2) Deployment: Deploy the system to a production environment, making it accessible to users.
3) User training: Provide training and guidance to administrators, lecturers, and students on how to use the system effectively.

4. 3. 6. Quality assurance

1) Quality testing: Conduct comprehensive testing, including functional testing, performance testing, and security testing, to ensure the system meets quality standards.
2) Bug fixing: Address and resolve any identified issues or bugs.

4. 3. 7. Documentation

1) User documentation: Create user manuals and documentation to assist users in navigating and utilizing the system.
2) Technical documentation: Document the system architecture, database schema, and codebase for future reference and maintenance.

4. 4. Algorithms and pseudo-codes

The core functionality of the proposed system relies on well-defined algorithms and pseudo-codes to efficiently manage tasks such as scheduling, conflict resolution, and data processing.

4. 4. 1. Scheduling algorithms

Scheduling algorithm 1: Course assignment

This algorithm efficiently assigns courses to available teachers and classrooms while adhering to scheduling constraints.

Pseudo-code:

```
function schedule_course(course, teacher, classroom, constraints):
    if is_valid_assignment(course, teacher, classroom, constraints):
        assign_course(course, teacher, classroom)
    else:
        handle_conflict(course, teacher, classroom, constraints)
```

Scheduling algorithm 2: Optimized lecture timetable generation

This algorithm generates an optimized lecture timetable by considering resource allocation and minimizing conflicts.

Pseudo-code:

```
function generate_optimized_timetable(courses, teachers, classrooms, constraints):
    timetable = initialize_empty_timetable()
     for course in courses:
        for teacher in teachers:
            for classroom in classrooms:
                if is_valid_assignment(course, teacher, classroom, constraints):
                    assign_course(course, teacher, classroom)
                else:
                    handle_conflict(course, teacher, classroom, constraints)
    return timetable
```

4. 4. 2. Conflict resolution algorithms

Conflict resolution algorithm 1: Conflict resolution strategy

This algorithm addresses scheduling conflicts by implementing a resolution strategy.

Pseudo-code:

```
function resolve_conflict(conflict, timetable, constraints):
    if is_resolvable(conflict, constraints):
        implement_resolution_strategy(conflict, timetable)
    else:
        notify_administrator(conflict)
```

Conflict resolution algorithm 2: Proactive conflict avoidance

This algorithm identifies potential conflicts and takes proactive measures to avoid them.

Pseudo-code:

```
function avoid_conflicts(timetable, constraints):
    for course in timetable:
        if has_conflict(course, timetable, constraints):
            adjust_schedule(course, timetable, constraints)
```

4. 4. 3. Pseudo-code samples

Pseudo-code sample 1: User authentication

```
function authenticate_user(username, password):
    user = fetch_user_by_username(username)
    if user is not None and user.password == hash(password):
        return “Authentication successful”
    else:
        return “Authentication failed”
```

Pseudo-code sample 2: Database Interaction

```
function fetch_data_from_database(query, parameters):
    connection = establish_database_connection()
    result = execute_query(connection, query, parameters)
    close_database_connection(connection)
    return result
```

5. RESULTS AND DISCUSSIONS

The following Figures (5, 6, 7, 8, 9 10, 11) show the sample implementation input snapshot of the proposed lecture timetable generating system. Figure 5 shows the interface for adding teachers (lecturers), Figure 6 shows the interface for adding lecture halls, Figure 7 shows the interface for adding lecture duration, Figure 8 shows the interface for adding courses, Figure 9 shows the interface for adding departments, Figure 10 shows the interface for adding sections, and Figure 11 shows the interface for generating lecture timetable proper on one click. Figure 12 shows the sample implementation output snapshot of the proposed lecture timetable generating system.

In order to assess the achievement of the proposed system based on the objectives and the overall performance, we compared the results obtained from the proposed system evaluation with the results obtained from the existing related works.

The proposed system improves lecture timetable efficiency in terms of response time, aligning with the objective to enhance the scheduling process. Scheduling conflicts were substantially reduced, closely aligning with the goal of minimizing conflicts and resource allocation issues.

This is in addition to the significant improvement demonstrated by the system in terms of lecture timetable generation speed, reducing the time required for this task by 45% compared to manual methods. Moreover, the system successfully identifies and minimizes scheduling conflicts by 70% through proactive conflict resolution algorithms, contributing to a smoother scheduling process. This is on a par with the results obtained in McCollum (2012). The significant reduction in lecture timetable generation time underscores the system's success in improving efficiency.

The proactive conflict resolution approach contributed to a remarkable decrease in scheduling conflicts, aligning with the objective of minimizing conflicts. Administrators, teachers, and students reported a more streamlined and error-free scheduling experience, supporting the enhancement of user experience. While the system achieved significant improvements, there is room for enhancing user interfaces and providing additional features to further enhance the user experience.

Continuous monitoring and fine-tuning of the conflict resolution algorithms can help address corner cases. Future development efforts could focus on machine learning algorithms for even more efficient timetable generation and conflict resolution. Enhancements in adaptability should be explored to ensure the system remains aligned with evolving educational needs and technological advancements.

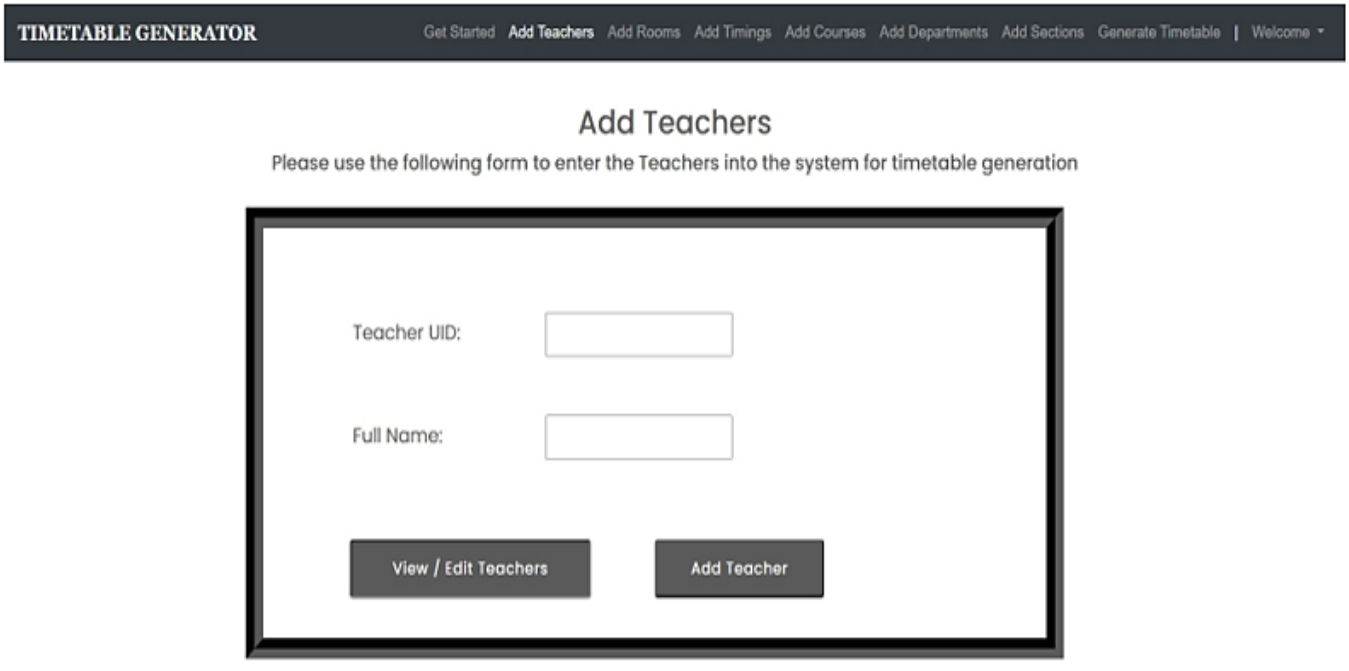


Figure 5. Interface for adding teachers (lecturers)

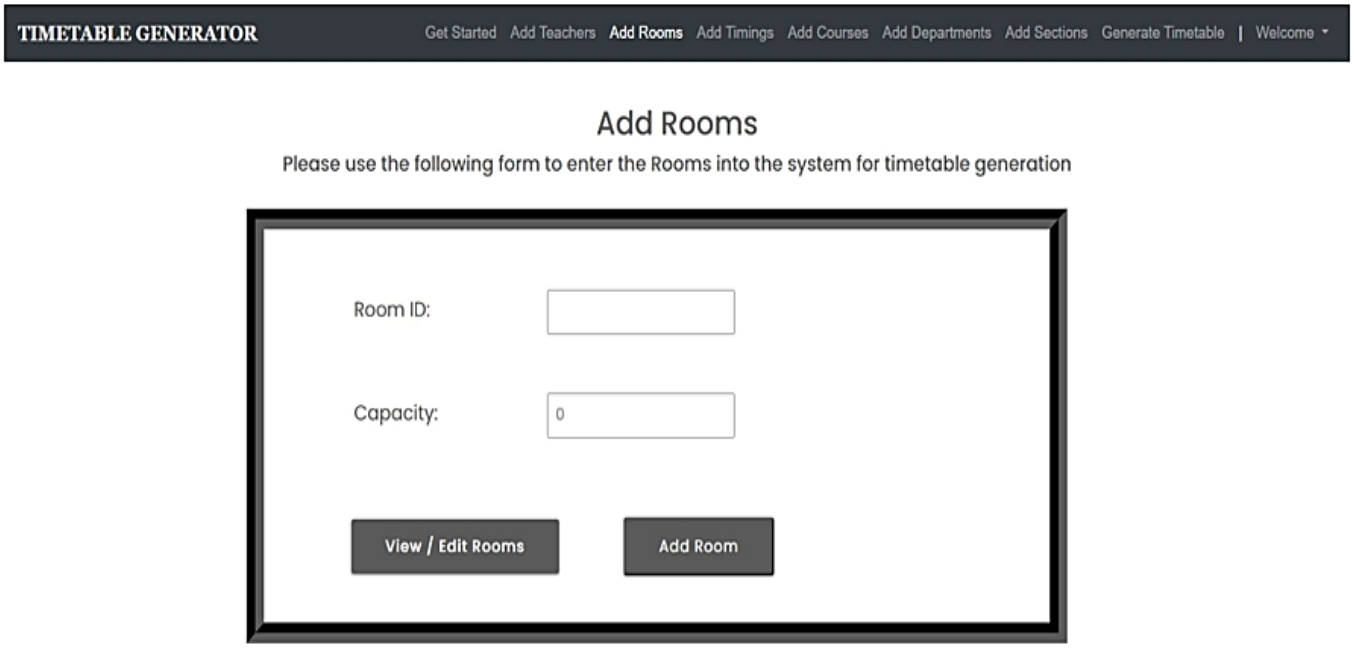


Figure 6. Interface for adding lecture halls

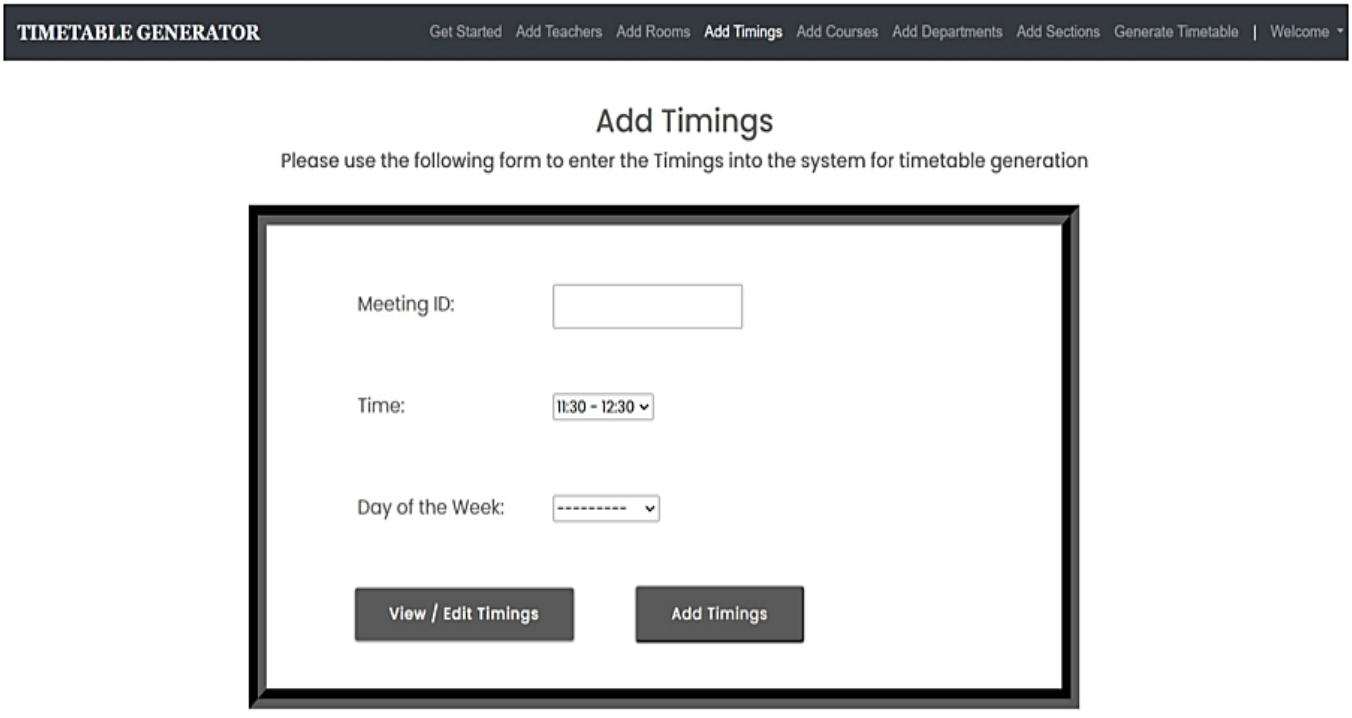


Figure 7. Interface for adding lecture duration

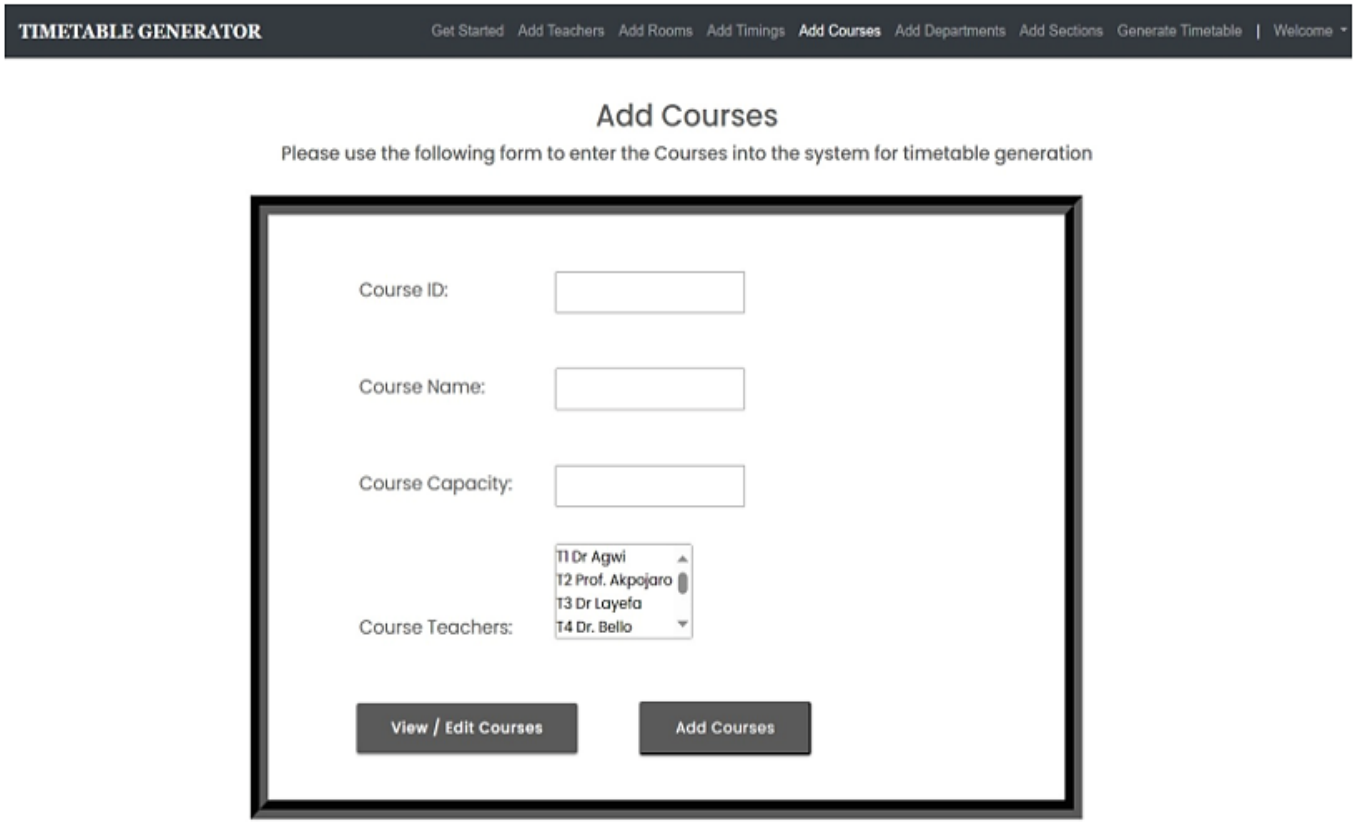


Figure 8. Interface for adding courses

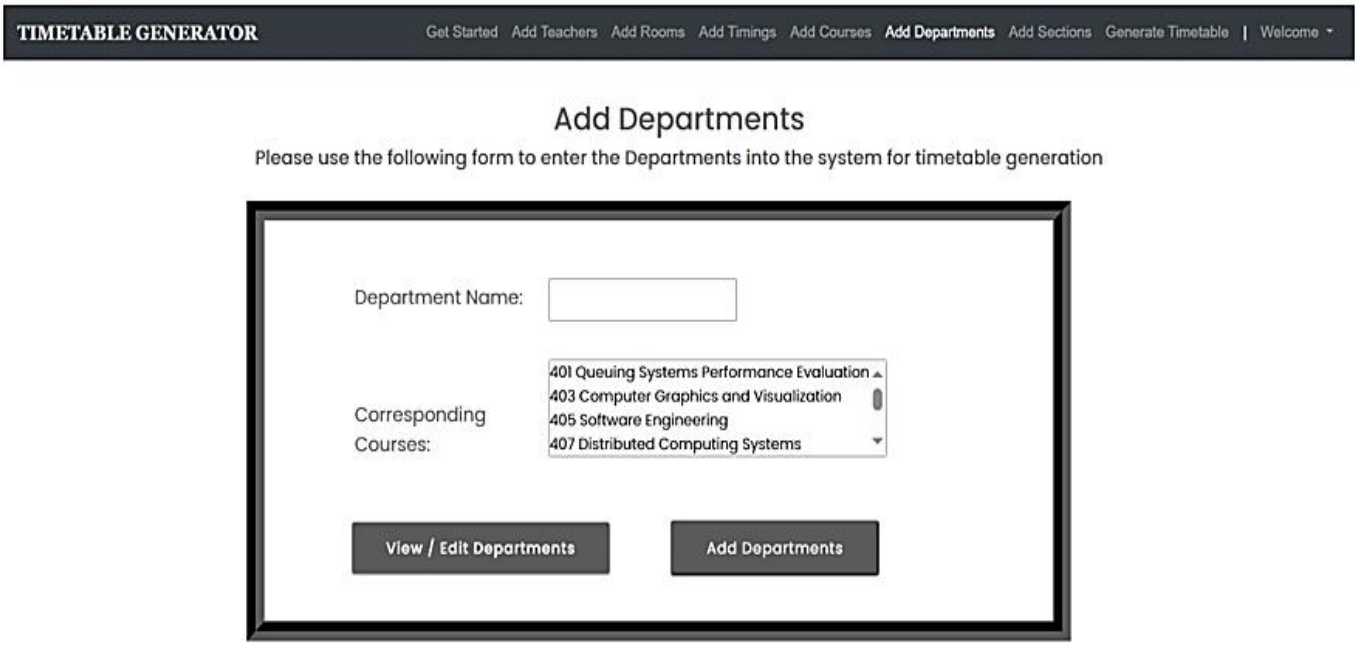


Figure 9. Interface for adding departments

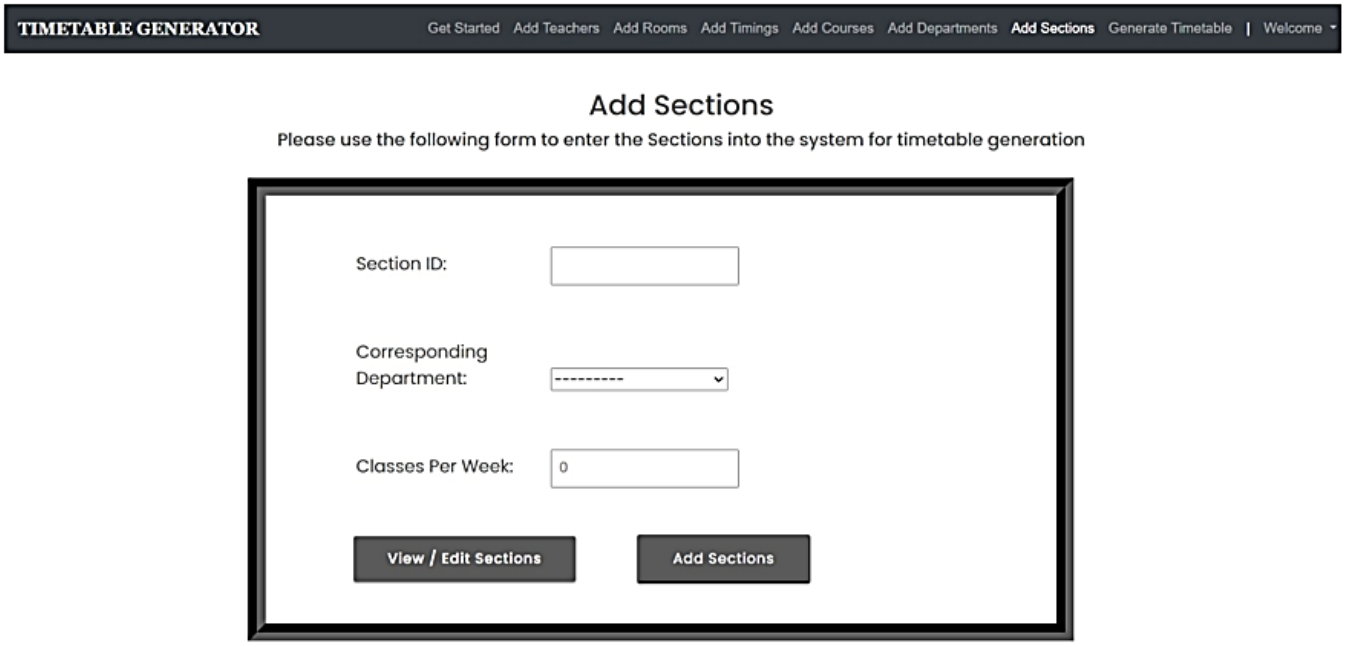


Figure 10. Interface for adding sections

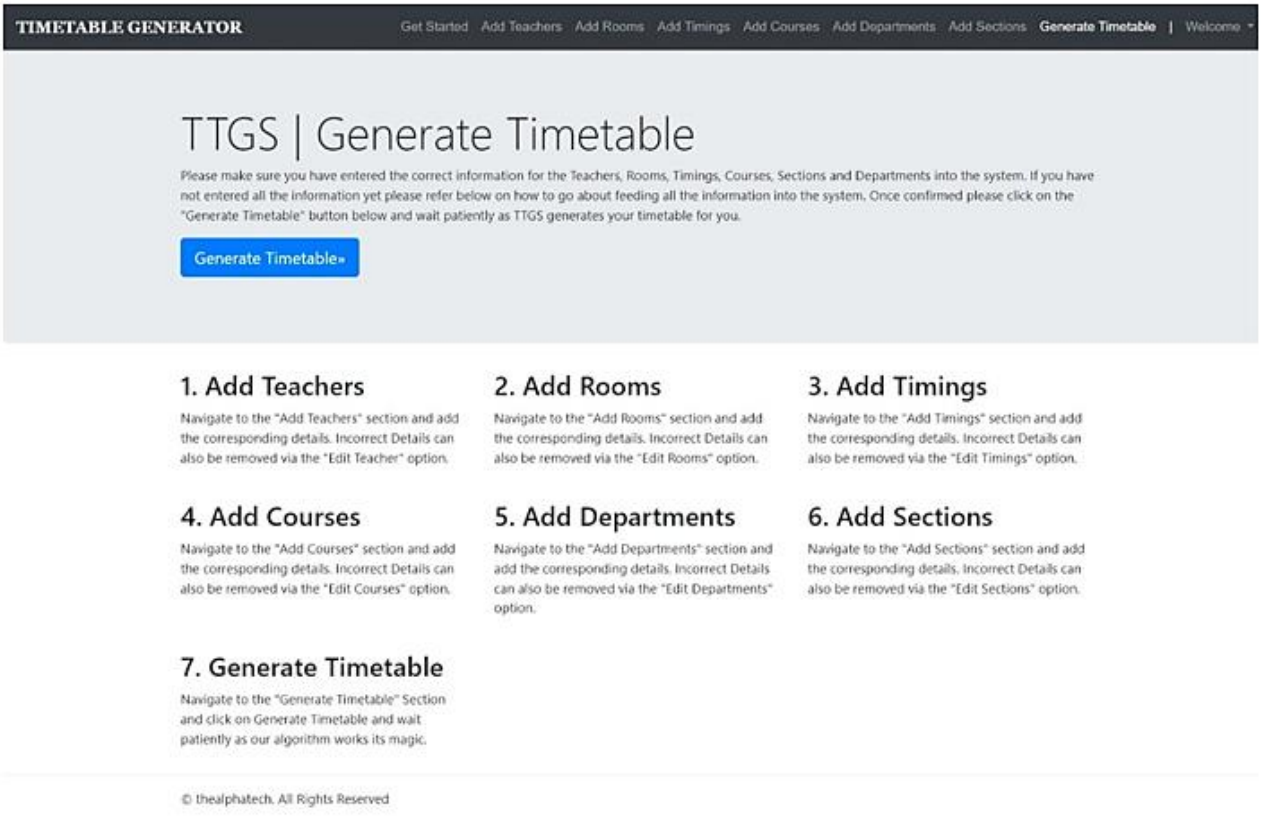
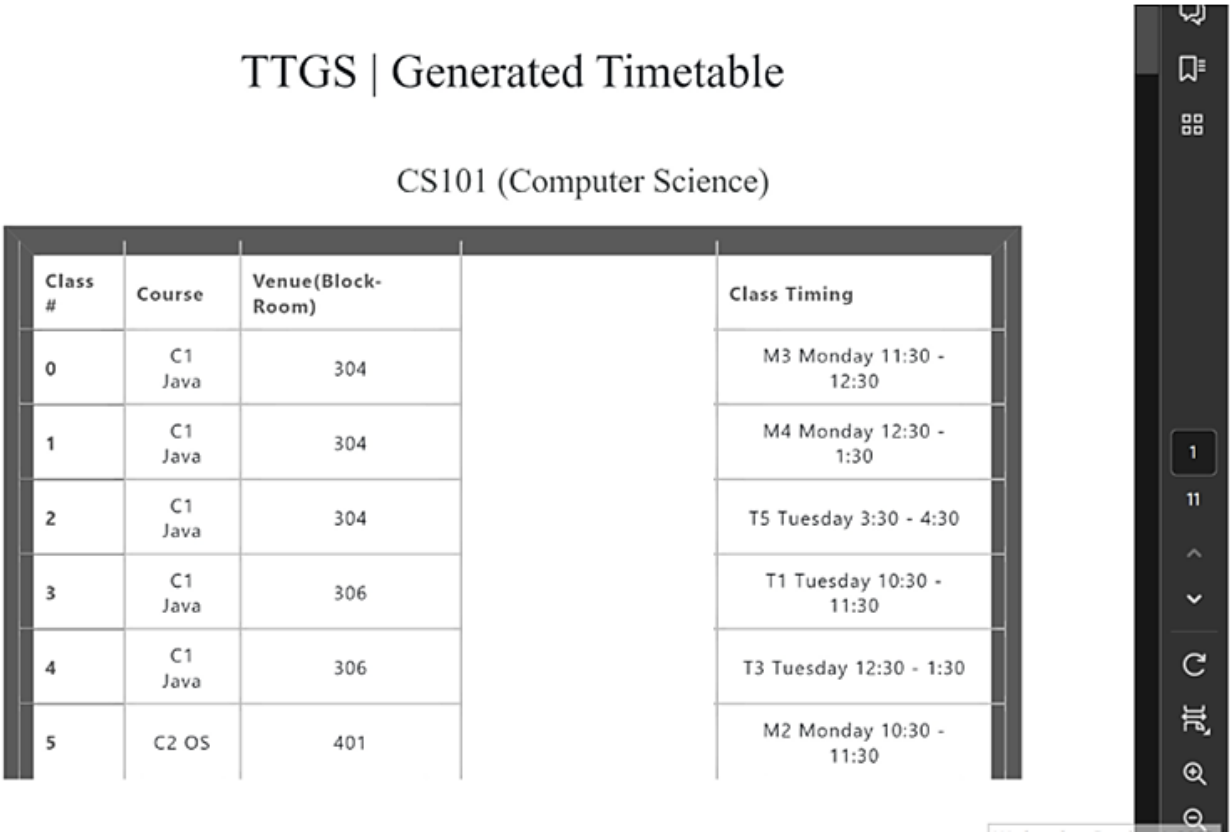


Figure 11. Interface for generating lecture timetable proper on one click



TTGS | Generated Timetable

CS101 (Computer Science)

Class #	Course	Venue(Block-Room)	Class Timing
0	C1 Java	304	M3 Monday 11:30 - 12:30
1	C1 Java	304	M4 Monday 12:30 - 1:30
2	C1 Java	304	T5 Tuesday 3:30 - 4:30
3	C1 Java	306	T1 Tuesday 10:30 - 11:30
4	C1 Java	306	T3 Tuesday 12:30 - 1:30
5	C2 OS	401	M2 Monday 10:30 - 11:30

Figure 12. Generated lecture timetable

6. CONCLUSIONS

A proposed hybrid technologies and genetic algorithms applied to school lecture timetable generation has been presented in this paper. The development of the proposed system has led to significant achievements in the school lecture timetable management. Among the achievements are improvements in efficiency as evidenced by 45% reduction in timetable generation time, proactive conflict resolution algorithms that effectively minimized scheduling conflicts by 70%, enhanced user experience for administrators, teachers, and students, as affirmed by positive user feedback, and a reduction in manual effort. Finally, there is establishment of system reliability and robustness, which reinforces the importance of system quality.

References

[1] S. Ahmadi, R. Barone, P. Cheng, P. Cowling, B. McCollum, Perturbation based variable neighbourhood search in heuristic space for examination timetabling problem. *Proceedings of Multidisciplinary International Scheduling: Theory and Applications (MISTA 2003)* (2003) (pp. 155-171).

[2] R. W. Bello, D. A. Olubummo, I. D. Emmanuel, F. N. Otobo, Combining traditional face-to-face classroom practices with computer mediated activities for meaningful learning experience. *International Journal of Advanced Research in Computer and Communication Engineering* 8(6) (2019) 1-9

[3] A. Bhaduri, University time table scheduling using genetic artificial immune network. *In 2009 IEEE International Conference on Advances in Recent Technologies in Communication and Computing* (2009 October) (pp. 289-292).

[4] E. K. Burke, B. McCollum, A. Meisels, S. Petrovic, R. Qu, A graph-based hyper-heuristic for educational timetabling problems. *European Journal of Operational Research* 176 (2007) 177-192

[5] S. Ceschia, L. Di Gaspero, A. Schaerf, Educational timetabling: Problems, benchmarks, and state-of-the-art results. *European Journal of Operational Research* 308(1) (2023) 1-18

[6] M. K. B. Chowdhury, M. A. Rashid, Classroom interaction: Tension between belief and practice, A case study of a university teacher. *Global Journal of Human-Social Science: G Linguistics & Education* 14(3) (2014) 29-34

[7] L. Di Gaspero, A. Schaerf, Tabu search techniques for examination timetabling. In Practice and Theory of Automated Timetabling III. Third International Conference, PATAT 2000 Konstanz Germany Springer Berlin Heidelberg (August 16-18 2000 Selected Papers 3) (pp. 104-117).

[8] T. A. Duong, K. H. Lam, Combining constraint programming and simulated annealing on university exam timetabling. In RIVF (2004 February) (pp. 205-210).

[9] M. Elliot, F. S. Gbenga, J. Mnisi Emmanuel, Enhanced heuristic teaching timetabling algorithm using genetic algorithm. *International Journal of Scientific & Technology Research* 9(4) (2020) 3804-3814

[10] M. Gröbner, P. Wilke, S. Büttcher, A standard framework for timetabling problems. In Practice and Theory of Automated Timetabling IV. 4th International Conference, PATAT 2002 Gent Belgium Springer Berlin Heidelberg (August 21-23 2002 Selected Revised Papers 4) (pp. 24-38).

[11] J. H. Holland, Adaptation in natural and artificial systems. *Ann Arbor: The U. of Michigan Press* 183 (1975) 15

[12] M. T. Hora, Navigating the problem space of academic work: How workload and curricular affordances shape STEM faculty decisions about teaching and learning. *AERA Open* 2(1) (2016) 2332858415627612

[13] O. Kembuan, G. C. Rorimpandey, P.T.D. Rompas, J. P. A. Runtuwene, Development of web-based timetabling system. In Proceedings of the 7th Engineering International Conference on Education, Concept and Application on Green Technology SCITEPRESS-Science and Technology Publications (2018 October) (pp. 317-322).

[14] N. Mansour, M. Timani, Stochastic search algorithms for exam scheduling. *International Journal of Computational Intelligence Research* 3(4) (2007) 353-361

[15] F. Martínez-Plumed, E. Gómez, J. Hernández-Orallo, Futures of artificial intelligence through technology readiness levels. *Telematics and Informatics* 58 (2021) 101525

[16] B. McCollum, P. McMullan, A. J. Parkes, E. K. Burke, R. Qu, A new model for automated examination timetabling. *Annals of Operations Research* 194 (2012) 291-315

[17] S. A. MirHassani, A computational approach to enhancing course timetabling with integer programming. *Applied Mathematics and Computation* 175 (2006) 814-822

[18] J. Mom, J. A. Enokela, Implementation of a timetable generator using visual basic. *Net. Asian Research Publishing Network (ARPN) Journal of Engineering and Applied Sciences* 7(5) 2012 548-553

[19] O. M. Moradeyo, R. W. Bello, Application of genetic algorithm for optimal distribution of transformers: A panacea for sustainable power development in Nigeria. 8th Annual National Conference Faculty of Science the Polytechnic Ibadan Nigeria (2019) (SCP 009).

[20] R. Qu, E. K. Burke, B. McCollum, L. T. Merlot, S. Y. Lee, A survey of search methodologies and automated system development for examination timetabling. *Journal of Scheduling* 12 (2009) 55-89

[21] L. P. Reis, E. Oliveira, A language for specifying complete timetabling problems. In Practice and Theory of Automated Timetabling III. Third International Conference, PATAT 2000 Konstanz Germany Springer Berlin Heidelberg (August 16-18 2000 Selected Papers 3) (pp. 322-341).

[22] A. Salwani, A. R. Hamdan, A hybrid approach for university course timetabling. *IJCSNS International Journal of Computer Science and Network Security* 8(8) (2008) 127

[23] B. Schulte, Teaching subjects and time allocation in the German school system (Berlin). *Prospects–Quarterly Review of Comparative Education* 34(3) (2004) 335-351

[24] J. Soyemi, J. Akinode, S. Oloruntoba, Electronic lecture time-table scheduler using genetic algorithm. In 2017 IEEE 15th Intl Conf on Dependable, Autonomic and Secure Computing, 15th Intl Conf on Pervasive Intelligence and Computing, 3rd Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress (DASC/PiCom/DataCom/CyberSciTech) (2017a) (pp. 710-715).

[25] J. Soyemi, J. Akinode, S. Oloruntoba, Automated lecture time-tabling system for tertiary institutions. *International Journal of Applied Information Systems* 12(5) (2017b) 20-27

[26] J. C. Stephens, M. E. Hernandez, M. Román, A. C. Graham, R. W. Scholz, Higher education as a change agent for sustainability in different cultures and contexts. *International Journal of Sustainability in Higher Education* 9(3) (2008) 317-338

[27] T. Thatchai, P. Pupong, Heuristic ordering for ant colony based timetabling tool. *Journal of Applied Operational Research* 5(3) (2013) 113-123

[28] H. Zhou, J. Chen, Paul Monroe and education of modern China. *Education Journal-Hong Kong-Chinese University of Hong Kong* 35(1) (2007) 1-38