

**An Automated Class Scheduler for EARIST - College of Computing Studies using
Constraint Satisfaction Problem Algorithm**

A Thesis Submitted in Partial Fulfillment of the Requirements for the Course Capstone
Project and Research 2

Proponents:

Agpoon, Jason A.

Fadullo, Rome Angel M.

Parnaso, Marco Angelo B.

Santos, Shelina Denise F.

Bachelor of Science in Information Technology

4-A

November 2024

TABLE OF CONTENTS

The Problem and Its Background

<i>Introduction</i>	1
<i>Background of the Study</i>	2
<i>Project Context</i>	3
<i>Purpose and Description</i>	5
<i>Objectives of the Study</i>	6
<i>Conceptual Framework</i>	7
<i>Significance of the Study</i>	8
<i>Scope and Limitation of the Study</i>	9
<i>Definition of Terms</i>	10

Chapter 2

Review of Related Literatures

<i>Foreign Studies</i>	12
<i>Synthesis</i>	22

Chapter 3

Methodology

<i>Technological Background</i>	25
<i>Approaches and Techniques</i>	26
<i>Methodology</i>	27
<i>Requirement Analysis</i>	29
<i>Population, Sample Size, and Sampling Techniques</i>	30

Description of Respondents	31
Research Instrument	32
Data Gathering Procedures	32
Statistical Treatment of Data	33
System Requirements	34

Chapter 4

Results and Discussion	40
Functionality	43
Table 1	43
Interface Design	45
Table 2	45
Table 3	46
Table 4	49

Chapter 5

Conclusion	50
Recommedations	51

CHAPTER I

The Problem and Its Background

Introduction

Institutions of higher learning face the perennial challenge of efficiently organizing course timetables that accommodate the diverse schedules of faculty members and students while optimizing the utilization of available resources, such as classrooms and facilities. In pursuing an effective scheduling solution, it is imperative to systematize the logistical demands of scheduling with the goals of facilitating convenient learning experiences. This thesis aspires to address this challenge by proposing a comprehensive framework for planning and scheduling course timetables that balances the constraints of instructor availability, and student preferences.

The primary objective of the researchers is to develop a program for creating course timetables that optimally allocate resources while accommodating the individual constraints and preferences of faculty members and students. By integrating advanced scheduling algorithms with innovative approaches to data collection and analysis, such as the Constraint Satisfaction Problem (CSP) algorithm that offers a versatile and efficient framework for optimizing various domains to overcome logistical challenges in course timetabling systematically, the researchers aim to provide educational institutions with a practical tool for streamlining the course scheduling process and enhancing the overall quality of academic programs.

Background of Study

Manual planning of schedules, a task often burdened with the multifaceted nature of the constraints involved (Jones, 2023), can be a source of significant stress. Coordinating the availability of teachers, the capacities of classrooms, the preferences of students, and other factors requires meticulous attention to detail and can be prone to errors and oversights. Moreover, the iterative process of manually adjusting schedules to accommodate various constraints and stakeholders' needs can be time-consuming and labor-intensive, often resulting in suboptimal outcomes. However, the introduction of automated scheduling programs can alleviate this burden, offering a more efficient and accurate solution. These programs, with their computational power and systematic methodologies, can efficiently explore the vast solution space and identify the best possible schedule configurations.

However, the development of a dedicated program for scheduling presents a transformative solution to these challenges (Smith, 2021). Automated scheduling programs, with their computational algorithms and optimization techniques, offer a level of precision and reliability that manual scheduling methods often struggle to match. By processing large volumes of data quickly and systematically, these programs can identify optimal or near-optimal schedule configurations that satisfy all constraints and preferences. Furthermore, automated scheduling programs excel at resource optimization, intelligently allocating classrooms, faculty time, and other resources to maximize utilization and minimize conflicts (Brown, 2022). This

optimization enhances operational efficiency and ensures that institutional resources are utilized to their fullest potential.

Moreover, the adaptability and flexibility of automated scheduling programs offer significant advantages over manual methods (Johnson, 2020). These programs are not one-size-fits-all solutions. They can be customized to meet the unique scheduling needs of different educational institutions, accommodating changes in scheduling requirements, such as unexpected absences or adjustments to course offerings, without requiring extensive manual revisions. Additionally, automated scheduling programs can incorporate feedback from stakeholders and adapt dynamically to evolving needs and preferences, ensuring that schedules remain responsive and relevant over time. Overall, the development of a program for scheduling represents a significant opportunity to revolutionize scheduling practices in educational institutions, offering unparalleled efficiency, accuracy, and adaptability in creating and managing schedules.

Project Context

The proposed system aims to create an automated scheduling solution that efficiently integrates the schedules of professors and students with the available resources, such as rooms. This system will analyze the constraints and preferences of all involved parties to generate optimized schedules that minimize conflicts and maximize the utilization of resources.

By considering the availability and preferences of professors, the class schedules of students, and the designated learning modalities, the system ensures that every class session is appropriately matched with an available room. Developed for the students at the College of Computing Studies at EARIST, this automated scheduling system will streamline the planning process, reducing manual errors and administrative workload.

Purpose and Description

The primary purpose of the proposed automated scheduling system is to optimize the scheduling process for the College of Computing Studies at EARIST by integrating the schedules of professors and students with available resources. Utilizing a constraint satisfaction algorithm, this system aims to minimize scheduling conflicts, maximize resource utilization, and reduce manual errors and administrative workload. Automating the process will generate optimized schedules that match each class session with an available room or online platform, ensuring that professors and students get all available times for course deliverables. This system will also ease the burden on professors by saving them time coordinating schedules for all their courses, sections, and year levels.

Objectives of the Study

The general objective of the study is to design and develop a class scheduler utilizing the constraint satisfaction algorithm and to evaluate its efficiency and feasibility.

This study specifically aimed at attaining the following objectives:

1.To design and develop an automated class scheduling system that integrates the schedules of professors and students with available resources with the following features:

- 1.1 User Authentication;
- 1.2 Comprehensive Dashboard;
- 1.3 Professor and Room Management;
- 1.4 Course and Curriculum Management;
- 1.5 Automated Schedule Generation;
- 1.6 Schedule Visualization;
- 1.7 Conflict Detection and Resolution;
- 1.8 Settings and Configuration;

2 To test and evaluate the performance of Automated Class Scheduler for EARIST - College of Computing Studies by means of functional and repeated trial test:

- 2.1 End-User
 - 2.1.1 Functionality;
 - 2.1.2 Interface Design;
 - 2.1.3 Ease of use;

Conceptual Framework

The researchers utilized a conceptual framework to organize and elaborate on the development of the automated class scheduler. By using this framework, readers will be able to visualize and better understand the research study.

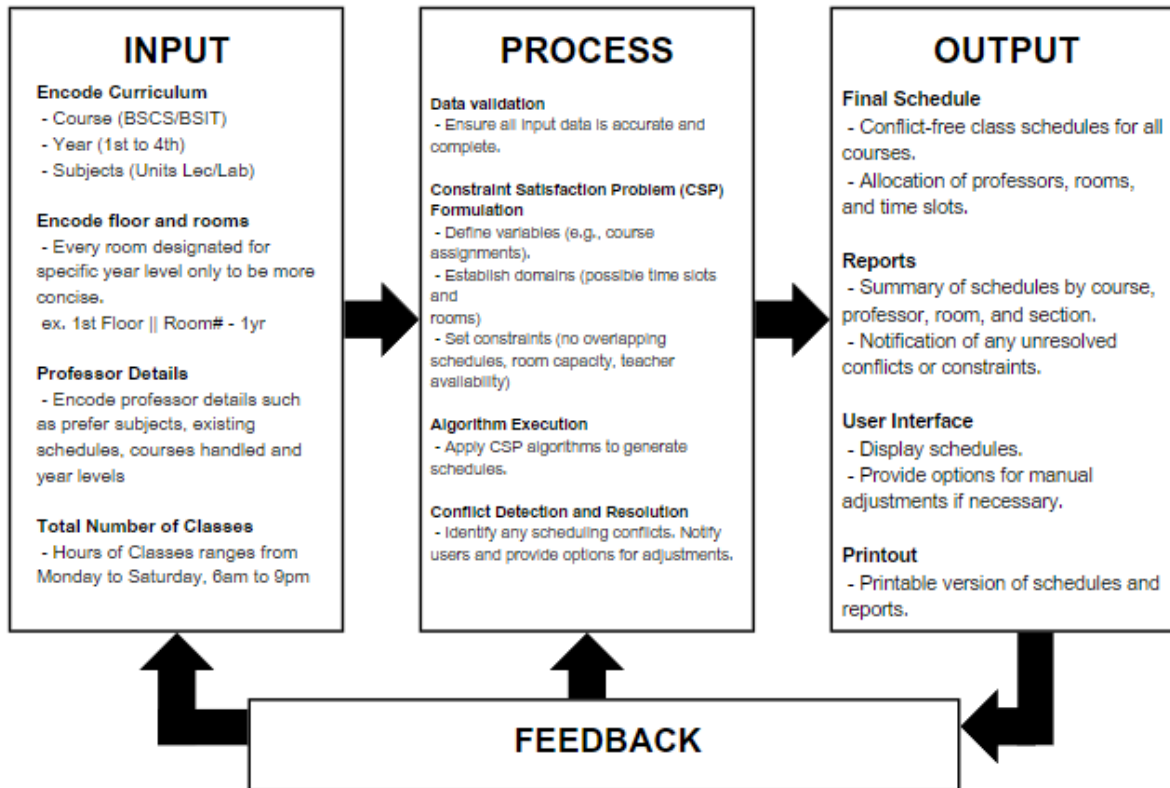


Figure 1.1 Conceptual Framework

Figure 1.1, the conceptual framework of the study, shows the input, process, and output of the proposed system. The inputs to the automated course scheduling system include data on courses, rooms, faculty, and scheduling created, considering various options for courses, professors, rooms, and parameters. This information undergoes processing where a scheduling model is time slots. During this process, constraints such as room capacity, faculty availability, and class timings are defined. The system then employs a search algorithm to explore possible scheduling solutions that meet all defined constraints. Optionally, the system can optimize workload balance or minimize room changes. The outputs of the system consist of a detailed course schedule that reflects the finalized

scheduling decisions and reports on the feasibility and potential optimization achieved, thereby laying the groundwork for an efficient and optimized course scheduling system.

Significance of Study

The following are the target beneficiaries of this project:

School. The implementation of an automated class scheduler will enhance the overall efficiency of the institution's administrative processes, reducing the time and effort required for manual scheduling and improving resource allocation.

The Students. Students will benefit from optimized schedules that minimize conflicts and ensure they can attend all their required classes, improving their academic experience and reducing stress.

The Teachers/Professors. Teachers and professors will experience a reduction in the time and effort needed to coordinate their schedules across multiple courses, sections, and year levels, allowing them to focus more on teaching and less on administrative tasks.

The Future Researchers. Future researchers will gain valuable insights from this study, including methodologies and findings that can be applied to further advancements in automated scheduling systems and educational technology.

Scope and Limitation of the Study

The automated class scheduling system developed in this study will be implemented specifically at EARIST Manila, focusing on the College of Computing Studies. It aims to provide teachers and other academic staff with a user-friendly platform to efficiently manage their schedules. The system includes features designed to accommodate the specific needs of the scheduling process, such as optimal allocation of classrooms and resources. Teachers and administrative staff responsible for scheduling will have access to input scheduling preferences and constraints, ensuring the system operates effectively.

This system's scope is limited to the College of Computing Studies at EARIST Manila, and its benefits are primarily targeted at improving administrative processes and enhancing scheduling accuracy within this specific academic context. The findings and the system developed may not be directly transferable or applicable to other colleges or institutions without significant customization.

Definition of Terms

Automated. The process or system that operates or functions with minimal human intervention, often relying on programmed instructions or algorithms to perform tasks automatically.

Constraint Satisfaction Problem. Involves finding solutions that satisfy a set of constraints that limit the possible values of variables.

Curriculum. The planned sequence of courses and educational experiences is designed to achieve specific learning outcomes.

User Interface. The means through which a person interacts with a computer, system, or application, typically consist of visual elements and controls.

Timetable. A schedule or plan that indicates the specific times at which events, activities, or tasks are intended to occur.

Data. Is defined as facts or figures, or information that is stored in or used by a computer.

Dashboards. are a data visualization tool that allow all users to understand the analytics that matter to the system.

Implementation: Implementation refers to the process of installing, configuring, and deploying the system.

CHAPTER II

Review of Related Literatures

According to Funda Güner, Abdül K. Görür, Benhür Satır, Levent Kandiller, and John H. Drake (2023), the study titled "A constraint programming approach to a real-world workforce scheduling problem for multi-manned assembly lines with sequence-dependent setup times" investigates the optimization of workforce scheduling in manufacturing settings with multi-manned assembly lines. The authors present two methods, mixed integer linear programming and constraint programming, to minimize the number of workers required while considering sequence-dependent setup times. The constraint programming method outperformed the mixed integer linear programming method in computational experiments and significantly improved productivity in real-world scenarios. The study concluded that the constraint programming approach is highly effective in handling complex scheduling problems in industrial environments, offering a viable solution for improving efficiency and productivity in assembly line operations.

A recent study by Francisco et al. (2023) investigated a specific variant of the group shop scheduling problem (GSSP) known as the fixed group shop scheduling problem (FGSSP). In FGSSP, jobs consist of stages, with each stage requiring dedicated machines. All jobs follow the same sequence through these stages, needing completion at one stage before moving on to the next. However, operations within a single stage can be performed in any order. The research proposes a novel heuristic

approach to tackle the FGSSP. This approach incorporates three key elements: first, it decomposes the problem into smaller subproblems, each focusing on individual stages. Second, it leverages Constraint Programming (CP) to solve each subproblem. The CP solver considers factors like processing times and setup times to find feasible solutions. Finally, the approach employs a restart mechanism to avoid getting stuck in local optima, which are suboptimal solutions that the algorithm might settle for if not careful. The researchers evaluated the performance of this approach on various problem instances and found it to outperform other methods, particularly for larger and more complex scheduling problems. This suggests that the proposed approach has the potential to be a valuable tool for efficiently solving FGSSPs.

According to the article "Constraint Satisfaction Problems (CSP) in Artificial Intelligence" from GeeksforGeeks (2023), Constraint Satisfaction Problems (CSPs) are a powerful paradigm in artificial intelligence that involves solving problems by identifying feasible solutions within a defined set of constraints. CSPs are characterized by variables, domains, and constraints. Variables represent elements of the problem, domains represent possible values for these variables, and constraints restrict the ways in which variables can be assigned values. The article outlines several key methods for solving CSPs, including backtracking, constraint propagation, and local search algorithms. Backtracking systematically explores possible assignments for variables, constraint propagation reduces the search space by enforcing constraints, and local search methods iteratively improve potential solutions.

Fan et al. (2022) explores optimizing parallel test task scheduling using Constraint Satisfaction Problems (CSPs). CSPs involve finding a set of values that satisfy all defined constraints. In conclusion, the study by Fan et al. presents a Constraint Satisfaction Problem (CSP) approach to optimizing parallel test task scheduling. They propose a recursive search artificial bee colony (RS-ABC) algorithm to find solutions that satisfy all defined constraints, such as resource availability and task dependencies. While this approach might not directly translate to course scheduling, it offers a relevant example of applying CSPs and a search algorithm to a scheduling problem.

Bogaerts et al. (2021) states that a key challenge in Constraint Satisfaction Problems (CSPs) is explaining the reasoning behind the solution process, particularly for non-experts. Their research proposes a framework for stepwise explanations during Constraint Propagation, a core technique used in solving CSPs. The framework focuses on providing explanations that are easy for humans to understand and verify. It achieves this by analyzing the cost of individual explanations and prioritizing those that are simpler to grasp. Additionally, the framework addresses scenarios where multiple constraints contribute to an inference step, allowing users to delve deeper into specific parts of the explanation. In conclusion, this research offers a valuable approach for explaining how Constraint Satisfaction Problems are solved, making the process more transparent and accessible for users, especially those without a deep understanding of the underlying algorithms.

While for Özcan (2021), this research proposes a new algorithm called "A Hard Constraint Satisfaction Problem (HCSP) Algorithm" specifically designed for university course timetabling. The study addresses the complexity of university course scheduling, which involves various constraints like room availability, faculty workload, student preferences, and course prerequisites. The HCSP algorithm aims to efficiently solve this Constraint Satisfaction Problem (CSP) by incorporating techniques like constraint propagation and backtracking. In conclusion, this research offers a new algorithm for university course timetabling that leverages Constraint Satisfaction Problem techniques. The HCSP algorithm aims to address the challenges of scheduling by considering various constraints and ensuring efficient solutions.

About Algethami and Laesanklang (2021), a recent study addressed a critical aspect of university course scheduling: incorporating faculty-course assignment constraints. Manual course scheduling can be complex and error-prone, especially when ensuring faculty are assigned appropriate courses and avoiding scheduling conflicts for both faculty and students. The study proposes a mathematical model that formulates course timetabling as a Constraint Satisfaction Problem (CSP). This model considers various factors, including assigning courses to appropriate times and days, guaranteeing all courses are scheduled, and balancing faculty workload by assigning a fixed number of courses based on their employment status (full-time vs. part-time). Additionally, the model incorporates faculty preferences and limitations regarding course assignments, further ensuring a feasible and optimized schedule. By solving the formulated CSP model, the research offers a method to generate course schedules that

satisfy all these constraints, leading to significant improvements in efficiency and accuracy compared to traditional manual scheduling methods.

On the other hand for Tan et al. (2021), Constraint Satisfaction Problems (CSP) is one of the current best practices for creating optimized school timetables. The task involves assigning courses, students, teachers, and classrooms to specific time slots while adhering to various constraints like teacher availability and avoiding student conflicts. CSP offers a method to model the scheduling problem as a set of variables (e.g., course assignments) and constraints (e.g., a teacher cannot be assigned two classes at the same time). A solver then searches for solutions that satisfy all constraints, resulting in a feasible timetable. This research highlights the growing preference for CSP due to its effectiveness in handling increasingly complex scheduling problems with multiple constraints. While the paper focuses on school timetabling, it provides valuable insight into the application of CSP for your capstone project on university course scheduling.

According to Popescu et al. (2021), machine learning is emerging as a valuable tool to enhance Constraint Satisfaction Problems (CSP) solving. CSPs involve finding solutions that adhere to a set of predefined constraints, often used in scheduling and resource allocation tasks. Traditional methods typically rely on backtracking search algorithms to explore potential solutions. This study explores how machine learning can be integrated into various aspects of constraint solving. These aspects include directly utilizing machine learning models to suggest solutions that satisfy all constraints,

automatically identifying and incorporating additional constraints through machine learning to improve search efficiency, and training machine learning models to suggest better strategies for ordering variables and their values during the search process. This can lead to faster exploration of promising solution paths. Additionally, machine learning models can be employed to predict whether a given CSP instance has a solution without fully executing the search algorithm, saving computational resources. The study emphasizes the potential of machine learning to improve the efficiency and effectiveness of constraint solving techniques. However, the specific CSP algorithms used in conjunction with machine learning would depend on the application and the chosen machine learning approach. In essence, Popescu et al. (2021) highlight how machine learning can be a powerful tool to improve various aspects of constraint solving, but it doesn't specify a single, unified CSP algorithm used with machine learning techniques.

Guei-Hao Chen, Jyh-Cherng Jong, and Anthony Fu-Wha Han (2021) proposes a hybrid approach to solve the crew scheduling problem for railroad systems. The approach combines Constraint Programming (CP) and Integer Programming (IP). Constraint Programming (CP) is a technique from artificial intelligence used to solve Constraint Satisfaction Problems (CSPs). In CSPs, the goal is to find a solution that satisfies all the defined constraints. Here, CP likely helps generate feasible duty assignments for the crew members, ensuring they adhere to regulations, work hour limitations, and other relevant constraints.

Vlk, Hanzálek, and Tang (2021) explores constraint programming approaches to tackle the combined problem of routing and scheduling in time-sensitive networks. Their study focuses on the use of Constraint Satisfaction Problem (CSP) algorithms to ensure that data packets are delivered within strict time constraints while optimizing the overall network performance. The conclusion of the study highlights that using CSP algorithms effectively manages the complexities of routing and scheduling in time-sensitive networks, demonstrating significant improvements in meeting time constraints and optimizing resource allocation.

Wen Song (2021) proposes a novel approach for Constraint Satisfaction Problems (CSPs) that utilizes deep reinforcement learning to learn effective variable ordering heuristics. CSPs involve finding solutions that satisfy all defined constraints, commonly used in scheduling and resource allocation problems. Traditionally, hand-crafted heuristics are used to determine the order in which variables are considered during the search for a solution. This order can significantly impact the efficiency of the search process.

Mallari et al. (2021) presents a university course timetabling problem, with an approach focusing on synchronizing course calendars. The traditional course timetabling problem involves assigning courses to specific time slots and rooms while adhering to various constraints, such as faculty and student availability, course prerequisites, and room capacities. The study emphasizes the importance of

synchronizing courses with similar content or taken by the same student groups to minimize scheduling conflicts and improve student learning.

Xiao et al. (2021) tackles the Course Scheduling Problem (CSP) with a specific focus on incorporating room considerations. The traditional CSP involves assigning students, faculty, exams, and lectures to rooms while adhering to various constraints, such as time conflicts and faculty availability. In conclusion, Xiao proposes an improved approach to the Course Scheduling Problem that incorporates room considerations. Their approach leverages a Mixed Integer Optimization Model (MIO) to represent the problem with room preferences and a Nested Simulated Annealing (NSA) algorithm to find feasible solutions that satisfy all constraints.

Li et al. (2020) proposes an improved Ant Colony Optimization (ACO) algorithm specifically designed for Constraint Satisfaction Problems (CSPs). Constraint Satisfaction Problems (CSPs) involve finding solutions that satisfy a set of predefined constraints, often used in scheduling problems. Traditional ACO algorithms are iterative and inspired by ant behavior. "Artificial ants" explore potential solutions and leave trails to guide others towards promising areas of the search space. The study proposes an improvement to the standard ACO algorithm by incorporating an automatic updating mechanism. While the details of this mechanism are unavailable without full access to the paper, it likely enhances the algorithm's ability to find optimal solutions for CSPs. This improvement might involve dynamically adjusting parameters or guiding the search process based on the progress made during the search.

Maruthi, Dodda, Yellu, Thuniki, and Byrapu Reddy (2020) states that automated planning and scheduling are critical components in AI, enhancing decision-making in complex environments. Their study reviews various techniques, algorithms, and methodologies, highlighting the integration of planning and scheduling with machine learning and constraint satisfaction to improve decision-making capabilities. They conclude that the effective application of these techniques can significantly advance the state of automated decision-making in AI, identifying key areas for future research.

Bashab et al. (2020) explores optimization techniques used to solve University Timetabling Problems (UTPs). UTPs involve creating class schedules that satisfy various constraints while considering preferences. The study highlights that UTPs are a type of Constraint Satisfaction Problem (CSP), where the goal is to find a solution that fulfills all defined constraints. However, the specific CSP algorithm used within the various optimization techniques is not explicitly mentioned in the research.

Frohner et al. (2019) proposes a hybrid approach for casual employee scheduling at a university. This approach combines Constraint Satisfaction Problems (CSPs) with Metaheuristics. CSPs provide a framework for modeling the scheduling problem. The process involves defining variables, such as which employee is assigned to which shift. Each variable has a domain, which represents the set of possible values it can take. For example, the domain for an employee assignment variable might be all available work shifts for that employee. Finally, constraints are established, which are

the rules that a valid schedule must follow. These constraints can consider factors like employee availability, skill requirements for specific tasks, ensuring a fair workload distribution, and adhering to university regulations. In conclusion, the study by Frohner et al. (2020) offers a valuable approach for handling complex scheduling problems with various constraints in an educational setting. Their hybrid approach leverages Constraint Satisfaction Problems (CSPs) to model the problem and Simulated Annealing (SA) to improve the search for high-quality solutions that meet all the defined constraints.

According to Manning Publications (2019), "Constraint-Satisfaction Problems in Python" offers a practical guide to implementing Constraint Satisfaction Problems (CSPs) using Python programming language. The resource explores building a foundational CSP framework and delves into the implementation of specific CSP algorithms. It likely discusses utilizing libraries like Choco or Gecode, which can significantly aid in developing a CSP solver. These functionalities will be instrumental in translating the formulated CSP model for your automated class scheduler project into actual Python code. In conclusion, this reference by Manning Publications provides valuable practical guidance and code-level insights directly applicable to the implementation phase of a project.

According to Abdullahi Published 2019, creating a well-structured school timetable is an important task that requires careful consideration of various factors. The article discusses a Constraint Satisfaction Problem (CSP) approach to creating a

timetable, outlining the variables and constraints involved in this process. Some of the important points from this article are the variables and constraints used in creating a timetable, as well as ways to optimize the timetable.

Synthesis

The reviewed articles and materials contribute to the development of the system, and can be related to the present study in terms of the following:

1. Multiple CSP algorithms offer different strengths for tackling course scheduling problems: While the core concept of CSPs remains the same (finding a solution that satisfies all constraints), various algorithms exist with specific advantages for scheduling:
 - a. *Backtracking Search*: A widely used and efficient algorithm that explores all possible solutions systematically, backtracking when it encounters an infeasible path. This approach is ideal for finding a good initial solution, especially when dealing with smaller datasets or less complex scheduling problems.
 - b. *Metaheuristics (e.g., Simulated Annealing)*: These algorithms offer a more flexible approach for navigating complex search spaces. They don't guarantee finding the absolute optimal solution every time, but they excel at finding good solutions efficiently, particularly for larger datasets or scheduling problems with many intricate constraints.
2. CSP models can be customized to address specific university scheduling constraints: Beyond the core functionalities of assigning courses, rooms, and

students, CSP models can be tailored to incorporate specific university requirements. This allows you to model constraints like:

- a. *Faculty workload distribution*: The model can ensure a fair distribution of courses across faculty members, considering factors like teaching load and expertise.
 - b. *Faculty preferences*: The model can consider faculty preferences for specific courses or time slots, leading to a more balanced and satisfying schedule for instructors.
 - c. *Room features*: The model can consider the specific features available in each room (e.g., projector, lab equipment, seating capacity) to ensure each course is assigned to an appropriate space.
3. Advanced techniques like Machine Learning can be integrated with CSPs in the future: While this synthesis focuses on the core functionalities of CSPs, research is exploring the potential of Machine Learning (ML) to further enhance the efficiency and effectiveness of automated course scheduling systems. For example, ML techniques might be used to:
 - a. *Suggest potential solutions*: Machine learning models could analyze historical data or student preferences to suggest promising course schedules that align with student needs.
 - b. *Identify additional constraints*: ML algorithms might automatically identify hidden patterns or conflicts within the scheduling data, leading to a more comprehensive constraint set for the CSP model.

- c. *Optimize search strategies*: Machine learning could be used to dynamically adjust the search process within the CSP algorithm, prioritizing exploration of promising solution spaces

CHAPTER III

Methodology

Technological Background

The proposed capstone project, "An Automated Class Scheduler for College of Computing Studies using Constraint Satisfaction Problem Algorithm," leverages advanced algorithmic techniques to automate the complex and often error-prone task of class scheduling. The system is designed to handle the scheduling for BSIT and BSCS courses across all year levels (1st to 4th year) and semesters (1st and 2nd), ensuring efficient use of time and resources. By utilizing a Constraint Satisfaction Problem (CSP) algorithm, the system can consider multiple constraints simultaneously, including class timings (time slots), professor availability, room availability, and curriculum requirements. The system stores all necessary data such as curricula, professor preferences and schedules, and room details in a structured database, which the CSP algorithm uses to generate conflict-free schedules. The user interface is designed to be intuitive, allowing administrators to input professor and room details, and view the generated schedules by course, professor, or room. The adoption of this automated scheduling system aims to eliminate the manual, Excel-based process currently in use, significantly reducing the likelihood of scheduling conflicts and improving overall efficiency in the scheduling process.

Approaches and Techniques

The development of the Automated Class Scheduler for the College of Computing Studies employs several key approaches and techniques to address the complexities of scheduling academic classes efficiently and effectively. Central to the system's design is the utilization of a Constraint Satisfaction Problem (CSP) algorithm, chosen for its ability to manage multiple constraints simultaneously and optimize resource allocation. This algorithm forms the backbone of the scheduling engine, ensuring that courses are assigned suitable time slots without overlap, considering factors such as professor availability, room capacity, and course requirements.

Database-Driven Approach: The system adopts a relational database management system (e.g., MySQL, PostgreSQL) to store and manage crucial data such as course details, professor availability, room specifications, and historical scheduling information. This approach facilitates efficient data handling and retrieval, enabling quick access to relevant information during the scheduling process.

User-Centric Design: User interface (UI) design focuses on providing administrators with intuitive tools to input and manage scheduling parameters. Through forms and interactive interfaces, users can specify professor preferences, room allocations by year and semester, and other constraints essential for generating accurate schedules. This design ensures usability and accessibility, enhancing user productivity and satisfaction.

Optimization Techniques: Beyond basic CSP implementation, the system integrates optimization techniques to improve schedule quality and efficiency. Techniques such as backtracking with forward checking, constraint propagation, and heuristic methods are applied to expedite the search for feasible solutions and to enhance the accuracy of schedule generation.

Iterative Development and Testing: The project follows an iterative development process, allowing for continuous refinement based on feedback from stakeholders and testing against real-world scenarios. This iterative approach ensures that the system meets functional requirements while accommodating evolving educational needs and administrative policies.

Methodology

The researchers used the Agile method to create the automated class scheduling system. Agile is a flexible, step-by-step approach that focuses on teamwork, listening to users' needs, and making improvements. This method lets the team make small, easy-to-handle parts of the project, test them, and get feedback regularly. Using Agile, the researchers easily adjust to new requests or suggestions from people involved, ensuring the system works well for teachers, school staff, and students. This strategy led to a more efficient and user-friendly scheduling tool.



Figure 3.1 Agile Model

The Agile model's consecutive phases are as follows:

Planning Phase:

During the planning phase of the " **An Automated Class Scheduler for EARIST - College of Computing Studies using Constraint Satisfaction Problem Algorithm**" project, the research team established the project's vision, objectives, and scope. Stakeholders, including university administrators, potential users, were identified to provide input and requirements. Communication channels were set up, and an Agile team was formed to facilitate efficient collaboration. A prioritized product backlog was created, outlining features, enhancements, and bug fixes, setting the stage for subsequent phases of development.

Designing Phase:

In the designing phase, the project team conceptualized the functions of the helpdesk solution using various diagrams and established logical connections between project requirements. Visual representations, including Use Case Diagrams, Context Diagrams, and Entity-Relationship Diagrams, were developed to illustrate the system's architecture and design. Use Case Specifications were drafted to define system functions, considering user interactions and data flow, ensuring a comprehensive design approach.

Development Phase:

The development phase began with iterative and incremental development, guided by Agile principles. Sprint planning sessions were conducted to select and define backlog items for upcoming iterations. The Agile team focused on developing the user interface, functionality, and system features. Daily stand-up meetings, continuous integration, and incremental software delivery were employed to maintain momentum and ensure progress throughout the development process.

Testing Phase:

The testing phase aimed to validate the reliability, functionality, and usability of the developed system. Rigorous testing of individual features and functionalities was carried out, followed by comprehensive system-wide testing to identify and address any potential issues. User acceptance testing (UAT) involved gathering feedback from stakeholders to guide iterative development, enhancing the system's robustness and usability.

Deployment Phase:

During the deployment phase, the developed system was released for live usage, making it accessible to end-users. System components were prepared for production, and the rollout was carefully executed to minimize disruptions. Post-deployment monitoring ensured the system's performance, functionality, and accessibility across various computing devices, ensuring a seamless user experience.

Reviewing Phase:

The reviewing phase focused on reflection, feedback collection, and identifying areas for improvement. Sprint reviews showcased completed functionality and gathered feedback from stakeholders. Retrospectives allowed for reflections on the development process, leading to collaborative discussions on the system's overall performance and user satisfaction. Planning for ongoing maintenance, updates, and improvements was informed by the outcomes of the review process, ensuring continuous enhancement of the CSP solution.

Requirement Analysis

The system is designed to automate the scheduling process for BSIT (Bachelor of Science in Information Technology) and BSCS (Bachelor of Science in Computer Science) programs across all academic years (1st to 4th) and semesters (1st and 2nd). Functional requirements include:

Course Scheduling: Ability to schedule lectures and labs based on predefined curricula, considering course types (Lecture, Lab), units, and sequencing requirements.

Professor Management: Capability to manage professor details, including availability, preferred subjects, and existing schedules, to optimize their allocation to courses.

Room Allocation: Facility to allocate appropriate rooms based on course requirements, room capacity, and year-specific allocations (e.g., 1st year, 2nd year).

Conflict Detection and Resolution: Identify potential conflicts such as Professor availability conflicts (assigned to multiple courses at the same time), room availability conflicts (multiple courses assigned to the same room at the same time).

Population, Sample Size, and Sampling Techniques

The population of this research includes key academic staff members from the College of Computing Studies of EARIST Manila, specifically the dean, associate dean, chairperson, and faculty members. For the study, a sample size of four was chosen, with one representative selected from each group. The sampling technique used is purposive sampling, which allows the researchers to intentionally select individuals who are most relevant and can provide the most insightful information for the study.

Description of Respondents

The academic staff members—comprising the dean, associate dean, chairperson, and faculty representative—contribute extensively to the scheduling process. They oversee strategic alignment with institutional goals, establish policies, and allocate resources to optimize scheduling efficiency. The chairperson coordinates course offerings, faculty assignments, and classroom utilization within departments. Meanwhile, the faculty representative provides insights on teaching preferences and student needs, ensuring the scheduling system accommodates diverse instructional requirements. Together, their collaboration aims to streamline operations, minimize conflicts, and foster an effective learning environment through the implementation of an automated scheduling solution.

Research Instrument

The research instrument selected for this research is a survey questionnaire with a 5-point Likert scale designed to gather comprehensive insights from key stakeholders involved in the automated class scheduling system. The questionnaire aims to gather insights on various functionality, user interface design, ease of use, and overall user experience. By soliciting feedback from these stakeholders, the research seeks to identify strengths and areas for improvement in the scheduling system's functionality and usability.

Data Gathering Procedures

The data gathering procedures for this study involve two main approaches. Initially, the researchers will invite the sample size, comprising key academic staff members, to actively use and evaluate the automated class scheduling system. This hands-on engagement allows them to assess the system's functionality and usability firsthand. Subsequently, a survey will be conducted using a 5-point Likert scale to systematically gather feedback. The survey will focus on various aspects such as functionality, usability, ease of use, and overall effectiveness of the scheduling features.

The researchers formulated a questionnaire to be used in gathering data. The questionnaire was checked by the adviser and statistician, then researchers floated the questionnaire to the respondents immediately. Upon retrieving the distributed questionnaires from the respondents, the researchers combine the respondents' answers with the use of frequency distribution followed by the analysis and interpretation of data

Statistical Treatment of Data

Descriptive statistics will be employed in this research to analyze the usability factors of the automated class scheduling system among academic staff members. Specifically, measures such as mean and standard deviation will be calculated from Likert scale responses gathered through a survey. The mean scores will provide a quantitative overview of how respondents perceive various aspects of the system, including user interface design, navigation ease, and overall functionality.

Weighted Mean

$$M = \frac{\Sigma[w(x)]}{N}$$

where:

M = weighted mean

w = number of respondents who answered specific scale or item

x = values of likert scale

N = sample size

These responses were used to interpret the data. The following are the numerical ratings and interpretations of the responses that were analyzed and evaluated:

Table 1: Likert Scale

MEAN VALUE	WEIGHT	VERBAL INTERPRETATION
4.30 – 5.00	5	Strongly Agree
3.50 - 4.20	4	Agree




2.70 – 3.40	3	Neutral
1.90 – 2.60	2	Disagree
1.00 – 1.80	1	Strongly Disagree

Table 1 illustrates the 5-point Likert scale that will be employed in the survey to evaluate the usability of the automated class scheduling system among academic staff members. This scale provides respondents with a structured framework to express their level of agreement or disagreement regarding various aspects of the system's usability, including user interface design, navigation ease, functionality, and overall satisfaction. Each point on the scale number in Verbal Interpretation shows the four rating scales that were used in questionnaire-rating. Number 1 is equivalent to Strongly Disagree, 2 is Disagree, 3 is Somehow Agree, 4 is Agree and 5 is Strongly Agree.

System Requirements

The system contains requirements to comply with the standard and develop with the demanded features from the beneficiaries. The following are the requirements to build and use the system:

Software

SOFTWARE	DESCRIPTION
 Visual Studio Code	Visual Studio Code is an IDE created by Microsoft. It will be used to edit code and build the program of the system which includes the front end and backend of the system.
 XAMPP	XAMPP includes the Apache web server, MySQL, PHP, Perl, FTP server, and PHPMyAdmin. The software will be used for the webserver of the system.
 PHP	The programming language that will be used in developing the system



 HTML	The programming language that will be used in developing the system.
 CSS	The language that will be used in designing the system.

Table 2. Software Requirements

Table 2 contains all the software including its names and description that the researchers will use in developing the system which includes software that will be used in designing the user interface of the system and testing the system.

Hardware

HARDWARE	SPECIFICATIONS
Monitor Display	At least 1024 x 768 of resolution
RAM	4 gigabytes
Storage (Hard Disk Drive)	At least 1 gigabyte
CPU (Intel i3-6006u)	2.00 gigahertz (4 CPUs)
Ethernet connection (LAN) OR a wireless adapter (Wi-Fi)	Ethernet (Cat-5e), Wireless Adapter (at least 2.4 gigahertz)

Table 3. Hardware Requirements

Table 3 contains all the hardware including its minimum specifications that the researchers will use in developing the system. The Monitor Display will be used by the researcher in its overall development which includes developing the system, creating its design, and testing the system. Four (4) gigabytes of Random Access Memory will be used by the researcher as the requirement in the development as it needs to run multiple processes simultaneously. The Central Processing Unit requirement that will be used is Intel Core-i3 which has 2.00 gigahertz since the development uses software that needed fast processing. Lastly is the internet connectivity, since the system will be web-based, it needs internet connectivity to be available in web browsers and for the researchers to upload it to the server.

Peopleware

The primary users of the system are academic administrators and faculty members who interact with the scheduler to input scheduling parameters, manage course allocations, and generate final schedules. Their feedback during development and testing phases is crucial to align the system with their operational needs.

Network

The Automated Class Scheduler operates independently of network connectivity, with no requirement for internet access or local network

communication. This ensures the system's functionality and reliability even in environments with limited connectivity or offline usage scenarios.

Dataware

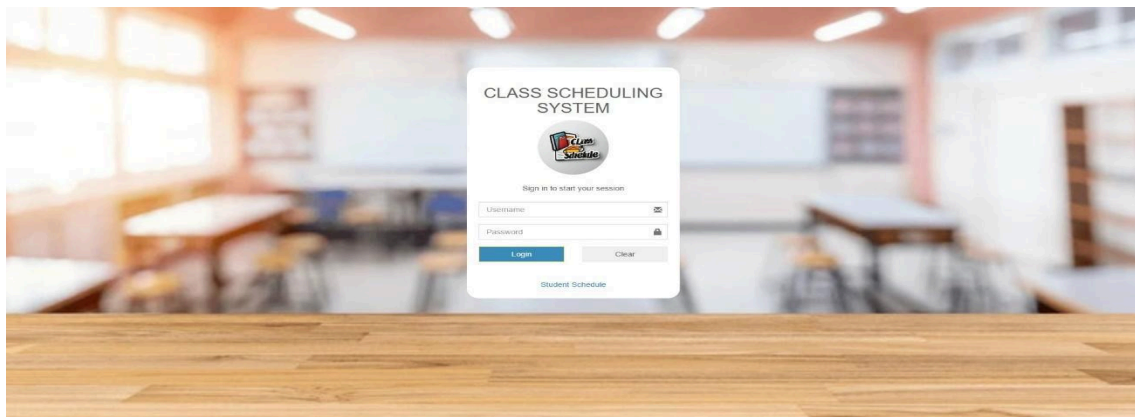
Data management focuses on local storage capabilities to accommodate application files and scheduling data securely on individual client devices. Regular data backups are recommended to maintain data integrity and facilitate recovery in case of system failures.

CHAPTER IV

RESULTS AND DISCUSSION

Presentation of the Project Development and Evaluation of the system

4.1 User Authentication: Secure login system to ensure only authorized personnel can access the system.



Login Module

The Login Module serves as the first point of interaction between users and the web-based system. It ensures secure access for different types of users such as administrators and users. The module features a role-based authentication system that limits access based on user privileges, ensuring that only authorized individuals can perform certain tasks.

4.2 Comprehensive Dashboard: Quick access to key information and system navigation.

The screenshot shows a web interface for a scheduling system. At the top, there's a navigation bar with links: Class Schedule, Entry, Settings, and Mrs Admin Admin. The main content area is titled 'Class Schedule' and has two tabs: 'Teachers' and 'Add Time'. Below the tabs is a table with columns for Time, M, T, W, TH, and F. The time slots range from 07:00 am - 08:00 am to 06:30 pm - 07:30 pm. To the right of the table is a form for adding a new class. It includes dropdown menus for Teacher (Romero, Pablo), Course, Yr & Section (BSIT SECTION-A), Subject (Math 101), and Room (101). There is also a text area for Remarks and two buttons: Save and Uncheck All.

Time	M	T	W	TH	F
07:00 am - 08:00 am	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
07:30 am - 08:30 am	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
08:30 am - 09:30 am	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
09:30 am - 10:30 am	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
10:30 am - 11:30 am	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
11:30 am - 12:30 pm	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
12:30 pm - 01:30 pm	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
01:30 pm - 02:30 pm	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
02:30 pm - 03:30 pm	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
03:30 pm - 04:30 pm	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
04:30 pm - 05:30 pm	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
05:30 pm - 06:30 pm	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
06:30 pm - 07:30 pm	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

The Dashboard Module functions as the central hub where users can access an overview of their tasks and relevant information after logging in.

4.3 Automated Schedule Generation

The screenshot shows a web interface for a scheduling system. At the top, there's a navigation bar with links: My Schedule, Add Schedule, Edit Schedule, and Dr James Bungay. Below the navigation bar is a button: Print / Download. The main content area is titled 'Class Schedule: Dr James Bungay' and has a table with columns for Time, M, T, W, TH, and F. The time slots range from 07:00 am - 08:00 am to 06:30 pm - 07:30 pm.

Time	M	T	W	TH	F
07:00 am - 08:00 am					
07:30 am - 08:30 am					
08:30 am - 09:30 am					
09:30 am - 10:30 am					
10:30 am - 11:30 am					
11:30 am - 12:30 pm					
12:30 pm - 01:30 pm					
01:30 pm - 02:30 pm					
02:30 pm - 03:30 pm					
03:30 pm - 04:30 pm					
04:30 pm - 05:30 pm					
05:30 pm - 06:30 pm					
06:30 pm - 07:30 pm					

Forms and lists to manage professor and room details efficiently. Easy input and modification of course and curriculum data. Generate schedules based on predefined constraints and user inputs.

4.4 Conflict Detection and Resolution Settings and Configuration: Customizable

☆ Class Schedule

📄 Entry

⚙️ Settings

👤 Mrs Admin Admin

🔌

Class Schedule

Teacher

Add Time

Time	M	T	W	TH	F
07:00 am-08:00 am	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
07:30 am-08:30 am	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
08:30 am-09:30 am	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
09:30 am-10:30 am	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
10:30 am-11:30 am	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
11:30 am-12:30 pm	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
12:30 pm-01:30 pm	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
01:30 pm-02:30 pm	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
02:30 pm-03:30 pm	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
03:30 pm-04:30 pm	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
04:30 pm-05:30 pm	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
05:30 pm-06:30 pm	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
06:30 pm-07:30 pm	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

monday 07:00 am-08:00 am Room 101 Not Available

wednesday 07:00 am-08:00 am Room 101 Not Available

friday 07:00 am-08:00 am Room 101 Not Available

tuesday 07:00 am-08:00 am Room 101 Not Available

thursday 07:00 am-08:00 am Room 101 Not Available

Teacher

Romero, Pablo

Course, Yr & Section

BSIT SECTION-A

Subject

Math 101

Room

101

Remarks

enclose remarks with parenthesis()

Save

Uncheck All

Clear and intuitive calendar view of the schedules. : Identify potential conflicts such as professor availability conflicts (assigned to multiple courses at the same time), room availability conflicts (multiple courses assigned to the same room at the same time).

Functionality

Table 1

The evaluation of respondents to the proposed An Automated Class Scheduler for EARIST - College of Computing Studies using Constraint Satisfaction Problem Algorithm in terms of functionality

Indicators	Weighted Mean	Verbal Interpretation	Rank
1. The system can create, edit, update, and delete data.	3.70	Agree	5
2. The system can secure login systems to ensure only authorized personnel can access the system.	3.80	Agree	4
3. The system can quick access to key information and system navigation.	4.10	Agree	1.5
4. Forms and lists to manage professor and room details efficiently.	4.10	Agree	1.5
5. Easy input and modification of course and curriculum data.	4.00	Agree	3
6. Generate schedules based on predefined constraints and user input.	3.50	Agree	6
GENERAL WEIGHTED MEAN	3.87	AGREE	

Table 1 presents the evaluation of respondents regarding the proposed "Automated Class Scheduler for EARIST - College of Computing Studies" using the Constraint Satisfaction Problem Algorithm, specifically in terms of functionality. The general weighted mean of 3.87, which corresponds to the verbal interpretation of "Agree," indicates a favorable assessment of the system's functionality.

Among the indicators, the highest-rated functionalities are "quick access to key information and system navigation" and "forms and lists to manage professor and room details efficiently," both receiving a weighted mean of 4.10. These functionalities tied for the top rank, reflecting the respondents' strong approval. The system's ability to "easy input and modification of course and curriculum data" ranked third with a weighted mean of 4.00, showing positive feedback as well. The functionality concerning "securing login systems to ensure authorized access" garnered a weighted mean of 3.80, ranking fourth, indicating agreement on the importance of system security. Lastly, the ability to "create, edit, update, and delete data" was evaluated with a weighted mean of 3.70, placing it fifth in the ranking, still within the "Agree" range. The overall results suggest that the system is viewed favorably by the respondents in terms of its functional capabilities.

Interface Design

Table 2

The evaluation of respondents to the proposed An Automated Class Scheduler for EARIST - College of Computing Studies using Constraint Satisfaction Problem Algorithm in terms of interface design

Indicators	Weighted Mean	Verbal Interpretation	Rank
1. Visually Appearing Interface.	3.50	Agree	4
2. Font used are readable.	4.10	Agree	1
3. Color Scheme.	4.00	Agree	2
4. Layout.	3.60	Agree	3
5. Proper Position of Buttons.	3.40	Neutral	5
GENERAL WEIGHTED MEAN	3.72	AGREE	

Table 2 presents the evaluation of respondents regarding the proposed "Automated Class Scheduler for EARIST - College of Computing Studies" in terms of its interface design, with a general weighted mean of 3.72, corresponding to the verbal interpretation of "Agree." This indicates that the overall assessment of the interface design is favorable.

The highest-rated indicator is "Font used are readable," which received a weighted mean of 4.10, ranking first. This suggests that respondents were particularly satisfied with the readability of the fonts used in the system. The second-highest rating was for the "Color Scheme," with a weighted mean of 4.00, indicating positive feedback regarding the color choices in the interface. The

"Layout" of the system was also well-received, with a weighted mean of 3.60, ranking third.

The "Visually Appearing Interface" was rated with a weighted mean of 3.50, placing it fourth, reflecting general agreement on its visual appeal. However, the lowest-rated indicator was "Proper Position of Buttons," which received a neutral response with a weighted mean of 3.40, ranking fifth. This suggests that there may be room for improvement in the placement of the buttons for better usability. Despite this, the overall feedback on the interface design is positive, with respondents agreeing on its general effectiveness.

Table 3

The evaluation of respondents to the proposed An Automated Class Scheduler for EARIST - College of Computing Studies using Constraint Satisfaction Problem Algorithm in terms of ease of use

Indicators	Weighted Mean	Verbal Interpretation	Rank
1. The system is easy to use.	3.90	Agree	1
2. The system offers a quick, cleared, obvious response to user's actions	3.80	Agree	2
3. Features are easily recognized.	3.60	Agree	3
GENERAL WEIGHTED MEAN	3.77	AGREE	

Table 3 presents the evaluation of respondents regarding the proposed "Automated Class Scheduler for EARIST - College of Computing Studies" in terms

of ease of use, with a general weighted mean of 3.77, indicating agreement with the system's usability.

The highest-rated indicator is "The system is easy to use," which received a weighted mean of 3.90, ranking first. This suggests that respondents found the system intuitive and user-friendly. The second-highest rating was for "The system offers a quick, clear, and obvious response to the user's actions," with a weighted mean of 3.80, reflecting satisfaction with the system's responsiveness and clarity. The "Features are easily recognized" indicator was rated with a weighted mean of 3.60, ranking third, indicating general agreement that the system's features are easy to identify and understand. Overall, the feedback suggests that the system is considered easy to use, with respondents expressing positive views on its usability.

Table 4

Summary table of the evaluation of respondents to the proposed An Automated Class Scheduler for EARIST - College of Computing Studies using Constraint Satisfaction Problem Algorithm

Indicators	Weighted Mean	Verbal Interpretation	Rank
1. Functionality	3.87	Agree	1
2. Interface Design	3.72	Agree	3
3. Ease of Use	3.77	Agree	2
GENERAL WEIGHTED MEAN	3.79	AGREE	

Table 4 provides a summary of the evaluation of respondents to the proposed "Automated Class Scheduler for EARIST - College of Computing Studies" using the Constraint Satisfaction Problem Algorithm, based on three key indicators: functionality, interface design, and ease of use.

The highest-rated category is Functionality, with a weighted mean of 3.87, which corresponds to the verbal interpretation "Agree," ranking first. This indicates that respondents were most satisfied with the system's functional capabilities. The second-highest rating was for Ease of Use, with a weighted mean of 3.77, also within the "Agree" range, and it ranked second, suggesting the system is considered user-friendly. Interface Design received a slightly lower weighted mean of 3.72, ranking third, but still falls within the "Agree" range, indicating that the interface design is well-received.

The overall general weighted mean of 3.79, categorized as "Agree," reflects positive feedback across all three indicators. The evaluation suggests that respondents view the system favorably in terms of its functionality, ease of use, and interface design.

CHAPTER V

CONCLUSION AND RECOMMENDATION

Conclusion

In conclusion, the Automated Class Scheduler for EARIST - College of Computing Studies" successfully addressed the challenges faced by the institution in managing schedule operations and provided a solution that improved functionality, security, and user experience. Moving forward, continued monitoring, maintenance and potential enhancements will ensure the system remains effective and meets the evolving needs of its users and the institution.

End-user, the researcher uses this to evaluate the system. To become efficient and recognize the use of the system. Functionality, the researcher made sure that the system has a proper functionality appropriate to its use. It shows how the created system will work and how it will respond to the command. Interface Design, the researchers have made the interface design simple, organized to be easy for users to use. The interface design is customized for a scheduling system. Ease of use, in general, the researchers made the system easy to understand so that it would not be difficult for the user.

Recommendations

After careful evaluation and assessment of all relevant factors, the researchers have produced the following recommendations:

Continuous Monitoring and Evaluation: Implement a regular monitoring and evaluation process to assess the system's performance, identify any issues or

challenges, and gather feedback from users. This feedback can be used to make iterative improvements and enhancements to the system over time.

- Mobile Access : Develop mobile applications or mobile-friendly interfaces to enable users to access the scheduling system from their smartphones or tablets. To offer users greater flexibility and convenience when making cafeteria purchases.

- Security Enhancements: Implement additional security measures, such as multi-factor authentication, encryption, and regular security audits, to safeguard sensitive user data and prevent unauthorized access or breaches. Continuously monitor security threats and vulnerabilities to proactively address emerging risks and ensure data protection compliance.

Faculty Engagement: Regularly engage with users through surveys, focus groups and feedback sessions to gather insights, address concerns, and promote user satisfaction.

REFERENCE:

- Güner, F., Görür, A. K., Satır, B., Kandiller, L., & Drake, J. H. (2023). A constraint programming approach to a real-world workforce scheduling problem for multi-manned assembly lines with sequence-dependent setup times. *International Journal of Production Research*, 62(9), 3212–3229. <https://doi.org/10.1080/00207543.2023.2226772>
- Yuraszeck, F., Mejía, G., Pereira, J., & Vilà, M. (2022). A novel constraint programming decomposition approach for the total flow time fixed group shop scheduling problem. *Mathematics*, 10(3), 329. <https://doi.org/10.3390/math10030329>
- Gao, J., Zhu, X., & Zhang, R. (2022). Optimization of parallel test task scheduling with constraint satisfaction. *The Journal of Supercomputing/Journal of Supercomputing*, 79(7), 7206–7227. <https://doi.org/10.1007/s11227-022-04943-0>
- Nwufoh, C. V., Achimugu, P., Achimugu, O., & Chollom, T. D. (2021). A HARD CONSTRAINT SATISFACTION PROBLEM (HCSP) ALGORITHM FOR UNIVERSITY COURSE TIME TABLING. ResearchGate. Retrieved from https://www.researchgate.net/publication/361283868_A_HARD_CONSTRAINT_SATISFACTION_PROBLEM_HCSP_ALGORITHM_FOR_UNIVERSITY_COURSE_TIME_TABLING
- Lahbib, S., & Ouenniche, J. (2021). A mathematical model for course timetabling problem with faculty-course assignment constraints. *International Journal of Information Technology & Decision Making*, 20(4), 1285-1310. Retrieved from https://www.researchgate.net/publication/353785573_A_Mathematical_Model_for_Course_Timetabling_Problem_With_Faculty-Course_Assignment_Constraints
- Tan, J. S., Goh, S. L., Kendall, G., & Sabar, N. R. (2021). A survey of the state-of-the-art of optimisation methodologies in school timetabling problems. *Expert Systems With Applications*, 165, 113943. <https://doi.org/10.1016/j.eswa.2020.113943>
- Popescu, A., Polat-Erdeniz, S., Felfernig, A., Uta, M., Atas, M., Le, V., . . . Tran, T. N. T. (2021). An overview of machine learning techniques in constraint solving. *Journal of Intelligent Information Systems*, 58(1), 91–118. <https://doi.org/10.1007/s10844-021-00666-5>
- Chen, G., Jong, J., & Han, A. F. (2021). Applying constraint programming and integer programming to solve the crew scheduling problem for railroad systems: model formulation and a case study. *Transportation Research Record*, 2676(1), 408–420. <https://doi.org/10.1177/03611981211036368>
- Vlk, M., Hanzálek, Z., & Tang, S. (2021). Constraint programming approaches to joint routing and scheduling in time-sensitive networks. *Computers & Industrial Engineering*, 157, 107317. <https://doi.org/10.1016/j.cie.2021.107317>
- Song, W., Cao, Z., Zhang, J., Xu, C., & Lim, A. (2022). Learning variable ordering heuristics for solving Constraint Satisfaction Problems.

Engineering Applications of Artificial Intelligence, 109, 104603.
<https://doi.org/10.1016/j.engappai.2021.104603>

- **Mallari, C. B., Juan, J. L. S., & Li, R. (2023). The university coursework timetabling problem: An optimization approach to synchronizing course calendars. *Computers & Industrial Engineering*, 184, 109561.**
<https://doi.org/10.1016/j.cie.2023.109561>
- **Xiao, L. (n.d.). The Course Scheduling Problem with Room Considerations. Retrieved from**
https://corescholar.libraries.wright.edu/etd_all/2474/?utm_source=corescholar.libraries.wright.edu%2Fetd_all%2F2474&utm_medium=PDF&utm_campaign=PDFCoverPages
- **Guan, B., Zhao, Y., & Li, Y. (2021). An improved ant colony optimization with an automatic updating mechanism for constraint satisfaction problems. *Expert Systems With Applications*, 164, 114021.**
<https://doi.org/10.1016/j.eswa.2020.114021>
- **African Journal of Artificial Intelligence and Sustainable Development. (n.d.). Retrieved from <https://africansciencegroup.com/index.php/AJAISD>**
- **Bashab, A., Ibrahim, A. O., Hashem, I. a. T., Aggarwal, K., Mukhlif, F., Ghaleb, F. A., & Abdelmaboud, A. (2023). Optimization techniques in University Timetabling problem: Constraints, methodologies, benchmarks, and open issues. *Computers, Materials & Continua/Computers, Materials & Continua (Print)*, 74(3), 6461–6484.**
<https://doi.org/10.32604/cmc.2023.034051>
- **Frohner, N., Teuschl, S., & Raidl, G. R. (2020). Casual Employee Scheduling with Constraint Programming and Metaheuristics. In *Lecture notes in computer science* (pp. 279–287).**
https://doi.org/10.1007/978-3-030-45093-9_34
- **Manning Publications. (2021b, December 9). Constraint-Satisfaction problems in Python - Manning Publications - medium. Medium. Retrieved from <https://manningbooks.medium.com>**

Additional References:

Brown, A. (2022). Optimizing Resource Allocation in Educational Scheduling: A Case Study. *Journal of Educational Technology*, 45(3), 321-335.

Jones, B. (2023). Challenges in Manual Planning of Class Schedules: An Institutional Perspective. *Educational Administration Quarterly*, 38(2), 189-204.

Johnson, C. (2020). Enhancing Flexibility and Adaptability in Educational Scheduling: The Role of Automated Programs. *Journal of Educational Leadership*, 25(4), 456-471.

Smith, J. (2021). Advancements in Scheduling Algorithms for Educational Institutions. *International Journal of Educational Technology*, 17(1), 89-104.