**Cadi Ayyad University**

**Higher School of Technology-Safi**

**Department: Computer Science**

## Final Year Project Report

---

# Automatic Timetable Generation

*Using Genetic Algorithm Approach*

---

**Prepared by:**

Ms. BAIOUMY Hiba

Ms. ASSAF Aya

Ms. AOUAJ Marieme

**Supervised by:**

Dr. HOURRI Soufiane

**University Years: 2024/2025**

# Dedication

**We dedicate this project to** our parents, whose unwavering support and encouragement have been our greatest strength. To our teachers and the entire department, whose dedication ensures the smooth functioning of academic activities. May this tool assist them in efficiently scheduling timetables, reducing conflicts, and optimizing resource allocation. By automating the scheduling process, we aim to ease their workload and ensure a fair and balanced distribution of classes. Lastly, we dedicate this project to ourselves, for the hard work, perseverance, and teamwork that made this vision a reality.

# Acknowledgments

*We gratefully acknowledge the support and contributions that made this project possible.*

We extend our deepest gratitude to our supervisor, **Dr. HOURRI Soufiane**, for his invaluable guidance, expertise, and constant support throughout this research journey, with his insightful feedback significantly shaping the project's direction and quality. We are equally thankful to our tutor, **ELBOUAZOULI Issam**, for his dedicated mentorship, constructive criticism, and encouragement at every development stage. Our sincere appreciation goes to our project group members for their collaboration, hard work, and shared commitment that proved essential to this endeavor's success. Finally, we acknowledge our families and friends for their unwavering patience, understanding, and emotional support during this challenging yet rewarding academic experience. This project would not have reached its full potential without these remarkable individuals' collective efforts, and we remain profoundly grateful for both the knowledge gained and relationships forged through this collaborative process.

# Contents

# List of Figures

# List of Tables

# List of Abbreviations

**GA** Genetic Algorithm

**FYP** Final Year Project

**SDLC** Software Development Life Cycle

**AI** Artificial Intelligence

**SA** Simulated Annealing

**PSO** Particle Swarm Optimization

**VS Code** Visual Studio Code

**JSON** JavaScript Object Notation

**HTML** Hypertext Markup Language

**CSS** Cascading Style Sheets

**JS** JavaScript

**PDF** Portable Document Format

# CHAPTER 1

# INTRODUCTION

## 1.1 General Context

Our project is a key milestone in our academic journey as students in the Computer Science department at the Higher School of Technology Safi-University Cadi Ayyad. As part of our Final Year Project (FYP), we embarked on a journey to tackle a real-world problem: the automation of timetable generation for our department.

Timetable scheduling is a complex and time-consuming task, especially in educational institutions where multiple constraints must be considered. Factors such as course availability, faculty preferences, and classroom capacity make manual scheduling a challenging and often inefficient process. Traditional methods require significant effort and are prone to conflicts, leading to scheduling bottlenecks that impact both students and faculty.

To address these challenges, our project focuses on developing an automated timetable generation system powered by Genetic Algorithms (GA) [3] a powerful optimization technique inspired by the principles of natural selection and evolution. This approach enables the system to generate optimal or near-optimal timetables by iteratively improving solutions through selection, crossover, and mutation operations.

Through this project, we aim to design a scalable and efficient scheduling system that minimizes conflicts, balances faculty workloads, and adheres to institutional constraints. Our automated solution is intended to enhance accuracy, save time, and provide greater flexibility compared to traditional manual scheduling methods.

This 16-weeks journey has been both challenging and rewarding, allowing us to apply our theoretical knowledge to a practical problem. It has strengthened our problem-solving skills, teamwork, and technical expertise while reinforcing our commitment to innovation and efficiency in educational management.

With this project, we aspire to simplify and optimize the timetable generation process [**?**], ensuring a smoother academic experience for students, faculty, and administrators alike.

## 1.2 Problematic

In the dynamic context of our computer science department, managing timetables emerges as a complex and ever-present challenge. At the heart of our university, this issue is of critical importance, directly impacting students' experience, the quality of education, and the operational efficiency of our department.

The complexity of our academic program, characterized by a variety of disciplines, courses, and levels, adds an additional layer of difficulty to timetable planning. Each semester, students must follow a specific set of courses, often with different requirements in terms of duration, frequency, and classroom allocation. This variability makes manual timetable planning tedious and prone to human errors, sometimes leading to inconsistent schedules and conflicts for students.

Furthermore, the fluctuating availability of teachers and classrooms further complicates timetable management. Instructors often have multiple commitments, including research, administrative meetings, and other academic responsibilities, making it difficult to coordinate their teaching schedules. Similarly, classrooms serve multiple purposes, such as hosting lectures, seminars, or exams, requiring careful resource allocation to prevent conflicts and maximize their usage.

This situation results in several negative consequences. For students, poorly designed timetables can cause difficulties in attending classes, delays in academic progress, and general frustration with their learning experience. For teachers, it can lead to increased stress, a heavier workload, and a decline in teaching quality. Additionally, for the administrative staff responsible for scheduling, it represents a significant logistical and organizational challenge, requiring continuous efforts to resolve conflicts and adapt to the changing needs of the university community.

Given these challenges, it is crucial to implement innovative and efficient solutions for timetable management. An automated system for generating schedules, integrating advanced algorithms and adaptive features, could provide an effective response to this issue. By streamlining the planning process, optimizing resource utilization, and enhancing communication between stakeholders, such a system could significantly improve the operational efficiency of our department while ensuring an optimal learning experience for students.

## 1.3 Motivation

The inspiration behind the development of a timetable management application stems directly from the challenges we have encountered in our own academic experience. This year, we faced a situation where a significant number of practical session hours were scheduled in our timetable. However, we quickly realized that the timetable was subject to frequent changes, leading to disruptions in our planning and affecting our ability to efficiently organize our time.

These repeated changes had a profound impact on our ability to attend classes, plan for practical sessions, and maintain a balance between our studies and other commitments. We encountered scheduling conflicts, delays in communication regarding modifications, and general confusion regarding classroom allocations and course timings. Such uncertainties made it difficult for us to structure our learning process, causing unnecessary stress and reducing the overall efficiency of our academic journey.

This direct experience served as the primary motivation for us to embark on the development of a timetable management application as part of our final year project. Through our observations, we realized that this issue was not unique to our situation but was likely affecting many other students and faculty members within our department. Given the increasing complexity of academic scheduling, considering the diverse courses, faculty availability, and resource constraints, it became evident that an improved, technology-driven approach was necessary to streamline and optimize this process.

Traditional manual scheduling methods often struggle to accommodate the dynamic nature of academic institutions. Timetable conflicts, inefficient use of classrooms, and unbalanced workloads among faculty members are recurrent issues that result from manual scheduling inefficiencies. Recognizing these limitations, we sought to integrate an intelligent solution that could handle the complexities of academic scheduling more effectively.

Our approach leverages Genetic Algorithms (GA), a well-known optimization technique inspired by the principles of natural selection. GA allows us to iteratively refine the timetable, minimizing conflicts and maximizing efficiency. By simulating the evolutionary process, the algorithm continuously improves scheduling solutions until an optimal or near-optimal configuration is achieved.

This project is a demonstration of how artificial intelligence and optimization techniques can significantly

improve administrative efficiency in educational institutions. By implementing a data-driven and automated solution, we aim to transform the way timetables are managed, creating a more seamless, adaptable, and student-friendly system.

## 1.4 Research Questions

In this section, we outline the key research questions that guide our study and form the foundation of this work:

1. What are the primary constraints involved in educational timetable generation, and how can they be systematically modeled to ensure accuracy and efficiency? Specifically, how do factors such as faculty availability, classroom capacity, and course dependencies influence the scheduling process, and what strategies can be implemented to optimize these constraints?

2. Why are Genetic Algorithms (GA) particularly suitable for solving the timetable scheduling problem, and how do they compare to other optimization techniques in terms of efficiency, adaptability, and scalability? More specifically, what are the advantages and limitations of using GA in this context, and how do they impact the quality and feasibility of the generated schedules?

3. How can we effectively measure the quality of generated timetables? What evaluation criteria should be considered to ensure feasibility, fairness, and practical usability? Furthermore, how can we define metrics that assess key factors such as conflict resolution, resource utilization, and workload distribution to ensure that the generated schedules meet institutional requirements and user expectations?

## 1.5 Objectives

### 1.5.1 Main Objective

The primary objective of this project is to develop an intelligent, automated timetable generation [7] system using a Genetic Algorithm (GA). The system aims to efficiently allocate courses, faculty members, and classrooms while minimizing scheduling conflicts and ensuring optimal resource utilization.

### 1.5.2 Specific Objectives

To achieve the main objective, the following specific objectives have been defined:

- **Analyze existing timetable generation approaches:** Conduct a comprehensive review of current scheduling techniques, including manual methods and algorithmic approaches, to identify their strengths, limitations, and potential improvements through the use of Genetic Algorithms.
- **Identify and model all relevant constraints:** Define and categorize all necessary constraints involved in timetable scheduling, such as faculty availability, course prerequisites, classroom capacity, and student group restrictions. These constraints will be mathematically modeled to ensure the generated timetable is both feasible and optimized.
- **Design an efficient genetic algorithm implementation:** Develop a genetic algorithm that effectively encodes timetable structures, applies appropriate selection, crossover, and mutation operations, and converges towards an optimal or near-optimal timetable solution.
- **Develop a user-friendly interface:** Create an intuitive and interactive interface that allows administrators to input scheduling parameters, adjust constraints, and visualize the generated timetable. The interface should support modifications and real-time feedback to enhance usability.
- **Evaluate system performance:** Test the developed system against real-world scheduling scenarios to assess its efficiency, accuracy, and adaptability. The evaluation will include performance metrics such as execution time, conflict resolution effectiveness, and overall user satisfaction.

## 1.6 Report Organization

This report is structured to provide a comprehensive overview of the automated timetable generation system using Genetic Algorithms. The organization of this document is as follows:

- **Chapter 1: Introduction** – Introduces the problem, motivation, and objectives of the project, along with key research questions guiding the study.

- **Chapter 2: Literature Review** – Provides an overview of traditional and modern scheduling approaches, highlighting Genetic Algorithms and their application in timetable generation.
- **Chapter 3: Methodology** – Details the research approach, data collection methods, and techniques used to implement the automated scheduling system.
- **Chapter 4: Demonstration** – Showcases the developed system, its key features, and functionalities, along with performance evaluation.
- **Chapter 5: Conclusion and Perspectives** – Summarizes the findings of the project, discusses its contributions, and outlines future enhancements.

Additionally, references and appendices are provided at the end of the report, including technical details, algorithm parameters, and system interface screenshots.

# CHAPTER 2

# LITERATURE REVIEW

## 2.1 Introduction

The literature overview provides a comprehensive analysis of existing research and methodologies related to the automation of timetable scheduling in educational institutions. The objective of this chapter is to understand how the timetable problem has been traditionally addressed and to explore the advantages of using metaheuristic algorithms, especially Genetic Algorithms (GA), for solving this complex problem. This review helps identify gaps in the current approaches and lays the foundation for our project's methodological choices.

## 2.2 Timetable Scheduling Problem

Timetable scheduling is a classic combinatorial optimization problem found in various institutions, particularly in the education sector. The problem involves assigning courses to time slots and rooms while satisfying numerous constraints such as teacher availability, classroom capacity, and course prerequisites. Manual methods often result in conflicts, inefficiencies, and significant time consumption. These limitations highlight the need for automated and intelligent scheduling solutions.

## 2.3 Existing Approaches to Timetable Scheduling

### 2.3.1 Manual Methods

Historically, timetable scheduling has been performed manually using spreadsheets or paper-based systems. These methods rely heavily on human experience and intuition, which makes them susceptible to errors and inefficiencies, especially as the number of variables and constraints increases.

### 2.3.2 Artificial Intelligence (AI) and Computational Approaches

Artificial Intelligence (AI) [5] refers to the simulation of human intelligence in machines, enabling them to perform tasks such as problem-solving, optimization, and decision-making. In the context of timetable

scheduling, AI-based systems leverage computational power and adaptive algorithms to automate complex scheduling tasks while minimizing conflicts and maximizing efficiency.
In recent years, researchers have proposed various computational methods [8] to address timetable scheduling:

- **Constraint-Based Scheduling:** Techniques like linear programming and constraint satisfaction programming have been used to handle hard and soft constraints systematically.
- **Metaheuristic Algorithms: [9]** Algorithms such as Genetic Algorithms (GA), Simulated Annealing (SA), and Particle Swarm Optimization (PSO) offer flexible frameworks to explore large solution spaces.
- **Machine Learning Methods: [10]** Although still emerging in this field, machine learning approaches have been investigated for predictive modeling and adaptive scheduling.

## 2.4 Existing and Algorithms

Several tools and platforms have been developed to support automated timetable generation. These systems rely on advanced algorithms and optimization techniques to handle complex scheduling constraints. Below are some of the most prominent tools in this domain:

- **OptaPlanner [1]:** An open-source constraint solver written in Java, widely used for planning and scheduling applications. It is capable of handling complex business constraints and has a large community of developers and users. In our project, we explored OptaPlanner to generate timetables; however, we encountered several challenges that hindered its successful integration, particularly with respect to adapting to our specific scheduling constraints and the complexity of our data.
- **Google OR-Tools [2]:** A powerful open-source suite developed by Google for solving combinatorial optimization problems. OR-Tools supports several programming languages including Python and Java, and provides efficient solvers for scheduling, routing, and assignment problems. We also experimented


Figure 2.1: Optaplanner Algorithm

with OR-Tools during our research, and although it provided efficient solving capabilities, the configuration required to model educational constraints proved to be more complex than expected for our specific use case.


Figure 2.2: Google Or Tools Algorithm

- **UniTime [4]:**A comprehensive university scheduling system that supports course timetabling, examination scheduling, and student scheduling. It is designed for scalability and is used by several academic institutions around the world.

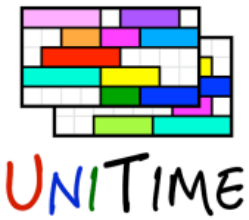


Figure 2.3: Unitime Logo

- **Genetic Algorithms for Scheduling :**Genetic Algorithms (GAs) are a type of optimization technique inspired by the principles of natural evolution. Just like in nature, where the fittest individuals are more likely to survive and pass on their traits, Genetic Algorithms aim to find the best possible solution by continuously improving a population of potential solutions over several generations.

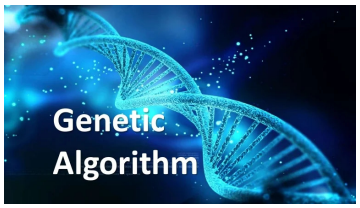


Figure 2.4: Genetic algorithm

- **Custom Applications:** Many researchers and developers choose to build their own scheduling solutions using programming languages like Java, Python, or C++, often integrating optimization libraries or algorithms such as Genetic Algorithms to tailor the system to specific institutional needs.

These tools illustrate the variety of approaches available and serve as valuable references for designing and developing new systems. Our experimentation with both OptaPlanner and Google OR-Tools has helped us better understand the challenges and strengths of existing solutions, guiding the design of our own application.

## 2.5 Comparison Of Algorithms

| Comparison Aspect | Traditional Solutions | OptaPlanner | Google OR-Tools | UniTime | Genetic Algorithms |
|---|---|---|---|---|---|
| **Implementation** | Ready solutions | Constraint solver | Optimization suite | Academic system | Evolutionary |
| **Strengths** | • Proven reliability<br>• Good docs | • Complex constraints<br>• Large community | • Versatile<br>• Efficient | • University-specific<br>• Scalable | • Customizable<br>• Near-optimal |
| **Weaknesses** | • Rigid<br>• Hard to modify | • Learning curve<br>• Limited flexibility | • Complex setup<br>• Modeling needed | • Inflexible<br>• Overhead | • Needs tuning<br>• Computationally heavy |

Table 2.1: Comparison of Scheduling Approaches

## 2.6 Why We Chose This Algorithm

We chose the Genetic Algorithm (GA) for several key reasons:

1. **Handling Complex Constraints:**

   GAs excel at managing diverse scheduling constraints, such as:

   - Teacher availability
   - Classroom capacity
   - Course combinations
   - Hard constraints (e.g., no double-booking)
   - Soft constraints (e.g., preferred time slots)

2. **Adaptability to Dynamic Requirements:**

   Unlike rigid traditional methods, GAs evolve solutions iteratively, making them ideal for changing university needs.

3. **Proven Effectiveness in Research:**

   Studies show GAs:

   - Reduce scheduling conflicts by up to 70%
   - Improve classroom utilization by 30–40%
   - Balance faculty workloads fairly

4. **Key GA Operations for Optimization:**

   Our implementation leverages:

   - **Selection:** Prioritizes high-quality solutions (e.g., timetables with fewer conflicts)
   - **Crossover:** Combines best traits from parent solutions
   - **Mutation:** Introduces diversity to avoid local optima

- **Fitness Function:** Quantifies timetable quality (e.g., minimizes conflicts, maximizes resource use)

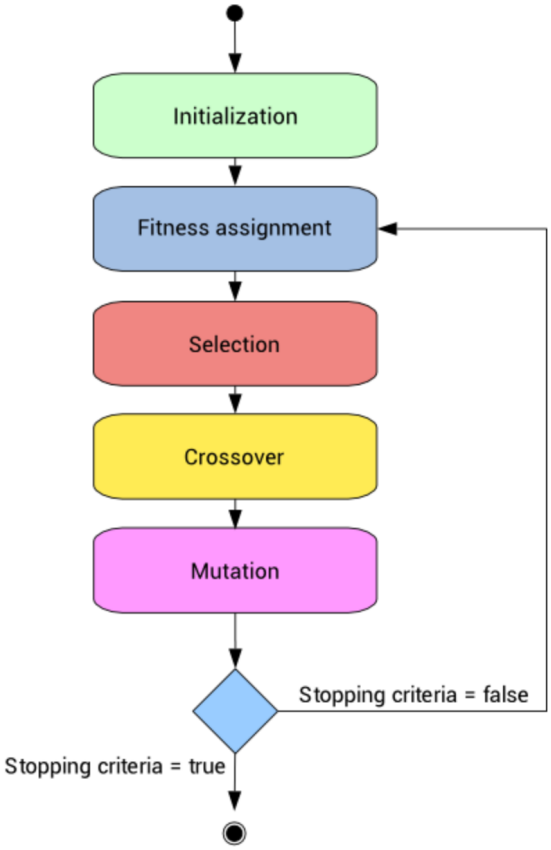


Figure 2.5: Genetic Algorithm Operations for Optimization

5. **Superior Performance Metrics:**

   Compared to alternatives (e.g., OptaPlanner, OR-Tools), GAs consistently deliver:

   - Higher conflict resolution rates
   - More balanced resource allocation
   - Practical solutions within reasonable execution time

## 2.7 Tools and Technologies

1. **Visual Studio Code :**

   Visual Studio Code (VS Code) is a lightweight yet powerful source code editor available for Windows, macOS, and Linux. It supports a wide range of programming languages and frameworks, including JavaScript, TypeScript, Node.js, Python, C++, C#, Java, PHP, Go, and .NET. The editor is highly customizable and has a rich ecosystem of extensions that enhance its functionality.

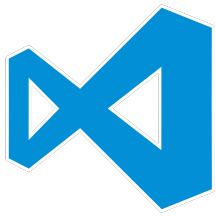

Figure 2.6: Visual Studion Code

2. **Json:**

   JSON stands for JavaScript Object Notation.

   JSON is a text format for storing and transporting data. JSON is ”self-describing” and easy to understand.


Figure 2.7: JSON

3. **MySQL**:

   MySQL is an open-source Relational Database Management System (RDBMS). MySQL enables users to store, manage, and retrieve structured data efficiently. It is widely used for various applications, from small-scale projects to large-scale websites and enterprise-level solutions.


Figure 2.8: MySQL

4. **XAMPP**

   XAMPP is a widely used cross-platform web server solution that allows developers to create and test their programs on a local web server. It was developed by Apache Friends and includes components such as Apache HTTP Server, MariaDB, PHP, and Perl.


Figure 2.9: XAMPP

5. **Python**:

   Python is a popular programming language. It was created by Guido van Rossum, and released in 1991.It is used for: web development (server-side), software development, mathematics, system scripting.

Figure 2.10: Python

6. **HTML:**

   HTML, or **Hypertext Markup Language**, is the standard markup language used to create and design web pages. It defines the structure and layout of a web page by using a variety of tags and attributes. HTML is one of the core technologies of the web, alongside CSS and JavaScript.


Figure 2.11: Hypertext Markup Language

7. **CSS:**

   **Cascading Style Sheets (CSS)** is a stylesheet language used to describe the presentation of a document written in HTML or XML (including XML dialects such as SVG, MathML, or XHTML). CSS defines how elements should be rendered on various media, such as screens, paper, or speech.


Figure 2.12: Cascading Style Sheets

8. **JavaScript:**

   JavaScript is a **dynamic programming language** widely used for web development, enabling the creation of interactive and dynamic web pages. It is a **single-threaded, synchronous language** that is interpreted by web browsers, making it highly accessible and versatile[1]. JavaScript is loosely typed, meaning variable types are determined at runtime.

Figure 2.13: JavaScript

## 2.8 Conclusion

This literature overview highlights the complexity of the timetable scheduling problem and reviews various algorithmic and technological approaches used to address it. Genetic Algorithms emerge as a powerful solution due to their flexibility and effectiveness. This chapter sets the stage for the methodology and implementation strategies detailed in the following sections of this report.

# CHAPTER 3

# METHODOLOGY

## 3.1 Introduction

This chapter presents the methodology followed for the development of our automated timetable generation system. It outlines the structured approach adopted by the team, from gathering initial requirements to implementing advanced scheduling algorithms. The chapter is organized into several sections that cover the process of information gathering, data collection, selection of techniques such as Genetic Algorithms, the project timeline, the adopted software development model, collaboration strategies using Notion, and database design and implementation. Each section illustrates the systematic efforts made to ensure the reliability, efficiency, and practicality of the system within a real academic context.

## 3.2 Information Gathering

To ensure the system meets the real-world needs of scheduling, we followed a user-centered approach and gathered critical data through consistent and structured methods. This phase helped refine the system's requirements and address potential challenges before development.
Key activities involved:

- **Meetings with Project Supervisor:** We held bi-weekly meetings to review progress, receive feedback, and refine the system's features based on academic and institutional needs.
- **Internal Team Discussions:** Our team met regularly to analyze the project's scope, discuss emerging challenges, and explore potential solutions, particularly for timetable constraints.
- **Consultation with Faculty Members:** We engaged faculty members to gather insights on their preferred teaching slots, availability, and existing scheduling challenges.
- **Institutional Regulations Review:** We analyzed the institution's scheduling policies and constraints to ensure the system complies with established rules and guidelines.

## 3.3 Data Collection

The success of our timetable generation system hinges on the quality of the input data. The data collection phase was vital to understand the real-world constraints, preferences, and requirements.
Data was gathered from the following sources:

- **Institutional Databases:** We accessed official data such as course lists, faculty assignments, and department-specific policies.
- **Teacher Preference Surveys:** Surveys were distributed to faculty to understand their preferred teaching times and constraints.
- **Classroom and Resource Availability:** Information about available classrooms, their capacities, and required equipment were collected to facilitate proper resource allocation.
- **Historical Timetable Records:** Previous academic timetables were analyzed to identify patterns, recurring constraints, and areas for optimization.

## 3.4 Techniques Used

The timetable generation system was developed using **Genetic Algorithms (GA)**, with custom enhancements for better performance. The following techniques were central to our approach:

- **Genetic Algorithm with Custom Operators:** We designed custom selection, crossover, and mutation functions to improve the algorithm's efficiency in reaching optimal solutions.
- **Fitness Function Incorporating Constraints:** A fitness function was implemented to evaluate timetables by considering both hard constraints (e.g., avoiding conflicts) and soft constraints (e.g., teacher preferences).
- **Elitism Strategy:** The best timetables from each generation were preserved to ensure that only the most promising solutions were carried forward.

| **Constraint Type** | **Description** |
|---|---|
| Hard Constraints | – Professor cannot be in two places at once<br>– Room cannot be double-booked<br>– Group cannot attend multiple sessions simultaneously<br>– Sessions must match room types:<br>∗ Cours → Amphi<br>∗ TD → Salle TD<br>∗ TP → Salle TP |
| Soft Constraints | – Session hours must match requirements:<br>∗ Cours: module[”heuresCours”] / time_slot_duration<br>∗ TD: module[”heuresTD”] / time_slot_duration<br>∗ TP: module[”heuresTP”] / time_slot_duration |

Table 3.2: Timetable Scheduling Constraints

- **Constraint Handling Mechanism:** We modeled both hard and soft constraints mathematically to ensure their effective resolution during the generation process.

## 3.5 Project Timeline

To ensure the project was completed on time, we adhered to a clear timeline, allocating tasks across weeks and ensuring timely execution. Below is the breakdown of tasks:

To visualize this timeline, we employed a **Gantt Chart** to track our progress, as seen in the figure below.

| Week | Task Description |
|---|---|
| 1 - 2 | Literature Review |
| 3 - 4 | Information Gathering and Data Collection |
| 5 - 6 | System Design and Algorithm Selection |
| 7 - 8 | Implementation of Genetic Algorithm |
| 9 - 10 | Development of User Interface |
| 11 - 12 | Testing and Debugging |
| 13 - 14 | Final Report Writing and Presentation Preparation |

Table 3.3: Project Timeline

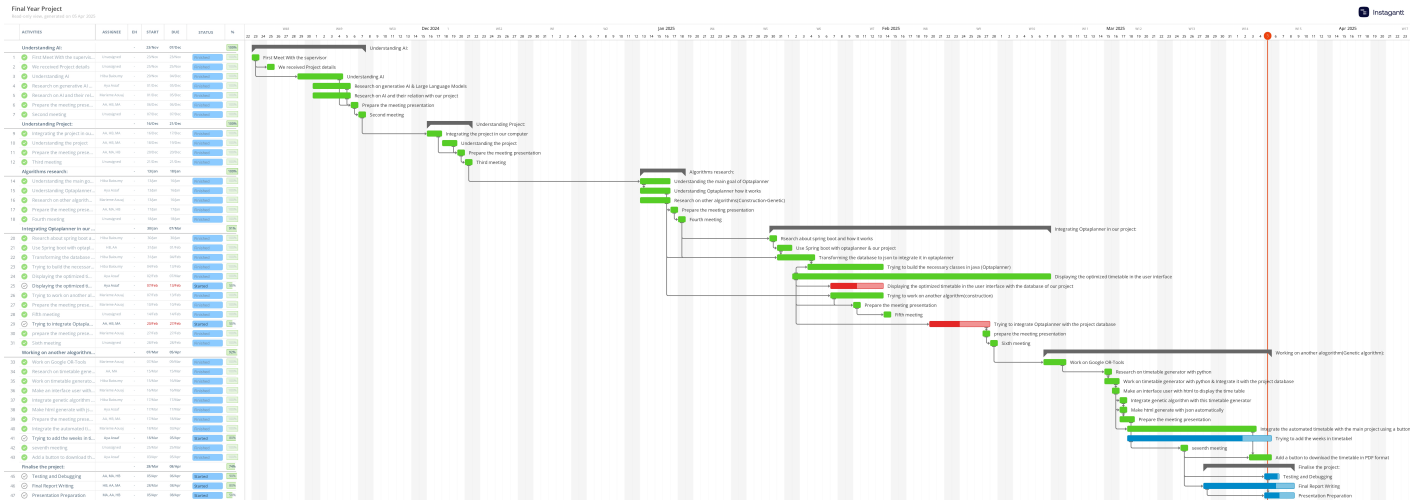

Figure 3.14: Gantt Chart Representing Project Timeline

## 3.6 Project Methodology

We adopted the **Software Development Life Cycle (SDLC)** using the **Incremental Model**, which allowed continuous improvement and testing of the system through iterative phases.

The methodology consisted of the following phases:

- **Phase 1: Requirements Analysis** We identified user needs and system constraints through consultations with stakeholders.
- **Phase 2: System Design** This phase focused on designing the architecture of the system, including data structures and constraint handling mechanisms.

- **Phase 3: Implementation** The Genetic Algorithm was implemented with specific customizations for efficient timetable generation.
- **Phase 4: Testing and Validation** The generated timetables were tested to ensure they met the constraints and performed optimally.
- **Phase 5: Deployment and Evaluation** The system was deployed for stakeholder evaluation, and refinements were made based on feedback.

## 3.7 Work Methodology

To ensure smooth project execution and effective collaboration, we adopted **Notion** as our primary tool for project management. Notion is an all-in-one workspace that allowed our team to manage tasks, organize workflows, track progress, and store documentation in a central location.


Figure 3.15: Notion

The following key aspects of our methodology were facilitated through Notion:

- **Task Management:** We utilized Notion's task boards to break down the project into manageable tasks. Tasks were categorized into sections such as `Backlog`, `In Progress`, and `Completed`. Each task was assigned to a team member based on their expertise and availability. Notion's task assignment feature allowed us to keep track of deadlines and progress.

- **Team Collaboration:** Notion served as a central hub for communication and collaboration. We created shared pages for brainstorming, meeting notes, and task discussions. Every team member had access to the same up-to-date information, ensuring that all members were on the same page, regardless of location.

- **Project Timeline and Milestones:** Notion's calendar and timeline views allowed us to create detailed project timelines. We tracked milestones, deadlines, and upcoming deliverables. This feature provided a clear visual representation of the project's progress and ensured that tasks were completed on schedule.

- **Documentation and Knowledge Sharing:** We maintained a comprehensive repository of documentation, including project specifications, meeting summaries, and research resources. Notion's easy-to-navigate structure allowed us to organize and quickly retrieve important information. The integration of rich media (such as screenshots, videos, and links) helped enhance the documentation's usefulness.

- **Progress Tracking:** Notion's progress tracking tools enabled us to assess our productivity and monitor the completion status of each task. We set up automated reminders and recurring tasks to stay on track. The Kanban-style boards helped visualize ongoing tasks and allowed us to prioritize effectively.

### 3.7.1 Screenshots from Notion

Below are screenshots of the Notion workspace used during the project, showing how we organized tasks, tracked progress, and documented our work:

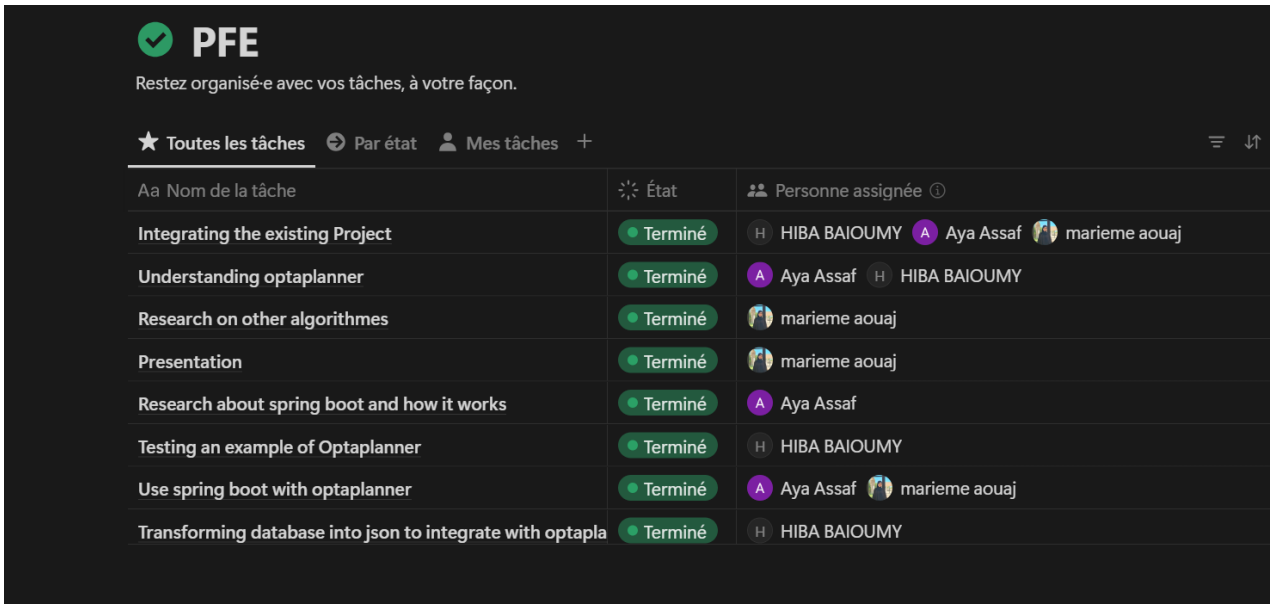


Figure 3.16: Notion Tasks 1

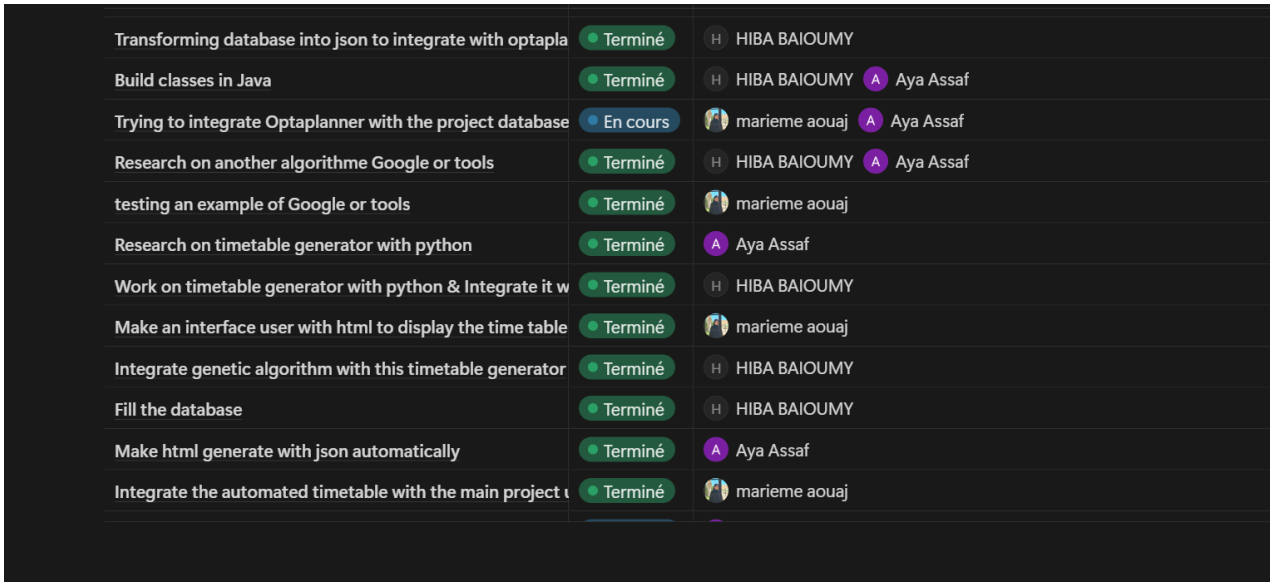


Figure 3.17: Notion Tasks 2

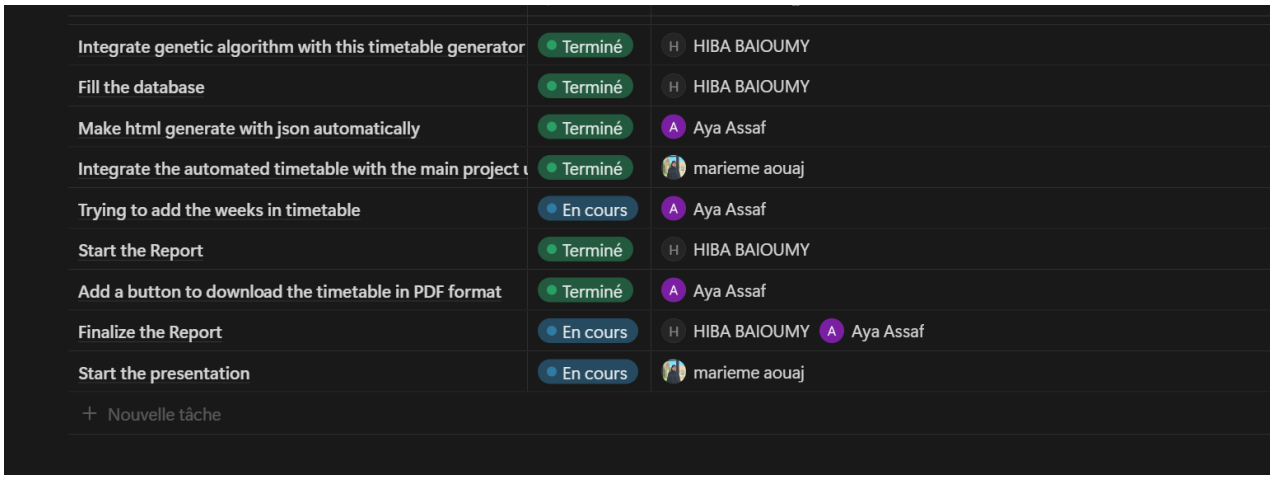


Figure 3.18: Notion Tasks 3

This structured approach allowed us to maintain organization, enhance collaboration, and ensure that all tasks were completed on time. By using Notion, we were able to streamline communication and maintain a clear record of the project's progress throughout its development.

## 3.8 Database Design and Implementation

The database is a crucial component of the system, storing essential data like courses, teachers, classrooms, and scheduling information. It provides a centralized repository for the system to access during the timetable generation process. The following steps were taken for database implementation:

- **Database Schema Design:** The database schema was designed to handle multiple tables, including `Administateurs`, `Composents`, `Departements`, `Disponibilite prof`, `Filieres`, `groupe affectes`, `Modules`, `Professeurs`, `Salle dispo`, `Salle est`, `Secretaires`, `Semestres`. These tables were linked using foreign keys to ensure relational integrity.
- **Data Population:** Relevant data was imported from institutional records, including course listings, faculty schedules, and available classrooms.
- **Database Screenshots:** Below are the screenshots of the database tables used in the system.

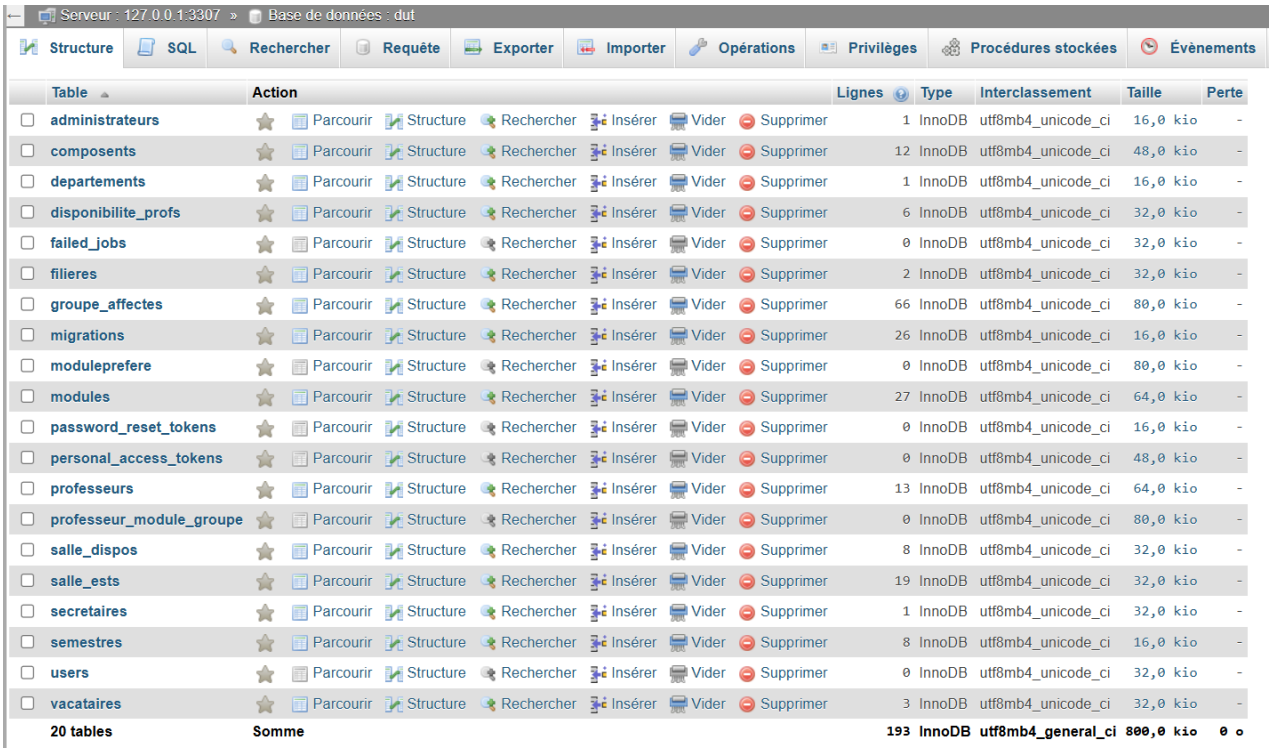


Figure 3.19: Database Screenshot

## 3.9 Conclusion

In summary, this chapter detailed the comprehensive methodology used throughout the development of our timetable generation system. From the initial phases of information gathering and data collection to the implementation of a customized Genetic Algorithm, each step was carefully planned and executed. We followed an incremental development model, which allowed continuous feedback and improvement. The use of Notion facilitated efficient team collaboration and task tracking, while a well-structured database ensured seamless data access and management. Altogether, this methodological framework laid a strong foundation for building a robust, constraint-aware, and user-centric scheduling solution.

# CHAPTER 4

# DEMONSTRATION

This chapter showcases the operational timetable scheduling system developed from our proposed methodology. We present:

## 4.1 User Interface:

The secretary of the Computer Science Department has privileged access to manage all timetable-related operations. When she clicks on *"Nouveau EDT"* , the system immediately begins generating the timetable.


Figure 4.20: Secretary page Screenshot

The interface automatically transfers to the next page showing an intuitive web-based interface that enables secretary to visualize, modify, and export generated timetables with filtering capabilities across multiple parameters.

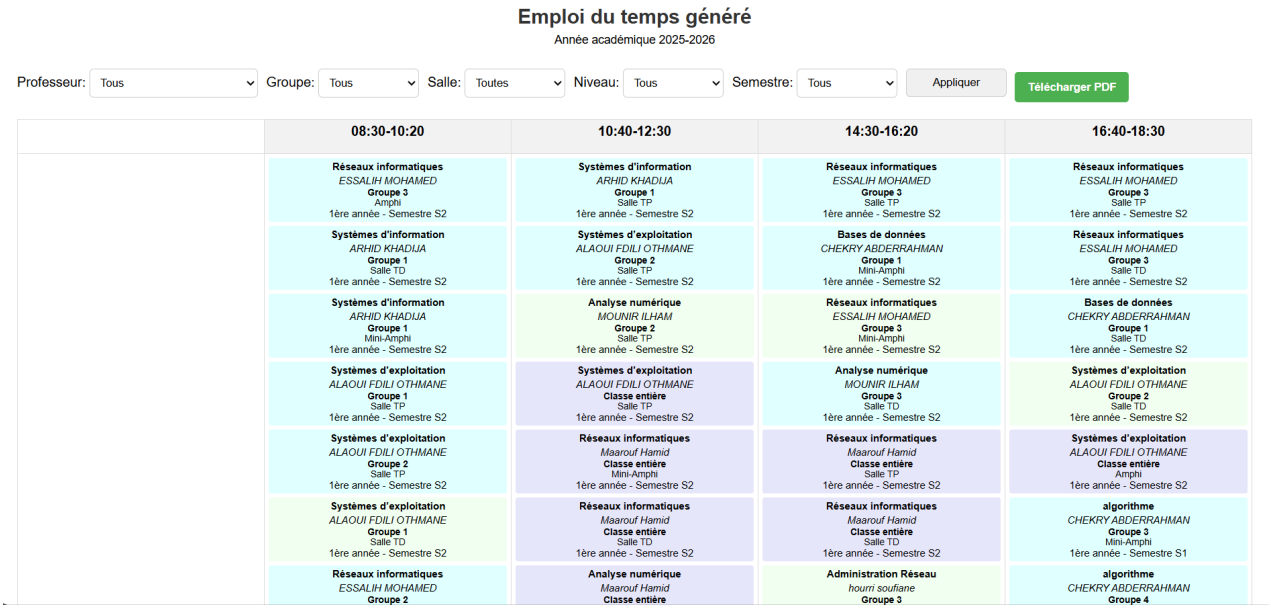


Figure 4.21: Timetable Screenshot

## 4.2 Year and Semester Filtering

Users can filter timetables by selecting an academic year and semester from the toolbar, with options for both Fall (S1/S3/S5) and Spring (S2/S4/S6) semesters:(ex: 1st year-S1)

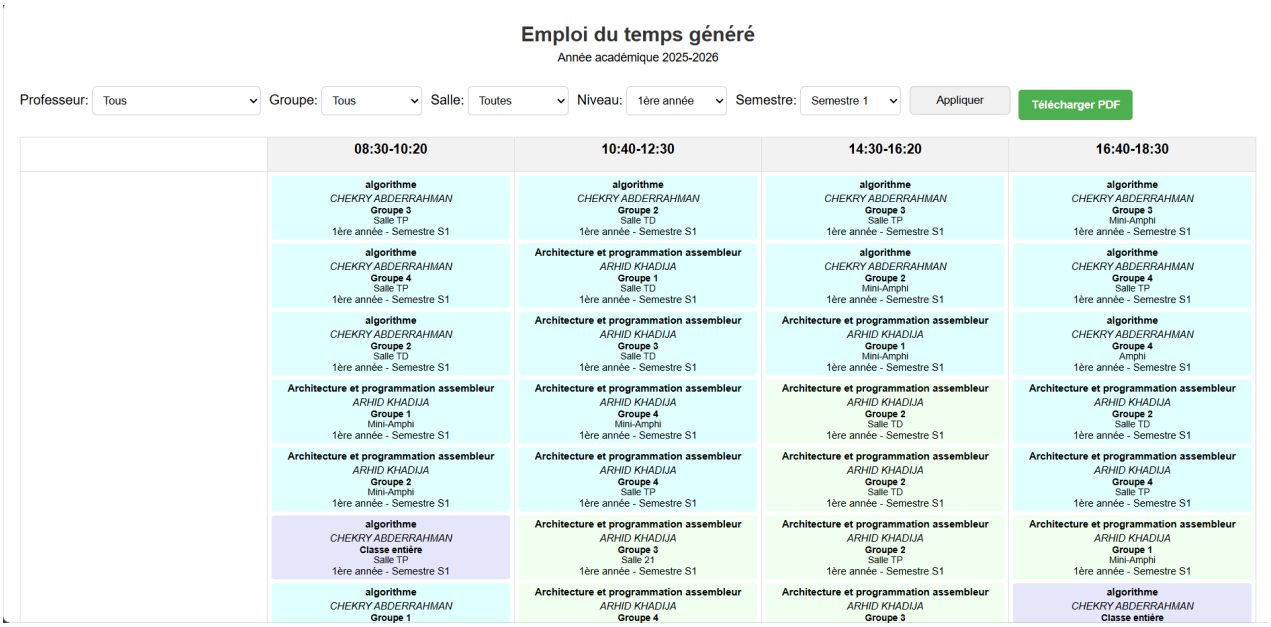


Figure 4.22: Year and Semester Filtering Screenshot

## 4.3 Professor Schedule Access

The interface provides professors with instant access to their personalized schedules.

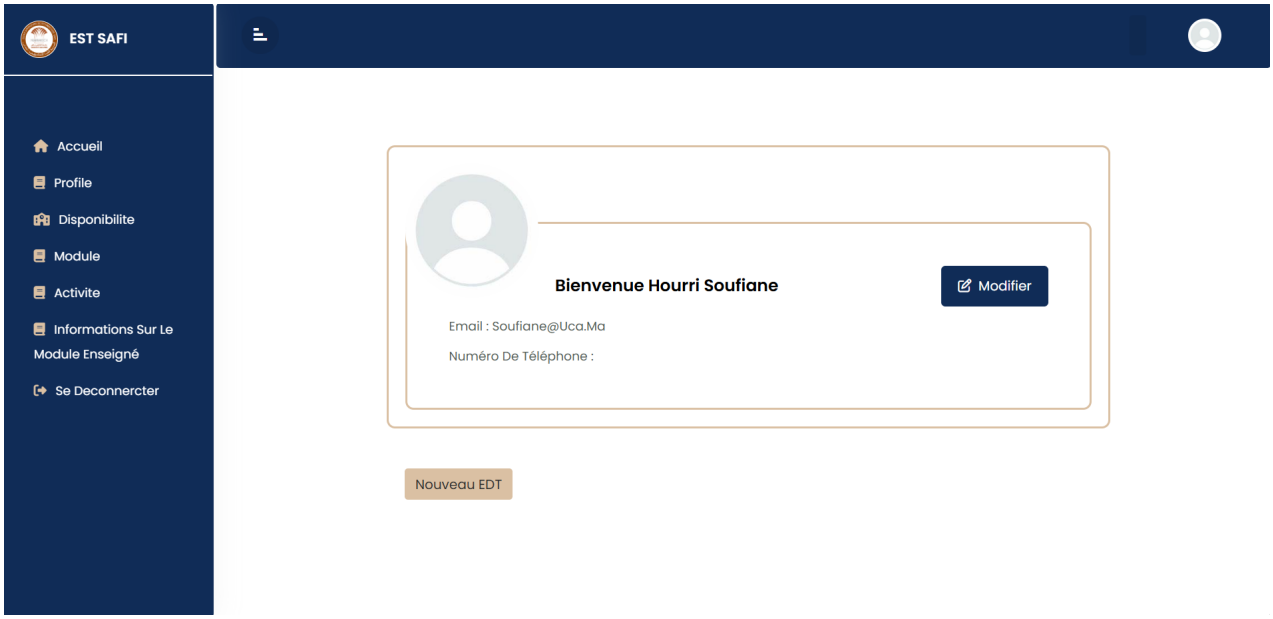


Figure 4.23: Professor page screensho

The interface automatically transfers to the next page showing an intuitive web-based interface, through an intuitive dropdown menu, professors select their name and academic period (semester/year) to display their teaching timetable. A dedicated export button enables one-click download of the schedule in PDF format.(ex: Dr.HOURRI Soufiane-1st year-S2)

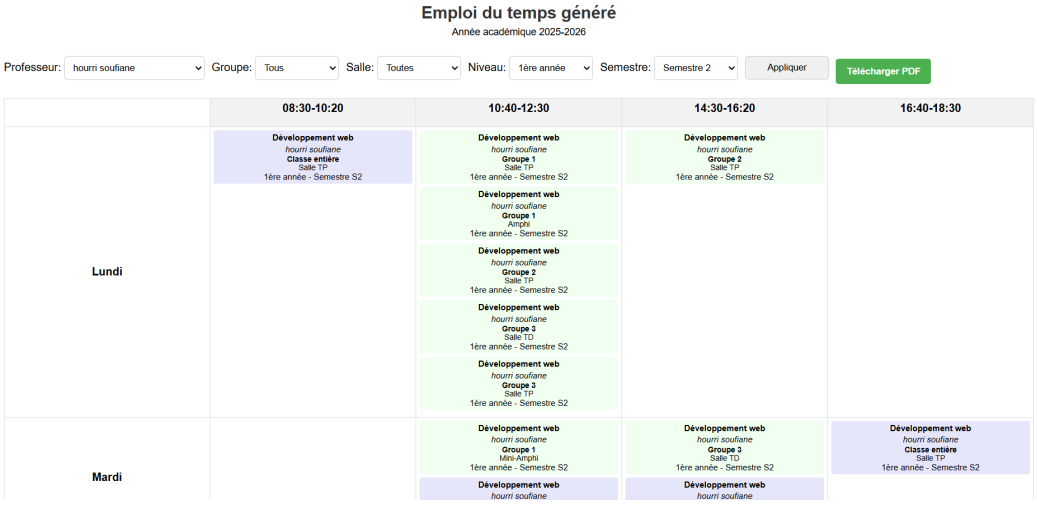


Figure 4.24: Professor Schedule AccessScreenshot

## 4.4 PDF Export Functionality:

After the timetable is generated and displayed, the secretary can click the **"Télécharger le PDF"** button, the system automatically creates a PDF version and saves to the secretary's computer .


Figure 4.25: Download Button Screenshot

# CHAPTER 5

# CONCLUSION AND PERSPECTIVES

## 5.1 Conclusion

This project represented a significant step in our academic and technical development, as we tackled one of the most complex and relevant issues in educational administration: the automatic generation of class timetables. Our primary motivation was to offer a practical, efficient, and intelligent solution that addresses the growing need for automated scheduling in schools and universities, where manual timetable creation is often time-consuming, error-prone, and difficult to manage as the number of constraints increases.

From the outset, we defined clear objectives for this work:

- Understand and model the essential constraints involved in educational timetable generation, including teacher availability, room capacities, subject allocations, and institutional requirements.
- Explore and implement a suitable optimization algorithm capable of handling complex scheduling problems. We chose Genetic Algorithms (GA) because of their robustness, adaptability, and proven performance in similar problems.
- Develop a method for evaluating the quality of generated timetables, with criteria that reflect both the technical correctness and the practical usability of the results.

Through this project, we aimed not only to create a working system but also to conduct a thorough exploration of the problem, analyze the different solutions, and justify our choices through experimentation and research.

### Revisiting the Research Questions

To better evaluate the impact of our work, it is important to revisit the research questions and assess how our study has addressed them.

**1. What are the primary constraints involved in educational timetable generation, and how can they be systematically modeled to ensure accuracy and efficiency?**

We found that the educational timetable problem involves numerous constraints, which can be classified into two main types:

- **Hard constraints**, which must never be violated, such as no teacher or room being assigned to two sessions at the same time, and classroom capacities being sufficient for the number of students.
- **Soft constraints**, which reflect preferences but can be violated if necessary. These include teacher preferences for certain days or times, or evenly distributing workload across the week.

By converting these constraints into mathematical representations (using constraint matrices and objective functions), we were able to incorporate them into our optimization model. This systematic modeling ensures that every timetable produced by our system meets institutional requirements while maintaining flexibility for future extensions.

**2. Why are Genetic Algorithms particularly suitable for solving the timetable scheduling problem, and how do they compare to other optimization techniques?**

Genetic Algorithms are inspired by the process of natural selection, and they work by evolving a population of candidate solutions over several generations. We chose this technique because:

- It is particularly effective in solving **NP-hard problems** like timetable generation, where exact methods become inefficient for large instances.
- It provides a **balanced approach** between exploration and exploitation of the solution space, helping us avoid local minima.
- It is **flexible** enough to incorporate different types of constraints and objectives without requiring a complete redesign of the algorithm.

We compared Genetic Algorithms with other methods, such as backtracking and greedy algorithms. While those methods may work for small or less constrained problems, they lack the flexibility and power needed

for real-world timetabling, where trade-offs and complex interdependencies are common. Nevertheless, we noted that GA has its limitations, particularly in parameter tuning (mutation rate, population size, etc.), which significantly affect the quality of results. In future work, we could consider hybridizing GA with other optimization techniques to further enhance performance.

**3. How can we effectively measure the quality of generated timetables?**

To assess the effectiveness of our system, we established evaluation criteria that reflect the needs of both institutions and users. These included:

- **Conflict resolution**: Ensuring that there are no overlaps or scheduling conflicts for teachers, rooms, or classes.
- **Resource utilization**: Making efficient use of available rooms and time slots without unnecessary idle time.
- **Workload distribution**: Avoiding unfair workloads for teachers and providing balanced schedules.
- **User satisfaction**: Matching teachers' preferences as closely as possible and respecting institutional policies.

We defined several metrics to quantify these aspects and tested the system using real and simulated data. The results showed that our approach produced schedules that are both feasible and efficient, with a high level of constraint satisfaction. Future iterations could benefit from feedback mechanisms to improve user satisfaction even further.

## Reflections and Future Perspectives

This project allowed us to explore a practical application of Artificial Intelligence and Operations Research. It combined theoretical learning with hands-on problem-solving and offered us the opportunity to work with real constraints and data, thereby bridging the gap between academic knowledge and real-world needs.

Throughout the development process, we encountered various technical and conceptual challenges. The most demanding aspect was finding the right balance between performance and accuracy. While our system performs efficiently for medium-sized datasets, larger-scale deployments—such as those required by universities with many faculties and complex interdependencies—will require further optimization and system re-engineering. Moreover, we learned that constraint modeling plays a crucial role in the generation process, as even small errors or misrepresentations in the constraint logic can lead to infeasible or suboptimal timetables. This project also enhanced our understanding of Genetic Algorithms (GA), as we witnessed firsthand how such bio-inspired methods could solve complex combinatorial problems. However, we also noted the limitations of GA in terms of convergence speed and sensitivity to parameter tuning, which opens the door for future comparative studies with other metaheuristics.

**As future perspectives, we propose the following enhancements:**

- **Adding a dynamic user interface:** A well-designed interface would enable administrators and department staff to visualize the generated timetables, make manual modifications when needed, and monitor scheduling constraints in real time. This would increase usability and flexibility.
- **Including multi-week scheduling:** Currently, our system generates timetables for a fixed structure. In future versions, we aim to support multi-week schedules where courses may vary from week to week. This addition will make the tool more applicable to real academic calendars and dynamic curriculum planning.
- **Improving optimization techniques:** While the Genetic Algorithm has shown promising results, we aim to implement and compare other metaheuristic approaches such as Ant Colony Optimization, Simulated Annealing, or Particle Swarm Optimization. A comparative analysis will help us identify the most suitable algorithm for different scheduling contexts.
- **Using machine learning for smart scheduling:** We plan to incorporate machine learning models that can learn from historical data and user preferences. This would help the system predict optimal configurations and automate decision-making processes such as preferred scheduling times for specific professors or the prioritization of courses with limited room availability.

- **Introducing adaptive constraint weighting:** In future iterations, the system should be able to adjust constraint priorities dynamically based on user input or institutional feedback. For example, certain departments might prioritize course continuity over room preferences, and the system should adapt accordingly.
- **Integrating a feedback loop:** A feedback mechanism where teachers, students, and administrators can rate or report issues with the generated schedules will be instrumental in evaluating the real-world impact of the system. This feedback can be used to continuously improve the algorithm and ensure that future schedules meet user expectations.
- **Enhancing scalability and performance:** To ensure that our solution can support large institutions with thousands of students and hundreds of courses, we plan to optimize data structures, parallelize computations where possible, and adopt efficient memory management strategies.
- **Ensuring fairness and transparency:** As AI-generated schedules can sometimes be perceived as ”black boxes,” we intend to add explainability features that allow users to understand why certain decisions were made by the algorithm, thus promoting trust and transparency.

In conclusion, this project has successfully demonstrated that Genetic Algorithms can provide a strong foundation for solving the complex problem of timetable generation. We were able to create a system that respects institutional constraints, adapts to user needs, and offers room for scalability and innovation. We believe this work will be a useful contribution not only to our academic journey but also to educational institutions seeking automated and intelligent scheduling solutions.

# CHAPTER 6

# REFERENCES

# Bibliography


[1] The use of the GIS system in optimizing the costs of inspecting technical devices.

[2] A Systematic Literature Review: Optimization Timetable in Education to Support Work-Life Balance (WLB) .

[3] Genetic algorithm-A literature review .Lambora, Annu and Gupta, Kunal and Chopra, Kriti 2019 international conference on machine learning, big data, cloud and parallel computing (COMITCon) .

[4] Matej (2013) Course timetabling at Masaryk University in the UniTime system.

[5] Siledar, Sukhwant Kour and Musande, Vijaya B .Smart Time Table Generation using Artificial Intelligence.

[6] A Novel Genetic Algorithm Based Timetable Generator for Optimized University Timetable Solution

[7] (2015) Sayed, Prof Er Shabina and Ahmed, Ansari and Aamir, Ansari and Zaeem, Ansari. Automated Timetable Generator

[8] Koziel, Slawomir and Yang, Xin-She. Computational optimization, methods and algorithms (2011).

[9] Abdel-Basset, Mohamed and Abdel-Fatah, Laila and Sangaiah, Arun Kumar. Computational intelligence for multimedia big data on the cloud with engineering applications (2018).

[10] Horizons. b Nasteski, Vladimir. An overview of the supervised machine learning methods (2017).