LanguageTeams Planning Tool

Project Plan

Bachelor's degree Applied Computer Science
Artificial Intelligence

Rob Verbeek

Academic year 2024-2025

Campus Geel, Kleinhoefstraat 4, BE-2440 Geel

LID VAN ASSOCIATIE KU LEUVEN

THOMAS MORE

Table of Contents

1 WHO 4

2 WHAT 5

2.1 Application Functionalities 5
2.2 Business Rules 6
2.2.1 General Business Rules 6
2.2.2 Business Rules based on Course 7
2.3 Smart Scheduling with AI 10
2.3.1 Agentic AI 10
2.3.2 Constraint Satisfaction Problem 11
2.3.3 Genetic Algorithm 11
2.3.4 Conclusion 11
2.4 Technology stack 12
2.4.1 Mandatory Technology 12
2.4.2 AI Technologies 12

3 WHY 13

4 PLANNING 2

4.1 Research Phase 2
4.2 Testing Phase 3
4.3 Implementation Phase 4

5 COMMUNICATION 5

1 WHO

This project takes place at **B_Robots**, a company specialized in Robotic Process Automation (RPA), Low Code, Intelligent Document Processing (IDP), AI automation, and more recently, Process Orchestration.

B_Robots was founded in 2017 and started with RPA. In 2019, they expanded into Low Code app development, in 2020 into Intelligent Document Processing, in 2021 into AI automation, in 2024 into Agentic AI, and since 2025 also into Process Orchestration.

These technologies automate repetitive tasks, allowing employees to focus on more valuable work. Additionally, these solutions can run 24/7, which improves clients' efficiency and productivity.

2 WHAT

The assignment for this internship is for the client LanguageTeams and its subsidiary Bxl Academy. This is a language school where students can learn a language at any level in small groups, with a strong focus on interaction and communication.

The problem lies in scheduling the class timetable for the teachers. Currently, this is done manually through coordination between teachers and students. This is because students themselves indicate when they are available for lessons, and the teachers' schedules need to be flexibly adjusted to accommodate this. In addition, teachers need to make arrangements among themselves regarding the use of classrooms.

To address this, an application will need to be developed to make this entire process more convenient for the teachers.

2.1 Application Functionalities

- Teachers and staff can log in and receive a personalized dashboard.
- Teachers can submit their availability for lessons.
- An algorithm generates a schedule based on matching availability between students and teachers.
- Teachers can approve or modify the proposed schedule.
- Once approved, the schedule becomes final and is added to the calendar of the assigned teachers.
- Upon approval, the schedule is also sent to the relevant students.
- Staff members have access to an overview of all:
    - teachers
    - students
    - courses
- Staff can manage teachers within the teacher overview (add, remove, edit).
- Staff have access to a Language & Room Management overview.
- Within Language & Room Management, staff can manage languages, classrooms, and locations (add, remove, edit).

2.2 Business Rules

When generating the class schedule using AI, several fixed business rules must be strictly followed. These rules define how students are grouped, how teachers and classrooms are assigned, and how communication is handled. Additionally, the different course formats and their associated scheduling requirements must be respected.
By integrating these guidelines into the algorithm, it ensures that the schedule is realistic, efficient, and aligned with the organization's requirements.
Below are the key business rules the AI must adhere to when creating the schedule.

2.2.1 General Business Rules

1. **Language Levels**
   The following language levels are used: A0, A0+, A1, A1+, A2, A2+, B1, B1+, B2, B2+, C1.

2. **Grouping Rules**
   Students with a half-level difference may be placed in the same group (e.g., A0/A0+ or A2+/B1). Groups should be formed as large as possible to minimize the total number of scheduled teaching hours.

3. **Long-Term Enrolments**
   For students enrolled for a longer period (e.g., 12 weeks), their language level progression must be considered. Reference should be made to the document outlining the number of lesson hours needed to advance to the next level.

4. **Group Size**
   A group may consist out of a maximum of 8 students.

5. **Student Enrolments**
   Students can only be scheduled once their payment status in Team leader is marked as "paid." A response from the web developer regarding integration with the ECWID web shop is still pending.

6. **Classroom Assignment**
   Classrooms are assigned based on availability.

7. **Teacher Assignment & Schedule Changes**
   Teachers are assigned based on their availability. Schedule changes are allowed if necessary, with a maximum shift of 30 minutes. However, the total number of scheduled hours must remain unchanged.

8. **Standard Email Communication**
   Every Thursday at 15:00, a standard email is sent out with the class schedule or a notice of a possible course delay.

9. **Attendance List**
   The final attendance list is sent to the language teacher based on the final schedule. For online classes, this should preferably be a digitally signed document, for example with YouSign.

2.2.2 Business Rules based on Course

1. **Intensive course (FR, NL, EN)**

   - 20 hours in 2 weeks (or 40 hours in 4 weeks).
   - Classes from Monday till friday, 9:30 - 11:30.
   - Schedule based on group size:
     - 1 student: 10:30 - 11:30 (notified by email).
     - 2 students: 10:00 - 11:30 (notified by email).
     - 3-8 students: 9:30 - 11:30.

2. **Intensive PREMIUM Course (FR, NL, EN)**

   - 20 hours in a small group + 4 hours of private lessons in 2 weeks (or 40 + 8 in 4 weeks).
   - Group lessons from Monday to Friday, 9:30 AM – 11:30 AM.
   - Private lessons on Tuesday and Thursday, 12:30 PM – 1:30 PM.
   - Group lesson schedule is identical to the standard intensive course.

3. **Hyper Intensive Course (FR, NL, EN)**

   - 40 hours in a small group in 2 weeks (or 80 hours in 4 weeks).
   - Morning schedule identical to the intensive course.
   - Afternoon lessons from 12:30 PM – 2:30 PM, based on group size:
     - 1 student: 12:30 - 13:30 (notified by email).
     - 2 students: 12:30 - 14:00 (notified by email).
     - 3-8 students: 12:30 - 14:30.

4. **Hyper Intensive PRO Course (FR, NL, EN)**

   - 40 hours in 2 weeks, with 20 hours of group lessons and 20 hours of private lessons (or 80 in 4 weeks).
   - Morning schedule identical to the intensive course
   - Private lessons daily from 12:30 - 14:30.

5. Avondcursussen (FR, NL, EN)

- 20 hours in 5 weeks (or 40 hours in 10 weeks).
- Monday & Wednesday or Tuesday & Thursday, 18:30 – 20:30.
- If fewer than 3 students enroll:
    - Course is postponed by 1 week (up to 4 times).
    - Guaranteed start on week 4.
- Schedule based on group size:
    - 1 student: 12:30 - 13:30 (notified by email).
    - 2 students: 12:30 - 14:00 (notified by email).
    - 3-8 students: 12:30 - 14:30.
- Progress tracked weekly in Google Sheets.

6. Online Progressive Courses (FR, NL, EN)

- 20 uur in 5 weeks (or 40 hours in 10 weeks).
- Monday & Wednesday or Tuesday & Thursday, 9:30 – 11:30 or 18:30 – 20:30.
- If fewer than 3 students enroll:
    - Course is postponed by 1 week (up to 4 times).
    - Guaranteed start on week 4.
- Schedule based on group size:
    - 1 student: 18:30 - 19:30 (notified by email).
    - 2 students: 18:30 - 20:00 (notified by email).
    - 3-8 students: 18:30 - 20:30.
- Online Teams meeting created based on the final schedule.

7. Online Intensive Courses (FR, NL, EN)

- 20 hours in 2 weeks (or 40 hours in 10 weeks).
- Monday & Wednesday or Tuesday & Thursday, 9:30 - 11:30 of 18:30 - 20:30.
- If only one student enrolls:
    - Ask if a 1-week delay is possible.
    - If not, schedule the course with a shorter lesson
- Schedule based on group size:
    - 1 student: 10:30 - 11:30 of 18:30 - 19:30 (notified by email).
    - 2 students: 10:00 - 11:30 of 18:30 - 20:00 (notified by email).
    - 3-8 students: 9:30 - 11:30 of 18:30 - 20:30.
- Online Teams meeting created based on the final schedule.

2.3 Smart Scheduling with AI

To automate and optimize the class scheduling process, artificial intelligence (AI) is being used. AI plays a crucial role in matching the availability of teachers and students, avoiding scheduling conflicts, and efficiently assigning classrooms. This reduces administrative workload and ensures a flexible and dynamic timetable.

To determine the most suitable AI technology, research has been conducted into various approaches. During this project, the most promising methods will be further evaluated and tested in order to implement the best possible solution.

2.3.1 Agentic AI

Agentic AI leverages agents which are autonomous software programs that perform specific tasks and communicate with each other to achieve a shared goal. In this case, agents assist in creating a class schedule by negotiating and resolving conflicts.

The following agents could be deployed:

- **Scheduling Agent**: Drafts the initial version of the class schedule based on provided availabilities.
- **Teacher Agent**: Represents the teachers and negotiates based on their availability and preferences.
- **Student Agent**: Does the same for students, considering their preferences and constraints.
- **Negotiation Agent**: Acts as a mediator in case of conflicts. If a teacher rejects a proposed schedule, this agent can engage in a chatbot conversation, process the feedback, and propose a revised schedule.

By using this **multi-agent system**, conflicts can be resolved automatically without manual intervention, significantly increasing the efficiency of the scheduling process. Additionally, the chatbot offers added value for teachers by allowing them to provide direct feedback in a user-friendly way.

2.3.2 Constraint Satisfaction Problem

Another solution is to use Constraint Programming (CP) with the CP-SAT solver to generate a class schedule. In this approach, the schedule is constructed through constraints that define which combinations of lessons, teachers, classrooms, and time slots are possible.

For this method, various constraints are defined, such as teacher and student availability, avoiding double bookings, and optimizing free time slots. CP-SAT then searches for a solution that satisfies all these constraints and tries to optimize it where possible.

To make this process more efficient, extensions like a warm start can be applied. This uses a previously generated schedule as a starting point so the solver doesn't have to start from scratch each time. This prevents repeatedly generating the same incorrect schedules and speeds up optimization.

With CP-SAT, large and complex scheduling problems can be solved efficiently. The solver combines Constraint Programming, Mixed Integer Programming (MIP), and SAT-solving, making it one of the most powerful techniques for generating class schedules.

2.3.3 Genetic Algorithm

Another possible solution for generating a class schedule is using a Genetic Algorithm (GA). This algorithm is inspired by natural evolution and searches for the best solutions through selection, crossover, and mutation.

The method starts with a population of random schedules. Each schedule is evaluated using a fitness function that measures how well it meets the given requirements, such as teacher availability and avoiding double bookings.

To generate better schedules, the highest-scoring solutions are selected and combined via crossover, where parts of two schedules are merged. Mutation is also applied, introducing small changes to maintain diversity and prevent the algorithm from getting stuck in a local optimum.

This process is repeated multiple times, with each new generation producing increasingly better schedules. Through iterative evolution, the algorithm can eventually produce a schedule that is optimal or at least acceptable.

2.3.4 Conclusion

These three approaches deploy AI and are compared with the traditional method (constraint programming) to generate class schedules in an efficient, flexible, and optimized way. During the project, an evaluation will be conducted to determine which method is best suited for the specific requirements of LanguageTeams.

2.4 Technology stack

2.4.1 Mandatory Technology

For this assignment, certain technologies are mandatory:

- **Django**: A web framework used for developing the application. With Django, we can build web pages in Python, enabling server-side Python code execution. Django acts as the backend framework and remains active as long as the application is online.

- **Celery & Redis**: Celery is a powerful asynchronous job queue that allows time-consuming Python functions to run in the background. During certain phases of the process, tasks are created and placed in a queue. These tasks can be assigned priorities and are executed in chronological order as computing resources become available. By combining Celery with Django, we can use the web application as an end user while time-intensive tasks run in the background.

- **PostgreSQL**: The database where all application data is stored. This includes both data entered through the application and data retrieved via the Ecwid API. LanguageTeams uses this to manage their sales, so registration information is also extracted from here.

2.4.2 AI Technologies

For the AI component of the application, various technologies are used to generate and optimize class schedules. The following technologies have been selected, allowing me freedom in my choices:

- **CrewAI:** A framework for implementing Agentic AI, a method where digital agents collaborate to perform tasks. CrewAI enables multiple agents to operate within a structured workflow. In this application, agents are used to create schedules, negotiate changes, and process feedback from teachers and students. CrewAI is flexible, scalable, and open-source, making it suitable for complex multi-agent systems.

- **Google OR-Tools (CP-SAT):** A powerful optimization library utilizing Constraint Programming (CP) and Mixed Integer Programming (MIP) to generate efficient class schedules. CP-SAT helps solve complex scheduling problems through mathematical constraints such as teacher and student availability and avoiding double bookings. This approach ensures an optimized schedule that meets all requirements.

- **DEAP (Distributed Evolutionary Algorithms in Python):** A library for Genetic Algorithms (GA), a method inspired by natural evolution. This algorithm generates multiple possible schedules and iteratively improves them through selection, crossover, and mutation. This allows finding the most optimal solution even for complex scheduling problems. DEAP offers flexibility for experimenting with various optimization strategies.

3 WHY

Currently, the scheduling is done manually: teachers have to coordinate with their students to arrange a schedule, which takes a lot of time. This automated scheduling app saves valuable time for the staff.

Additionally, much administrative work is currently done using Excel files. With this new application, new data (such as teachers, students, and courses) can be managed more efficiently through a central database. This not only improves the speed of the workflow but also enhances usability and clarity.

4 PLANNING

4.1 Research Phase

The first three weeks will focus on research. During this phase, the most suitable techniques and methodologies for generating a class schedule will be investigated.

Additionally, research will be conducted on the mandatory technologies that must be used in the application:

- **Django** as the web framework for the backend.
- **Celery & Redis** for processing background tasks.

For Django, a course has been arranged, which I am already following during this phase since it is a new technology for me.

Besides the technical research, the first visual and structural designs will also be created during this phase:

- Mock-ups of the application have been made to provide a clear view of the user interface and the interaction within the app. These mock-ups help define the user experience (UX) and user interface (UI).
- Database models have been developed and implemented in Django, establishing the basic structure for data storage. This makes it easier to integrate AI solutions and manage data efficiently in later phases.

4.2 Testing Phase

In this phase, the various AI solutions will be developed and tested to determine which one is best suited for generating class schedules. At the same time, development of the application will begin, with the first functionalities implemented and tested to lay a solid foundation for integrating the AI solution.

This includes:

- **Implementing and testing CrewAI** for an Agentic AI solution, where multiple agents collaborate to generate and optimize schedules based on negotiations between teachers and students.
- **Applying Google OR-Tools (CP-SAT)** to generate schedules through constraint satisfaction, defining and solving hard and soft constraints.
- **Implementing DEAP for a Genetic Algorithm solution**, using evolution-based optimization to find the best possible schedules.
- Setting up the application, converting the previously created mock-ups into working web pages, and developing and testing the necessary functionalities.

During this phase, the performance, accuracy, and efficiency of each method will be evaluated. Factors such as speed, scalability, and flexibility will be compared to make a well-informed decision on the best solution.

4.3 Implementation Phase

After the testing phase follows the Implementation phase, during which the chosen solution is actually integrated into the application. In this phase, the AI solution is connected to the web application, and all necessary functionalities are linked to both the front-end and back-end.

The implementation includes, among other things:

- **Integration of the chosen AI technology** (Agentic AI, CP-SAT, or Genetic Algorithm) into the Django environment. This means adapting the application's backend to support the selected AI solution, ensuring smooth interaction between the scheduling logic and data processing.

- Further refinement of the user interface (UI) based on feedback from the testing phase, optimizing the interaction between users (teachers, staff) and the application.

- **Completion of database integration**. The previously created database models are refined and optimized for the production environment, ensuring efficient storage and retrieval of all data (class schedules, teachers, students).

- **Configuration of Celery & Redis** for asynchronous processing of time-consuming tasks, such as generating and adjusting schedules. This ensures the application remains responsive even when complex calculations are running in the background.

During this phase, necessary tests will also be performed to guarantee the application's stability, performance, and freedom from errors. The application will be continuously optimized to improve user experience and overall functionality.

5 COMMUNICATION

For communication within the team and with the internship mentor, **Trello** is used to track progress. There are two Trello boards: one for the internship itself (documentation, presentations, etc.) and one for the LanguageTeams project. This board tracks the tasks that need to be done, who is working on them, what still needs revision, and what is completed. This board only contains tasks for the interns and not for other employees.

Weekly stand-ups are held **three times a week** on Monday, Wednesday, and Friday. During these stand-ups, everyone discusses what they are working on and whether anyone is stuck. The Trello board is also reviewed during the stand-ups to provide a clearer overview.

To keep the internship supervisor informed, a **weekly report** is sent explaining what I have done that week.