WORK INTEGRATED LEARNING
COMP3851B

FINAL REPORT

Covid-19 Extension Used

Develop an Application Programming Interface (API) for Staff Scheduling for Disability and Community Care Service Providers

SOS Technology Group Pty Ltd.

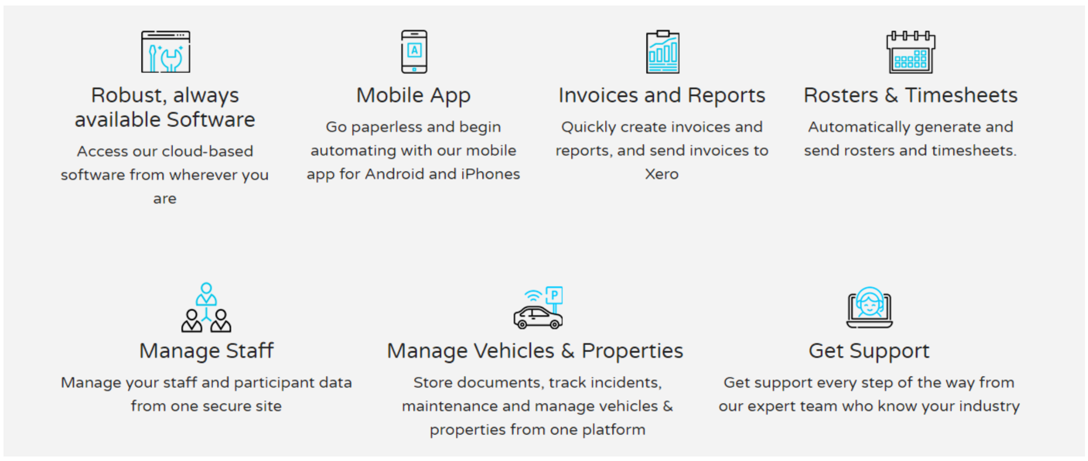
AUTHOR: SHARJEEL SOHAIL
SUPERVISOR: REGINA BERRETTA
PROJECT MANAGER: MEHDI ABEDI

SECTION 01

BACKGROUND

SOS Technology Group Pty Ltd *(SOSTG, 2018)* offers outstanding IT services to businesses and is one of the first Newcastle based IT providers to offer fixed cost, proactive IT support, and has quickly evolved to offer a complete range of IT services.

One of their projects is CareMaster *(CareMaster, 2020)*, a web-based National Disability Insurance Scheme *(NDIS, 2021)* service management platform. It assists NDIS service providers successfully manage their business with variety of services (as in Figure 1), one of them is Rosters & Timesheets.



Robust, always available Software

Access our cloud-based software from wherever you are

Mobile App

Go paperless and begin automating with our mobile app for Android and iPhones

Invoices and Reports

Quickly create invoices and reports, and send invoices to Xero

Rosters & Timesheets

Automatically generate and send rosters and timesheets.

Manage Staff

Manage your staff and participant data from one secure site

Manage Vehicles & Properties

Store documents, track incidents, maintenance and manage vehicles & properties from one platform

Get Support

Get support every step of the way from our expert team who know your industry

Figure 1: CareMaster Services

Due to the complexity of businesses, staff scheduling takes hours and still provides inaccurate and inefficient final schedule. A few problems with scheduling are: overscheduling, disorganisation, and incompetent scheduling.

In order to deal with this problem, SOSTG aims to develop a web API equipped with optimisation algorithm which receives data such as, business constraints, work regulation, personnel skill, and their availability, and provides an optimised solution satisfying clients' as well as business requirements.

One of the main motivations of this project is to design a system that not only satisfy clients' requirements but also satisfy a high number of specific personnel constraints of NDIS providing businesses, in quick and efficient way.

SECTION 02

AIMS

The main objective of the project is to enhance the system to achieve cost efficiency and high accuracy when providing services to NDIS service providers. Following aims are:

2.1) Identify the staff rostering problem and its related parameters, decision variables, constraints, and objectives.

2.2) Develop a web API which communicates with CareMaster to receive data and send back rostering solution using JSON files.

2.3) Equip the API with staff rostering optimisation algorithm which schedules the staff for daily routine tasks by implementing:

a) Fitness Evaluation for nurse scheduling problems *(Burke et al., 2004)* which generates a random schedule and add penalties based on the given constraints

b) Local Search optimisation *(Meignan & Knust, 2019)* which propose a neutrality-based iterated local search metaheuristic for reoptimizing the solution in efficient and short time

2.4) Satisfy clients' requirements (for e.g. specific healthcare, homecare, or transportation requests) by forwarding their requests to the qualified staff member in the most efficient way using these algorithms.

2.5) Roll out the schedule on an interactive webpage implemented on .NET platform using C# and HTML/CSS/JavaScript which also allow admin staff to modify solutions if needed, and then finally sending schedule to the CareMaster using API

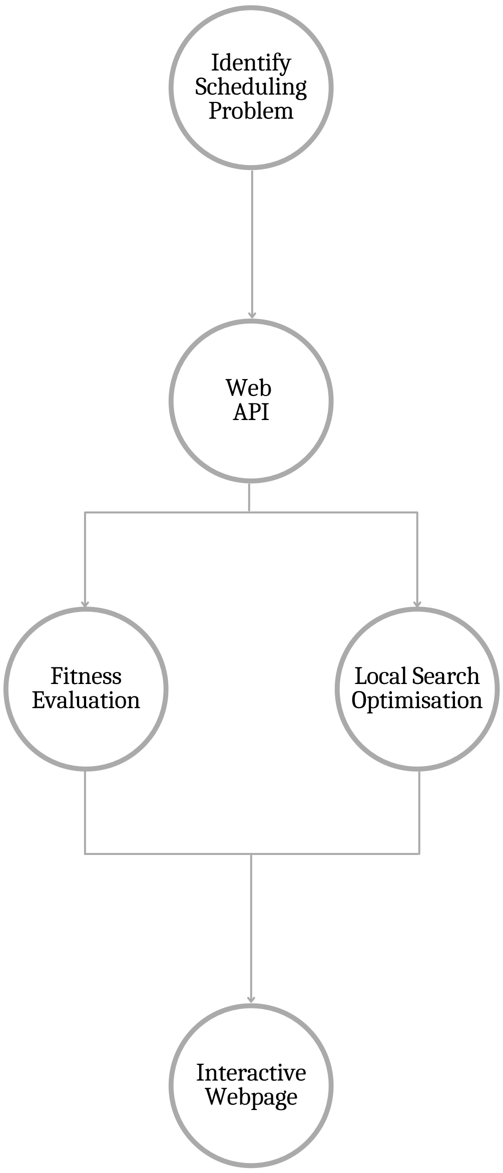


Figure 2: Aims

SECTION 03

METHODS

It was crucial to identify the current problem of the rostering system and how it produces an incorrect schedule, what business and staff constraints are required, and what the objectives are. Once identified, following methodologies were used in ASP.NET C# to achieve the results:

3.1) Random schedule

Creating an initial schedule randomly for each employee personnel based on their availability, skills, and business requirements.

3.2) Fitness Evaluation

Evaluating our initial schedule by applying fitness function using the approach given in “Fitness Evaluation for Nurse Scheduling Problems” *(Burke et al., 2004)* which considers a set of soft and numbering constraints, previous planning period, current cost, and counters.

3.3) Local Search algorithm

Implementing an optimisation algorithm using the approach given in “A neutrality-based iterated local search for shift scheduling optimization and interactive reoptimization” *(Meignan & Knust, 2019)* which move or swap the no. of assignments (shifts) of a person with their neighbour in order to satisfy the violated constraints to improve the cost of our evaluated schedule.

3.4) Web API

Creating a Web API which communicates with CareMaster to receive and send back rostering solution using JSON files

3.5) Interactive Web page

Implementing a basic web page using MVC architecture which asks for user input and displays the initial schedule, and then the optimised schedule to the admin

Iterative testing was carried out at each step to achieve the required results, and then finally finishing the project with supporting documentation and delivery.

SECTION 04

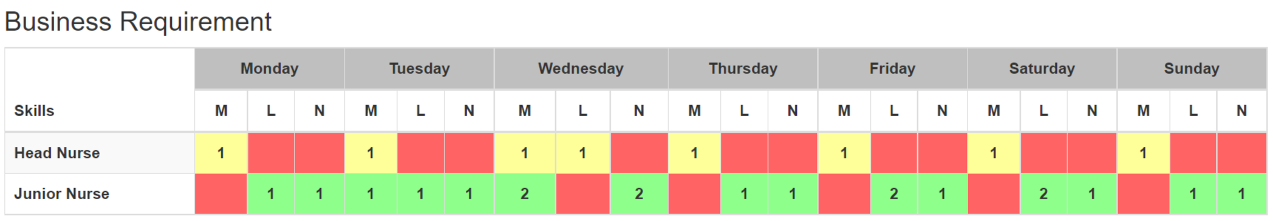
IMPLEMENTATION AND RESULTS

In this section, a demonstration is given of the aims achieved to develop an API equipped with optimisation algorithm for staff scheduling for disability and community care service providers. Methods used for implementation and how they could easily be extended to suit a similar problem are also explained throughout.

Before we examine the results, it is important to understand the implementation process. After analysing the staff scheduling problem of the system, we applied the following main approaches to achieve the desired results:

4.1) Initial Schedule

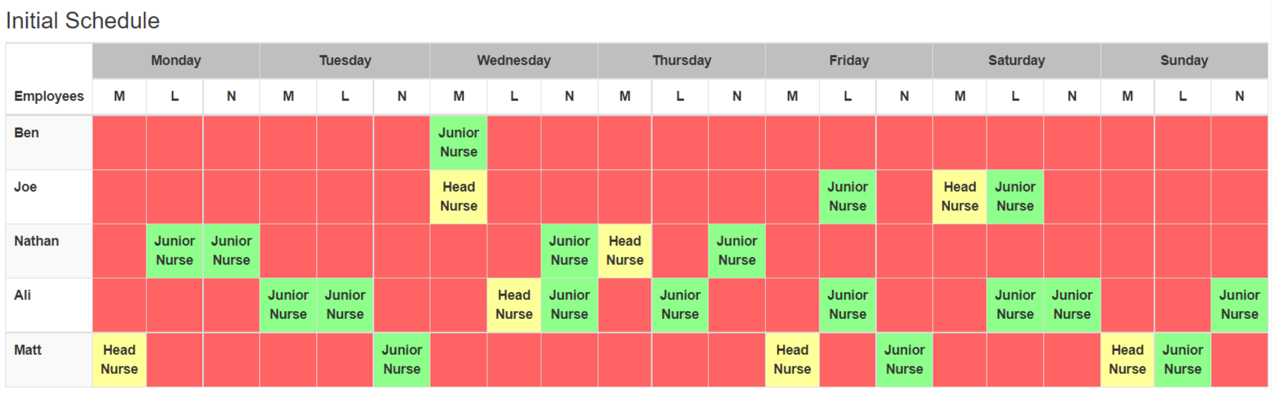
Based on the business requirement i.e. number of different skilled employees required for each shift of the day for the planning period *(Figure 3)*, we generated an initial schedule *(Figure 4)* randomly for every employee in the system.



Business Requirement

	Monday			Tuesday			Wednesday			Thursday			Friday			Saturday			Sunday		
Skills	M	L	N	M	L	N	M	L	N	M	L	N	M	L	N	M	L	N	M	L	N
Head Nurse	1			1			1	1		1			1			1			1		
Junior Nurse		1	1	1	1	1	2		2		1	1		2	1		2	1		1	1

Figure 3: Business requirement (Skills required)
M,L, and N represents Morning, Late, and Night Shift



Initial Schedule

	Monday			Tuesday			Wednesday			Thursday			Friday			Saturday			Sunday		
Employees	M	L	N	M	L	N	M	L	N	M	L	N	M	L	N	M	L	N	M	L	N
Ben							Junior Nurse														
Joe							Head Nurse							Junior Nurse		Head Nurse	Junior Nurse				
Nathan		Junior Nurse	Junior Nurse						Junior Nurse	Head Nurse		Junior Nurse									
Ali				Junior Nurse	Junior Nurse			Head Nurse	Junior Nurse		Junior Nurse			Junior Nurse			Junior Nurse	Junior Nurse			Junior Nurse
Matt	Head Nurse					Junior Nurse							Head Nurse		Junior Nurse				Head Nurse	Junior Nurse	

Figure 4: Initial random schedule
M,L, and N represents Morning, Late, and Night Shift

The requirement for each day is fulfilled, however, since no other considerations were taken while generating other than business requirement, the allotment of shifts are inaccurate. For e.g., Employee Ali is working Wednesday late, night, and again on Thursday late, which in real world in non-viable. Let's go to our next approach i.e. Fitness evaluation and see what steps we consider to make it better.

4.2) Fitness Evaluation

We know that our initial schedule is random and not usable, but how far off is it? To determine how fit our solution is, we apply a fitness function, to evaluate how incorrect or how close our initial solution is to the optimum solution of our shift scheduling problem. We will be following an approach given in “Fitness Evaluation for Nurse Scheduling Problems” *(Burke et al., 2004)*. The method is fast and is extremely useful for evaluating all neighbourhood solutions for the heuristics applied. To apply the fitness function on our current solution, we considered:

4.2.1) <u>Set of Soft & Numbering constraints</u>

Soft constraints, also known as work regulation, for each personnel member *(Table 1)*, and numbering constraints, which are based on real world constraints and represents all days of a week, only night shifts, and only weekends respectively (N1, N2, and N3)

Soft Constraints	**N1**	**N2**	**N1**
Maximum no. of shifts in a week	6	3	
Minimum no. of shifts in a week	2		
Maximum no. shifts in one day	1		
Minimum no. of shifts in one day	0		
Maximum no. of days off between two shifts	8		
Minimum no. of days off between two shifts	2		
Maximum no. of consecutive shifts in a week	4		
Minimum no. of consecutive shifts in a week	2	2	2

Table 1: Soft Constraints (Burke et al., 2004)

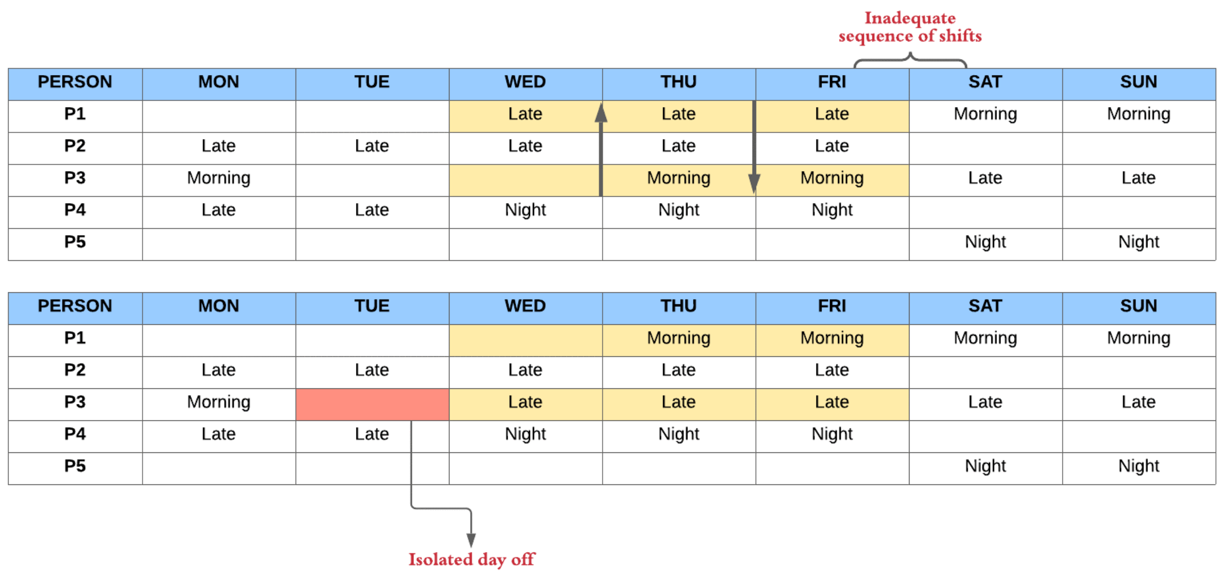
4.2.2) <u>Previous planning period, Counters & Cost</u>

We also considered the previous schedule of each personnel and initialise a set of counters to which are then induced if required for constraints like, maximum no. of consecutive shifts. Costs for each penalty is set to 1 for this method.

The evaluation process then starts, and each personnel’s schedule is evaluated separately. The method goes through each shift of each day and check if the person is schedule, if so, check the constraints with counters initialised and if any violation has occurred, it increases the respective penalty by the given cost. Finally, adding up all the violated penalties giving a total cost which represents the fitness of our current solution.

4.3) Optimisation Algorithm

In our previous sub-heading, we applied fitness function to our initial schedule, and calculated the total cost by adding up all the penalties against the required constraints. The goal is to get the best possible solution (schedule) and so, we implemented an approach given in "A neutrality-based iterated local search (NILS) for shift scheduling optimization and interactive reoptimization" *(Meignan & Knust, 2019)* to minimize the cost.


Inadequate sequence of shifts

PERSON	MON	TUE	WED	THU	FRI	SAT	SUN
P1			Late	Late	Late	Morning	Morning
P2	Late	Late	Late	Late	Late		
P3	Morning			Morning	Morning	Late	Late
P4	Late	Late	Night	Night	Night		
P5						Night	Night

PERSON	MON	TUE	WED	THU	FRI	SAT	SUN
P1				Morning	Morning	Morning	Morning
P2	Late	Late	Late	Late	Late		
P3	Morning		Late	Late	Late	Late	Late
P4	Late	Late	Night	Night	Night		
P5						Night	Night

Isolated day off

Figure 5: K-Swap neighbourhood (k no. of assignments move/swap between two employees). Satisfy constraints but may also violate some constraints that were satisfied before

We go through the following three phases in NILS framework which uses the k-swap neighbourhood structure i.e., adjusts an employee solution by moving/swapping k-assigned shifts of an employee to another employee who is not working on that day *(Figure 5)*:

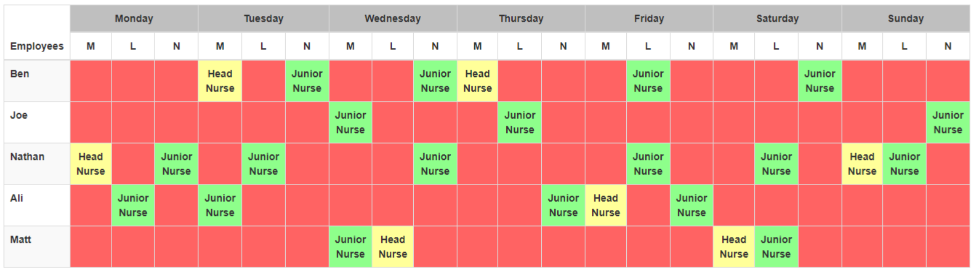
Local Search
This procedure progressively explores the k-swap neighbourhood of the current solution from a block size (no. of shifts) from 1 to 7. It starts with 1-swap neighbourhood and evaluates in a random order and the first neighbour that improves the cost is accepted, if no better solution is found, then the block size is increased. The local search procedure ends when no improving solution is found in 7-swap neighbourhood.

Plateau Exploration
In this procedure, the moves performed are selected according to the number of unsatisfied soft constraints. The procedure is also guided by two heuristics: (1) At least one individual violated constraint must be satisfied after the move. (2) The exploration must move away from the solution on which the plateau exploration started (i.e., finding an improving solution).

Perturbation
Finally, perturbation is applied on the solution selected by the acceptance criterion. It accepts the solution resulting from the plateau exploration if its cost is less or equal to the cost of the best-known solution. Otherwise, the exploration returns to the best-known solution.

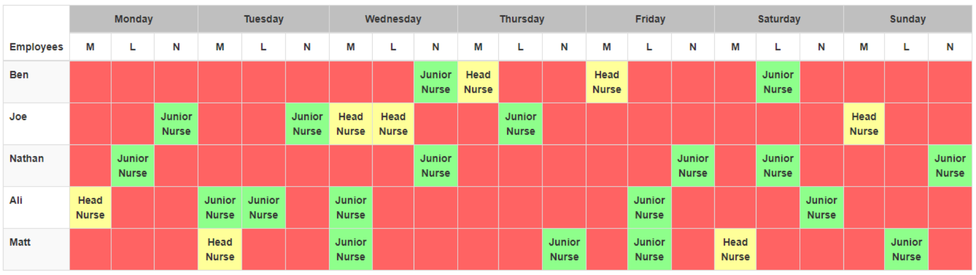
4.4) Final Optimised Schedule

Following screenshots are the Optimised schedule results based on different computational times:



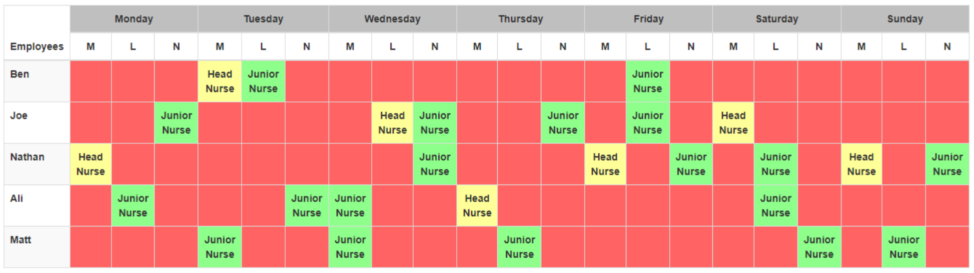
	Monday			Tuesday			Wednesday			Thursday			Friday			Saturday			Sunday		
Employees	M	L	N	M	L	N	M	L	N	M	L	N	M	L	N	M	L	N	M	L	N
Ben				Head Nurse		Junior Nurse			Junior Nurse	Head Nurse				Junior Nurse				Junior Nurse			
Joe							Junior Nurse				Junior Nurse										Junior Nurse
Nathan	Head Nurse		Junior Nurse		Junior Nurse				Junior Nurse					Junior Nurse			Junior Nurse		Head Nurse	Junior Nurse	
Ali		Junior Nurse		Junior Nurse								Junior Nurse	Head Nurse		Junior Nurse						
Matt							Junior Nurse	Head Nurse								Head Nurse	Junior Nurse				

Figure 6: Results on 2 minutes computational time | Cost: 36



	Monday			Tuesday			Wednesday			Thursday			Friday			Saturday			Sunday		
Employees	M	L	N	M	L	N	M	L	N	M	L	N	M	L	N	M	L	N	M	L	N
Ben									Junior Nurse	Head Nurse			Head Nurse				Junior Nurse				
Joe			Junior Nurse			Junior Nurse	Head Nurse	Head Nurse			Junior Nurse								Head Nurse		
Nathan		Junior Nurse							Junior Nurse						Junior Nurse		Junior Nurse				Junior Nurse
Ali	Head Nurse			Junior Nurse	Junior Nurse		Junior Nurse							Junior Nurse				Junior Nurse			
Matt				Head Nurse			Junior Nurse					Junior Nurse		Junior Nurse		Head Nurse				Junior Nurse	

Figure 7: Results on 4 minutes computational time | Cost 29



	Monday			Tuesday			Wednesday			Thursday			Friday			Saturday			Sunday		
Employees	M	L	N	M	L	N	M	L	N	M	L	N	M	L	N	M	L	N	M	L	N
Ben				Head Nurse	Junior Nurse									Junior Nurse							
Joe			Junior Nurse					Head Nurse	Junior Nurse			Junior Nurse		Junior Nurse		Head Nurse					
Nathan	Head Nurse								Junior Nurse				Head Nurse		Junior Nurse		Junior Nurse		Head Nurse		Junior Nurse
Ali		Junior Nurse				Junior Nurse	Junior Nurse			Head Nurse							Junior Nurse				
Matt				Junior Nurse			Junior Nurse				Junior Nurse							Junior Nurse		Junior Nurse	

Figure 8: Results on 8 minutes computational time | Cost 17

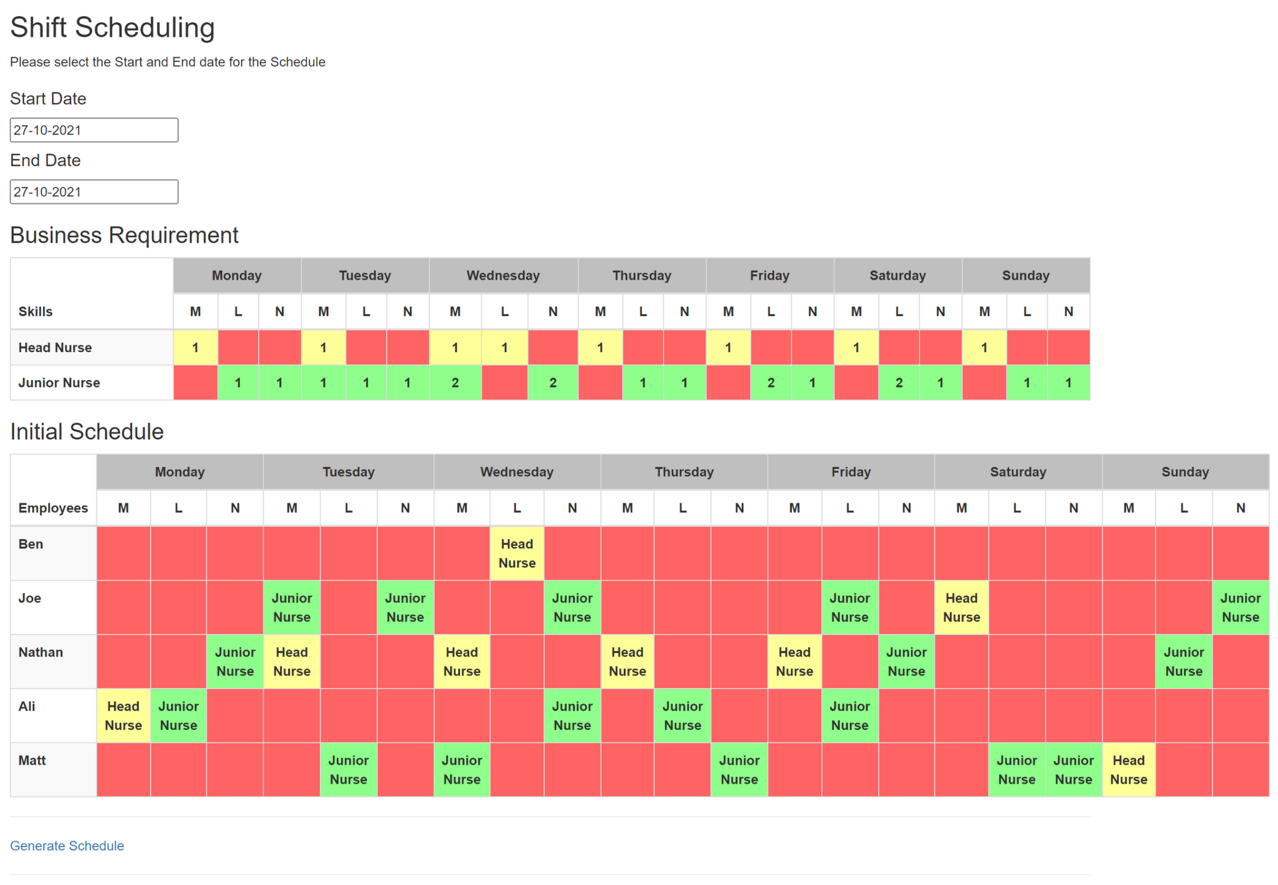
Above optimised results can be seen running on different computational times, and also shows the cost of each solution. It is evident as we increase the time limit, the cost of the solution comes down which means in each iteration, the solution satisfies some of the violated constraints using the optimisation algorithm and evaluates the fitness of the solution to provide the best possible solution with minimum violations. In *(Figure 8)*, we can see that there are still some violations, but it is much better and improved (as represented by cost) that the schedule in *(Figure 6)*.

4.5) Web API

To be able to communicate with CareMaster to get the data such as employees availabilities, business requirement, planning period etc., we created a Web API using ASP.NET, which receives the data using POST method in JSON format, convert it into parameters, and send back the optimised solution using GET method in JSON format. The API is equipped with the Fitness Evaluation and Optimisation algorithm implemented above.

4.6) Web Page

To be able to test and see the solution in an interactive manner, two webpages were implemented using ASP.NET MVC architecture which enables the admin staff to make adjustments (webpage 1) and generate random schedule *(Figure 9)* if necessary and then finally send the optimised schedule (webpage 2)*(Figure 10)* using Web API.



Shift Scheduling

Please select the Start and End date for the Schedule

Start Date: 27-10-2021

End Date: 27-10-2021

Business Requirement

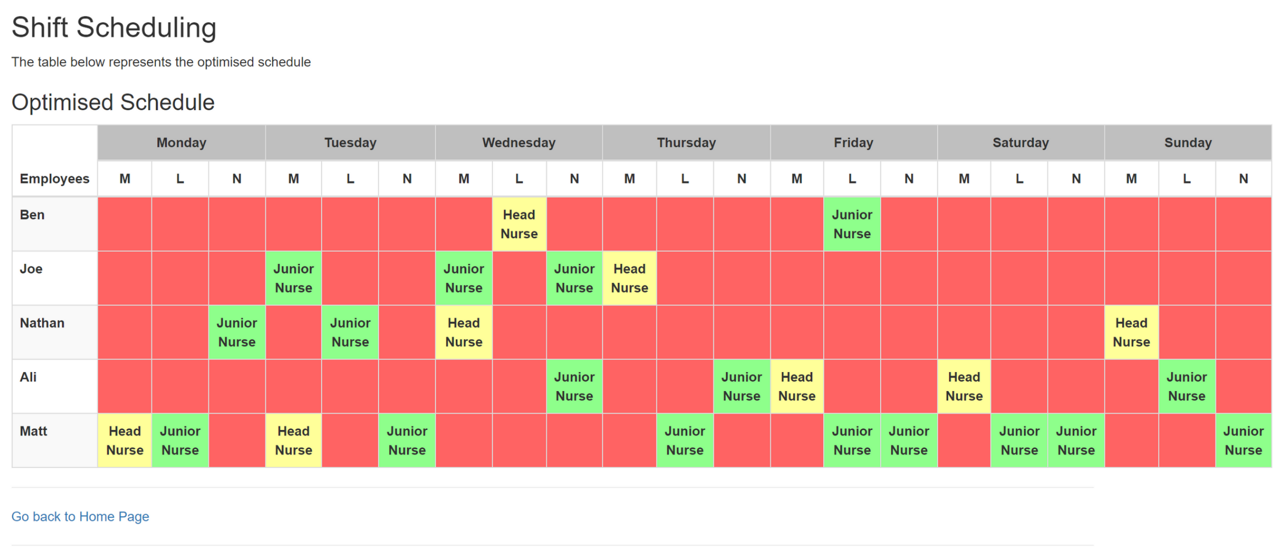
	Monday			Tuesday			Wednesday			Thursday			Friday			Saturday			Sunday		
Skills	M	L	N	M	L	N	M	L	N	M	L	N	M	L	N	M	L	N	M	L	N
Head Nurse	1			1			1	1		1			1			1			1		
Junior Nurse		1	1	1	1	1	2		2		1	1		2	1		2	1		1	1

Initial Schedule

	Monday			Tuesday			Wednesday			Thursday			Friday			Saturday			Sunday		
Employees	M	L	N	M	L	N	M	L	N	M	L	N	M	L	N	M	L	N	M	L	N
Ben								Head Nurse													
Joe				Junior Nurse		Junior Nurse			Junior Nurse					Junior Nurse		Head Nurse					Junior Nurse
Nathan			Junior Nurse	Head Nurse			Head Nurse			Head Nurse			Head Nurse		Junior Nurse					Junior Nurse	
Ali	Head Nurse	Junior Nurse							Junior Nurse		Junior Nurse			Junior Nurse							
Matt					Junior Nurse		Junior Nurse					Junior Nurse					Junior Nurse	Junior Nurse	Head Nurse		

Generate Schedule

Figure 9: Homepage



Shift Scheduling

The table below represents the optimised schedule

Optimised Schedule

	Monday			Tuesday			Wednesday			Thursday			Friday			Saturday			Sunday		
Employees	M	L	N	M	L	N	M	L	N	M	L	N	M	L	N	M	L	N	M	L	N
Ben								Head Nurse						Junior Nurse							
Joe				Junior Nurse			Junior Nurse		Junior Nurse	Head Nurse											
Nathan			Junior Nurse		Junior Nurse		Head Nurse												Head Nurse		
Ali									Junior Nurse			Junior Nurse	Head Nurse			Head Nurse				Junior Nurse	
Matt	Head Nurse	Junior Nurse		Head Nurse		Junior Nurse					Junior Nurse			Junior Nurse	Junior Nurse		Junior Nurse	Junior Nurse			Junior Nurse

Go back to Home Page

Figure 10: Optimised schedule page

SECTION 05

ETHICS

Data is one of the most important assets of any organisation. Losing critical information could be extremely harmful and could even result in demise of the company. Throughout the project duration, sample data was used for every method implemented. However, when successfully delivered, the Web API would be collecting the actual data from the CareMaster, we have considered the following data security techniques:

5.1) Encrypted JSON files

The system files containing solution have made encrypted and in JSON format to keep information inaccessible without authority, giving hackers no chance to crack the solution. These JSON files are shared with an API equipped with optimisation algorithms, which converts the JSON data into input parameters, go through the different evaluation phases, and then finally returns the solution in a JSON format to send it back to CareMaster.

5.2) Secured Socket Layer (SSL)

The interactive webpage uses Secured Socket Layer (SSL) which offers secure communication between web servers and browsers (web clients) to keep the solution and the user data protected

SECTION 06

SELF-REFLECTION & PROJECT MANAGEMENT

6.1) Personal Experience

My personal experience while creating a Web API for Staff Scheduling for SOS Technology Group Pty Ltd. was very exciting and engaging the whole time. It was one of many lifetime experiences which weighs quite a lot in my professional career. Before this project, I had little to no knowledge in implementing optimisation algorithms, and when applying for this project, I knew what was coming for me. I was scared but I was also excited to experience new skills which I can not only add into my resume but also apply those skills in my career. I acquired many technical and soft skills which I have talked about in the following section.

I worked with an amazing team of 2, Regina Berretta, the supervisor, and Mehdi Abedi, the project manager. This was an exciting opportunity for to learn from the best, both of them. They both were always ready to help me at any day of the week and were supportive throughout. I believe the most important thing is that when you enjoy the process thoroughly, and not get bored, and I can happily say that this was one of those experiences.

6.2) Self-directed Learning

In order to carry out certain tasks to achieve the aims of the project, following self-learning was undertaken throughout the course of the project:

- **Python** – For this project, a prototype was provided to implement some external functions that were not provided in the research papers, into C# language, in ASP.NET. Before this, I had no experience working with Python and learning a new language was very exciting journey for me, and was carried out throughout the duration of the project.
- **Model-View-Controller (MVC) Architecture and Web API** – One of the aims were to implement a Web API as well as a couple of web pages, using MVC architecture in ASP.NET. To learn MVC, I found an amazing YouTuber, Tim Corey, who has detailed courses on Web API and MVC up on his YouTube channel. The full course was about 5-hours long and was explained nicely for beginners. I was able to learn both skills in a week and was successful in implementing these methodologies in the project.
- **Visual Studio Code** – To be able to test the prototype provided in Python, I chose VS Code to run the project, and since I had no experience with this environment, I used YouTube to watch a few videos for some basic tasks that would help me run and analyse the prototype. A few challenges were to create a virtual environment and importing libraries like 'NumPy' etc.

6.3) Skills Acquired

Following skills were acquired during the course of the project:

- **Sustainability** – As the approach used for the project was Agile, sustainable development was the main idea for implementing this project. I have always believed that quality of code matters a lot than achieving an output in a poorly designed technique. A few examples are given in the Challenges sub-heading below (Section 2.3). A constant iterative approach helped me think better at every iteration, and so, applying the best and sustainable approach to provide the best results possible.

- **Creativity** – Implementing the project from scratch gave me enough room to think creatively and outside of the box to achieve the aims. I wasn't restricted or given a set of methodologies to use, but was given a total freedom to use any approach to get the results. I implemented some of the best functions that I'm very proud of, which solves a complex problem in the process. An example is given in the Challenges sub-heading below (Section 2.3)
- **Other soft skills** like problem-solving, communication skills, flexibility, and work ethics were also acquired throughout the course.

6.4) Challenges

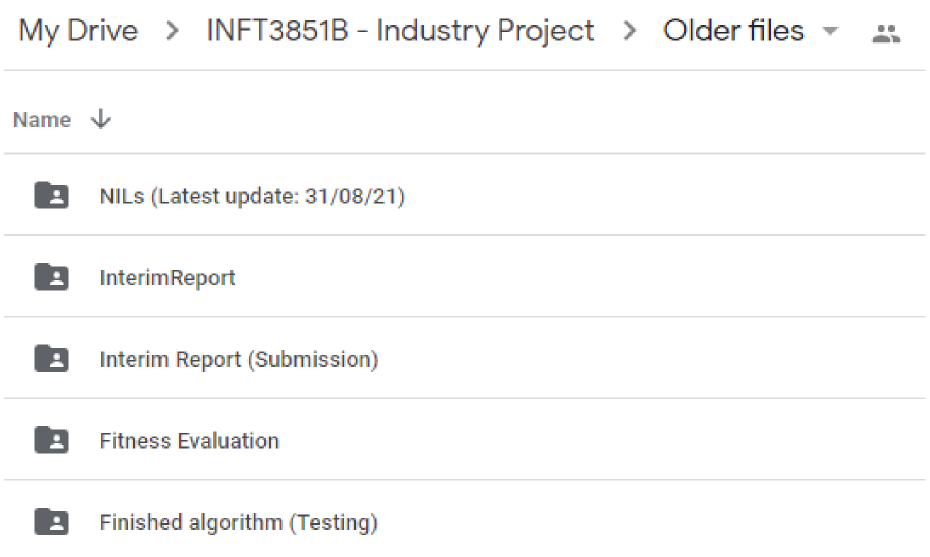
Following challenges were faced and how I overcame them:

- **Covid-19 Restrictions** – The pandemic led to an unexpected experience in the learning process, especially when NSW was under strict lockdown. The lockdown was announced on a short notice and I had all my project files saved in my University Student Drive, which I was unable to transfer into my personal laptop. I contacted IT Services to map my Student Drive to my personal laptop, but that process took almost about 3 weeks, resulting my workload being piled up. Being a student who is always dependent on University resources from the beginning, it was very challenging for me to focus at home. While scrolling my Instagram, I came across this influencer called 'Software_Journey1' who was posting his daily routine and encouraging others to do the same. Next thing I did was to order a desk and few other accessories which really helped me focus and get back on track.
- **Technical challenges** – There were a lot of technical challenges but to mention a few, let's start with analysing the results. During the final stages of the project, the task was to generate the best possible schedule, and when displayed on the webpage, I noticed that the algorithm was producing the exact same schedule for every employee. I debugged over and over again but couldn't figure out why it was doing such thing. I discussed this issue with my project manager, Mehdi Abedi, and we went through the code together via Zoom. He pointed out that I was only 'referencing' the classes by using the equal '=' sign, it wasn't actually copying the class and its reference type variables. This resulted in being used the same solution over and over again, which made every employee's schedule the same. I wasn't aware of such thing before as I had never experienced with classes with so many reference type variables in it. I used my technical skills so work my way around it and finally fixed it by making a constructor and few overloaded copy functions to make a deep copy of a class. Another interesting challenge was about computational time when producing the results. In the prototype that was provided, the computational time was set to 360 seconds. I assumed that this might be a mistake, why would a project run for 6 minutes? Isn't this a lot of wait time? I changed it to 3.6 seconds thinking that this is what it's supposed to be. When running it for only 3.6 seconds, I noticed that there's only a few changes and the violation cost is still high. I consulted this with Mehdi and my supervisor, Regina Berretta, and we discussed this and they told me that algorithms take time to product working results and that was quite an eye-opening fact for me.

6.5) Project Management

To avoid any delays and failure, and to ensure that project is delivered on time, following project management tools were used throughout.

- **Google Drive** *(Figure 11 and 12)*– Sharing of coding files, meeting minutes, research papers and any project related material, was done using Google Drive so that the rest of the team members, in this case, my project manager and supervisor, are updated of all the tasks and deliverables to keep the project up and running.
- **Outlook** – To communicate with each other, Outlook was used to send out emails and meeting announcements. It was also used to send drafts of the Interim and Final report and presentation for reviews and feedback.
- **Zoom** – Since we all couldn't meet in person every week because of the pandemic and other work-related assignments, our team of three would meet every Thursday from 14:00 to 15:00 to discuss any ideas and sharing of work done by me. This was also a good time for me to ask any questions that I had and shared my screen to show the weekly progress of the project.
- **Trello** *(Figure 13 and 14)*– It was used to manage the deliverables every fortnight. For every task or function that needed to be done in order to achieve the end-result, a list of items was created under four tabs, i.e., To-Do, Doing, Done, Code Review, and Testing. This was shared with the rest of the team to be able to track the progress.



My Drive > INFT3851B - Industry Project > Older files

Name ↓

NILs (Latest update: 31/08/21)

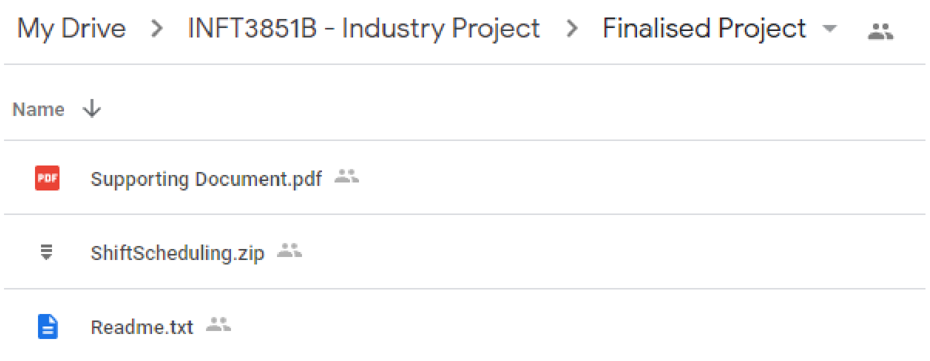
InterimReport

Interim Report (Submission)

Fitness Evaluation

Finished algorithm (Testing)

Figure 11: Google Drive Screenshot (Shared Folder)



My Drive > INFT3851B - Industry Project > Finalised Project

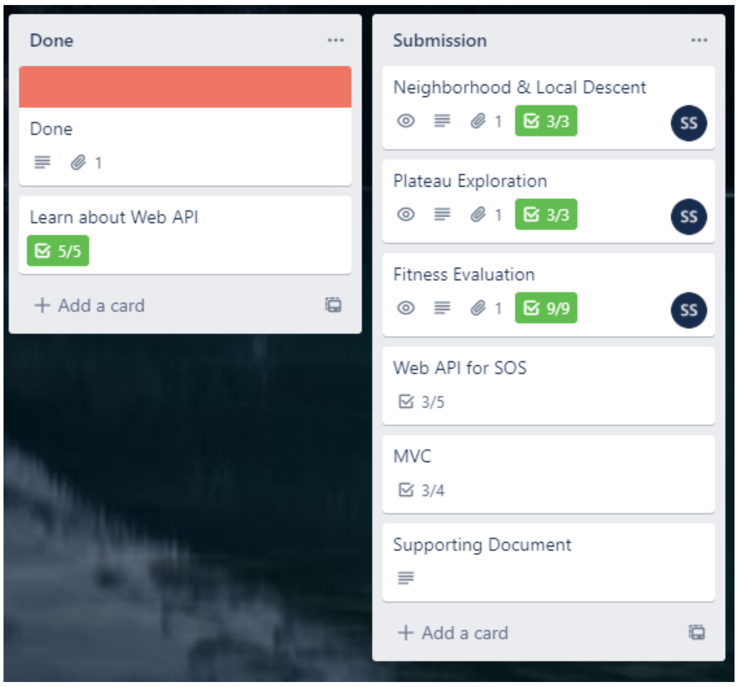
Name ↓

Supporting Document.pdf

ShiftScheduling.zip

Readme.txt

Figure 12: Google Drive Screenshot (Shared Folder)



Done

Done

Learn about Web API

5/5

Submission

Neighborhood & Local Descent

3/3

Plateau Exploration

3/3

Fitness Evaluation

9/9

Web API for SOS

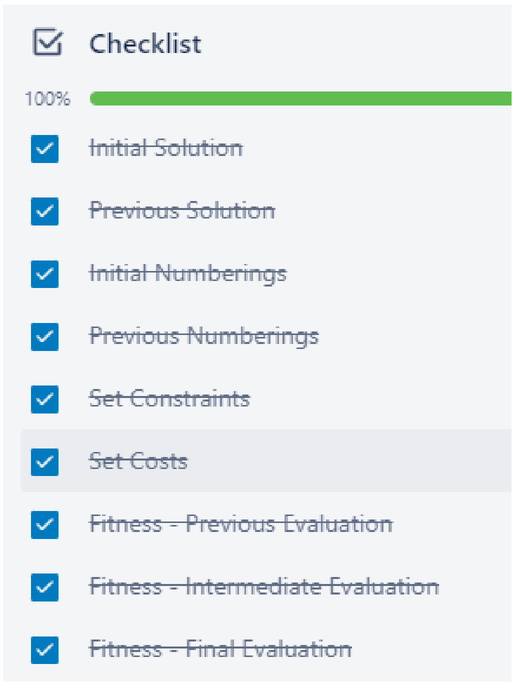
3/5

MVC

3/4

Supporting Document

Figure 13: Trello Board Screenshot



Checklist

100%

- ✓ ~~Initial Solution~~
- ✓ ~~Previous Solution~~
- ✓ ~~Initial Numberings~~
- ✓ ~~Previous Numberings~~
- ✓ ~~Set Constraints~~
- ✓ ~~Set Costs~~
- ✓ ~~Fitness - Previous Evaluation~~
- ✓ ~~Fitness - Intermediate Evaluation~~
- ✓ ~~Fitness - Final Evaluation~~

Figure 14: Trello Task List

6.6) Things I Would Do Differently

Although I am very proud and satisfied with the project overall, but if I had to do all over again, I would do the following things differently:

- **The first thing** would be to understand the specifications clearly, ask as many questions in the beginning as I can, about the aims and methodologies to elicit any confusion I have sooner. The reason for this is because I was a little lost in the first couple of months as I thought that I'll be collecting real data and would schedule them as I go. However, later in the implementation process, I realised that we would only use sample data to test the project and the implementation with CareMaster will be done later.
- **Second** would be using project management tools from the beginning. I say this because in Part A of Work Integrated Learning, I could only implement one function, which was Fitness Evaluation and took a while in that, although it was quite challenging alone but now I believe that I could've done more, and using project management tools could have been helpful to achieve the tasks.

SECTION 07

REFERENCES

Burke, E. K. Causmaecker, P. De, Petrovic, S, and Berghe G. V., "Fitness evaluation for nurse scheduling problems," Proceedings of the 2001 Congress on Evolutionary Computation (IEEE Cat. No.01TH8546), 2001, pp. 1139-1146 vol. 2, doi: 10.1109/CEC.2001.934319. https://ieeexplore-ieee-org.ezproxy.newcastle.edu.au/stamp/stamp.jsp?tp=&arnumber=934319&isnumber=20224

Care Master. (n.d.). Home. Retrieved October 14, 2021, from https://caremaster.com.au/

Meignan, D., and Knust, S. (2019). A neutrality-based iterated local search for shift scheduling optimization and interactive reoptimization. European Journal of Operational Research, 279(2), 320–334. https://doi.org/10.1016/j.ejor.2019.06.005

National Disability Insurance Scheme (NDIS). (2021, October 14). NDIS. https://www.ndis.gov.au/

SOS Technology Group Pty Ltd. (2018, April 19). About Us. SOS Technology Group. https://sostg.com.au/what-we-do/