A SMART TASK MANAGEMENT AND SCHEDULE SUGGESTION MOBILE PLATFORM FOR EFFICIENCY IMPROVEMENT USING ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

Keith Cao[1], Joshua Wu[2]

[1]Issaquah High School, 700 2nd Ave SE, Issaquah, WA 98027
[2]Computer Science Department, California State Polytechnic University, Pomona, CA 91768

ABSTRACT

Task management is a dilemma many people are plagued with in their daily lives. Ironically, this itself is a massive task for many, especially if they suffer from depression or other mental illnesses. In developing this app, we used ChatGPT 3.5-Turbo to receive a list of tasks from a user and return a completed schedule for them to follow. It contains a scheduling system that receives a token from the user and returns a token of its own in the form of JSON data that the program then parses and displays, a cloud storage that stores every user's data individually and can be drawn on easily, and a tasks page where users may create new tasks and complete old ones with three different display settings relating to the range of time the tasks are in: Today, Week, and All. This application will enable users to save time by crafting a schedule for them, help improve the time management skills of users, and help them finish goals before a set deadline.

KEYWORDS

Artificial Intelligence, ChatGPT, OpenAI, Flutter

1. INTRODUCTION

Making a schedule is a time and energy-consuming task that many people suffering from mental illnesses like depression or anxiety are unable to complete easily [1]. In fact, in the last two decades, over 19% of US adults suffer from some form of anxiety and over 8% suffer from clinical depression [2][3]. Those affected by these mental illnesses also experience other detriments such as fatigue, concentration problems, and headaches [4].

Some of the existing assistive applications for anxiety and depression are mindfulness apps and hotlines, especially for suicide. However, these apps have other issues such as privacy violations and a sense of remoteness [5]. This lack of personal connection means that online therapy does not work as well as in-person therapy. Also, another major risk of using these applications is the possibility of false or misleading information within the application. Around half of the apps surveyed utilized principles of Cognitive Behavioral Therapy (CBT) and around a fifth were based upon mindfulness and meditation [5]. Notably, none of these applications are offered as a cure for mental illness, and are rather for helping people in conjunction with proper healthcare.

David C. Wyld et al. (Eds): DMSE, CSEIT, NeTCoM, SPM, CIoT, NCS, NLPD, ArIT, CMLA – 2024
pp. 199-210, 2024. CS & IT - CSCP 2024 DOI: 10.5121/csit.2024.141416

As for treating the illness, a common practice today is to use Selective Serotonin Reuptake Inhibitors (SSRIs) to counteract the effects of mental illness on someone's brain. SSRIs are intended to raise the serotonin levels of the brain, which can help symptoms of depression and help with other treatments [6]. However, these usually take several weeks for any effects to show, and often must be taken up to 6 months after depression lifts. Additionally, there are many limitations of SSRIs; some pregnant women cannot take it, along with those who have diabetes, epilepsy, or kidney disease [6]. On top of that, 38% of those surveyed in a 2009 study reported side effects after taking SSRIs, with 26% of those listing the side effects as very bothersome to their daily life [7]. Perhaps due to these side effects, we see that this method is often paired with other treatment methods like CBT because it acts as more of an assistive measure rather than a cure for the illness.

Cognitive Behavioral Therapy is another common method of treating mental illnesses like depression and anxiety [8]. It works through a collaborative problem-solving process with a trained therapist to gradually challenge and modify a person's internal maladaptive patterns in a direction such that they no longer suffer from these mental illnesses. However, since CBT works on an individual basis, must be modified according to an individual's needs by a specially trained therapist, and must persist for a long time for any effects (6-12 sessions), it is very inefficient on resources and not very suitable for many working adults due to the lengthy time commitment that must be made [9]. As such, there must be a way to support more people at once on their therapeutic journeys.

We present one such support in the form of an application. As an assistive method for depression and anxiety, IntelliPlan helps to soothe the stress over schedules these people have. Although it is a mobile application, it does not suffer from a majority of the flaws of other applications for mental health. There is no misleading information within because it is user-driven and it does not ask for sensitive personal information. It also does not have the intent to provide therapy to an individual. Instead, the app receives user-inputted tasks, fully customized however the user requires, and outputs a cohesive, ready-to-use schedule through the use of ChatGPT. This schedule is saved and used for a week, and the user may generate a new schedule at any time if they are unsatisfied with their current one. By taking the initiative to give the user a schedule based on their tasks, the app ensures that they do not have to force themselves to come up with a plan and can instead follow a ready-made one. Through relieving this pressure, Intelliplan assists the user in feeling more able to take on their daily responsibilities.

The organization of the paper is as follows: Section 2 discusses the challenges we faced in our development of the application and how we solved them, Section 3 discusses the main aspects of our application and the most important systems integrated into it, Section 4 describes the experiments we did with our application and the results obtained, Section 5 is our methodology and summarizes related works along with a short analysis comparing it to our application, and Section 6 is our conclusion, with a discussion of limitations and summaries of other sections.

2. CHALLENGES

As artificial intelligence is a relatively new field and ChatGPT is not built as a schedule processor, there were several challenges we had to face in order to remodel the AI to act as one. The schedulers currently out there force the user to schedule their tasks themselves, choosing time slots and days. However, our AI takes that burden off the user so they have more time and energy to actually do the tasks within their schedule.

2.1. The Performance of the Scheduler

The performance of the scheduler was highly dependent on the quality of the ChatGPT response received. We first had to consider how to set the base guidelines for the artificial intelligence to follow, such as the formatting required and how to list the scheduled items in order. To do this, we wrote a long prompt that comprehensively covered these topics and more, such as what prompts to schedule and what the range of the schedule was. The response had to make sense and fit within the allotted tokens but also be a high-quality schedule that could be reasonably followed by an average person.

2.2. Wording the Prompt

Another challenge was simply wording the prompt in a way that enabled the AI to understand what its task was and how to format its response. We first attempted to have it directly return a JSON file, but when that proved difficult we settled for the AI giving us text in the format of JSON data. We also had to repeatedly warn ChatGPT not to add words before or after the JSON data it provided because doing so would mess up the input and cause the schedule to fail to load. We eventually found a prompt that it followed consistently and so the failing has been decreased drastically.

2.3. The Ranking System

One more problem was getting the ranking system in our app to be functional, as it was extremely difficult and had many roadblocks along the way. We had trouble with formatting the rank in a way that was appealing and had to spend quite a while searching online for solutions. Furthermore, the ranking page often would not properly display the completion percentage of users or even show any users at all. This was solved through many hours of debugging and testing to see where the list of usernames was getting dropped. We then found that the user's percentage would display as NaN instead of 0 occasionally if they had no tasks in their task log. This was unprecedented because we had coded it earlier such that those usernames simply would not be displayed on the ranking page; them being shown meant that the if-then statements we had done earlier had some issues. However, it was solved relatively quickly and was overall one of the more straightforward issues with the ranking system.

3. SOLUTION

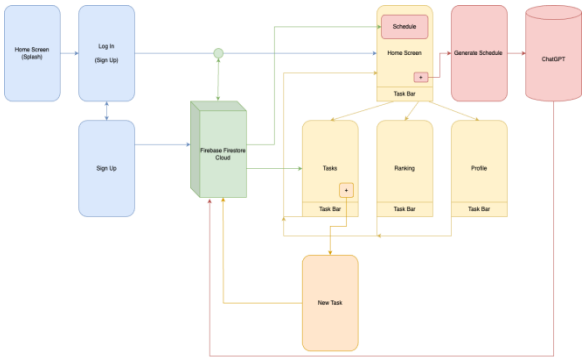

Figure 1. Flowchart

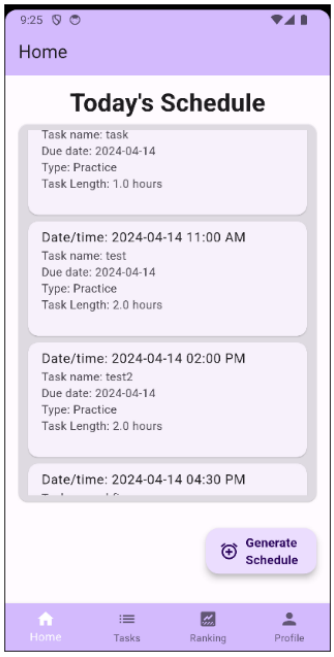

Figure 2. Home page

Figure 3. Schedule Generator Code Snippet

When the app starts up, it displays a splash art greeting the user while loading the rest of the app. It then enters a login screen, where the user is prompted to enter their email and password or create a new account if they do not have an account. Regardless of the choice selected, the information inputted is passed through the Firebase cloud and verified. The app then enters the home screen where we have the option to create a schedule or navigate to other tabs. The other tabs include the task screen, where user-created tasks are displayed and new ones can be made; the ranking screen, where all users are displayed on a scrollable leaderboard based off of what percentage of their tasks they have completed; and the profile screen, where the user can enter some of their personal data -- their name, age, gender, and a short biography. The profile data is

saved into the Firebase cloud and will be pulled from each startup to automatically display when the app is opened. New users will be prompted to enter this information into the cloud before creating a schedule. They create new tasks by tapping on a "New Task" button in the bottom right of the screen on the task page. This navigates the user to a screen in which they are prompted to enter the task's name, type, duration, and priority, alongside an optional short description of it. Once filled out, the user can confirm the creation of the task and it will be stored in the Firebase cloud to be displayed in the task screen. The user may complete tasks by tapping a check box on the right side of the screen next to the task displayed, which will update its status within the cloud to be marked as completed. A schedule for the following seven days can be generated by tapping the "Generate Schedule" button in the bottom right of the home screen. This will bring the user to a pop-up box in which they are asked what date and time to begin generating the schedule. This affects which tasks are selected for scheduling and what times the tasks are scheduled for. After this confirmation, ChatGPT will receive a prompt containing this data and the tasks to be scheduled, outputting a response that is parsed by the app and displayed on the screen for the user to look through. If the user likes the schedule, they may tap a save icon in the bottom right of the screen and the schedule will be saved into the cloud. On the home screen, this schedule will be displayed for the user to view both after saving and on startup of the app.

The schedule generator is the main aspect of the application and is what interfaces with ChatGPT to produce the schedules sent to the user. The app utilizes the OpenAI API as well as the Firestore cloud to first gather a prompt to send to the API, which then saves into the cloud, which is then pulled from to display on the user's home page (fig. 2).

We first initialize the class _GenerateSchedulePageState (fig. 3) with an OpenAI API Key and give a maximum response time of thirty seconds. We then call handleInitMessage(), which pulls data from the user's Firestore data in order to complete a prompt that is sent to ChatGPT for processing. The language model is given a maximum response length of 3000 tokens and told to return a weekly schedule in the form of JSON data, scheduling each task given once and designating specific hours as unavailable. The length of the prompt is long, totaling 330 words, due to how ChatGPT works; each edge case must be found and prevented such that the schedule is properly returned. We use the jsonDecode() function on ChatGPT's response to parse it into separate tasks, which is then displayed in a cohesive weekly schedule and saved into the user's Firestore Cloud to be drawn on from the home page of the app.

4. EXPERIMENT

4.1. Experiment 1

Trial No.	Average Length (hr)	Each Task Scheduled?	Duration Accurate?	Breaks Included?	Exact Task Duration
1	0.5	Y	Y	Y	0.5, 0.5, 0.5, 0.5, 0.5
2	0.5	N (error)	N/A	N/A	0.5, 0.5, 0.5, 0.5, 0.5
3	0.5	N (error)	N/A	N/A	0.5, 0.5, 0.5, 0.5, 0.5
4	0.5	Y	Y	N (3/4 breaks)	0.5, 0.5, 0.5, 0.5, 0.5
5	0.5	Y	Y	N (3/4 breaks)	1, 0.5, 0.5, 0.25, 0.25
6	0.5	N (error)	N/A	N/A	1, 0.5, 0.5, 0.25, 0.25
7	0.5	Y	Y	Y	1, 0.5, 0.5, 0.25, 0.25
8	0.5	N	Y	Y	1, 0.5, 0.5, 0.25, 0.25
9	0.5	Y	Y	N (3/4 breaks)	1, 0.5, 0.33, 0.33, 0.33
10	0.5	Y	N	Y	1, 0.5, 0.33, 0.33, 0.33
11	0.5	N	Y	Y	1, 0.5, 0.33, 0.33, 0.33
12	0.5	N	Y	N (1/3 breaks)	1, 0.5, 0.33, 0.33, 0.33
13	0.5	Y	N	Y	1.33, 0.33, 0.33, 0.25, 0.25
14	0.5	Y	Y	Y	1.33, 0.33, 0.33, 0.25, 0.25
15	0.5	N (error)	N/A	N/A	1.33, 0.33, 0.33, 0.25, 0.25
16	0.5	Y	Y	Y	1.33, 0.33, 0.33, 0.25, 0.25
17	0.5	Y	Y	N (3/4 breaks)	0.75, 0.75, 0.5, 0.25, 0.25
18	0.5	N (error)	N/A	N/A	0.75, 0.75, 0.5, 0.25, 0.25
19	0.5	Y	Y	Y	0.75, 0.75, 0.5, 0.25, 0.25
20	0.5	Y	Y	N (1/4 breaks)	0.75, 0.75, 0.5, 0.25, 0.25

Figure 4. Experiment 1 Data

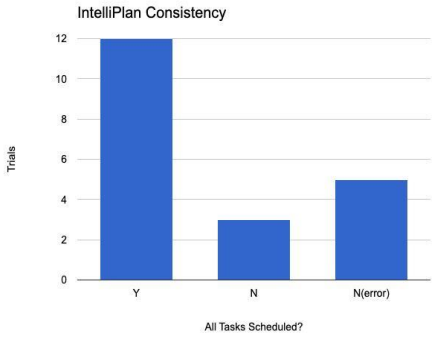

Figure 5. Task Scheduled Data

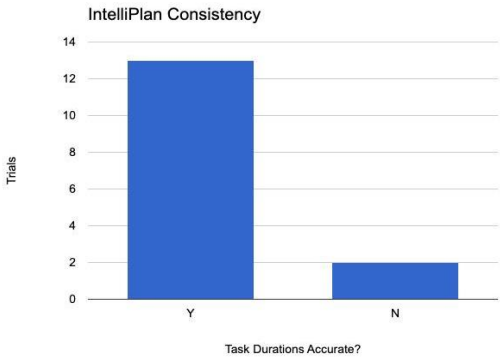

Figure 6. Task Duration Data

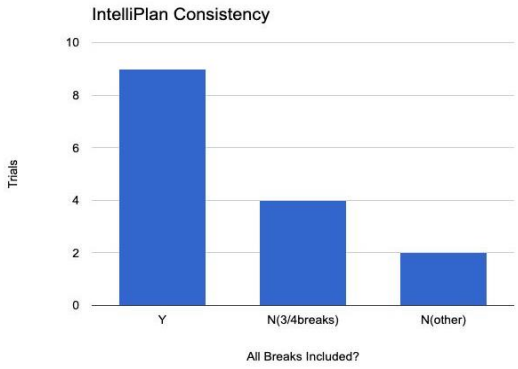

Figure 7. Break Data

The consistency of AI scheduling is a critical component of our app as it is the feature that enables our app to function as a scheduler and give results to users. If it did not work, then the app would have no way of giving schedules to the user since the AI is our only source of scheduling. To test the consistency, we will run a loop where a fixed number of tasks are created with varying durations, all due on the same day. Then, we call the AI scheduler and see if it schedules the tasks properly. The schedule should include all tasks, should schedule each task only once, should fully cover the duration, and include breaks in between tasks. The data will measure the individual task lengths, whether each aspect of a successful schedule was met, and which trial is run.

ChatGPT is inconsistent across the same data, often succeeding in some trials and failing in others due to slight modifications to the start time and inherent temperature. However, it was very consistent in scheduling tasks for the correct durations and scheduling breaks in between tasks, succeeding 86.7% of the time in scheduling the correct duration and scheduling at least one break every single time (fig. 4) (fig 6). It scheduled a break for every task only 60% of the time though, which is another thing to look into (fig. 4) (fig. 7). Many of the breaks it scheduled were very short too and as such are not very realistic or feasible as breaks.

We believe that the main issue causing the errors that happen in 25% of trials is the built-in "politeness" of ChatGPT. This politeness makes it so the AI is obligated to provide a preface to the data, usually in the form of "Certainly! I can make a schedule…". This extra text means that the JSON data parser no longer receives pure JSON data, and instead outputs an error because these prefacing sentences outputted by ChatGPT disrupt the program. This could likely be fixed by additional fine-tuning of the prompt, possibly through forcing the AI to charade as a taciturn individual skilled in computer science, but implementing such a prompt would take up many of the tokens reserved for the AI's response and make the application worse due to a shorter possible schedule.

4.2. Experiment 2

Duration	Iteration	# of Tasks	Each Task Scheduled?	Duration Accurate?
30 min	1	1	Y	Y
	2	2	Y	Y
	3	3	Y	Y
	4	4	Y	Y
	5	5	Y	Y
	6	6	Y	Y
	7	7	Y	Y
	8	8	Y	Y
	9	9	Y	Y
	10	10	Y	Y
	11	11	Y	Y
	12	12	Y	Y
	13	13	Y	Y
	14	14	N (error)	N/A
	15	14	Y	Y
	16	15	Y	Y
	17	16	N (token limit)	N/A
	18	16	N (token limit)	N/A
	19	16	N (token limit)	N/A
60 min	1	1	Y	Y
	2	2	Y	Y
	3	3	Y	Y
	4	4	Y	Y
	5	5	Y	Y
	6	6	Y	Y
	7	7	Y	Y
	8	8	Y	Y
	9	9	Y	Y
	10	10	N (error)	N/A
	11	10	N (error)	N/A
	12	10	N (error)	N/A
90 min	1	1	Y	Y
	2	2	Y	Y
	3	3	Y	Y
	4	4	Y	N (3/4 right)
	5	4	Y	Y
	6	5	Y	Y
	7	6	N (error)	N/A
	8	6	Y	Y
	9	7	Y	N (6/7 right)
	10	7	N (error)	N/A
	11	7	Y	N (6/7 right)

Figure 8. Experiment 2 Data

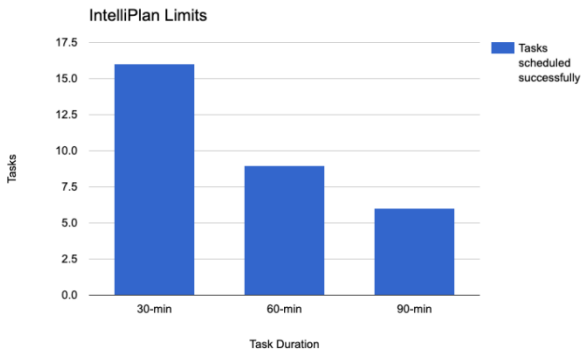

Figure 9. Tasks Scheduled Data

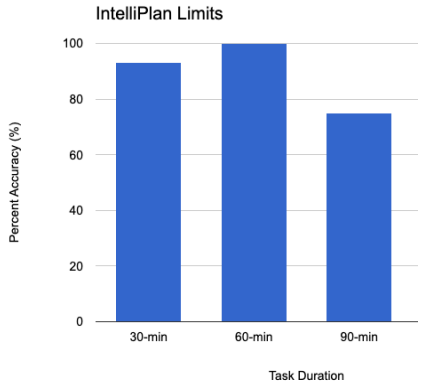

Figure 10. Task Consistency Data

The second experiment was conducted in order to find the limit of how many tasks the AI can schedule at once, so that we could properly limit the amount and length of user responses to the AI. Understanding how many tasks it can consistently schedule is a major component of that as it enables us to set a limit on tasks per schedule to preserve the integrity of each schedule. For this experiment, we ran a loop where the amount of tasks the AI is ordered to schedule increases by one each time it successfully completes a schedule, starting with one task. Once the AI failed a schedule, we continued to run it until it failed that number of tasks twice more, at which point we stopped the experiment. We considered a valid schedule as one that scheduled each task once and fully covered the duration of each task. Due to how the duration of the tasks may affect the number scheduled successfully, we tested various different durations.

With 30-minute tasks the AI was extremely consistent, working almost 100% of the time and only stopping due to a hardcoded token limit being crossed (fig. 10). Interestingly, the AI occasionally gave out breaks but was very inconsistent in doing so. This is likely because the prompt does not explicitly tell the AI to give breaks, but instead implies that if the AI does give breaks to not schedule them as tasks in the schedule. Consequently, the AI often simply ignores breaks outright or, in the case of a few schedules here, gives breaks of only a couple minutes at a time. We also noticed that when the AI wraps around to a new day, it does not start at midnight but rather at the start time ${widget.startDate} mentioned in the code snippet above (fig. 3). The 60-minute tasks were very consistent up to a point as well until it hit 10 tasks and consistently placed words before the JSON data (fig. 10). However, the AI failed much earlier than the 30-minute tasks, only scheduling about half of the tasks successfully (fig. 9).

The 90-minute tasks were far more inconsistent than each of the others, and we observed a strange behavior when the AI interacted with tasks supposed to be scheduled at midnight. The AI would schedule these tasks at 11:59pm instead of 12:00am, which led it to fail the duration accuracy test. This is what led to the failures where only one task was not accurate in terms of duration in this section of the data (fig. 8). We believe this has to do with the data ChatGPT was trained on, as 11:59pm is a common time to set due dates because it is essentially the same time as 12:00 am is but is less confusing due to keeping the same day that it is due on rather than going to the next like midnight does. The 90-minute tasks also failed far more times than the other durations and scheduled even less tasks than the others, only successfully scheduling six tasks (fig. 9) (fig. 10). Surprisingly, the total duration of the tasks scheduled in all three cases was rather consistent: the 90-minute and 60-minute tasks both had a total duration of 9 hours of tasks successfully scheduled, and we estimate that had the 30-minute tasks not run into the token limit it would have done similarly. As the duration of the tasks decreased, the accuracy increased dramatically, with the AI being able to handle over twice the number of 30-minute tasks as 90-minute tasks (fig. 9).

This information will be helpful in determining future adjustments to user token limits and extrapolating the AI's limits for any duration of task. For example, we can assume that the AI has a general baseline of around 9 hours of tasks it can schedule per schedule, and as such can plan around that accordingly. We can also set better limits on individual task durations, setting it to two to three hours from the current 23 hours and 59 minute limit.

5. RELATED WORK

Zhou et al. has compiled a list of multiple scheduling algorithms for construction [10]. One such algorithm is the Critical Path Method (CPM). CPM works as an optimization algorithm to finish a construction task in the shortest time possible. It works by finding critical activities that cannot be delayed, because they are the tasks which take the longest and therefore what the finish date is most reliant on. All other activities are called float activities and can be freely adjusted however the user wants due to how they are independent of the deadline as long as they are not completely forgotten. It is fairly effective and is a very commonly used algorithm for determining how to undertake a construction process, but it does not take into account resource constraints and production bottlenecks, instead only factoring in time. Although CPM will generally produce a better result than our AI, the AI is far faster than CPM and better fitted for general users as well as being able to handle a higher quantity of tasks.

Etschmaier and Mathasiel have also tackled the problem of scheduling tasks, but for a broader audience: airline services[11]. Airline Scheduling generally works in one of two ways: a step-wise approach and a direct approach. We will focus on the former due to how it showcases an interesting aspect that our app does not have: group scheduling. The step-wise approach works by first determining how frequently each flight route is used, known as Frequency Optimization, then determining departure time based on demand, and finally checking each time for feasibility. For our app to contain these features would require a major rework of many systems, including the addition of a friend system so that a collaboration could be verified as intentional. It would also mean that we would have to compare schedules as well, making the sent token to ChatGPT far longer and cutting down on the tokens available for a response, which is not ideal.

Giffler and Thompson discuss algorithms for solving Production-Scheduling problems, focusing on Gantt Charts as a method for solving them [12]. Gantt Charts work by first creating a matrix based upon the initial conditions of a schedule, with actions being called "facilities" and tasks to be completed called "commodities". The matrix is rearranged so that there is the minimum amount of "idle" time for all facilities, thereby being the most efficient procedure for the schedule. The paper gives several usable definitions of an "optimal" schedule, with one being the schedule with a "Gantt chart length at least as short as any other feasible schedule", meaning any schedule with the shortest possible Gantt chart length among all possible schedules. Similar to CPM, despite Gantt charts providing a higher quality schedule to follow, actually finding such a schedule is computationally impractical at best and statistically impossible at worst. Since there is no real way to "solve" a Gantt chart other than brute force, the best one can do is find a relatively efficient schedule from it and slowly improve on that; however, there is no guarantee it will be the most efficient. ChatGPT provides a close to optimal schedule without needing an extremely powerful computational device or brute force, making it far more practical for the everyday user.

6. CONCLUSIONS

The project had severe issues with large amounts of tasks due to the inherent limit of 4096 tokens between the prompt and the response for ChatGPT [14]. This meant that any prompt that was too long would cause the AI to be unable to generate a proper schedule, defeating the purpose of the

app. Also, the ranking page did not have full functionality, which we will need to fix in future updates. Additionally, the tasks page could be updated to display a blurb with all of a task's info when tapping on a task; this would be a significant QOL update to the app [14]. A password recovery option and a built-in account deletion option are two more features we wish to add in the future for a better user experience. These would allow a user to regain access to their account without having to manually fill out a Google form and wait for us to reset their password or delete their account by hand, saving both us and them time.

Despite the obvious improvements in schedule efficiency, the goal of the app is not to create the most optimized schedule. It is to quickly output a schedule that the user can easily adapt to and assist the user in accomplishing tasks that would otherwise be left untouched. As a mental health aid, the app is not a cure-all for a user's anxiety or depression, instead relieving the stress of needing to create a schedule for themselves constantly.

The app exists as an aid to help people do their work on time, as they may be unable or unwilling to create a schedule for themselves. Having this app means that they no longer need to spend time worrying over creating a schedule and can instead just follow the one given by the AI [15]. Ultimately, Intelliplan's functionality could be extended well beyond scheduling and to anything related to increased efficiency. In the future, further experiments involving this AI could be on a much larger scale such as scheduling factory shifts so that the maximum amount of workers are working the most efficiently, scheduling student's schedules so that they may complete their homework in the easiest manner, or making deadlines for a project's components so that it may go into production on time. No matter how rudimentary a prototype this app may be, the process behind its creation is sound and will be further developed in the future to facilitate more advanced creations.

REFERENCES

[1] Ping, W. A. N. G., and W. A. N. G. Xiaochun. "Effect of time management training on anxiety, depression, and sleep quality." Iranian journal of public health 47.12 (2018): 1822.

[2] Hasler, Gregor, et al. "Discovering endophenotypes for major depression." Neuropsychopharmacology 29.10 (2004): 1765-1781.

[3] Miyasaka, Lincoln Sakiara, Álvaro N. Atallah, and Bernardo Soares. "Passiflora for anxiety disorder." Cochrane Database of Systematic Reviews 1 (2007).

[4] Centers for Disease Control and Prevention. "Mental health conditions: Depression and anxiety." (2020).

[5] Firth, Joseph, et al. "The efficacy and safety of nutrient supplements in the treatment of mental disorders: a meta-review of meta-analyses of randomized controlled trials." World Psychiatry 18.3 (2019): 308-324.

[6] Carrasco, J. L., and C. Sandner. "Clinical effects of pharmacological variations in selective serotonin reuptake inhibitors: an overview." International journal of clinical practice 59.12 (2005): 1428-1434.

[7] Cascade, Elisa, Amir H. Kalali, and Sidney H. Kennedy. "Real-world data on SSRI antidepressant side effects." Psychiatry (Edgmont) 6.2 (2009): 16.

[8] Hofmann, Stefan G., et al. "The efficacy of cognitive behavioral therapy: A review of meta-analyses." Cognitive therapy and research 36 (2012): 427-440.

[9] Hoge, Elizabeth A., Ana Ivkovic, and Gregory L. Fricchione. "Generalized anxiety disorder: diagnosis and treatment." Bmj 345 (2012).

[10] Zhou, Jingyang, et al. "A review of methods and algorithms for optimizing construction scheduling." Journal of the Operational Research Society 64 (2013): 1091-1105.

[11] Etschmaier, Maximilian M., and Dennis FX Mathaisel. "Airline scheduling: An overview." Transportation science 19.2 (1985): 127-138.

[12] Giffler, Bernard, and Gerald Luther Thompson. "Algorithms for solving production-scheduling problems." Operations research 8.4 (1960): 487-503.

[13] Uysal, Muzaffer, et al. "Quality of life (QOL) and well-being research in tourism." Tourism Management 53 (2016): 244-261.
[14] Kalla, Dinesh, et al. "Study and analysis of chat GPT and its impact on different fields of study." International journal of innovative science and research technology 8.3 (2023).
[15] Barr, Avron, Edward A. Feigenbaum, and Paul R. Cohen, eds. The handbook of artificial intelligence. Vol. 1. HeurisTech Press, 1981.