Skedule: An Adaptive Planner

Zachary Inn, Brynn McGovern, Ronan Kearns, Dylan Inafuku

skedule®
~ An Online & Chill Studios App

Why Our Project Matters

- 50% of college students have been found to partake in some form of procrastination (Day, 2000)
- Stress caused by procrastination is the result of a student's perceived control of time, in other words if you feel like there is less time to study, you stress more.
- Especially without a structured schedule it is very easy to get off-task and lose focus on school work
- Our focus was to build an app that would assist college students, with time management through a user-specific scheduling app
- Great time management skills correlate with academic success (Martin et al. 2020)
- The app creates personalized study schedules to ensure that students can keep track of their studies in a timely manner in order to limit the amount of stress in their lives

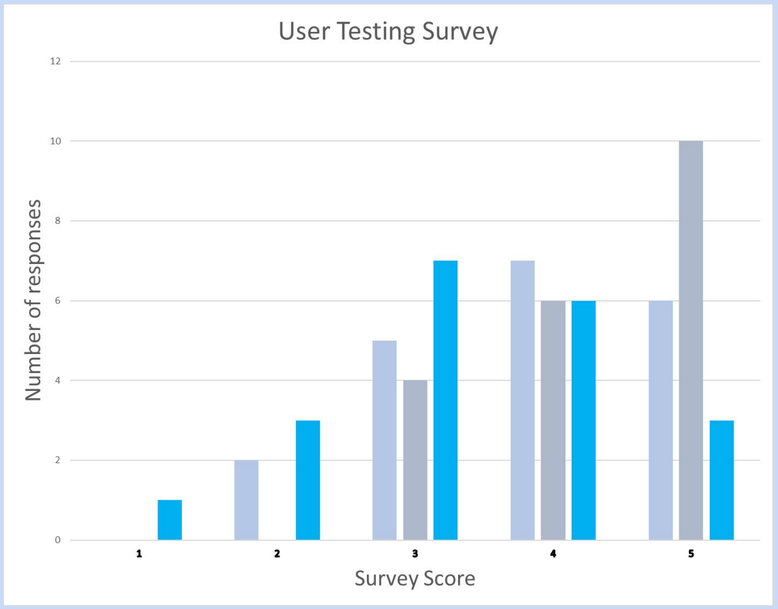

Figure 1: 1-5 scale from "completely disagree" to "completely agree" on how users felt about the user interface of Skedule

Legend (Questions asked on User Testing Survey):

Periwinkle: The user interface is easy to use and understand

Gray: The idea of an algorithm creating your study schedule is appealing to apply to everyday life

Blue: The schedules created by the algorithm are fair and balanced enough to be implemented into everyday life

Introduction

Our grand challenge is finding a way to manage your time wisely, as most cases of procrastination in college students are the result of poor time management (Macan et al. 1990). Our goal is to create an app that would help college students with their time management by creating study schedules based on their preferences. We believe that a study schedule fit for higher level students would actively cut down procrastination and stress amongst them.

Backend/Frontend of App

```
System.out.println("Hello, welcome to Skedule.");
System.out.println("Before we begin, you must answer a few questions about your studying habits.");
System.out.println("How many days a week would you like to study?");
Scanner input = new Scanner(System.in);
int num = input.nextInt();
int total = 0;
total += num;
System.out.println("How many times a day would you like to study?");
num = input.nextInt();
total += num;
System.out.println("Do you prefer short study sessions or long study sessions? (Short = 1; Long = 2)");
num = input.nextInt();
total += num;
```

Figure 2: Algorithm for user-specific scheduling (score method). The algorithm assigns values to user answers, and uses the total value to design a schedule. If a session was poor then the score would decrease and the algorithm would either make the next session shorter or add another break in the session.

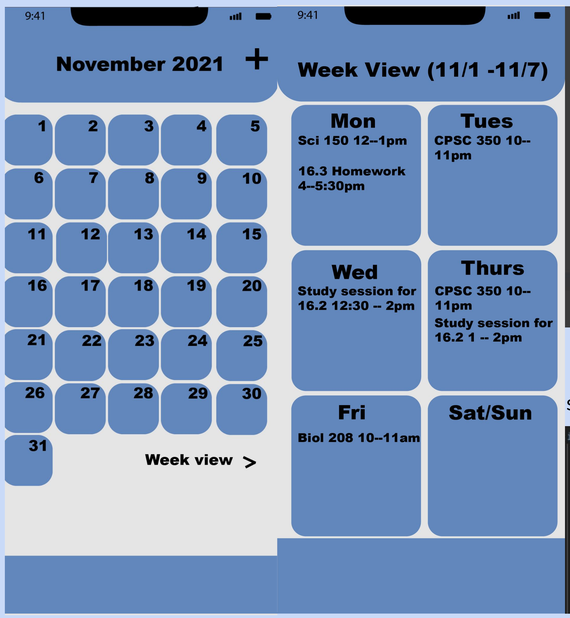

Figure 3:
Home screen, displays any upcoming study sessions

Figure 4:
Weekly view, displays upcoming due dates/tests

```
if (score < 25)
{
 subtractStudySession();
 addBreak();
 userClock(34.8, 42.0);
}
else if (score >= 25 && score <= 30)
{
 subtractStudySession(15);
 addBreak(15);
 userClock(42.0, 56.1);
}
else if (score > 30 && score < 35)
{
 subtractStudySession(30);
 addBreak(10);
 userClock(56.1, 69.3);
}
```

Figure 5:
Study session algorithm, higher score correlates to shorter session

```
int break::countdownClock(){
 int minutes;
 int seconds;
 int i;
 for(i = timeInSeconds; i >= 0; --i){
  minutes = i / 60;
  seconds = i % 60;
  cout << "Time left in break: " << minutes << " : " << seconds << endl;
  usleep(1000000);
  system ("clear");
}
```

Figure 6:
Clock Operations, used when session is active

Methods

1. Algorithm Research
2. Organizing Class Structure
3. Implementing Algorithm for Assisted Planning
4. Debugging and Testing
5. User Testing and Feedback

Tools: C++, Java, Illustrator, React, VSCode, Atom

Current Progress/Next Steps

We created a terminal based app with some UI elements. The next steps for this project would be to connect the frontend to the backend and implement a more user-specific algorithm than our score algorithm. This algorithm would more accurately adjust to each user's studying preferences but will take much more time to implement.

Conclusion

- **Big Picture Finding:** Using this application is an effective and efficient way to manage workload
- Users generally enjoyed the functionality of the app and user specific adjustments
- Our focus onwards would be improvements on the frontend of the application (UI)
- More user testing would have been beneficial to our application's usability
- Future work would involve improvement of the UI, increased user testing, and additional features

Acknowledgments

We would like to thank Kosta Kassarass for giving us advice on how to plan and build our app. He gave us invaluable insight on the steps that should be taken on our project.