Andrew Biehl, Eva Grench, Calypso Leonard, Yingying Wang
Software Design, Amy Csizmar Dalal
Due February 26th, 2018

Prøductivity+: A New Kind of Task Manager

Goals

We want to create a desktop app that allows the users to keep track of the tasks they do throughout the day and then analyze their productivity. There will be manual input mode for individual events or automatic input mode *Set Schedule Template* for recurring events on different weekdays. The users will be able to input their ideal schedule for the day in the morning or the day before and then update the app to track their activity throughout the day to keep a log of how their time was actually spent. A *Task* bar on the side will display all the scheduled events of the day. Once user clicks the activity or event's icon, our app will record the starting time from system/machine time. At the end of the day, the user will be asked to rate how productive they felt they were. As the app accumulates data, it will show the user where they tend to get off track and offer tips to help them be more productive. This is an app that would likely be useful to individuals who want to either maximize their productive habits, or who want to analyze their productivity. Some such individuals would be college students looking to find the best work/life balance, or managers for companies which want to track the best practices of their employees.

Prioritized List

1. Implement the main GUI page, consisting of the two time management bars. The bar on the left will display the "schedule goal", created by the user. The bar on the right will track the "real schedule" that the user inputs in real time via task buttons to the right of the bar.
2. Implement a GUI page in which the user is able to create new activities and add them to customizable "schedule goal" templates. These templates will then be usable in the main GUI page to compare against the real schedule.
3. Implement a productivity gauge, in which the user is able to rate how productive they thought they were after completing either a certain activity or after the end of the day.
4. Allow the user to save schedules as templates and select them as their schedule goal.
5. Allow for the user to import and/or export schedule templates or data that they have recorded in the app.
6. Create a GUI page that tracks the user's data and provides some basic data visualization.

7. Allow for the user to manually edit the actual schedule in order to account for mistakes that were made in adding data in real time (e.g. the user forgot to notify the app that they started or finished an activity).
8. Add personalized tips that respond to the user's activity.

Design Pattern

We plan on using MVC pattern to implement the overall structure of our application. This makes sense because the we will have a front-facing user interface which will communicate with a back-end regulation of user-inputted data. These sections will communicate using the observer pattern. If the user clicks on the GUI at some button or menu, this will be "pushed" to the controller, which will push the appropriate information to the model. The model will update and push its response back up through the chain of command to the view.

Partition of Work

Our team does not plan to adopt the pair-programming style for this project because it would be hard to share the screen among four group members, and it's inefficient to have 3 people watch while one person driving the computer. We will work mostly on our own or with one other person and communicate with other group members through messages/emails to update our progresses. Our plan is to create a list of task goals, which we will divide up by mutual agreement, to ensure we meet the deliverable timeline. Completed tasks will then we available in our GitHub for us to debug or reconfigure as needed. Whoever completed the task will be the primary person responsible for maintaining that section of code as we adjust our application, but ultimately we will all be responsible to make our code cohesive.

Task List:

Main Page (View) - Andrew, Eva

- Selected goal schedule
- Real schedule updated in real time
- Activities that can be selected
- Menu

Set Schedule (View) - Calypso, Yingying

- Adding a task to the schedule template
- Visualization of the schedule template

- Saved templates
- Save as template button and use as current template buttons

Report Productivity Pop-up (View) - Calypso

- Question
- Productivity slider (or something like it)

Trends (View) - Andrew

- Graph - dummy data to start
- Tips button

Productivity Tips Pop-up (View) - Eva

- List of suggestions - dummy data to start

Timer Logic (Model) - Yingying

- Start/stop timer function
    - Ask for productivity when stop is called
    - Store task, start, stop, and productivity
- Find total time

Average Time Spent (Model) - Calypso

- Keeps track of average time spent on each task actually and expected
- Adjust the average after a task is finished

Add to list of data (Model) - Andrew

- Information sent from view → controller → model that stores it in a list of list of lists where each time a button is pushed on the view that stops a task the model stores it as a list of length 4 things (task, start time, stop time, and productivity). These are grouped by day.

Find difference between real and expected (Model) - Eva

- For each task the user said they wanted to do and actually did, find the difference (if they do a task multiple times a day, find the sum first and then the difference).

Analyze productivity (Model) - Calypso, Yingying

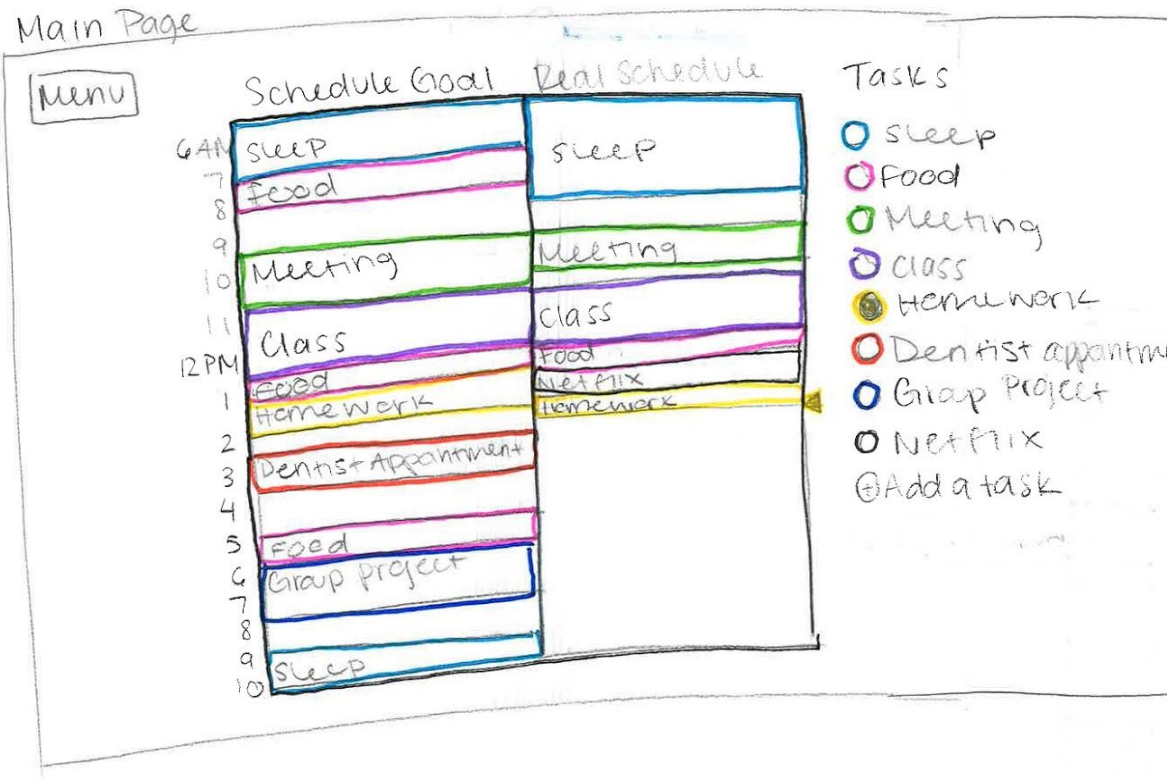
- Find top three tasks with the biggest difference between real and expected averages.
    - Send different messages based on whether or not they spent more or less time than expected.
- Find top three tasks with lowest productivity
    - Send to view and suggest they focus on these
- Find top three tasks with highest productivity
    - Send to view and suggest they cut these out

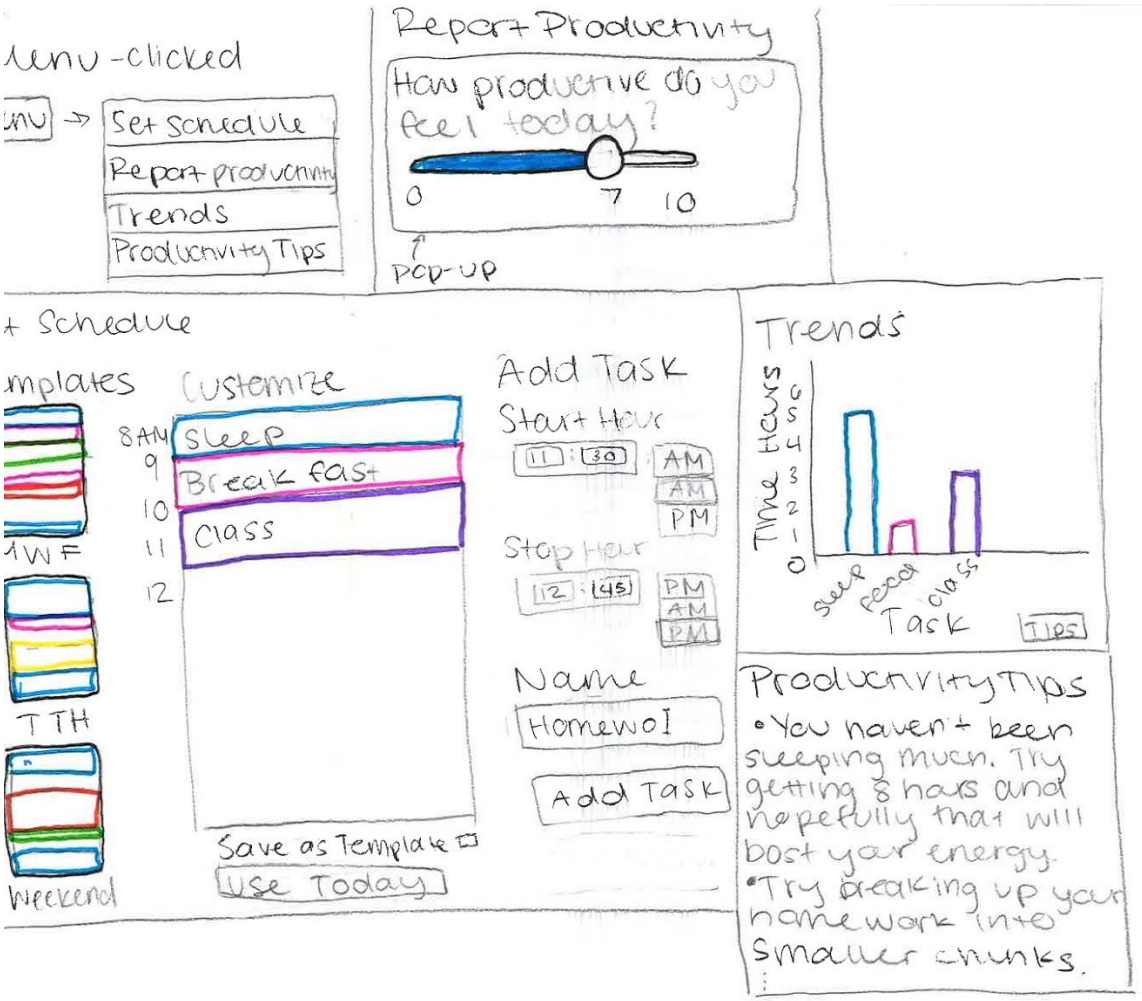
Controller Implementation - Everyone

- The controller will be implemented in response to how the model and the view send and receive data. The controller will be used to modify the data received from one end so that it can be sent out to the other end in an acceptable format.

<u>Sketch</u>

Fig. 1 shows the basic functional homepage of our app, while fig. 2 shows some of the possible extended features which we will attempt to implement once our basic functionality is working correctly.




Menu -clicked
Menu
Set Schedule
Report Productivity
Trends
Productivity Tips
Report Productivity
How productive do you feel today?
0
7
10
POP-UP
+ Schedule
Templates
Customize
Add Task
Start Hour
11 30 AM
Stop Hour
12 45 PM
Name
Homework
Add Task
8AM
9
10
11
12
Sleep
Breakfast
Class
MWF
TTH
Weekend
Save as Template
Use Today
Trends
Time Hours
Task
TIPS
Productivity Tips
• You haven't been sleeping much. Try getting 8 hours and hopefully that will boost your energy.
• Try breaking up your homework into smaller chunks.