MOMENTUM

1. MOTIVATION

Programmers have a reputation for being terrible at estimating how long it will take to complete a project [1] [2] [3]. This is because developing software is inherently uncertain: there is no way to forecast every detail of a design from the outset, understanding the full requirements of the project as well as the hurdles that an individual will encounter along the way. (If this were the case, developing software would be a trivial task). This is only exacerbated by the fact that a huge number of programming projects are not completed by a single person, but by teams — if an individual can't predict how long something will take for themselves, what hope does a team have of estimating the completion time for an entire project? Ultimately, this is a problem that can have huge monetary, emotional, and interpersonal costs [4]. The reality is that programmers are usually much too optimistic [1] [2] [3] — they try to fit too many features into a release, and some have to be cut; they think that they can focus effort in one area of a project because it will be quick, but it ends up taking two, three, four times as long as they had thought initially. Having better time estimates lets us choose how we focus our efforts [4] [5].

Additionally, existing effort estimation tools are unintuitive to use (*cf. section 2.2*) and many require extensive knowledge of effort estimation processes (*cf. section 2.3*) in order to use them properly. While learning these processes would not be inherently detrimental to a software developer — in fact, this may be in the developer's best interest — the reality is that many people (including software developers) are not good at making decisions that will benefit themselves in the long term [14]. Thus, Momentum aims to reduce the entry cost of accurate effort estimation by removing the need to have in-depth knowledge of effort estimation processes.

2. APPROACH

2.1. Goals

Our project involves creating a tool that uses information gathered about a user (and their team members) to predict a project's ship date. Our planned implementation approach focuses on the a single component of an effort estimation tool: the user interface (Fig. 2.1.1).

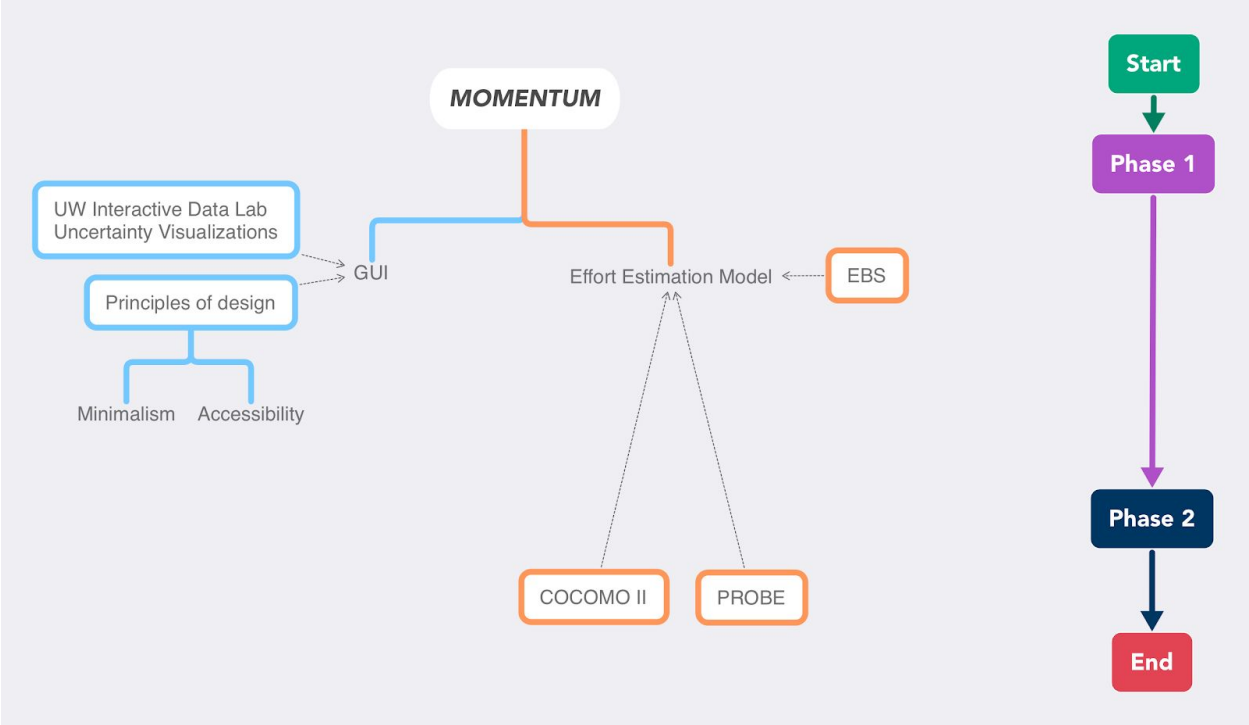

Fig. 2.1.1 A representation of of the two phases of our design process, as well as the two components we will be addressing.

As seen in Fig. 2.1.1 Phase 1, we plan to begin by focusing on implementing an accessible, simple, and intuitive GUI, using Evidence-Based Scheduling (*cf. section 2.3.2*) as our initial effort estimation model. Momentum will be the only tool that is simple to use, does not require high maintenance or training, and is cheap and accessible for the public. Phase 1 is our primary focus.

As seen in Fig. 2.1.1 Phase 2, we no longer plan to refine our effort estimation model (*cf. section 2.3*) in order to improve accuracy or ease-of-use. We do not have enough time to finish Phase 2.

In section 2.2, we will address existing GUI tools for software development effort estimation. In section 2.3, we will discuss existing processes for effort estimation.

2.2. Existing GUI Tools

Existing tools related to our project include Manuscript, ProcessPAIR, and Process Dashboard. Manuscript is outlined below.

1) Manuscript Overview

Manuscript is a web app for software teams that assists in the processes of project management, issue tracking, and support [8]. A subscription for a team of just five members can cost from $62 to $75 per month. We decided to use this competing tool for our user research rather than other competing tools like ProcessPAIR and Process Dashboard for the following reasons:

1. Manuscript produces estimates for team-wide estimations, which aligns to the goals of our tool
2. Manuscript uses EBS (*cf. section 2.3.2*), which is the effort estimation model we will use for Phase 1 of Manuscript development, and
3. Manuscript's audience are software engineers who do not need prior knowledge of PSP model and processes, which most closely aligns to the audience that we want to build our tool for.

2) User Research

Because a large goal of our project is to make an extremely accessible tool, we are substantiating our accessibility decisions based on data collected from user research. Our first phase of this research is to see what users think about our biggest competitor, Manuscript. This will help us make our tool more user-friendly and achieve our goal while also eliminating our own bias in usability of other projects. At this point, we have completed three informal user tests. The users are detailed below and will be referred to as “User 1”, “User 2”, and “User 3” for the rest of this portion of the report. We plan to perform five user tests on Manuscript as “testing with 5 people lets you find almost as many usability problems as you'd find using many more test participants” [15].

> User 1: User 1 is a fourth-year UW student in both the CSE and HCDE departments that has previous user interface and experience development experience through a past internship as well as software development experience on project teams. User 1 has never used Manuscript before the test and has been programming for about six years.
>
> User 2: User 2 is a third-year UW student in the CSE department that has previous software development experience on project teams through a past internship. User 2 has never used Manuscript before the test and has been programming for about four years.
>
> User 3: User 3 is a second-year UW student in the CSE department that has previous software development experience through 7 internships, 6 of which have included working on project teams. User 3 has never used Manuscript before the test and has been programming for about eight years.

The chosen tasks are summarized below as well as our intention behind choosing each task.

> Tasks 1-3: These tasks are presented as they represent a direct bridge between Manuscript and our tool. Both tools should and do offer the ability to create a space to hold related tasks, create and modify tasks, and start, stop, and resolve these tasks. Because these tasks are shared by both Manuscript and our tool, we believe that we can directly measure our success in our usability improvements by comparing user experience in this task across both tools.
>
> Overarching Questions: In this section, several overarching questions are asked of the user to conclude and summarize the user’s experience with the tasks given. We ask these questions to break away from the structure of a task and allow users to voice any

other concerns or compliments of the tool after they have finished completing their task. Cross-referencing the responses from this section with the performance of users in each task allows us understand how much ease or complication in each task contributed to the user's overall view of the product.

Task 1: Creating a new project

In this section, users were given the following scenario: "You are a part of a software project in which you have already broken down the tasks that are assigned to you and your fellow peers. Create a place to store these tasks and fill out any forms to your best ability." The users were started on the "Main Iteration Planner" screen of Manuscript. The most important takeaways from this task from each user are detailed below.

User 1: User 1 did not create the correct item. The intended purpose of this task was to create a card similar to those seen on the Iteration Planner Screen which are named "Intro to Manuscript" and "Set up your account" in the figure. Instead, the user created a "case" which does not fulfill the requirement of the task as a case does not have the ability to store tasks objects as a "case" in Manuscript is the equivalent of a task. User 1 did not understand how to fill out more than half of the form options in the New Case screen and wondered about their necessity. In terms of forms we are interested in including, the user was confused with both the "Priority" and "Estimation" forms, remarking that the priority system didn't seem to make sense as several different priority numbers had the same textual description and that they didn't understand what "Estimate current" meant in terms of a project.

User 2: User 2 did not create the correct item. The intended purpose of this task was to create a card similar to those seen on the Iteration Planner Screen which are named "Intro to Manuscript" and "Set up your account" in the figure. Instead, the user created a "case" which does not fulfill the requirement of the task. User 2 did not understand how to fill out more than half of the form options in the New Case screen. In terms of forms we are interested in including, the user was confused with only the "Priority", but filled the "Estimation" form in incorrectly. In the "Priority" form, the user was confused why different priority numbers were given the same description and did not know which of the three priorities with descriptions of "Must Fix" would be the highest priority. The user also wondered why the priority descriptions talked about "fixing" when the case was set to "Feature" not "Bug" in the "Category" section. The user understood what the "Estimation" form was asking for, but was confused as to what metric to use in the estimation (minutes/hours/days) and ended up entering the metric in days which corrected to hours. The user suggested that Manuscript add a small description of what each form prompt means to help users.

User 3: User 3 did not necessarily create the correct item. The intended purpose of this task was to create a "Milestone" similar to those seen on the Iteration Planner Screen which are named "Intro to Manuscript" and "Set up your account" in the figure; however, the user created a "Project" which, while not inherently wrong, was not the intention of the task. This indicates that the task instructions need to be clearer. Regardless, the

user's first instinct was to click "case" which does not fulfill the requirement of the task. Unlike other users, User 3 realized while filling out the "New Case" screen that this would not constitute a project and exited from this screen. The user then could not find a corresponding button and almost gave up before finding the "Projects" tab. When found, the user remarked that they did not know why it was so hidden. When entered into the "Project View" screen, the user remarked that the screen was "ugly" and "needed fixing".

Task 2: Creating a new task

In this section, users were given the following scenario: You are a part of a software project in which you have already broken down the tasks that are assigned to you and your fellow peers. Create a task and assign it to yourself, filling out any forms to your best ability.

The users were started on the "Main Iteration Planner" screen of Manuscript, shown. The most important takeaways from this task from each user are detailed below.

User 1: User 1 created the correct item. There was some confusion at the beginning of the task due that was due to the failure to complete the first task correctly. While scrolling around the "Main Iteration Planner" screen, the user remarked that they could not find the project that they created in the previous task. The user also suggested that they didn't want to click the "Plus" button at first because it was creating a "case" which is the same object that was created in the previous task. Once the user was on the "New Case" screen, the user realized that the last task had been completed incorrectly and did not attempt to fill any forms that were left untouched during the last task.

User 2: User 2 created the correct item. The user clicked on several different buttons on the "Main Iteration Planner" screen before the correct button, saying that they did not see another button for creating the correct object. Once the user was on the "New Case" screen, the user did not realize that they were on the same page as in the previous task, but remarked that the screen was very familiar. The user filled out the same forms as in the previous task, but realized this time that estimation should be filled out in hours and not days.

User 3: User 3 created the correct item. The user remarked on the inability to distinguish between different priority levels due to their identical descriptions; on this note, the user also commented that "the priorities should not be labeled as 'must fix' when I've set the type to 'feature' ". In terms of estimation, when the user estimated the task's time in days, Manuscript translated this information into hours by assuming 7.5 hours/day. This could be inconsistent with the perceived number of hours per day from the user's assumptions.

Task 3: Altering created tasks

In this section, users were given the following scenario: You are a part of a software project in which you have already broken down the tasks that are assigned to you and your fellow peers. Enter into the task that you have previously created, indicate how you would begin working on the task, stop working on the task, and resolve the task.

The users were started on the “Main Iteration Planner” screen of Manuscript. The most important takeaways from this task from each user are detailed below.

> User 1: User 1 did each of the tasks correctly. It took the user a bit to find the correct case as they did not know where in the “Main Iteration Planner” screen that the case would be located. This is most likely due to the failure to complete Task 1 correctly. The user understood the “Estimate” metric when it was presented on the “Main Iteration Planner” screen with the unit of “hours” appended to the end. The user remarked that after clicking “Start Work” on the “Case View” screen, the “Stop Work” button should appear without needing the exit out of the “Case View” screen and enter again to see it. Overall, this user did much better on this task than any other task.
>
> User 2: User 2 did each of the tasks correctly. It took the user a bit to find the correct case as they did not know where in the “Main Iteration Planner” screen that the case would be located. This is most likely due to the failure to complete Task 1 correctly. The user thought that the progress bar on the case was the button to press to access the “Case View”. The user was confused as to why they need to leave the “Case View” page and re-enter the page in order to access the “Stop Work” button after clicking the “Start Work” button. Overall, this user did much better on this task than any other task.
>
> User 3: User 3 did each of the tasks correctly after they had been re-explained. The user took a moment to find the correct case as they had not specified which milestone to add the case to in the previous task. The user then correctly found all buttons specified in the task though in two separate attempts. To clarify, the user did not understand the task in the first attempt. The user thought the task was to only start work and resolve.

Conclusion: Overarching comments

In this section, users were asked questions about their experience using Manuscript. The questions are repeated below as are the answers from each user.

Question 1: Did you have any troubles using Manuscript? If so, what were they?

> User 1: Yes. I did not figure out how to fill out a lot of the forms in the creation stage and I think I created the wrong thing in the first part of the test.
>
> User 2: Yes. I had trouble filling out the case thing and didn’t really know if I was doing the correct thing the whole time.
>
> User 3: Yes. The “case” thing is confusing. It was hard to find “Project”.

Question 2: What did you like about Manuscript?

> User 1: I liked that it had a lot of helpful fields to fill out like due date and priority, I wish they would’ve been more intuitive though. I also liked that you could attach things in the description part.

User 2: I liked that it looked like Trello and I've used that before so I kind of knew what was happening on the main page.

User 3: It has a nice visual layout of seeing what needs to be done. It does a good job at showing complexity.

Question 3: What did you dislike about Manuscript?

User 1: There seemed to be a lot of unnecessary bits that confused me when I was trying to complete the tests.

User 2: I didn't like that there was no explanation on any of the things you needed to fill out. That didn't really make me feel confident in knowing what I was doing.

User 3: It's too complicated for new users. I wasn't sure what everything was.

Question 4: Would you use Manuscript for future projects if you needed to keep track of project-wide estimations?

User 1: No.

User 2: I don't think so, no.

User 3: Yes for a team for any decent amount of complexity. Small teams not as much.

Question 5: Would you recommend Manuscript to others?

User 1: No.

User 2: No.

User 3: Depends on the use case. New teams definitely not.

Question 6: How would you rate your experience with Manuscript with 1 being the lowest and 5 being the highest?

User 1: 1.5

User 2: 2

User 3: 2.5

Concluding Remarks

Overall, users found Manuscript hard to use and were not always sure that they were completing the tasks correctly. They found that many fields and interfaces were not explained well enough for a first time user to fill out. The users would rather use other methods to estimate their projects than to use Manuscript, which leads us to believe that there is room for vast improvement in the usability of a tool that can help with project estimation. We strongly believe that a tool with a seamless user interface is a very novel contribution that drives users to

begin using and continue using our tool. Fixing the problems that users had with Manuscript will lead users to want to use our tool for their projects and benefit their overall workflow.

2.3. Existing Effort Estimation Processes/Models

There are many existing effort estimation processes embedded into larger software project management products. Three notable existing methods for effort estimation are PROBE, EBS, and COCOMO II.

1) PROBE: PROxy-Based Estimation (PROBE) is a component of a larger system known as the Personal Software Process (PSP) [6]. Developers using PROBE first break down a large project into smaller subprojects, then relate the size and type of each of these subprojects to subprojects (known as proxies) the developer has completed before. This knowledge is used to estimate the size, complexity, and time required for a large-scale project.

PROBE focuses primarily on how to break down and estimate large software projects, but does not emphasize integration into a larger team process. Additionally, PROBE is often embedded in the larger PSP, which has a high barrier to entry. Developers using the PSP often must read long books or take classes before they are able to effectively use the system. PROBE is often introduced as a whitebox process, and developers are consciously aware of the underlying algorithm driving estimation.

2) EBS: Evidence-Based Scheduling (EBS) is a method and piece of software for effort estimation developed by Joel Spolsky at FogCreek and has been integrated into their product called Manuscript [7]. Like PROBE, EBS first begins with the team of developers breaking down a large software project into smaller subprojects. Individual developers then cut these up into tasks they think they can complete in no more than a day of work. Developers estimate the amount of time each component will take. The EBS software keeps track of how much time the component actually takes and creates a model over time of how good an estimator the developer is. The EBS software then uses this model to predict how long the rest of the project will take and produces a probability distribution for the estimated ship date of the project.

EBS focuses less on how to break a project into smaller pieces and more on how to use effort estimates to produce projections for project completion that vary over time. While this is useful, we argue that the process of breaking a project down is just as important as estimating the time it will take a project to ship since it forces developers to think deeply about the larger structure of their project. Additionally, existing tools for EBS have poor visualizations of the data they produce.

3) COCOMO II: COnstructive COst MOdel II (COCOMO II) is a method for estimating software cost and time, generally for large organizations [12]. It consists of two models for estimating the cost and schedule of software projects: Early Design and Post-Architecture. The Early Design model is used to investigate alternative designs before a project's shape has been fully realized. The Post-Architecture model is for projects that are entering the coding stage of development. COCOMO II takes a collection of factors such as team cohesion, project novelty, and specification flexibility and produces an estimate of how much a project will cost and how long it will take to complete. Although detailed, the estimation process places a significant burden on those who use it. COCOMO II bases its numbers on 23 different factors [13], all of which must be considered by developers. While developers likely benefit from considering these

factors, COCOMO II's complexity creates a significant barrier to entry for new users.

2.4. Our Approach

In our approach to software development effort estimation, we plan to first focus on creating a better user interface. Our goal is to create a new UI that would have a low barrier of entry, and would be easy to maintain. For example, each page allows the user to perform a singular main function. There is a page that focuses on organizing the users' projects, a page focusing on details of a task, and a page focusing on giving an overview of running tasks. Current estimation tool have a high barrier of entry. For new users, it could take an hour or two to first learn how to navigate the tool and enter their data correctly before being able to actually use the tool to estimate a project's completion time. We plan to improve the interface so that users can quickly learn how to use our product to estimate their projects' completion times. Additionally, our tool would have low overhead. Once the project is set up using our tool, it should only take a few minutes each day to manage the tasks, when the user is working on the project.

We also plan to improve the way the data is visualized. The current tools provides a few graphs of the data that are not necessarily intuitive. Our plan is to improve upon the data visualization in our user interface using research on visualizing uncertainty from UW's Interactive Data Lab so that the analyzed information (e.g. ship date distribution) is easier to understand for the users [11].

Below are specific examples of the improvements we will make to the GUI. We see our direct competitor as Manuscript (see section 2.3), so we will compare our GUI mockups with Manuscript to display differences in design ideology. Note that Manuscript uses EBS (see section 2.3), and Momentum will also initially use EBS.

Fig. 2.4.1. The image above is the task board for Manuscript while the image below is the task board for Momentum. Momentum's task board is improved in that the interface is significantly simplified as our task board doesn't overwhelm the user with a multitude of non-important features. Momentum's task board has data visualizations easily accessible on the page, which summarize the project's time projections and estimations. It shows tasks ranked by the order that the user chooses (such as ordering by newest tasks first, or ordering by deadlines). The user can also easily see the severity/priority of each task. An additional feature included in Monument is that the user can visually see the progress/time bar of each task.

Fig. 2.4.2. Another example where Manuscript lacks in user friendliness is that the user is required to click on a small link in the navigation menu in order to view their projects, and then require two separate screens just to create a new project. Their screens also provide too many options which require reading the manual to discern. For example, there is no clear indication on what the difference of Milestones vs Tasks are. There is also no explanation for the usage and differences of Areas vs Case Groups either. Momentum, on the other hand, avoids these confusions as its project page involves only one screen which elegantly displays all the user's projects after logging in, and then allows the user to add information about a new project without being redirected to a new page.

Depending on the amount of time remaining after we complete the user interface, we will attempt to improve the effort estimation accuracy of our product (Fig. II-A-1, Phase 2). This may involve taking cues from the COCOMO family of estimation tools, PSP/TSP, etc.

The user manual for our tool may be viewed at this link:
https://github.com/cse403-MOMENTUM/MOMENTUM/blob/master/user-manual.md

2.5. GUI Mockup

A limited interactive experience with this GUI is available at the link below:

https://xd.adobe.com/view/b2653ac4-d8e2-4d64-90e0-d5cb276f5fb3/

All current GUI screens are provided and described below:

Fig. 2.5.1 An example landing page.

A landing page (Fig. 2.5.1) will allow the user to manage multiple project at a time. The user can click on any one of projects to get to project page of the project he/she have selected. To create a new project, the user can click on the plus icon and it will direct him/her to our new project page. To delete a project, the user can click on the X icon and then click on the project he/she wants to delete. The down arrow allows the user to scroll down his/her list of project. In each of our pages, there is an up arrow on the bottom right that will take the user back to the top of the page.

Fig. 2.5.2 An example project page.

A project page (Fig. 2.5.2) will allow the user to manage the project's tasks. The user can click on any one of tasks to get to task page of the task he/she have selected. To create a new task, the user can click on the plus icon and it will direct him/her to our new task page. To delete a project, the user can click on the X icon and then click on the task he/she wants to delete. The down arrow allows the user to scroll down the list of task. The icon above the progress bars allows the user to sort the project based on various parameters. The left arrow will take you back to the landing page. The name of the project is displayed at the top. Right below it is the names of the people who are part of this project. The graphs at the bottom of the page displays information about the project such as the distribution on when the project will be ready to be deployed.

Fig. 2.5.3 An example net task page & estimated task time

To create a new task (Fig. 2.5.3, upper), the user fills in the task name, selects a priority level, gives an estimate for how long the task will take, provides a description of the task, and clicks the “create this task” button. After the user finishes creating the task, they will be redirected back to the project’s page. If the user decides to cancel the task creation, they can click the left arrow button to return to the project’s page.

Fig. 2.5.4 An example task page.

A task page (Fig. 2.5.4) shows a task that is currently being worked on. This page shows the estimated time needed to complete the task as well as the time already spent on the task. The user can press the “start working on this task” button to begin or resume the timer when the timer is paused. Similarly, the user can press the “stop working on this task” to stop the timer when the timer is running. The button underneath is used to “mark this task as completed” after the task is complete. The name of the task is displayed at the top. The name of the person assigned to the task is displayed underneath the task name. The description of the task is displayed at the bottom of the page. The left arrow button takes the user back to the project’s page.

Fig. 2.5.5 Uncertainty visualization in Manuscript vs. Momentum (examples from [11]).

Rather than visualizing ship dates as a line representing the cumulative probability of project completion, we propose visualizing times as a quantile dotplot. Researchers have found [11] that users are able to better interpret the uncertainty of a predicted time when visualized as a quantile dotplot than as a cumulative distribution.

2.6. Architecture

Fig. 2.6.1 Architecture

Right now, we plan on using two primary Javascript frameworks to build Momentum: React and Redux. React will be used primarily to create user-interface components, displays, etc., and will handle sending event-based actions to Redux. Redux will be used to handle the model/state to be displayed in React. Currently, we plan on embedding EBS modeling into the Redux component of the application (Fig. 2.6.1), as well as using an external SQL database to store relevant information, synthesized from the state in Redux. Information in the SQL database will likely include information related to users, projects, and tasks.

3. Challenges and Risks

3.1. Anticipated Risks

There are three large sources of risk in this project that we anticipate could cause us major problems:

1. Creating something too similar to another product.
2. Creating *another tool*. This is an issue that is core to our vision: we want to make something that people would actually use in the software development process. That being said, we don't want our project to be another high-maintenance tool for developers and project managers to have to constantly maintain. This is, in many ways, antithetical to the process proposed in EBS and PROBE: for estimation to work correctly, you often have to manage historical data as well as break down specifications into anticipated tasks — things that are large tasks in and of themselves that may detract from a low-maintenance product.
3. Measuring the success of our product. Since our project focuses heavily on helping teams estimate the time needed for development of a particular software project, observing the actual usage of the deliverables we produce may be difficult (as the nature of this estimation is that it occurs over a longer period of time).

3.2. Anticipated Cost

We anticipate that we may be able to complete the basis of extending a better interface by week 7, which should leave us several weeks to add in additional features to the project.

3.3. Measurement of Success

TABLE I

ANTICIPATED TABLE WITH MEASUREMENTS FOR FINAL REPORT

Individual A results:

	Estimated Time	Actual Time	Accuracy
Task 1			
Task 2			
...			

Because an important feature of our tool is to tailor itself to the individual, it is best to conduct experiments that involve case studies with programmers.

To critique whether our project sufficiently solves time tracking and estimation problems for programmers, we intend to conduct informal user studies on peers in the computer science department and observe their behavior working with Manuscript, a competing tool. This way we can gauge what functionalities Manuscript have and lack. We have already outlined our user study forms here, and plan to conduct research with at least three engineers.

In order to evaluate the success of our project, a possible experiment to check for success would be using small test cases to check that our project works as intended. For example, we can conduct experiments where the user passes in input, does their work as usual, and then we check whether the estimation given by the project is fairly close to how long it actually took for the user to complete the project. The table for measurements we could take is shown in Table 1. We would need to conduct this experiment multiple times with the same user to test whether the accuracy of the estimations improve over time, as well as experimenting with multiple individuals to test whether the program can fit engineers with various work styles. We plan to give ourselves ample time to evaluate our project by starting our user research by week 7. Additionally, we are applying our estimation model to our own project during our implementation phase by recording our estimated and actual times taken to complete implementation tasks. . Our project would be successful if it is shown that Momentum's predictions of task has less than an average of 20% error after at least 100 inputs. The midterm "exam" would only test that Momentum would progressively estimate time accurately for one type of engineering work, while the final "exam" could include a variety of engineers who are completely new to the project and have different work styles. This would test both the usability of the project as well as its robustness. Ideally, our project would be tested by companies however this is not within the scope of the course.

4. Proposed Schedule/Updated Schedule

Week	**Milestone**
Week 3	**- Locate case studies for project.** - Gain access to source code for software management tools that employ time prediction. - Continue research on existing time estimation strategies.
Week 4	- MVP (minimum viable product) for new user interface and time estimation backend - Investigate user research and evaluation methods. Conduct user research on competing tools
Week 5	**- MVP for additional workflow enhancements.** - Establish metrics for software success and schedule user research.
Week 6	**- Continue working on MVP's additional workflow enhancements.**
Week 7	**- Complete "draft" of project.** **- Conduct first round of user testing to evaluate the effectiveness of our tool compared to existing solutions.**
Week 8	**- Decide on the "vertical" direction to take beyond first MVP.** - Vertical directions: Potentially domain-specific - Take lessons from MVP and improve it. - Project draft refinements based on user testing
Week 9	- Begin project presentation work - Second round of project draft refinements based on user testing
Week 10	**- Finalization of refinements** **- Finalization of project presentation**
Week 11	**- Complete project presentation**

5. Initial Results

5.1 User Research

The initial results of our user research includes the complete semi-formally user tests for Manuscript as well as informal user testing that was not recorded on our initial mockups of Momentum. Semi-formal user tests will be conducted on our first mockup shortly and added to the report document.

The user research on Manuscript indicates that there is much room for substantial improvements in user experience, as Manuscript's average experience score with users was a two out of five with one being the lowest and five being the highest. Most importantly, we see that Manuscript has non-intuitive presentation and access to project, milestone, and case creation buttons as well as broadly confusing form instructions without proper explanations. Outside of actual usability, users found that the user interface was boring and cluttered.

The user research on Momentum indicates that many of Momentum's user experience problems have been solved. When asked for a experience score, users on average reported a score of four out of five. This is a massive improvement already when we consider that Manuscript received an average of two out of five from the same user group. The users also found Momentum had a more aesthetically pleasing user interface and tasks were completed much faster. On the negative side, users were confused on one form instruction considering the "priority" of the task. This will be fixed in later mockups.

Semi-formal user tests will be conducted on our first Momentum mockup shortly and added to the report document. After each iteration of user tests on the Momentum mockup, the mockup with be tweaked according to user responses and a new round of user testing will begin. We plan to do this cycle at least twice given the amount of time we have to complete our project. We believe this is imperative to the success of our project as our user interface is our novel contribution to EBS. Following this, User research is also very important in evaluating our success in this project as is backs up our claims that Momentum's user interface is more accessible and usable that Manuscript.

5.2 Implementation

For our implementation, we have finished the bulk of our front end interface, including the login, project list, and project details page as well as a rough implementation of EBS. The interface is best visually seen by running the application. The technologies we are working with are mainly React and Redux which are libraries that help create web-applications using JavaScript (see below for more). After much research we have refined the architecture of our files by following this organization, which allows us to effectively organize between presentational code, logical code, and Redux vs React components.

Currently, we are using React as a user-facing, front-end library for creating user interfaces and handling any GUI events. Many of these components are "dumb," in the sense that they only store ephemeral data and handle extremely small computations (e.g. dispatching actions in response to events). Instead of storing and manipulating data themselves, these components send "actions" (*cf. Redux documentation*) to our Redux "backend" in response to

events initiated in React components, and then receive an updated state from Redux, after Redux has performed this given action. In other words, Redux is acting as a client-side, serializable database *for a single user*. We plan on implementing multi-user functionality after these single-user components are complete.

5.3 Limitations

Due to time constraints, there are a few objectives that we will not be able to complete. One of the stretch goals that we planned on was to improve upon the EBS estimation model by incorporating elements from COCOMO II and PROBE. However, improving on the GUI is taking more time and effort than we initially planned as we need to firmly validate our claims about usability and accessibility across both Manuscript and Momentum. Unfortunately, this means that we will not be able to complete that phase of our project. A part of this reason being that we are all learning new technology in implementing our project.

Another potential limitation due to the time constraint is to conduct studies on Manuscript and our GUI with small teams rather than just individuals. We will prioritize our user studies on individuals, but if we manage to find the time and a willing team to conduct the studies on at the end of the project, we will attempt to do so.

REFERENCES

[1] M. Jørgensen, "What We Do and Don't Know about Software Development Effort Estimation," *IEEE Software*, vol. 31, no. 2, pp. 37–40, Mar. 2014.

[2] M. Jørgensen, "The effect of the time unit on software development effort estimates," in *2015 9th International Conference on Software, Knowledge, Information Management and Applications (SKIMA)*, 2015, pp. 1–5.

[3] H. Barki, S. Rivard, and J. Talbot, "Toward an Assessment of Software Development Risk," *Journal of Management Information Systems*, vol. 10, no. 2, pp. 203–225, Sep. 1993.

[4] I. Benbasat and I. Vessey, "Programmer and Analyst Time/Cost Estimation," *MIS Quarterly*, vol. 4, no. 2, pp. 31–43, 1980.

[5] K. Srinivasan and D. Fisher, "Machine learning approaches to estimating software development effort," *IEEE Transactions on Software Engineering*, vol. 21, no. 2, pp. 126–137, Feb. 1995.

[6] W. Humphrey, *The Personal Software Process (PSP)* (Technical Report CMU/SEI-2000-TR-022). Pittsburgh: Software Engineering Institute, Carnegie Mellon University. 2000.

[7] J. Spolsky, "Evidence Based Scheduling," Oct. 2007

[8] "Manuscript - Project Management for Software Teams." Manuscript - Project Management for Software Teams, www.manuscript.com/.

[9] "Introduction to ProcessPAIR." ProcessPAIR, www.blogs.fe.up.pt/processpair/tutorials/intro/.

[10] "Functionality for Individuals." The Software Process Dashboard Initiative, http://www.processdash.com/functionality-personal.

[11] Kay, Matthew, et al. "When(Ish) Is My Bus? User-Centered Visualizations of Uncertainty in Everyday, Mobile Predictive Systems." UW Interactive Data Lab | Papers, ACM Human Factors in Computing Systems (CHI), 2016, idl.cs.washington.edu/papers/when-ish-is-my-bus/.

[12] "COCOMO II Model Definition Manual." USC, http://csse.usc.edu/csse/research/cocomoii/cocomo2000.0/cii_modelman2000.0.pdf

[13] B. Boehm *et al.*, "COCOMO Suite Methodology and Evolution," April 2005. http://csse.usc.edu/TECHRPTS/2005/usccse2005-509/usccse2005-509.pdf

[14] Hershfield, Hal. "Future self-continuity: How conceptions of the future self transform intertemporal choice. Annals of the New York Academy of Sciences." 2011.

[15] J. Nielsen, "How Many Test Users in a Usability Study?", *Nielsen Norman Group*, 2018. [Online]. Available: https://www.nngroup.com/articles/how-many-test-users/. [Accessed: 27- Apr- 2018].

META

A. Hours Spent on Assignment:

```
    Leon Pan: 6 hours
  Haley Ruth: 6 hours
 Anita Leung: 6 hours
Josh Pollock: 6 hours
   Austin Ha: 6 hours
```