Team B-2:

DROP TABLE ‘teams’;--

CPSC 4140, Spring 2018

Milestone 3 Report

Hannah Schilling, Alex Six, Thomas Randall, Lucas Wilson

Table of Contents

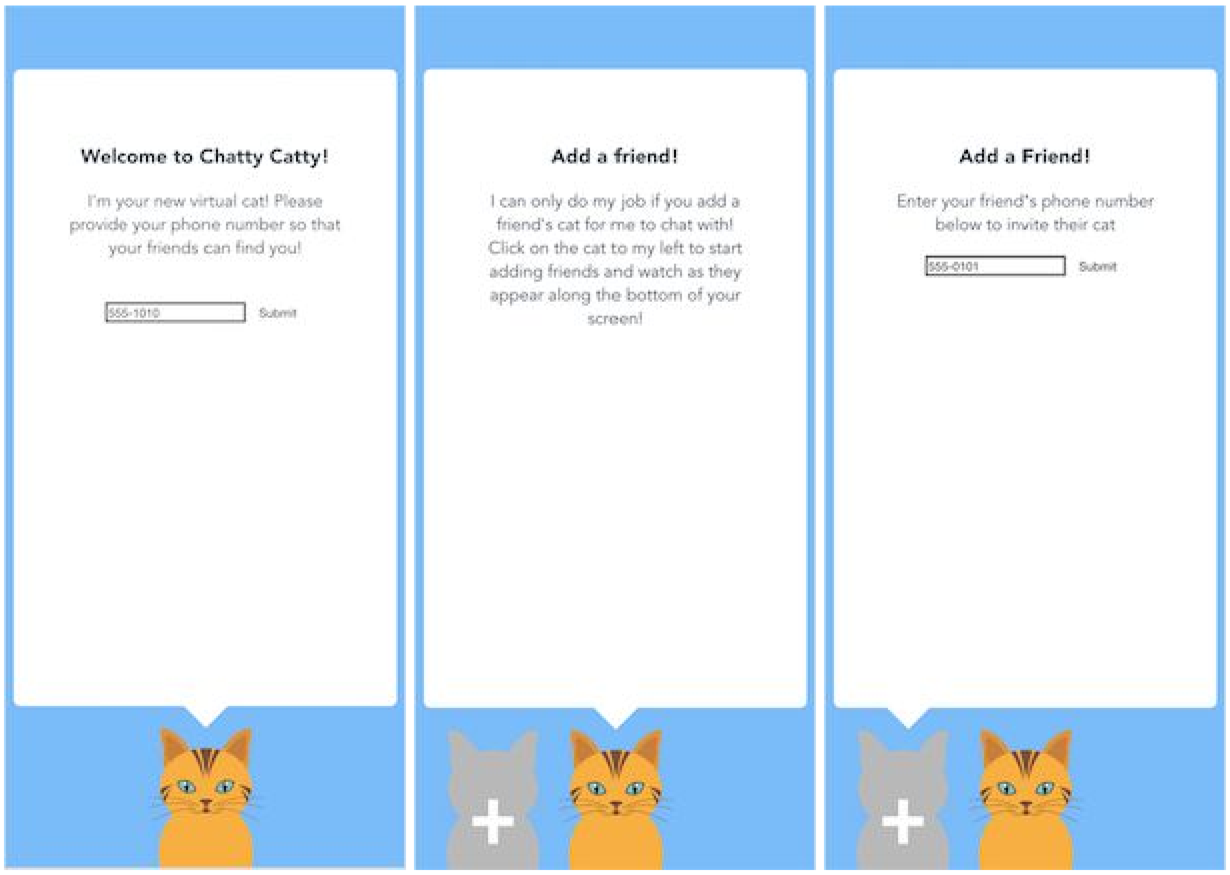
Design Choice

Studio 2 provided us with feedback to help choose which front-end design we should go with. Classmates were more receptive to the idea of having two impersonal objects communicating rather than asking for help from friends directly. There seemed to be an abundance of journaling and game apps, and the feedback we heard from Studio 2 was that there was just not as much room for us to improve those existing solutions. Most of the concerns about the Chatty Catty app stemmed from security, but the overall idea of the app was well received as innovative in the field of mental health.

From the current market of journaling applications available, there aren’t many options that allow for connectivity between users. While there are some social channels for people with mental illness, we felt that there was more room to expand here, as well as a wider audience to reach. The Chatty Catty app was also a personal favorite of the group, making us eager to bring the design to life. The process of implementing the app would provide meaningful learning opportunities and practice for our group members, but was not so far fetched as to seem unrealistic to implement within our time constraints.

We decided to make Chatty Catty a web application because web apps are nearly universally accessible and our application’s potential users span an extremely diverse population. With a web app, our users will be able to choose virtually any device that they prefer to access the prototype, as well as the final release version once it is available. The Vue.js framework was used in the back end of the project as half of our team members are experts with javascript frameworks, and the other half are familiar enough to aid the development process. Vue.js is great for prototyping and offers a high degree of portability to any technology built in the framework, meaning that we would be able to work quickly and transfer our work to new platforms if that seemed appropriate. There is also the ability to turn the web application into a “progressive web application,” which can run pseudo-natively on phones as well as on the web, which means that it can reach the widest potential audience of users on the planet right now. The code for our project was hosted on Github.com, as the site offers free student accounts and allows for version control and made collaboration easy for our group.

Description of the Prototype


Welcome to Chatty Catty!

I'm your new virtual cat! Please provide your phone number so that your friends can find you!

555-1010 Submit

Add a friend!

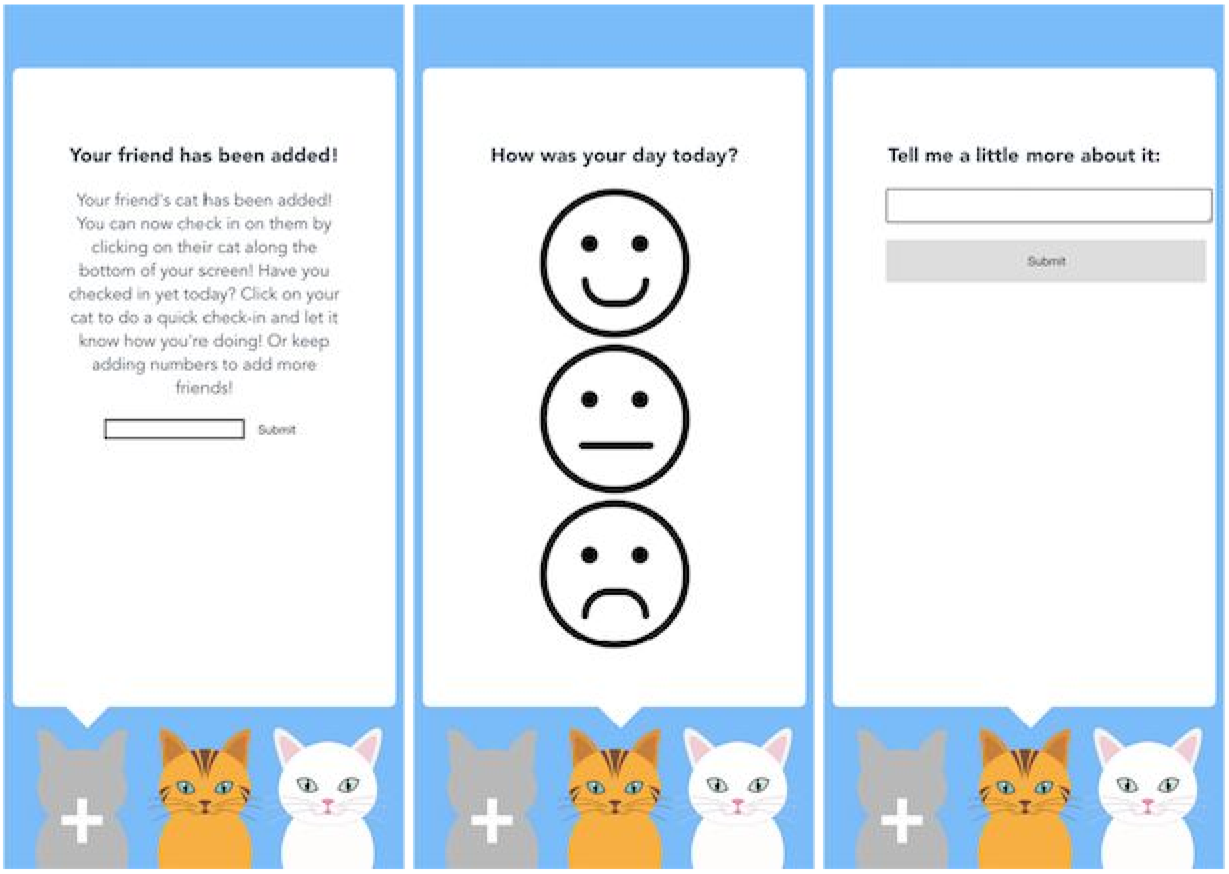
I can only do my job if you add a friend's cat for me to chat with! Click on the cat to my left to start adding friends and watch as they appear along the bottom of your screen!

Add a Friend!

Enter your friend's phone number below to invite their cat

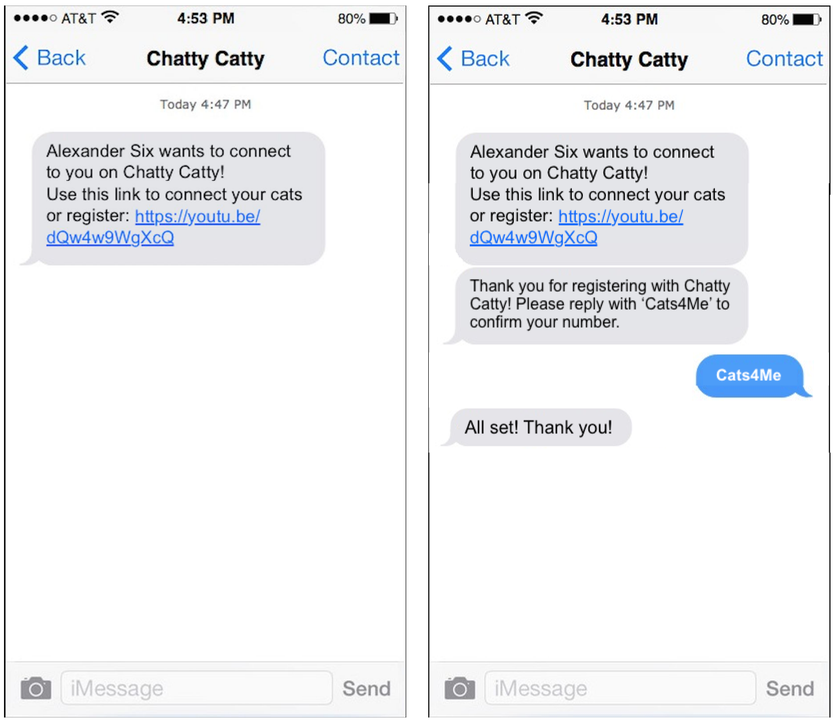
555-0101 Submit

New users are greeted with a simple introductory monologue from their cat (left), explaining how it will represent them in the app. It asks for the user's phone number so that the system can identify this person when they return (middle).
Because the app does not work in isolation, the user's cat asks them to add a friend, stepping them through the process to invite a friend to the app or to link their cats--the processes are identical (right). Friends are identified through their registered phone numbers.


Your friend has been added!

Your friend's cat has been added! You can now check in on them by clicking on their cat along the bottom of your screen! Have you checked in yet today? Click on your cat to do a quick check-in and let it know how you're doing! Or keep adding numbers to add more friends!

Feedback is given by the system when their friend is added (left). Users can continue to add friends or click either cat to utilize the app's other features.
Clicking the user's own cat will prompt the user to give an overall rating for the day (middle), as well as a short summary as to why they gave the day that rating (right).



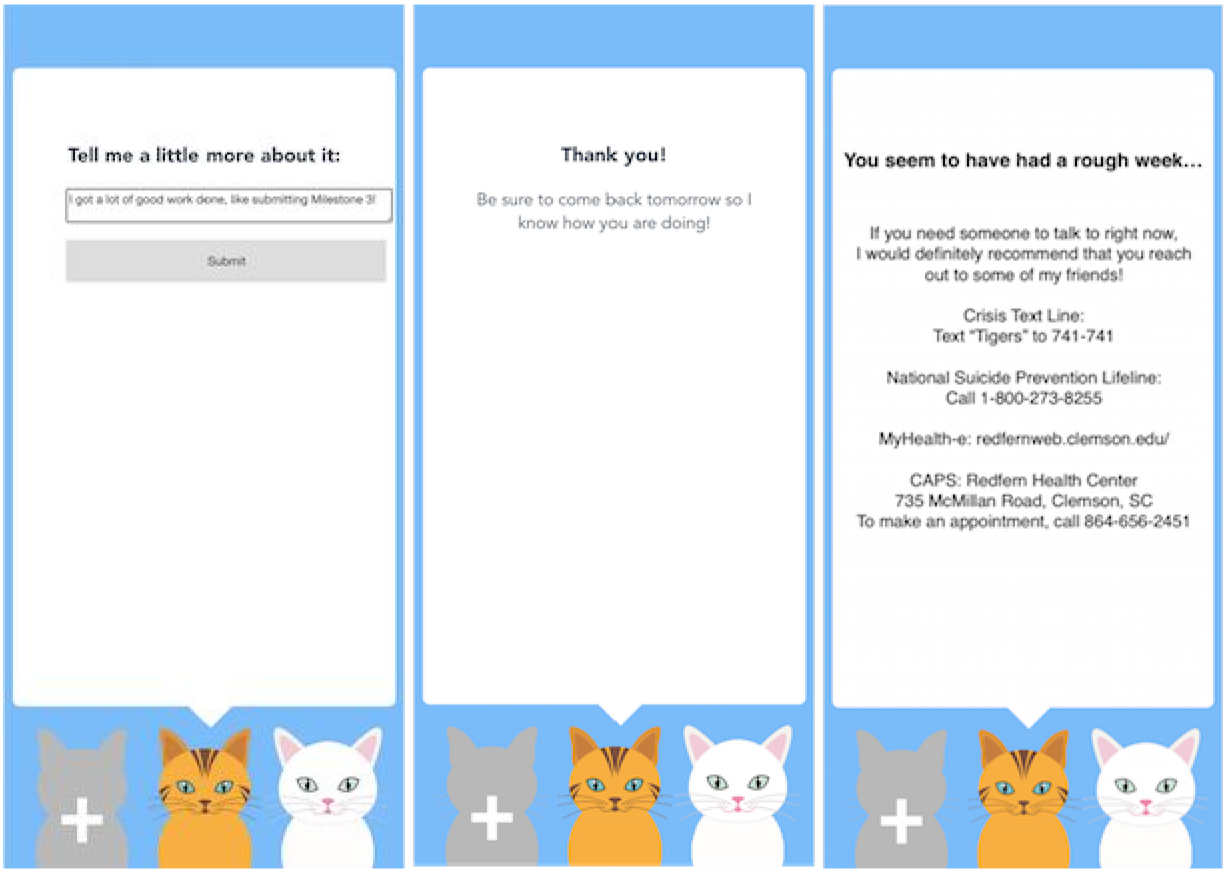
Alexander Six wants to connect to you on Chatty Catty!
Use this link to connect your cats or register: https://youtu.be/dQw4w9WgXcQ

Thank you for registering with Chatty Catty! Please reply with 'Cats4Me' to confirm your number.

Cats4Me

All set! Thank you!

When users are invited to the app to be someone's friend, they receive a text message from the system (left). After registering, they will confirm their number with the system (right).


Tell me a little more about it:

I got a lot of good work done, like submitting Milestone 3!

Thank you!

Be sure to come back tomorrow so I know how you are doing!

You seem to have had a rough week...

If you need someone to talk to right now, I would definitely recommend that you reach out to some of my friends!

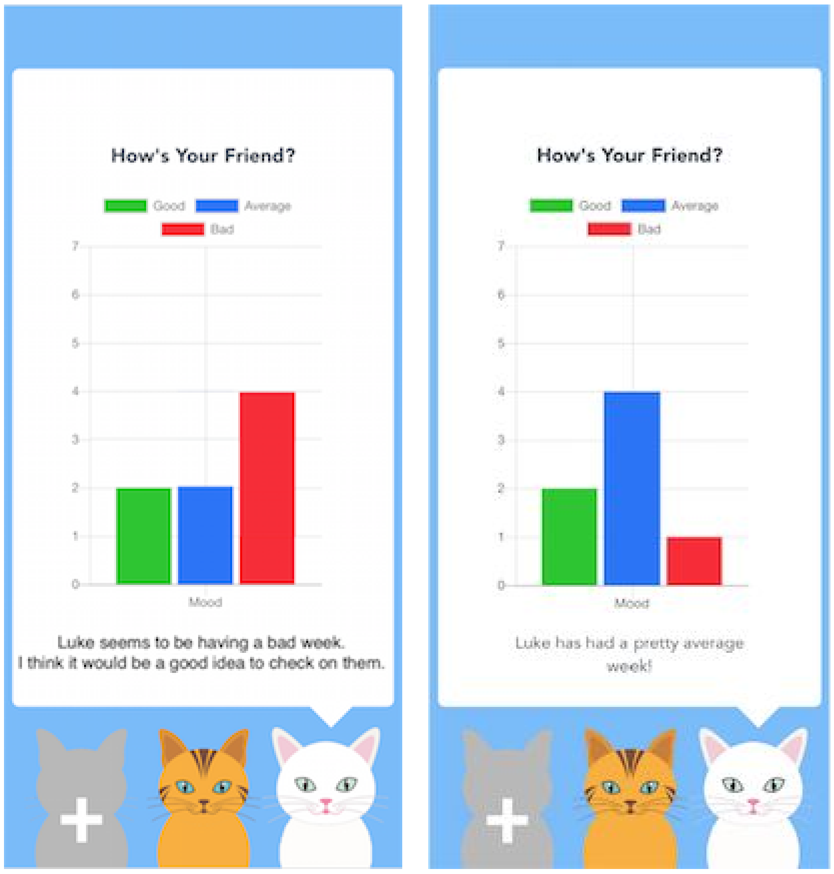
Crisis Text Line:
Text "Tigers" to 741-741

National Suicide Prevention Lifeline:
Call 1-800-273-8255

MyHealth-e: redfernweb.clemson.edu/

CAPS: Redfern Health Center
735 McMillan Road, Clemson, SC
To make an appointment, call 864-656-2451

After users submit their explanation (left), the system provides feedback (middle). When the system detects that someone is consistently reporting below-average days or otherwise may be depressed or stressed, it immediately recommends available resources to the user (right).


How's Your Friend?

Luke seems to be having a bad week.
I think it would be a good idea to check on them.

How's Your Friend?

Luke has had a pretty average week!

Finally, when the user clicks on one of their friends cats, they see a visualization of how the system believes their friend to be doing over the last few days (right). This allows friends paired with the app to better look after one another without disclosing any private information directly. If the friend has consistently reported below-average days or shown other signs of stress, the user will be shown this and advised to initiate a conversation with this friend.

Implementation Challenges During this Milestone

During this milestone, the team encountered a few snags that slowed down progress slightly, but we were able to solve the problems and finish the front-end prototype. The following were the difficulties we encountered and how we overcame them:

JavaScript Framework Quirks

Initially, when the team was choosing a production-level framework, we took some time deciding which would suit our application's needs the best. Eventually, we settled on Vue.js due to its speed compared to the other enterprise-grade frameworks and lightweight codebase. However, like all frameworks, we encountered a few quirks that had us confused for a short time.

One of the biggest problems that we ran into regarding the framework was in relation to the eventing system. Vue.js has a powerful eventing system that allows us to keep the whole app in sync across multiple views, but we were having problems getting certain events to be caught by our listeners. Eventually, through some experimentation, we figured out that the events would only be caught if they were named using all lowercase letters, because when Vue is converting the shadow DOM into the real DOM when the components are mounted, it changes all the HTML attributes to lowercase, which is where the listener is placed. Once we changed all of our events to lowercase, our eventing system worked perfectly.

Another problem we ran into was finding a way to create the graph in a lightweight, attractive way. Initially, we were intending on using the p3.js library, but it interacted very oddly with Vue, and was a very heavy codebase to include, so we needed to find a different way to create the charts. Upon further research and experimentation, we determined that Chart.js would be the package to use, since it was lightweight and integrated exceptionally well with Vue in the form of a component.

Interface Decisions

Another difficulty that we encountered was trying to figure out an intuitive way for users to interact with the app and quickly determine a general feeling of how they felt about their day. We decided that instead of leaving it up to the computer to determine user intention, we passed that duty off to the user by having them click a specific smiley face depending on how their day went. From there, the application can respond accordingly right away with a pseudo-personalized inquiry for more information from the user. This method also allows us to quickly and easily track what kind of days the user is having because again, the computer will not have to determine any implied intent. Instead, the user will provide it for the application, removing ambiguity and improving the system's ability to accurately gauge how users are doing.

Another interface decision that we had to make was in deciding how to design the smiley faces. We settled on three faces, a happy face, an “average” face (more on that later on in the feedback section), and a sad face. The idea was that the user would see the faces and choose the one that they were feeling most at the moment in order to give us a better idea of what the user was expecting.

Interface Design Implementation

The final big difficulty that we had to overcome was the implementation of the design that we planned in the beginning stages of the application development process. We designed a very clean, intuitive application, but this design proved difficult to implement in CSS.

The biggest difficulty in styling came when we were trying to manipulate the speech bubble depending on what cat was “speaking” at any given point in the application. If the speech bubble was simple just an image, the styling wouldn’t have been tremendously difficult, because we could have live-swapped in a specific image depending on which cat was speaking, but since the speech bubble was part of a component that included a wide variety of content, we needed to keep the bubble consistent.

In order to fix the problem, we ended up live-swapping CSS classes instead of images, each class moving the speech bubble relative to the specific cat that was “speaking.”

Addressing Studio 3 Feedback:

We made sure to add to our list of usability specs and evaluation plan (detailed completely below) based on feedback given to us during Studio 3. It was pointed out that we need to ask users how they interpreted visual information provided by the system, such as our smiley faces to rate how your day was (overall) and bar graph feedback on other users' days have been going. These visual shortcuts for information may not translate to common users in the intended manner, and if that is the case we absolutely need that feedback from our evaluation!

From Studio 3, we also learned from our peers that they would like more customization of "their" cat, in order to connect more with it and distinguish it from their friend's icons. Customization would let the user quickly identify their cat from a glance, and would provide incentive to continue coming back to the app if the user was able to gain more customization options through app usage.

Along with the feedback for our design, we received feedback on how useful the bar graphs for our design actually are. Most of our peers seemed to think that just showing a graph was too passive of an experience, and that it would not provide much incentive for users to continue using the application. Our peers thought it might be more helpful to to convey how the user was doing, along with how their connected friends were doing, in other ways, such as showing a line graph for over time, or implementing a system such as the Eco Leaves from Ford, or Apple's close the rings tag for their fitness motivation. We are continuing to think of different options for representing our data to users, but have not decided on any changes at this point.

Usability Specifications and Evaluation Plan

In order to evaluate the usability of our interface, we have developed the following specifications and evaluation plan:

For quantitative benchmarks, sample users will perform the demo that we presented in Studio 3, with a human observer or computer tracking the following information:

- The number of clicks and field selections necessary to complete the demo
- The time necessary to complete the demo
- The number of clicks and field selections necessary to complete the demo after already completing it at least once
- The time necessary to complete the demo after already completing it at least once
- The number of errors committed before completing a demo
- The number of clicks and field selections necessary to correct errors

The average time to complete the demo and number of clicks and field selections will allow us to determine how easily the interface is by new users. We expect that a newcomer to the system can complete the demo in less than two minutes, and anyone who has completed the demo at least once before should be capable of completing it again in less than one minute with about ten combined clicks and field selections. Performing at this level should be indicative that the system's prompts are easily recognizable and understood with little to no exertion from the user, performing worse than this may indicate that our design is less navigable or less easily understood than expected. These heuristics are critical for our system because users will not want to regularly complete tasks in our system if they are tedious, mentally taxing or confusing. Therefore it is of highest importance to ensure that the causes of subpar performance with regards to these measures are examined and improved upon.

There is also a decent likelihood that users of our app will not be in at their best mentally or emotionally at the time of use, meaning that error prevention and highly effective error recovery are necessary to not frustrate the users our app can potentially offer the most aid to. We expect that a newcomer to the system can complete the demo with less than two errors, and someone who has already completed the demo to be capable of completing it without any errors. We also expect that in the event that an error is made, the user can recognize that error and recover from it in about three combined clicks and field edits, but no more than six per error. Since we will not be intentionally causing our users distress as a course of the usability test, we understand that opportunities to measure this data in the cases where it matters most may be underrepresented. Despite this, we want to consider the error-making and recovery of our sample users as best-case scenarios for the user and system, so success for our usability trials will mean that users commit few to no errors during their time with the system and recovery from any errors rapidly. A failure to meet these standards with our sample users will almost certainly indicate poor performance for our stressed, anxious, or otherwise unstable users, and will require special attention to better prevent the observed errors or communicate to the user how to recover from the error.

We also intend to evaluate the system using qualitative benchmarks, by having each sample user from our quantitative benchmark tests fill out a short post-demo survey, either in person with an interviewer or by paper or electronic means via a form containing the questions below:

- What do each of the smiley faces (pictures included) used on the "How Was Your Day" page mean to you?
- What did the representation shown in the friend's data visualization mean to you?
  - Would you bring the data up in later conversation with them?
  - Would you proactively talk to this friend because of how you interpreted the data?
- Did any series of actions or prompts in the demo confuse you or feel odd? Please explain.
- Did you make any mistakes? If so, did the demo help you to realize that you made a mistake?
- Did the demo feel unresponsive or tedious to use at any moment? Please explain.
- What would you add to or remove from the current demo?
- Is there anything else you would like to share about your experience with the demo?

It is extremely important that all visual metaphors supplied by the system are understood by users. This is subjective, so we will directly ask our sample users how well or poorly they believe the metaphors work, as well as how they interpreted the metaphors. If there is a mismatch in how we as designers understood the metaphor and how our users perceive it, this will indicate that the visual metaphor is too ambiguous to remain as it currently exists. Equally important is how prospective users respond to feedback from the system, so we ask how they would respond to the data visualizations for their friends in the system. If the visualization is not compelling when we believe it should be, or is compelling when it is not intended to be, then the visualization is too ambiguous to remain as it currently exists.

Similarly, the wording used in prompts, questions, and feedback supplied by the system should never be confusing to users. As stated before, we expect that our sample users may be in a better state of mind than the users who need our system the most, so confusion and ambiguity must be minimized. To ensure that this is the case, we will ask users to share any design failures of this kind that they notice in the survey. Along similar lines, we want to understand why errors are committed by users, if indeed they notice their own mistakes as they use the system. A user's lack of noticing their mistakes indicates that our feedback is not strong enough, but otherwise the sample users' feedback will help us determine if our current responses are appropriate.

Finally, we ask users if they would have expected some features in the demo but could not find them, as well as if they failed to recognize the usefulness of some features present in the demo. This will help us to make sure that our design is minimalist without compromising on things that users find important, and may bring along other insights to the user's perception of the system as well. The final question ensures that users have a chance to provide any and all information that we have neglected to ask about but they find important. We would prefer to have this feedback if it exists, so it is important to offer users the chance to share it with us.