mWork

Re-imagining the Future of Mobile Work for the Masses

Medium-Fi Prototyping Assignment Write Up

Lea Coligado | Design + Development

Andrea Sy | Managment + Design

Allen Yu | Documentation + User Testing

John Yang-Sammataro | Development + Management

October 31st, 2014 | CS147 HCI + D

Problem and Solution Overview

Problem

Poverty and underemployment are two of the biggest global problems in our day and age. One of the starkest examples is what we call the “micro-task gap”: On one side, companies and individuals are willing to pay to complete millions of small tasks - such as determining the content of a picture - that still can only be performed well with human intelligence. On the other side, over 25 million people[1] in the United States and over 202 million people[2] around the globe are unemployed and over 3 billion people live on less than $2.50 a day.[3] These people could make multiples of their current income by completing micro-tasks. However, existing solutions such as Amazon Mechanical Turk and Samasource only allow workers with full computers to bridge this gap and pass over the increasing number of global smartphone users in all levels of society.[4]

Solution

mWork is “micro-tasks for the masses.” Its goal is to allow anyone to work from anywhere by connecting ***clients*** with micro-tasks that require human intelligence to mobile device carrying ***workers*** who complete this simple work. This allows clients to crowdsource important functions and allows workers to earn disposable income in their spare time.

The platform’s initial target worker demographic is underemployed workers in the developed and developing world experienced in mobile applications. Eventually, the application could expand to any potential worker anywhere. Our solution also focuses on the worker micro-task experiences which are severely lacking on mobile devices since existing client interfaces currently meet most task requester needs.

Tasks

We used three main tasks to guide the creation of our medium fidelity prototype broken into simple, moderate, and complex tasks that represent key touch points in the worker task completion cycle:

1) **Categorizing content** (*simple*) - This represents a simple task executed by the worker on their mobile interface. A worker is given the task of determining the category for a specific piece of content to answer discrete question. For instance, a user may be displayed a picture and asked to categorize it as a dog or cat.

2) **Matching client tasks to workers** (*moderate*) - This tasks represents connecting a worker to a task on their mobile interface based on their current context and desired objectives. There are a variety of micro-tasks that might be requested. A worker needs to navigate the types of tasks available to something appropriate for them to complete.

3) **Payments** (*complex*) - This involves a worker using their mobile interface to cash out earnings as a reward for a given task so that they can be compensated for their work.

Our low fidelity prototype user experience studies confirmed that the tasks above were well chosen experience milestones with which to test our application. The only changes made from our earlier user

---

[1] Source: http://data.worldbank.org/country/united-states
[2] Source: http://www.theguardian.com/business/2013/jan/22/ilo-unemployment-numbers-rise-2013
[3] Source: https://www.dosomething.org/facts/11-facts-about-global-poverty
[4] Source: http://readwrite.com/2013/05/13/mobile-is-taking-over-the-world

testing relate to focusing the tasks exclusively on the worker rather than both the worker and client, since we have decided to focus on the worker experience.

Revised Interface Design

We took the feedback we received from testers and from the teaching team to redesign our mWork mobile app's interface. The table below outlines the major usability issues we focused on.

Problem	Location	Severity Rating	Possible Fix	Task Number
Cannot login because user does not have an email account and doesn't use computers often.	Signup / Login	4	Refocus and redefine our target audience to ones with access to email and online payment	1
User is confused about what certain task icons mean	Task Page	2	Simpler and less task icons	2
User tries to go backward by clicking on the checkbox on the bottom right corner	Task Page	1	Have clearly labeled text-based buttons instead of symbols	2
User has not heard of bitcoin	Payment	2	Make sure to educate the user through an initial tutorial. Even if they do not use bitcoin they can use the bank account option.	3
User thought linking to bank account was confusing	Payment	3	Make flow easier to understand by having clear directions	3

Figure A: Table outlining usability problems.

Task Selection

We made the most significant changes in the task selection process as we realized that this was key in building a good user experience for the taskers, which ideally will lead to greater tasker loyalty. We wanted to understand what context or environment our taskers were currently in to best provide tasks that would fit their situation. For example, for individuals who are standing on a bus during their commutes might have some down time to conduct tasks but will only have one hand available to complete tasks. We then selected the types of tasks (such as image identification) that could be completed with one hand.

After testing our low-fi prototype with other individuals, we realized that the task selection process with the buttons was not intuitive and that the symbols were confusing. Please refer to Figure D. We selected better icons to indicate what kind of tasks were available and included more descriptive text as well. Please refer to Figure E in the appendix.

Other options that we considered were categorizing tasks by length of time or by monetary compensation of each task. However, in the end we wanted to optimize for the user experience and, thus, wanted to make sure that the tasks offered made the most sense to the user's current context.

Payments Flow

The main feedback that we received for the payments flow is that there were too many options in the payment page and there was no clear flow or next steps for the user to take.

We decided to lessen the number of buttons available per page to only the ones necessary and to provide "back" buttons to allow users to move forward and backward through the payment process. This was done to give the users flexibility to correct mistakes or switch cash out methods. We also provided clearer signals as to the success of an account verification or cash out process to indicate progress to the user. We

tested this flow with new users who provided positive feedback on the ease in which they were able to process payments. Please refer to Figure H and I.

Footer Menu

We received feedback that the icons on our footer menu were confusing and not intuitive. Some people thought that the “checkbox” on the low-fi prototype was a back button while others thought that it was the task selection button. Please refer to Figure J.

As we were thinking of redesigning the footer menu, we identified the key processes and workflows the tasker would conduct most often as 1) task selection, 2) task completion, and 3) payment. We decided to only include these main workflows in the footer menu and place the less relevant details (profile, get qualified, etc) in an “others” button. In addition to honing in on these three workflows, we decided to include text to make sure that the icons would not be misunderstood. Please refer to Figure K.

In summary, we wanted to create a more enjoyable user experience for the taskers and make the task selection process like a portal of discovery. The user is “shopping” for tasks so we want to make this process as seamless and easy as possible. Our belief is that making a higher quality user experience on mWork will attract a better range and level of workers.

Upcoming interface designs we want to implement include:

- Allowing users to add specific tasks to a “favorites” list
- Add a summary or list of available tasks per category with images and key words to quickly show the tasker a high level overview of tasks

Scenarios

Signup/Login

Low-Fi Prototype

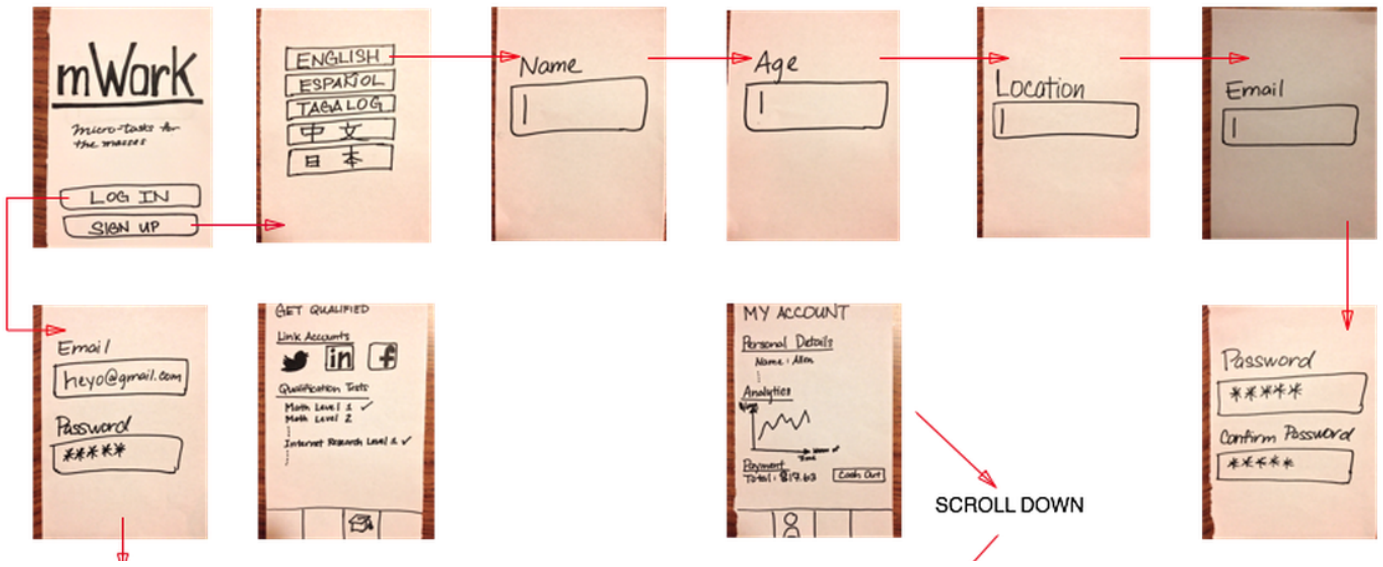


Figure B: Signup process flow in first iteration.

Medium-Fi Prototype

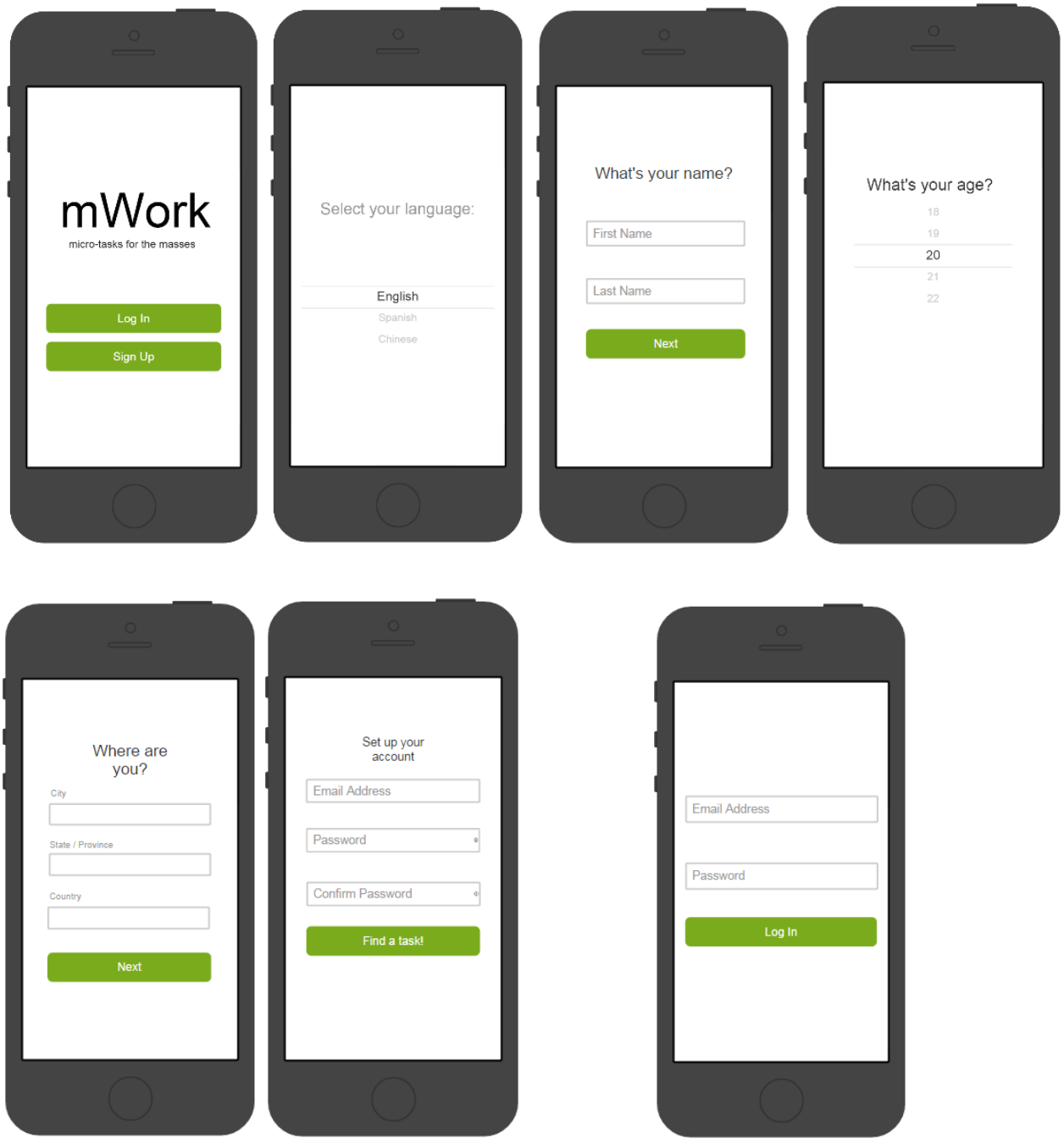


Figure C: Signup/login process flow made using Proto.io in second iteration.

Main Screen

Low-Fi Prototype

Figure D: Main task screen in first iteration.

Medium-Fi Prototype

Figure E: Main task screen with improved UI.

Completing a Task

Low-Fi Prototype

Figure F: Simple task screen in first iteration.

Medium-Fi Prototype

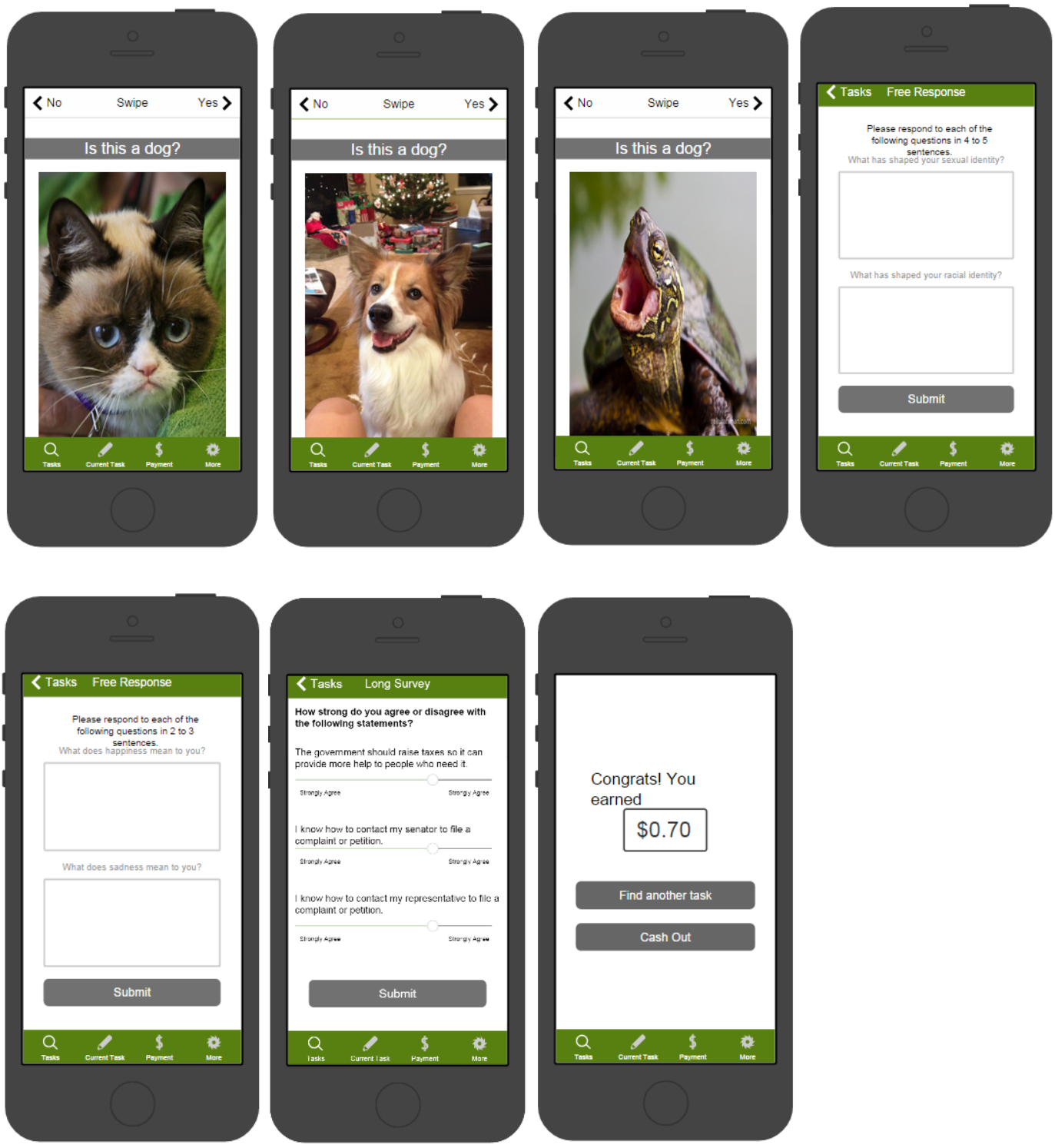


Figure G: Process flow of various tasks that users can complete.

Payment

Low-Fi Prototype

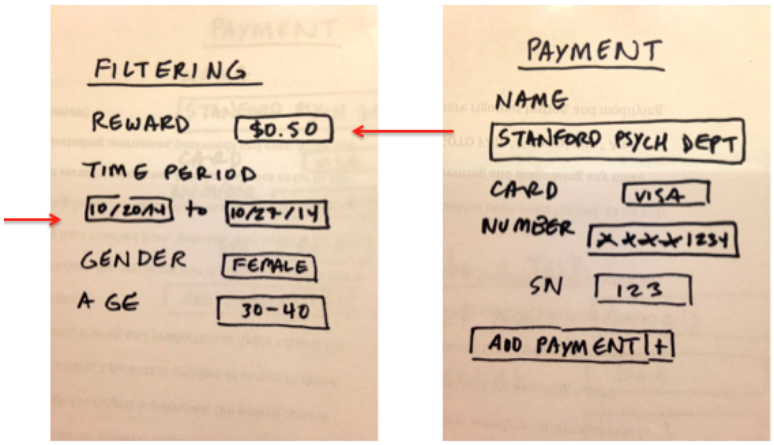
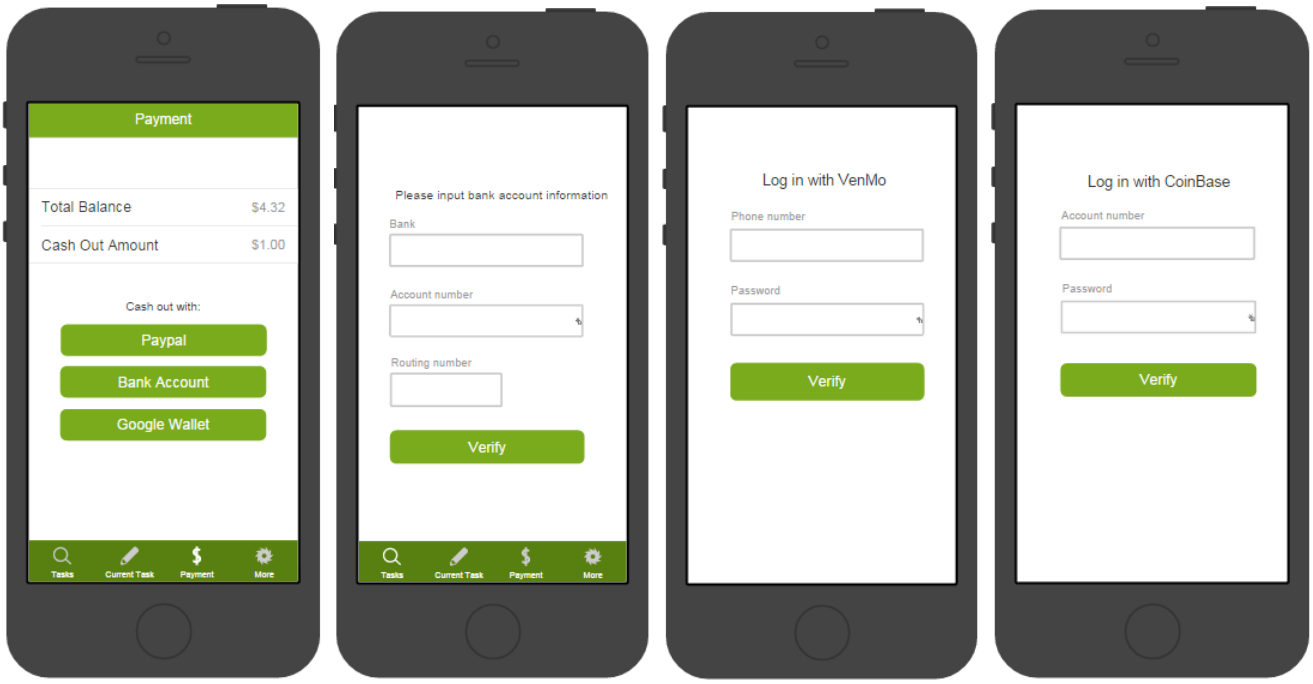


Figure H: Payment screen in first iteration.

Medium-Fi Prototype



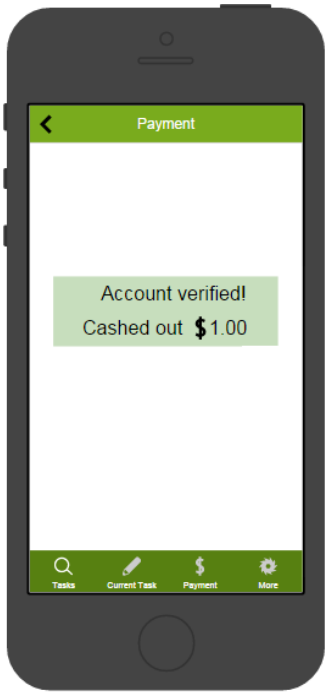


Figure I: Payment process with improved UI in second iteration.

Footer Menu

Low-Fi Prototype

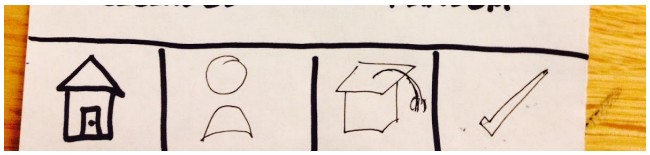


Figure J: Footer in first iteration.

Medium-Fi Prototype

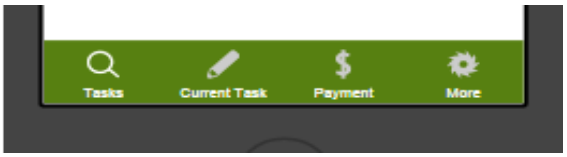


Figure I: Footer with improved UI.

Prototype Overview

Tools

For the medium-fi prototype we used Proto.io, a pre-built platform that helps users build their application's user interface and functions. It contained very useful tools that helped us design our app's screen layouts and UI Flow in a quick and visually professional manner. It helped us develop displays that were clean, consistent, and functional. Since the application's design and layout were more important in this iteration, we focused more on specific details that would optimize user experience such as the placement of buttons and the color scheme of backgrounds, fonts, and images.

However, despite its convenience of having pre-built tools, Proto.io had some major robustness issues that caused great inconvneience. The interface was clunky and tend to frequently freeze, which made the designing process choppy and frustrating at times. Its greatest drawback was perhaps its collaboration feature, where it did not update in real time and often resulted in corrupt files and lost work. We had to redo a lot of work due to saving issues and as a result wasted time that we could have spent in other areas such as better app design. Looking back it may have been more convenient to custom built the prototype with HTML, CSS, and JavaScript since that would have given us more controls in layout and functionality. Overall, the experience would have been more enjoyable and productive had the website's interface worked smoother.

Limitations and Tradeoffs

Throughout the process we had to consider the tradeoff between polishing our visuals and implementing functionality. More polished visuals typically meant more complex layout and process which would be more difficult to build. We aimed to focus on only the key defining features that would have the most impact on user experience and leave some miscellaneous functions for the next prototype. As a result, we did not implement the client side user interface so the questions were hard-coded and not real questions from clients. Since we did not have many questions, we also could not simulate a real environment where a user would do multiple tasks at once.

Wizard of Oz Techniques

Since Proto.io had limited functionality, we had to hard-code some features to provide the user with an idea of what to expect. Our app's backend was practically non-existent so we could not include conditional form support, validation for reasonable answers, and a backend results analytics system. Although these are important, they do not impact the user experience as much so we decided to save it for our next iteration. Some other features we have yet to implement include connecting to bank accounts for payment and creating qualifying tests for users to do more difficult tasks. We hard-coded the bank payment process since it would be mostly linear with some calculations. The qualifying tests are used less frequently so we thought it would be more appropriate to include it in the hi-fi prototype as well.

README File

This app is a micro-task platform for workers to perform micro-tasks for clients for pay. It is created using Proto.io and is only a prototype. To begin, create an account by clicking signup and follow the instructions. The fields actually do not take in any input since there is no backend so the inputs are irrelevant. It is meant to simulate what the actual signup process will be like. In the main screen, there will be various task categories that the user can choose to do and at the bottom there is a footer menu for ease of access of tasks, current task, payment, and settings. In this prototype, only one handed, two handed, and single session tasks have functional tasks.

Once the user clicks a category, there will be hard-coded tasks that will appear and users can follow the instructions on screen to complete them. After completion, a pop-up screen will indicate that the user has earned a pre-determined amount of pay. The payment function is all hard-coded and does not actually link to a bank account so there will be no monetary transaction. This prototype aims to test how smooth and pleasing the user experience is. More features will be implemented in the high-fi prototype.