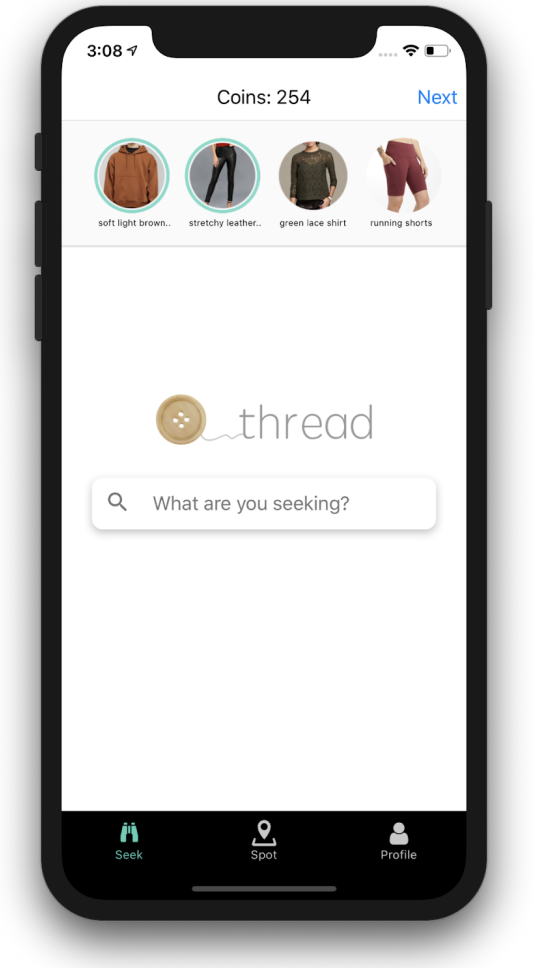
CS 147 Assignment 8
Final Report

thread

get your look, together



Team Members & Roles

Jihyeon L.
App Developer

Stephanie N.
Designer &
App Developer

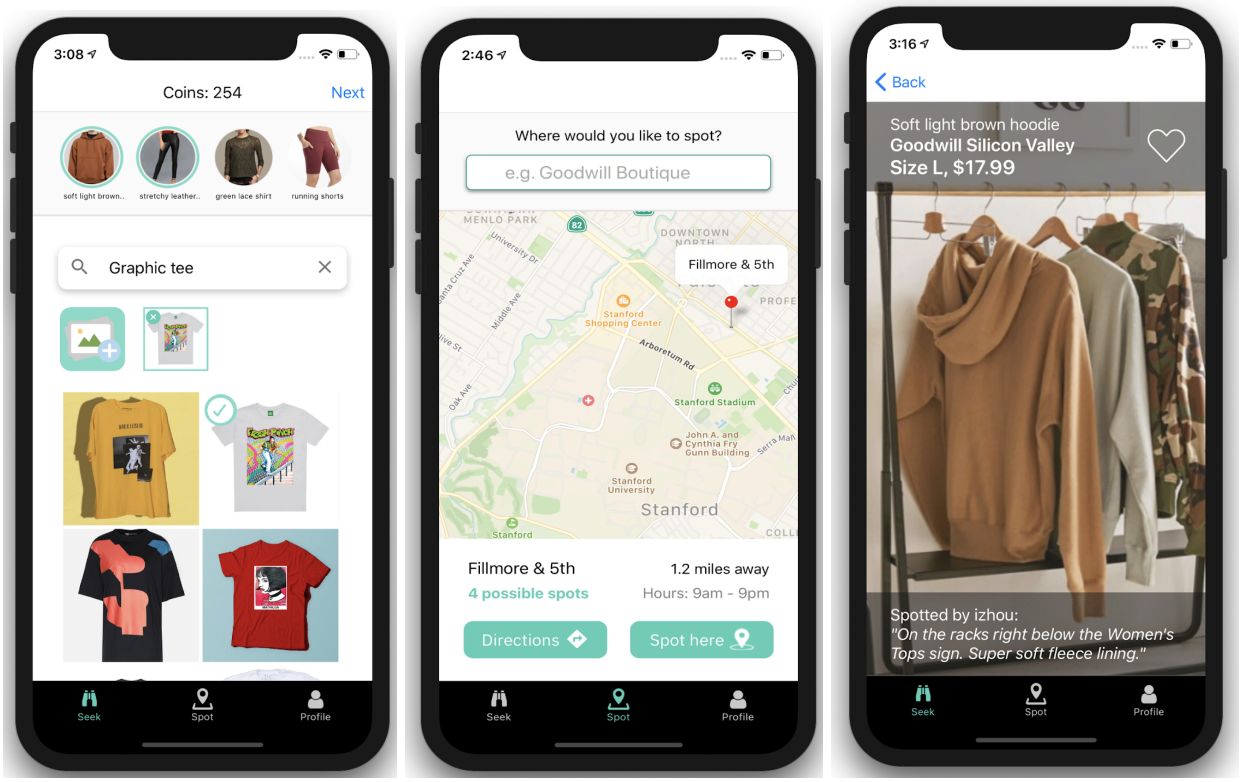
Michelle P.
Designer &
App Developer

Isabelle Z.
Web Developer &
App Developer

Problem & Solution Overview

Without the financial freedom to shop at mainstream retail stores, low-income shoppers often **spend hours at multiple thrift shops** in search of a particular clothing item that fits their style, size, and fit. Even with this high effort, they frequently **don't find what they're looking for**.

Our goal is to **connect low-income shoppers to their desired look in less time** by pooling together the browsing capacities of other thrift shoppers in the area. To do so, shoppers can submit "seeks", or requests for specific items of clothing that they hope to find nearby. They can also submit "spots" when they find clothing that matches other shoppers' requests. With Thread, shoppers can get what exactly what they want faster, because their entire community is shopping with them.



Seek desired clothing through requests

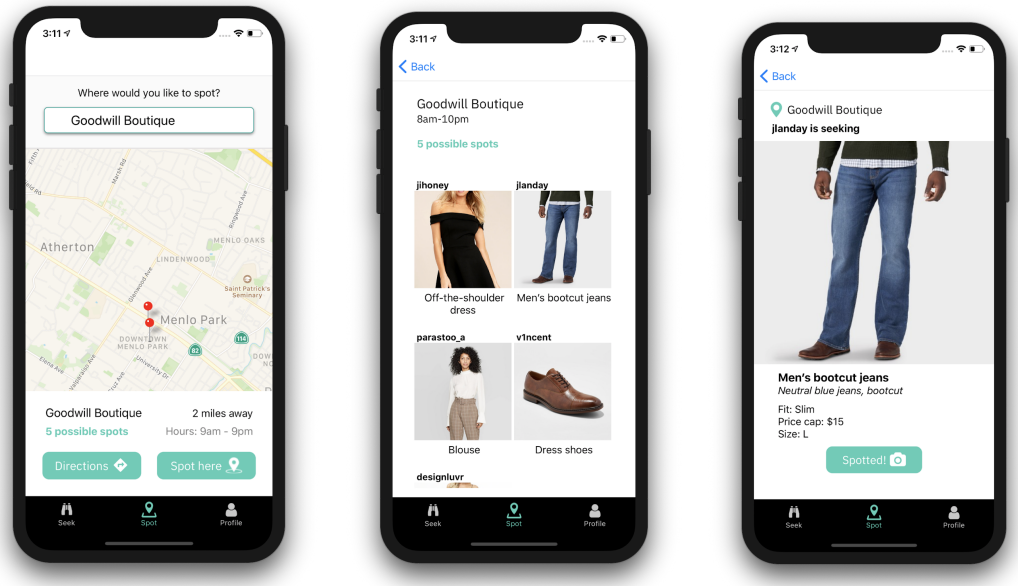
Spot others' desired items in stores and show them

Browse community finds of the clothes you want

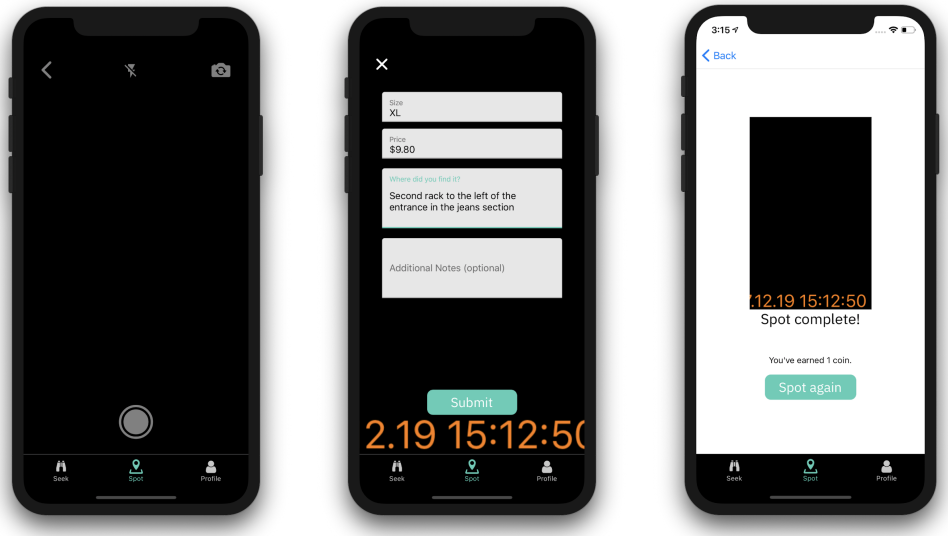
Tasks & Final Interface Scenarios

Task 1: Spot (simple). Spot someone else's desired item and forward it to them.

We chose this as our simple task because it is the easiest task to complete on our app and also has the lowest coin value (1 coin earned per spot). Based on our user research, thrift shoppers are already looking out for friends and family when shopping -- this concretizes their efforts by allowing them to submit photos of itmes they spot for others.



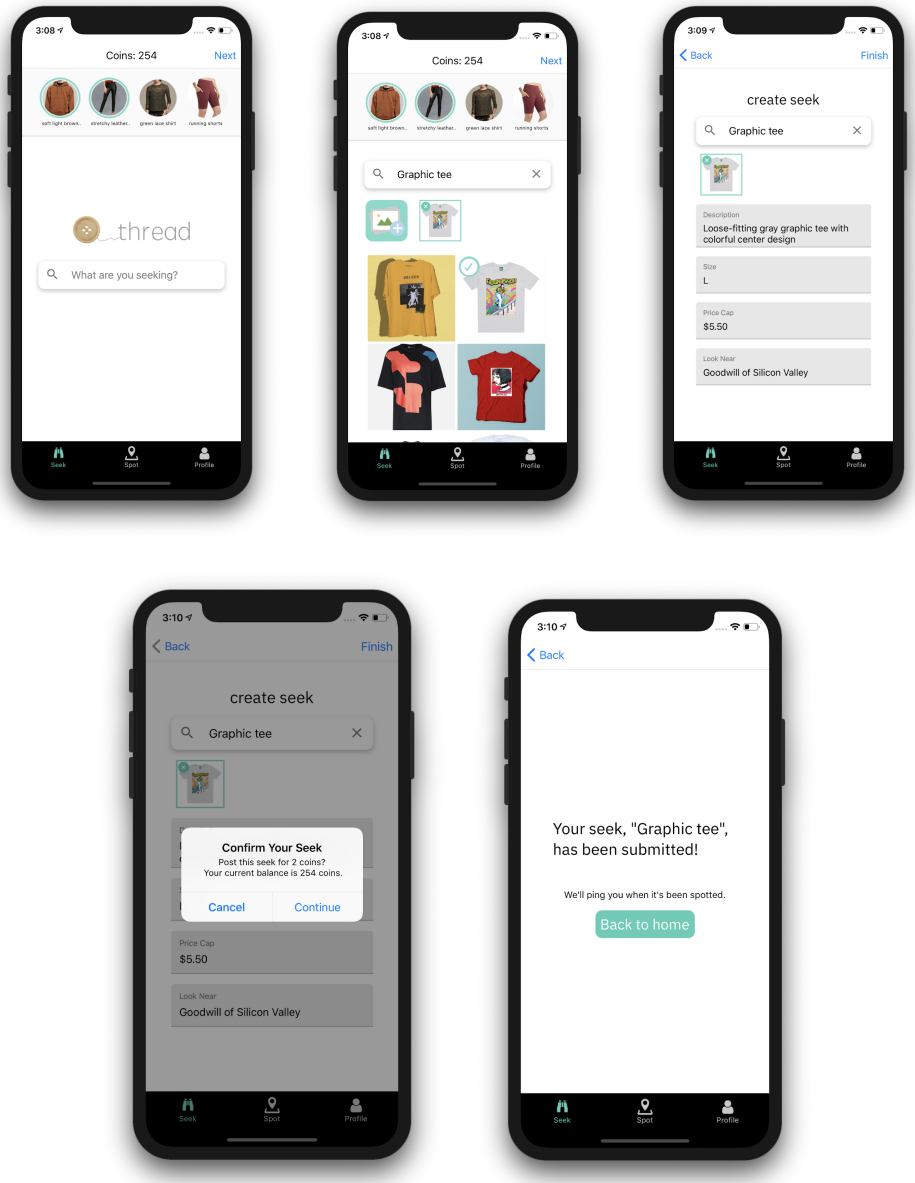
Task 1: Spot (simple)



Task 1- Photographing clothing item (camera is black because shown on emulator)

Task 2: Seek (medium). Seek your desired clothing item.

We chose this as our medium task because it introduces the “seek” request portion of the app, where users are able to specify the clothing they want by providing a reference photo and a nearby thrift store location where they usually shop. A “seek” request costs users 2 coins.

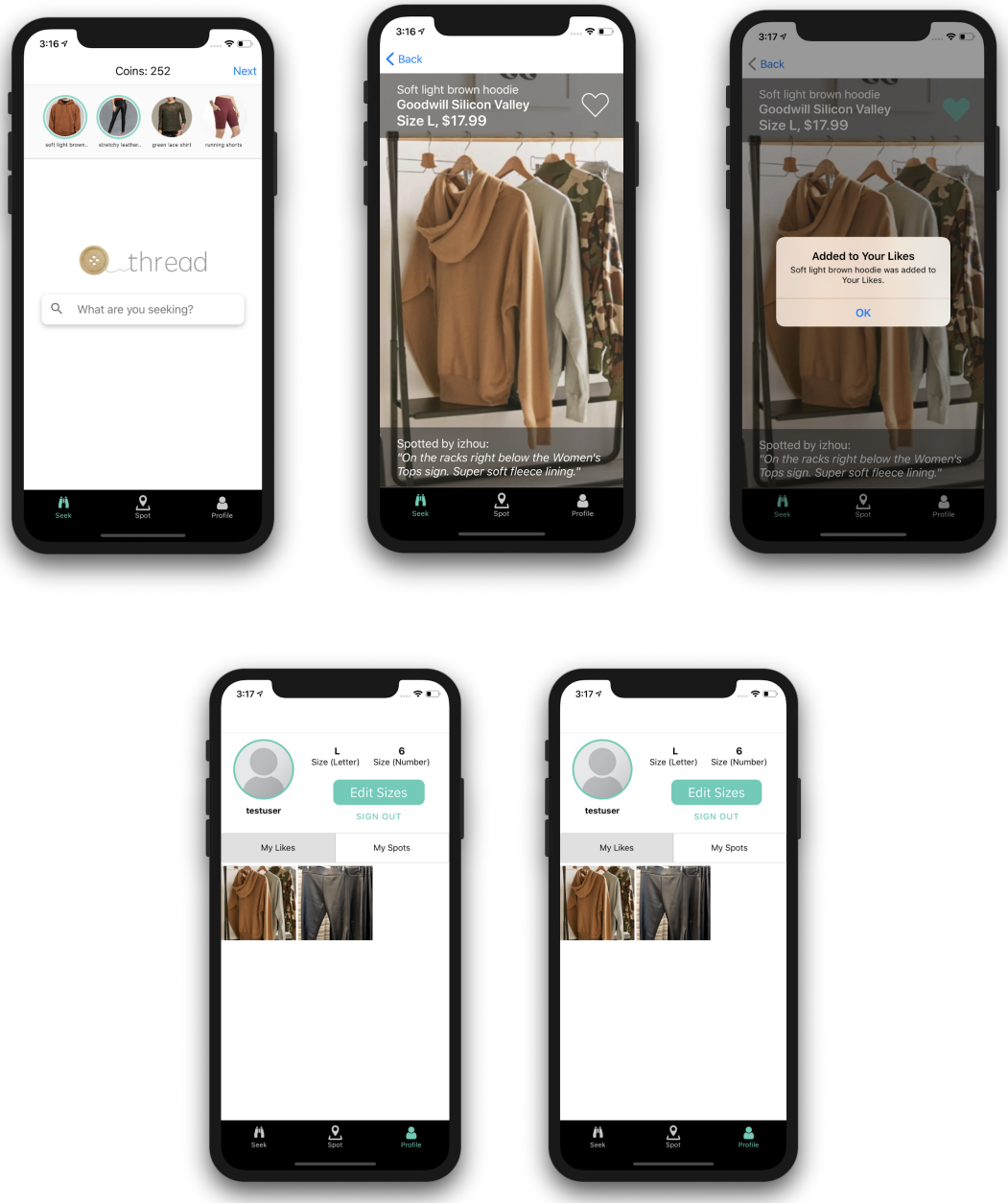


Task 2: Seek (medium)

Task 3: Browse (complex). Browse and select between the results of the item you were seeking.

We chose this as our complex task because it is the most difficult to complete - users have to look through and select the items they would like to purchase every time a new spotting of their seek is posted (and do so before the photo expires in a month) as well as remember to travel to the thrift store to buy the item. In terms of implementation, this was also the most difficult -- not only did we have to sync all

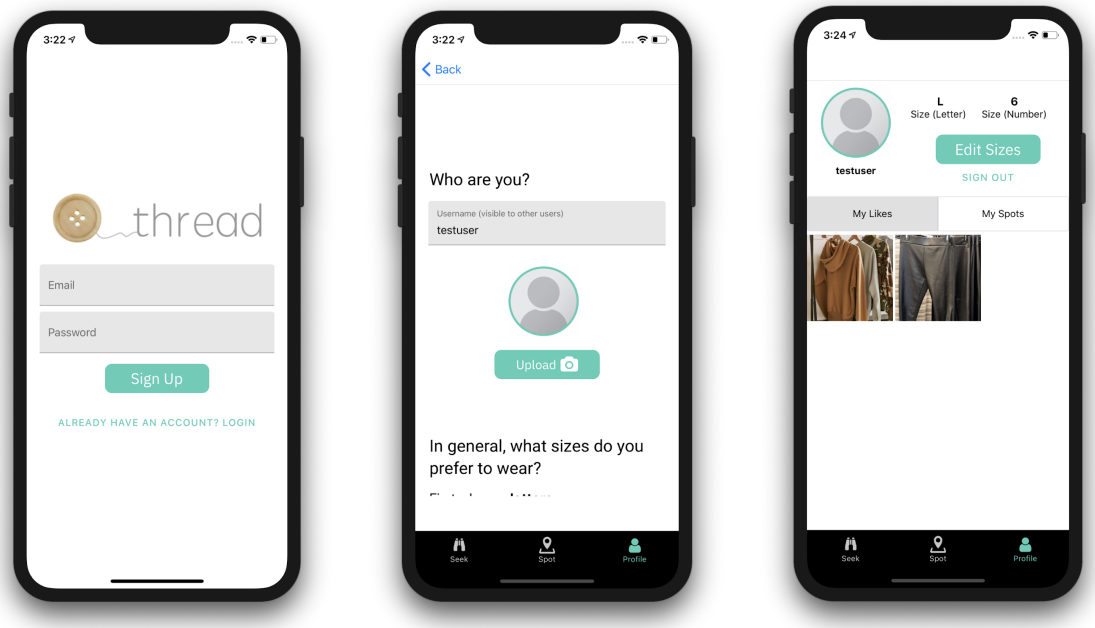
seeks and spots with our backend database across users, but we also had to track each user's "likes" and "spots" within their account. Users can "like" a spot to add it to a "My Likes" section of their profile, or delete a spot that they are unlikely to purchase.



Task 3: Browse (complex)

Additional Task: Profile (complex). Create and use your own unique profile with saved likes and spots as well as sizes information.

Because so much of the functionality of our other three tasks was tied to unique user profiles and the ability to interact across profiles, we chose to implement an additional complex task as a fully functioning user profile tied to a fully functioning database. Users can create an account, login, create seeks, create spots, view likes, and view spots while having these changes reflected in our database and across the app for other users. Any seeks and spots are also directly reflected in the internal currency for each profile (so the correct amounts are subtracted and added each time, with appropriate warnings/limitations if the amounts become too low).



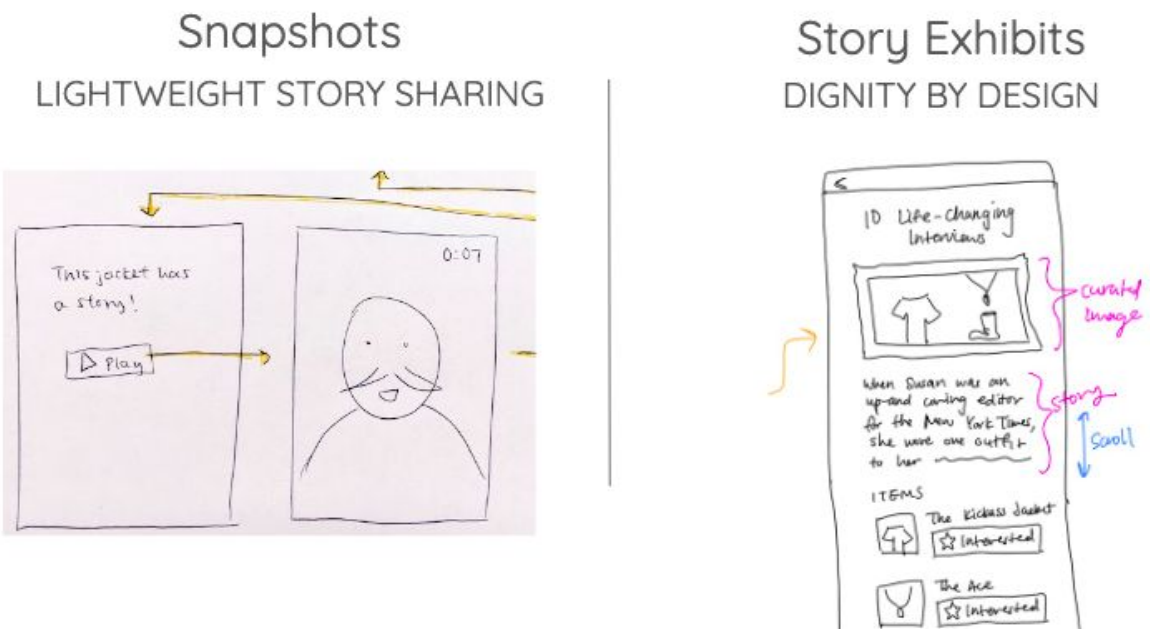
Task 4: Profile (complex)

Design Evolution

Our design has evolved greatly over the course of the prototyping process. In particular, we pivoted to a different user, problem, and solution after testing our low-fi prototype on shoppers. The resulting design evolution contains two low-fi sketches to reflect this change.

Initial Sketches

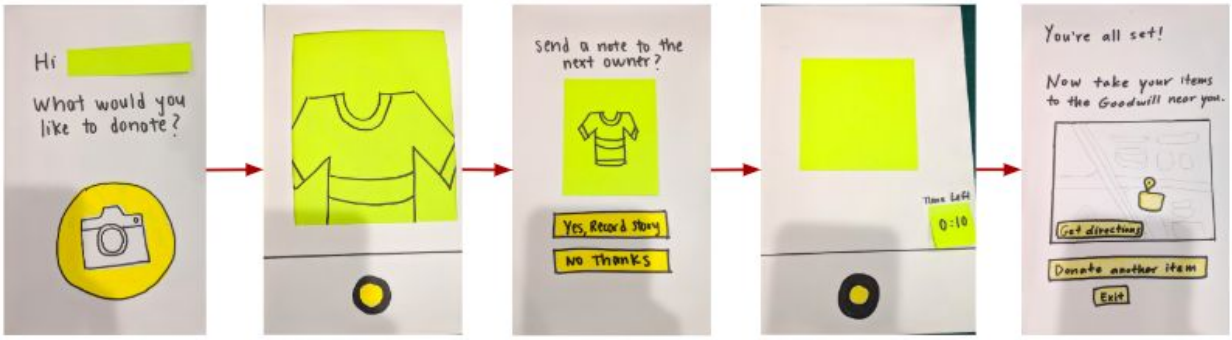
Our initial sketches focused on two modes of exploring stories behind clothing items. Story Snapshots focuses on lightweight, Snapchat-like sharing that uses brief videos of donors as the primary medium. By contrast, Story Exhibits presents a selection of curated stories that are more laborious, but intentionally, presented. We decided to proceed with Story Snapshots, because it requires far less labor from both store workers and donors and enables a more personal exchange of stories.



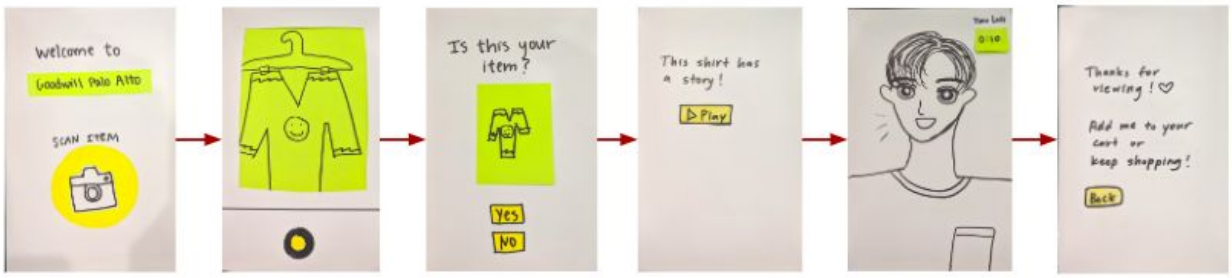
Initial sketches: Story Snapshots (left) and Story Exhibits (right)

Low-fi

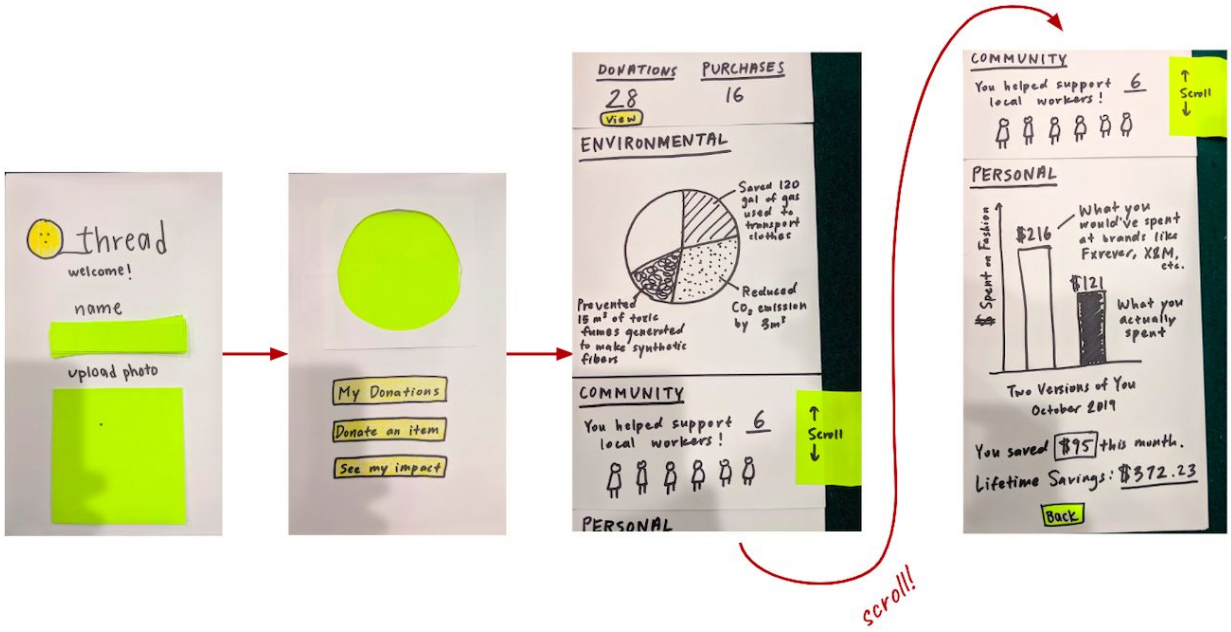
Our low-fi prototyping tested a lightweight way for users to share and view stories, using a timer-based camera for donors to record stories and an image-recognition scanner for shoppers to view stories associated with items. We added a third task, viewing impact, to allow both donors and shoppers to see measurable social, environmental, and financial impacts that their donating and shopping had.



Task 1: Pass on stories (camera interface for donors to record 10 second video)



Task 2: Indulge curiosity (camera for shoppers to scan item and view video)



Task 3: View impact (long scroll with visualizations of impact over time)

We tested these tasks on 4 users representation a range of thrifting frequency, from someone who seldom thrifts to a consignment business owner. Our low-fi users had strong reactions to this prototype. As donors, users felt a deep aversion to having to record themselves, and felt that sharing a story was an extra step in an already cumbersome donation process. As shoppers, they felt indifferent toward positive stories, but were wary of negative stories *decreasing* the value of the item to them. Lastly, they found the impact metrics we used, like number of thrift workers supported and money saved, to be confusing and indicative of privacy concerns.

Testing interactive components of our low-fi.

These results consistently violated our hypotheses that

1) Donors need a way to share stories about their items
2) Stories make items more valuable

Seeing these results, we decided to go back to the drawing board and need-find at Savers of Redwood City, a store that we suspected would represent a broader range of shoppers, both in terms of income level and reason for shopping thrift, than Goodwill of Silicon Valley.

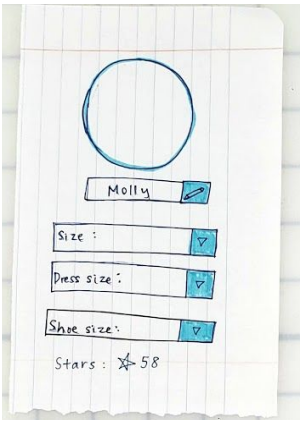
Low-fi Part 2

At Savers, we spoke to 7 shoppers. From hearing about these speakers' pain points, we decided to change not just our UI but our problem. Rather than focusing on item stories, we wanted to solve the following problem: that low-income shoppers often spend hours in thrift stores looking for the right item, often not finding what they want.

This led us to a new solution: a crowdsourced shopping platform where users can request items they want and find items for others. Our three tasks were:

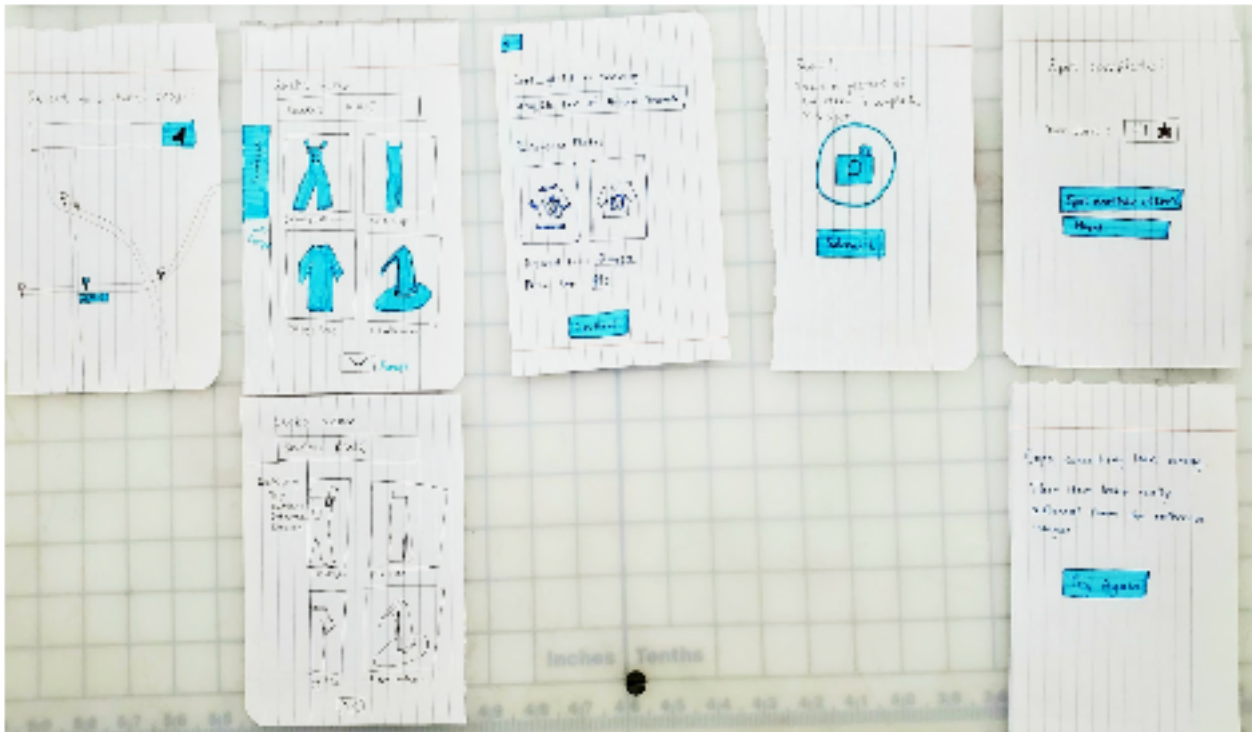
1) Build a profile
2) Request an item (Seek)
3) Find a requested item (Spot)

Our new interface relied on form completing to create a profile, photo upload and a text form for submitting seeks, and location-based spotting of existing seeks.



Task 1: Create a profile (dropdown selection of size information)

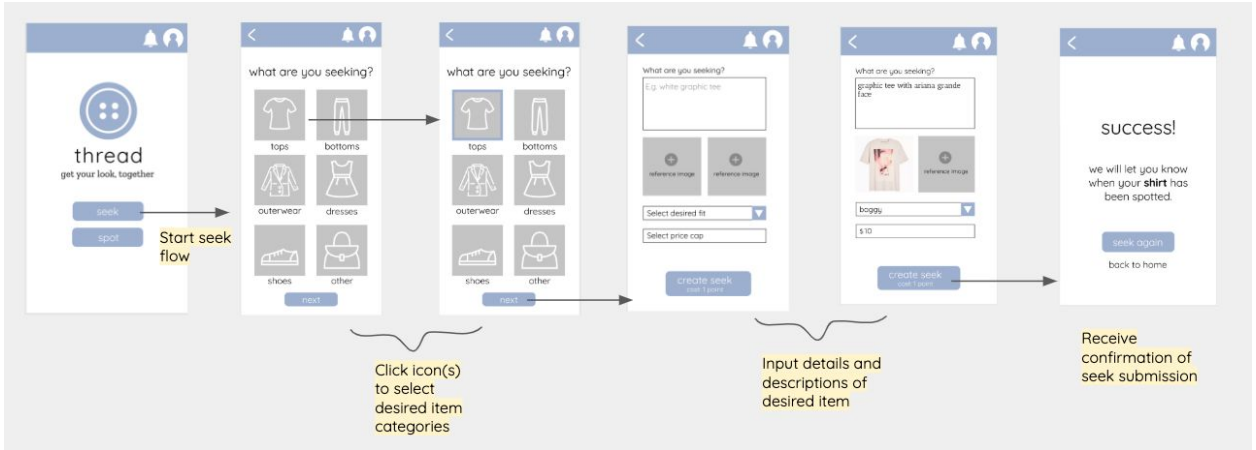
Task 2: Seek (upload reference photos, fill our text or dropdown fields for remaining information)



Task 3: Spot (map interface to select store, gallery view of existing seeks, camera to capture item you've spotted, AI for image recognition matching)

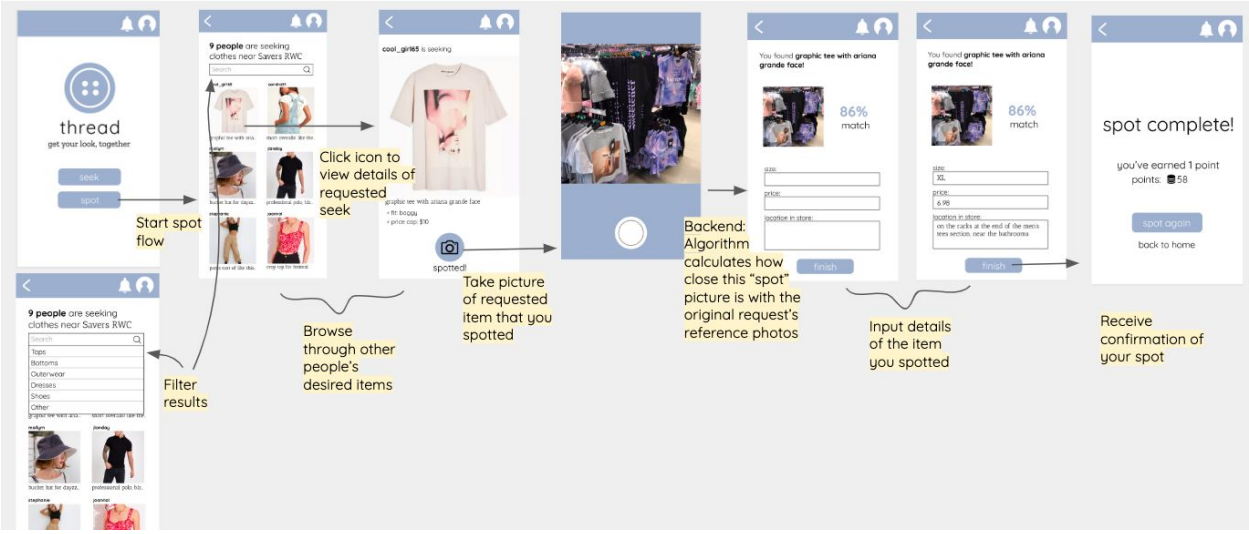
Med-fi

From our low-fi we made several changes to UI details and changed our first task. Consistently, users were confused about how they would view results of a requested seek. From this feedback, we changed our first task ("Build a profile") to "View your results," or "Browse," including UI for a notification system.



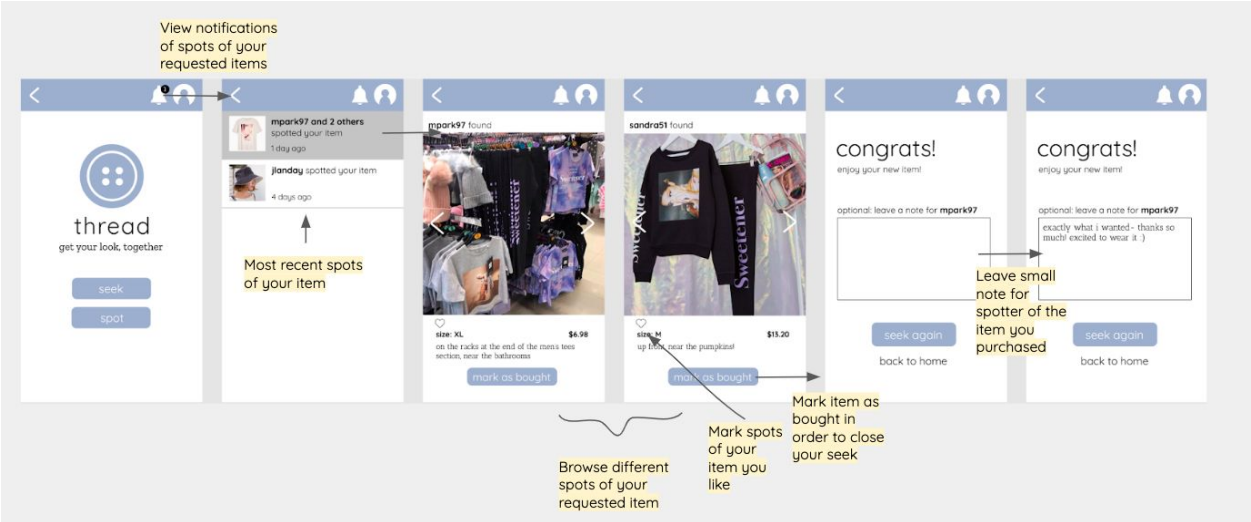
Task 1: Seek

The seek flow was generally easy to understand; many users made several seeks of items they actually wanted, so we did not make significant changes to this UI.



Task 2: Spot

In the Spot flow, users were confused where seeks were coming from. Rather than "Seeks Near You," we changed our interface to say "9 people are seeking from x" to more directly link to the people requesting items. We also more clearly indicated our in-app currency by using a coin icon rather than the start icon we were using previously.



Task 3: Browse

Lastly, in the Browse flow, we added a notification system that would allow users to see who had spotted an item, and which item it was for. This interface allows users to view multiple photos from a spotter and leave feedback in the form of a note to thank the spotter or otherwise communicate to them.

Major Usability Problems Addressed

Transitioning from med-fi to hi-fi, we faced a number of design decisions, like what task to default to as a homepage (seek or spot), how prominently to display usernames in task flows, and how users can interact on the platform. In weighing these changes, we made a critical decision between two values that guided the rest of our design process: 1) **sense of community** and 2) **saving time**. In thinking back to our target users' needs, we agreed that low-income shoppers' sharpest pain point is in the amount of time they spend looking for what they want at thrift stores. Therefore, we decided to focus on saving our users time. This focus on saving time is reflected in many of the changes we made in this section, and can be seen particularly in our design of our homepage.

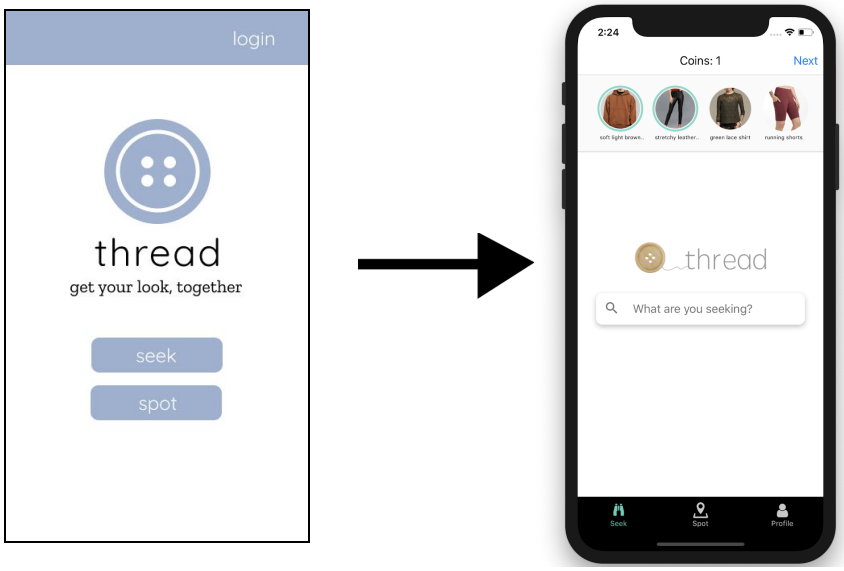
In this section, we number and detail the changes we made to our hi-fi prototype.

Level 3-4 Heuristic Violations

We fixed all the level 3-4 heuristic violations we received. We detail them below.

1. H1: Visibility of System Status / Severity: 3
There's no way for users to see previous seeks they've created.

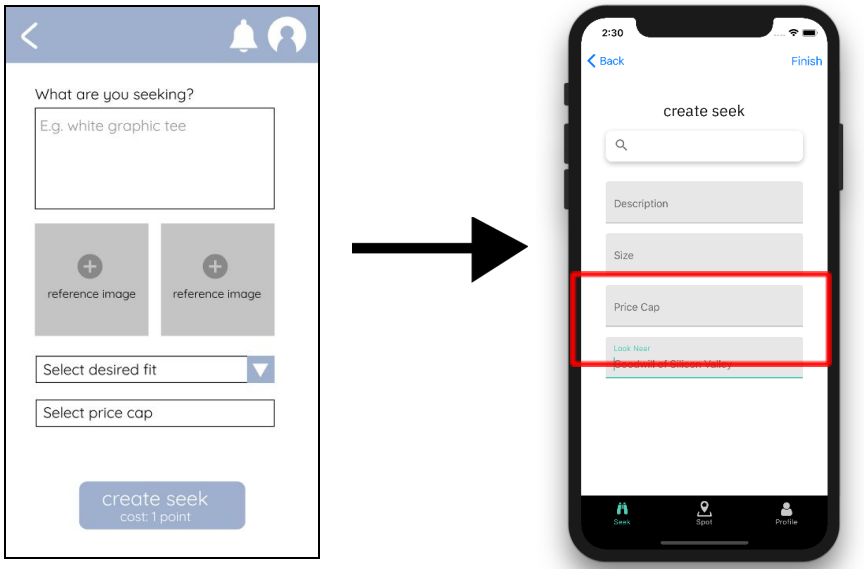
Change: We added bubbles at the top of the "Seek" start view to show users the previous seeks they created, making sure to clearly mark each bubble with the reference image they submitted and the title of the item they requested.



2. H3: User Control and Freedom / Severity 3:

The app doesn't allow the user to specify what location and radius they are currently submitting the seek request for.

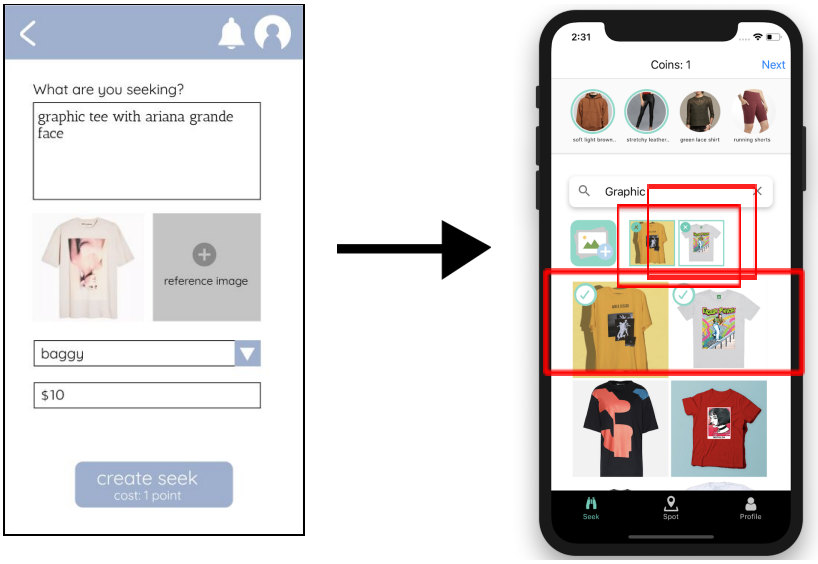
Change: We added an input box for the user to specify what location they wanted to shop near when submitting their seek request. Because adding an additional "radius" parameter might be confusing to users, we decided to tune a radius on our own backend according to what spots users actually "liked" (wizard-of-oz-ed in our hi-fi).



3. H3. User Control and Freedom / Severity: 3

After adding a reference image on the Seek page, there is no way to remove or change the image.

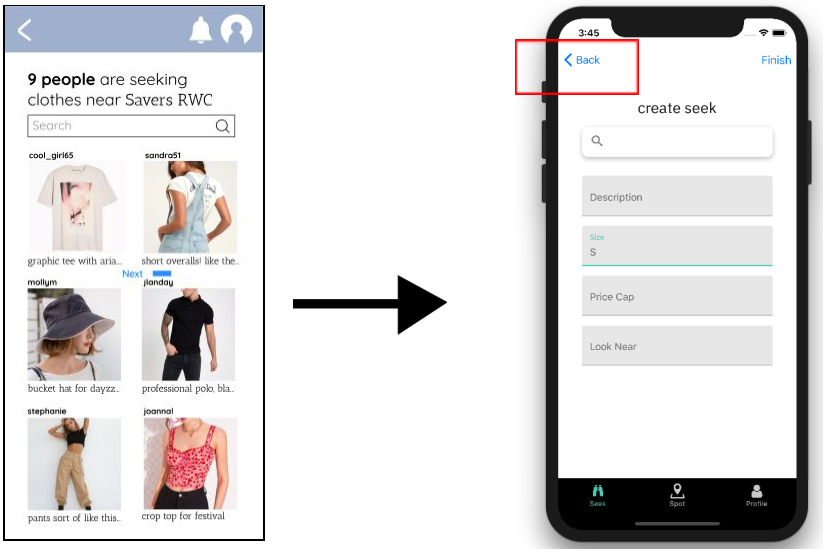
Change: Tap a selected reference image to remove it. We indicated this remove functionality by adding an “x” button to the panel of selected images right below the search bar.



4. H3. User Control and Freedom / Severity: 3

There is no ability to go back a screen and returning to the app home screen is confusing. In the seek/spot process, the back button takes you home instead of back one screen.

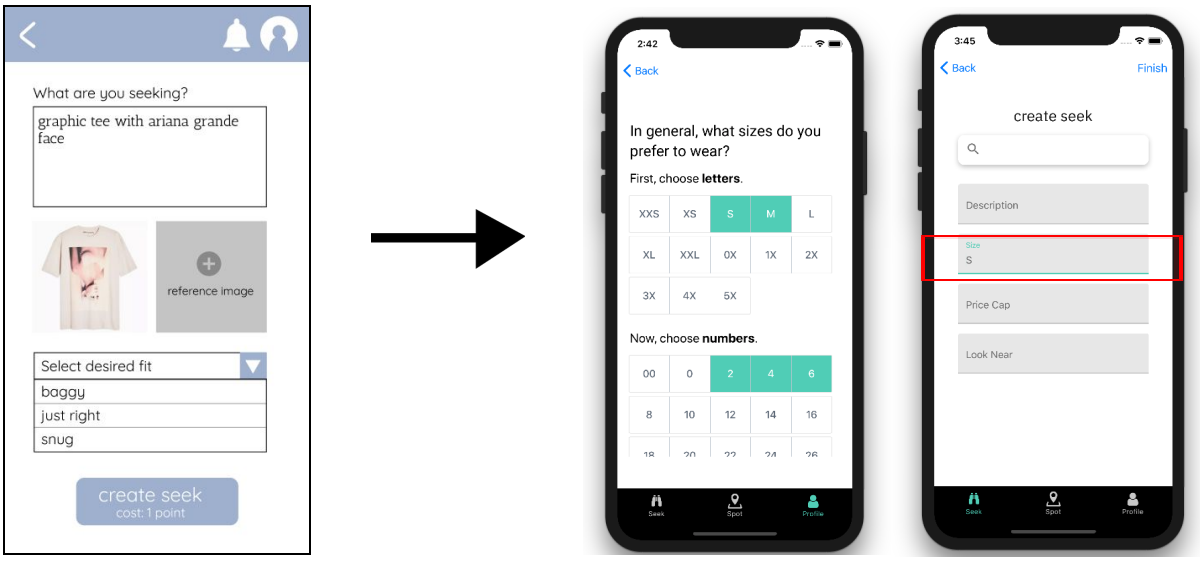
Change: We added the ability to go back on all screens.



5. H2. Match between System and Real World / Severity: 4

The sizing options of “Baggy,” “Just Right,” or “Snug,” don’t fit standard sizing conventions, making it difficult for the spotter since they have to input a size. This also does not account for children, adult, male, or female cuts which are sized differently. Furthermore, user’s size, height and inseam may also be important to the seeker, and those parameters aren’t currently reflected in that item seeking flow.

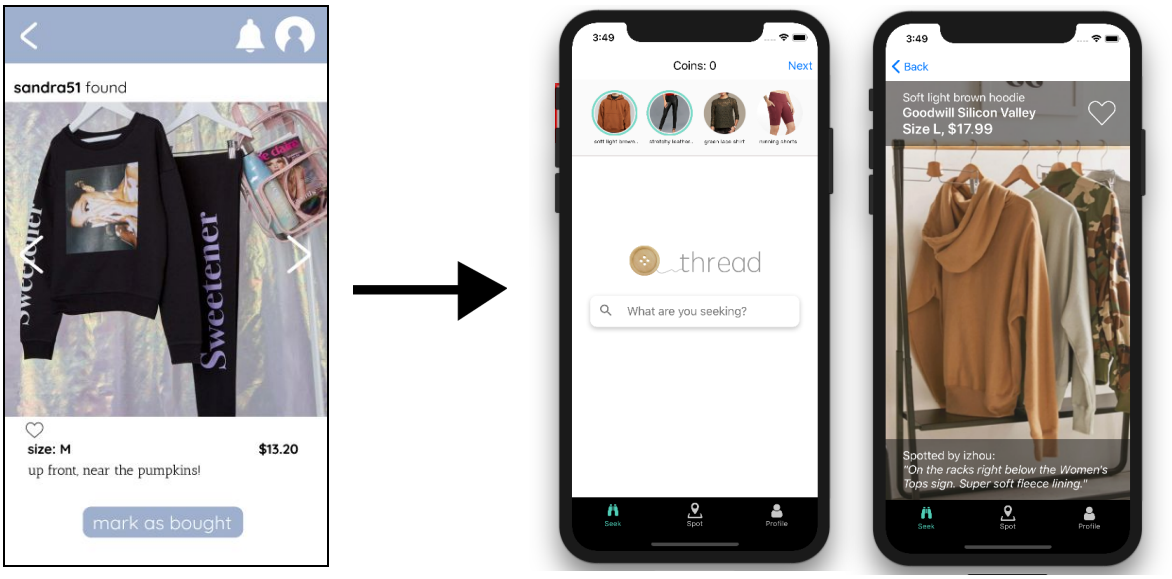
Change: We changed the sizing options to letter sizes (XXS-5X) and number sizes (00-32). We default to the sizes specified in the user’s profile for every seek, but allow this size to be changed by the user for a particular seek if they wish. We also removed the fit options of “Baggy”, “Just Right”, or “Snug” as an input parameter altogether, since this is inherently captured in the user’s manipulation of size for a given seek, as well as their description and reference image.



6. H2. Match between System and Real World / Severity: 4

If seekers view an item that someone spotted for them, they're provided with an in-store location of an item. However, the store that the item is located in is not listed. Additionally, users never select a store to seek during the seek process.

Change: We added the store that the item is located in to the browsing task flow, where users can tap on their seek bubble to view the spots of that seek along with the store it was located in. We also added an input parameter for the user to specify a store when submitting a seek (discussed in 2).

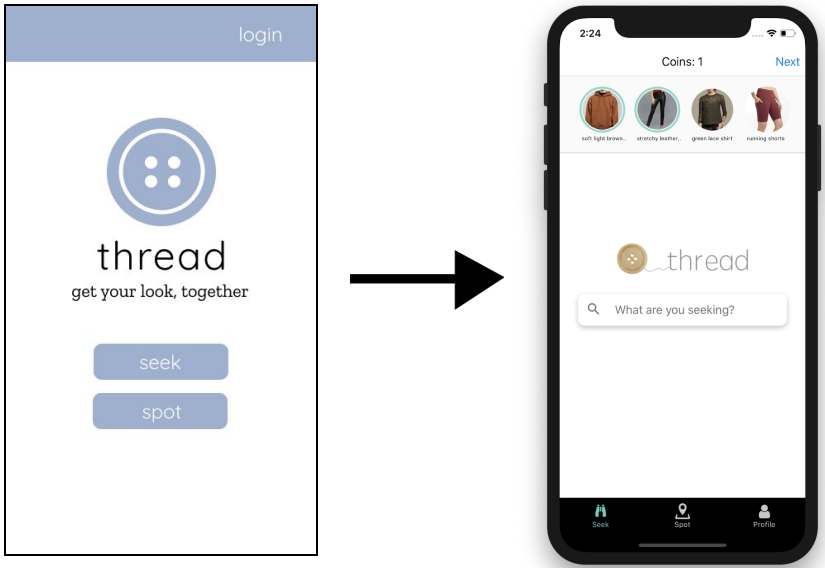


Additional Major Changes

7. Color palette, look & feel of app

Studio Feedback: UI is “Too Minimal” and the flat blue-gray color lacks personality

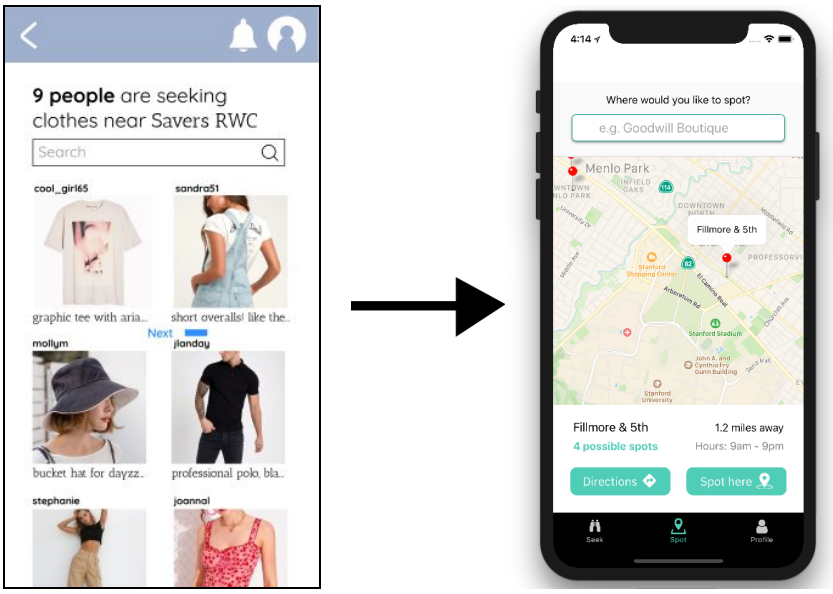
Change: We adopted a sea green color for highlighting in conjunction with a black and beige color palette. Keeping our primary goal to save shoppers time in mind, we wanted to maintain the minimal nature of our app. Thus, we used color and our logo to add personality to the app, while maintaining the clean whitespace around the seek search bar.



8. Added map view to browse possible spots nearby

In our medium-fi, we assumed that user was already at a thrift store when starting the spot flow.

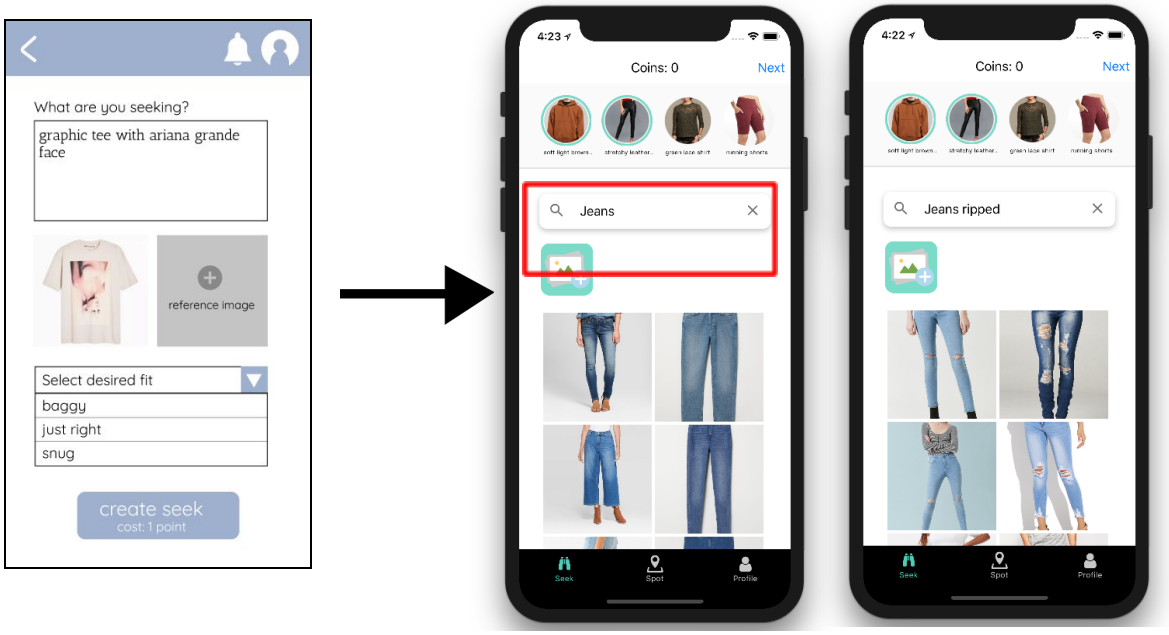
Change: We realized users could browse possible spots nearby before heading to a thrift store, and added a map view to allow them to browse thrift stores nearby with possible spots.



9. Added Google image search to help find reference photos

We previously only gave users the ability to upload their own reference photos.

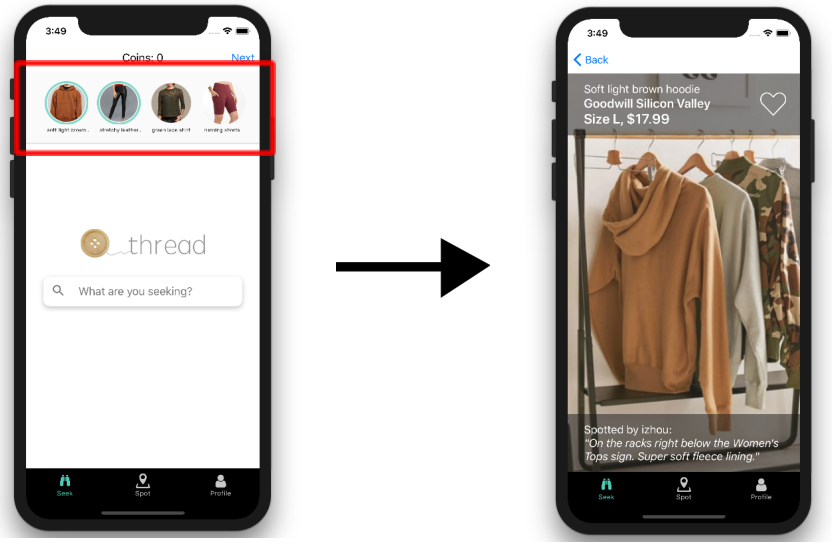
Change: In order to help users find reference images for what they want, we dynamically surfaced images from Google search that matched what the user was searching for.



10. Story bubbles to browse spots of your currently active seeks

We previously designed the “browse spots of your seeks” flow to start from a notification.

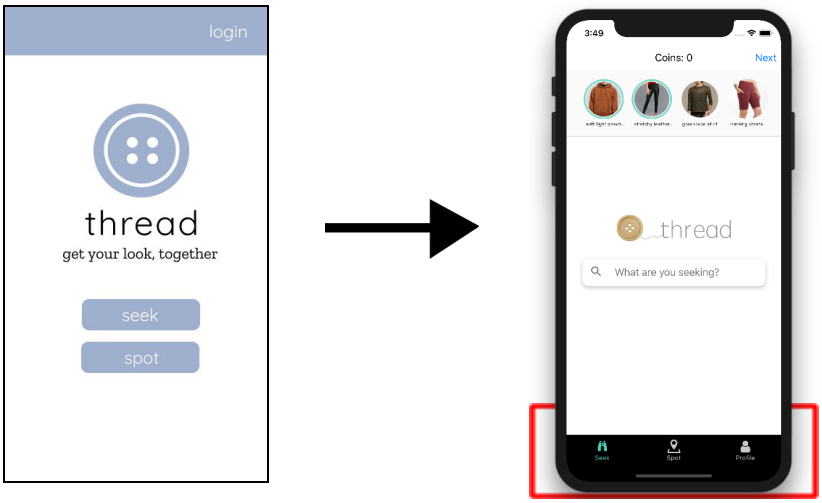
Change: We implemented an Instagram-style story bubble to start the “browse” task flow. This allowed us to leverage an existing design pattern to flip through photos of the same category (in our case, a given seek request) as well as enable users to go back to older seeks after a notification disappeared.



11. Switched to tab bar for navigation flow

We previously allowed users to enter either the seek or spot flow from the home page through a button.

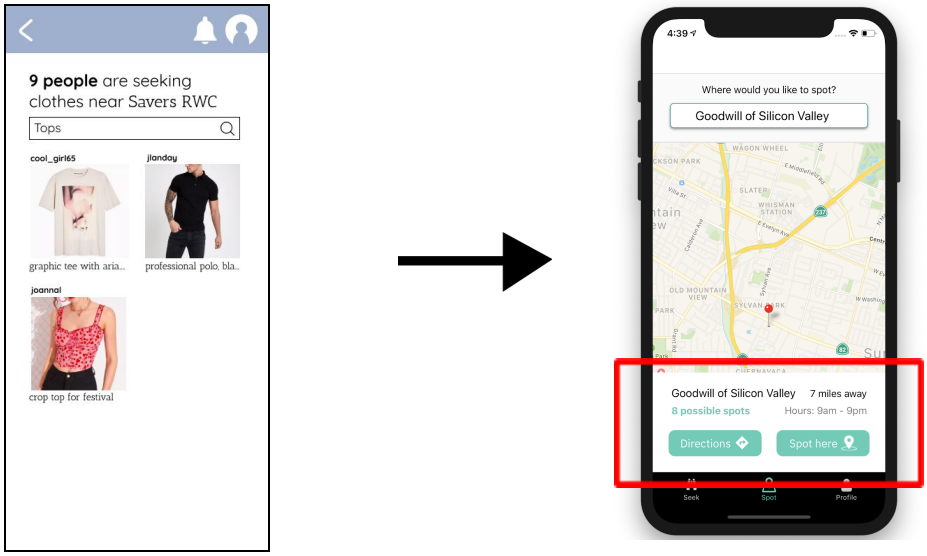
Change: We added a tab bar for easier navigation to a different task.



12. Shop Preview Panel

We previously did not give users a snapshot of a given store (hours open, number of spots available) before launching them directly into a view with a gallery of possible spots.

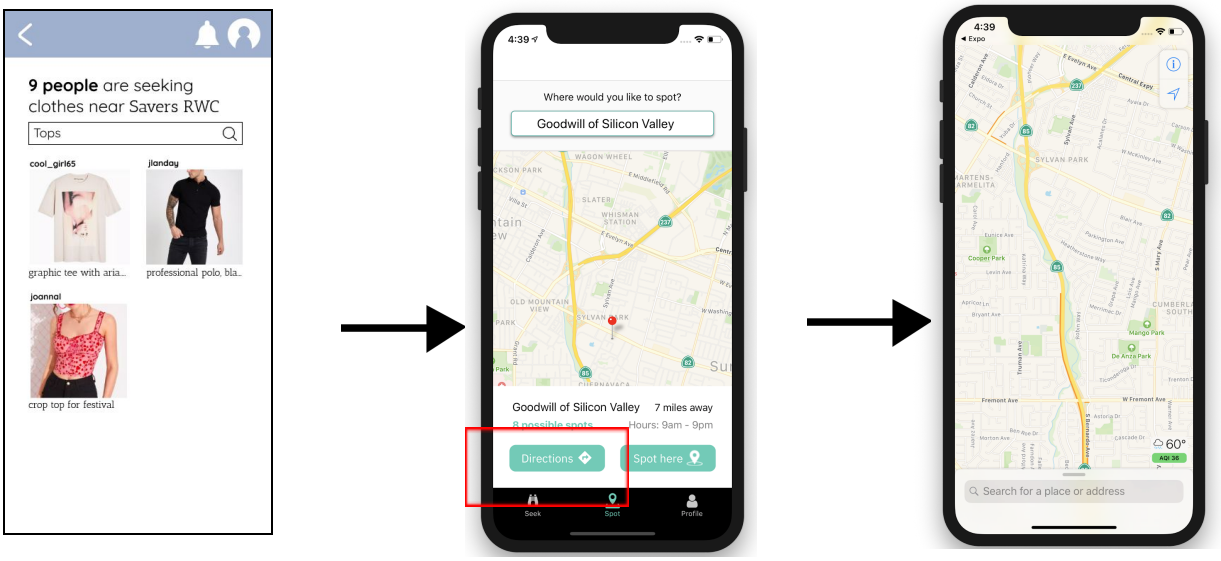
Change: We added a preview panel in our spot map screen to give users a snapshot of a given store before giving them more details.



13. Integration with Native Maps for Directions

We previously did not link users with Google Maps (or Maps on iPhone) for directions to a given thrift store.

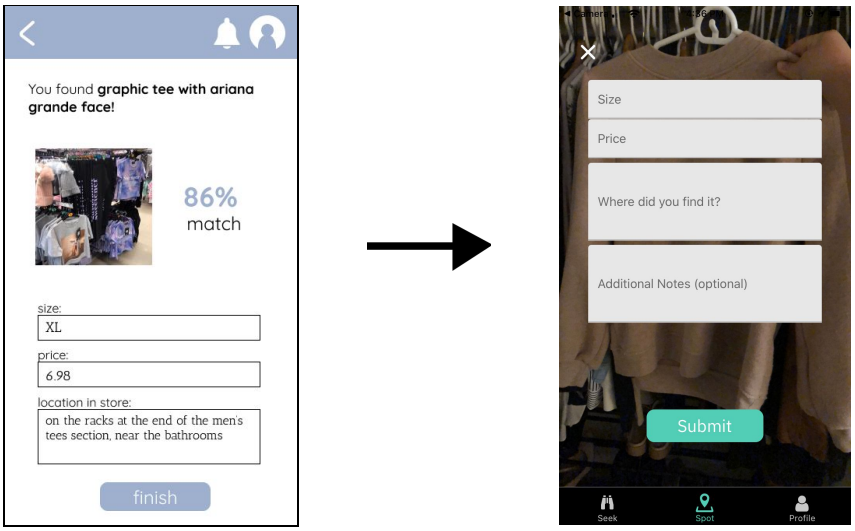
Change: We added integration with the native Maps app from our spot map screen to open Maps when the user clicked “Directions”.



14. Spot Inputting Detail Design

We previously had the details of a spot be inputted in a regular-looking form with a white background.

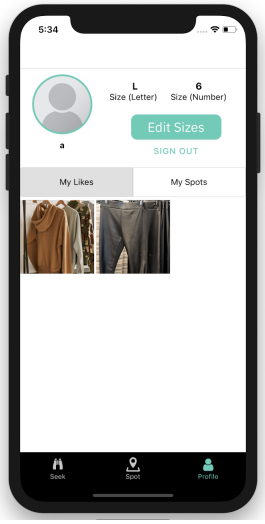
Change: To draw a closer tie between the spot that is submitted and the spots users browse through in the story bubbles, we designed the spot detail screen to place the picture that was just taken by the user as the background of the form, and have slightly transparent overlays for the user to input details about their spot.



15. Sign in, sign out, storing user state in Firebase

We previously did not have a way for users to sign in or out.

Change: We now store the user's information (sizes, spots, seeks, username) in Firebase and dynamically fetch this information upon sign-in. Users can also sign out.



Note that we also document some of the lower severity heuristic violations we addressed in the appendix.

Prototype Implementation

Tools

We used the tech stack and tools in the below figure to build our prototype.

We also used the following external packages for dynamic functionality:

- Google Image Search
- Apple/Google Maps
- Camera

Tools Analysis

We found React Native and Expo particularly helpful in its use of components because we had many similar features/designs across our app that would have been very difficult to code multiple times. The use of components enabled us to standardize our design, build our pages well, and then use the same standardized refined code across different features. React Native's syntactical similarity to JavaScript and web development also provided easier development paradigms for us to understand and use while building our app.

Because none of us had prior experience with React Native, this particular set of tools resulted in a steep learning curve for all of us. Additionally, because React Native is meant to be a platform-agnostic language, we weren't building with a particular phone in mind. When we found out our demo phone was going to be an

iPhone, we had to go in and manually re-style our application to be optimized for iPhone X rather than the default adaptations put in place by React Native.

Hard-Coded Content & Wizard of Oz

Seek and Spot

The seeks present in our app were actual seeks that each of our team members submitted through the app UI. However, parts of the seek and spot flow are hard-coded.

Currently, our seek image search is limited to the following terms: 'graphic t', 'dress', 'shirt', 'sweater', and 'jeans'. To search the Google Images API, you can use any phrase that contains one of the keywords, like “little black dress.” To avoid hitting our free daily query limit, we display only the first 10 results of the search. If the search is not working (this means we hit the daily query limit), we implemented a backup for the search for “graphic tee” by pulling from a local database in this case.

We also hard-coded nearby locations of 4 thrift stores (Goodwill of Silicon Valley, Goodwill Boutique, The Shop, Fillmore & 5th), since we currently do not have a classification algorithm to quickly determine which locations on a map are thrift stores. The shop preview panel on the spot map also pulls the number of possible spots at a given store from a local json file, while the shop detail view dynamically queries Firebase to fetch the real number of possible spots. Further work can be done in editing the preview panel to also query Firebase. Passing the form validation for creating a seek requires listing one of these stores as the location. The map view and search for thrift stores is fully functional.

Browse

Because we do not have thousands of live users on our app, we pre-coded some “seeks” and “spots” to help demonstrate usability features. Specifically, some of the top bubble icons on our Seek flow homepage come with pre-coded spot submissions for the user’s desired items. Some of the images of items that can be spotted at local stores are also pre-coded. However, users do have the ability to submit their own additional seeks and spots (on top of these pre-coded ones) that will be reflected in the interface through its live connection with our functioning database; this functionality helps the app testing experience feel authentic.

Profile

Our user profiles are fully-functional, including password setting, sizes, and the ability to signout. Users are stored in Firebase. Within the “Profile” view, the gallery view under “My Likes” is hard-coded to display the brown hoodie and black leggings. The “My Spots” tab is fully functional and will update every time the user spots an item (though this may take some time to load).

Future Features

Due to feature prioritization and time restrictions, we had to omit features that would be included in a fully functional version of the app, such as:

Seek

- Learning preferred thrift shops or radius of travel from user’s behavior to “like” items from certain stores

Spot

- A machine-generated “match percentage” describing the similarity rating between a user-submitted spot and the seek it was for
- Sharing spot status with spotters (“liked,” “purchased,” “expired”)
- Push notifications whenever a user steps into a thrift store to remind them to open the app and help spot items
- Automatic input of store address into Google Maps when user clicks “Directions”, currently only opens Google Maps

Browse

- Seek bubbles with multiple results (displayed in a sequential, story-like view)
- Ability to delete unwanted seek results from seek bubbles
- Automatic expiration and removal of “likes” after a month

Note that in order to pre-populate user information in our hi-fi for our demo, our hi-fi currently does not explicitly show what some of the screens look like for a brand new user (no seeks yet). We attach a screenshot of how we currently designed this real-world version of the new user interface in the appendix, and future work may involve clearer guidelines of use for new users

Summary

From our needfinding, we realized that despite the growing popularity of thrift shopping, there are many thrift shoppers who shop out of necessity rather than to follow a trend. We decided to focus our product on the sharp pain points of

low-income shoppers who pay with their time by spending hours looking through multiple stores for a particular item, often with no success. We designed Thread with the goal to lower the time these shoppers spend at thrift stores, and developed a new system for crowdsourced collaborative shopping where users can post requests for clothing items they want while spotting clothing items for others. After many rounds of concept testing and low-fidelity prototype testing, we reached our basic app design with our medium-fidelity prototype created in Figma. Through extensive review and adaptation of our app based on heuristic evaluation feedback for our medium-fidelity prototype, we developed our high-fidelity prototype using React Native, Firebase, and Expo. This prototype included ability for users to create their unique profile and dynamically submit seeks and spots as well as browse/save their favorite item findings as well as submitted spots.

Throughout this process, we have learned the importance of grounding a design in user empathy, and focusing on the sharpest pain point experienced by our users. By questioning and revisiting our assumptions about thrift shoppers, we were able to build a product that we believe truly gets at the heart of the thrifting experience for low-income shoppers, and that we hope can make their lives better.

Appendix

Additional Lower Severity (Level 0-2) Heuristic Violations Addressed

These changes were relatively minor, so we did not attach explicit screenshots for each one.

H1. Visibility of System Status / Severity: 1

There was no way to view whether the items I spotted had been purchased unless the user that purchased them sent me an optional note. Having a record might give users that use the app frequently a sense of fulfillment.

Change: We added a “Your Spots” view where users could see a record of what they spotted for others. We decided not to allow users to view if their spot had been purchased by another user to focus on our primary mission to save shoppers time, rather than create a social experience. We also did not want to discourage shoppers whose spots were not actually purchased, since only one out of many spots would likely be purchased for each seek.

H1. Visibility of System Status / Severity: 2 / Found By: B, C

After clicking through the “item found” notification, each found item has a clickable heart button. However, after toggling that button on/off, there isn’t any confirmation from the system as to what that action did and the app lacks documentation on this functionality.

Change: We added an alert clarifying that clicking the heart icon (“liking”) adds a particular spot of the user’s requested item to a shortlist of items they’ve liked for easy viewing. We also use color to indicate whether the spot has been liked or not.

H1. Visibility of System Status / Severity: 2

After marking a found item as bought, a screen comes up that asks the user to add an optional note to the spotter. However, there is no explicit “submit” button for that note, and there isn't any confirmation from the app after clicking “Seek again” that the note was actually delivered.

Change: We added a “Submit” button and designed the spot description flow to be more uniform, to make it clear that clicking “Submit” submits the entire spot (including the optional note) to the system. We also added a submit success screen to confirm that the spot was successfully submitted.

H2. Match between System and Real World / Severity: 2

After clicking through an “item found” notification, the various found matches for a particular sought item are displayed as a horizontal-swipe carousel. However, I would assume that the left/right arrows would scroll to different photos of a single item (e.g. matching the user flow from Instagram multi-photo posts).

Change: We created a more intuitive design for browsing through spots of users’ requested items, where clicking on a seek bubble opens an Instagram-style story view of the different spots of your item.

H3. User Control and Freedom / Severity: 1

When seeking an item, the input form only allows the user to upload two reference images and give two descriptors (fit and price upper-bound), which might not be sufficient detail for spotters to find a good match.

Change: We allow users to upload more than two reference images when starting a seek request, and help them by allowing them to sift through Google image search results for their seek as well.

H4. Consistency & Standards / Severity: 1

Sometimes you state “you found” or “found” instead of “you spotted” or “spotted.” The text input fields in the “Seek” request are capitalized while all the other text in the app is lowercase.

Change: We standardized to using “spot” instead of “find” to refer to spotting another user’s seek, as well as standardizing capitalization.

H4. Consistency & Standards / Severity: 2 / Found By: A

The top notification had an image of the item I was seeking, but I wasn’t sure if the lower notification was an image of a hat or the profile picture of the user who had “Spotted” my item since their name was bolded. If it is the profile picture that would make more sense to me, since the profile picture is not shared with other users anywhere else in the app.

Change: To avoid this confusion, we completely removed the ability to see another user’s profile picture and just referred to another user by their username.

New User UIs