CS 147 ShopKeep Final Report

Tyler Yep, Joy Yuzuriha, Alex Fu, Adam Halper

ShopKeep - Shop With Experts

Our team name, ShopKeep, conveys the service we are providing: bringing the knowledge of a shopkeeper to the customer. Our value proposition, “Shop With Experts”, highlights the conceptual experience we had in mind - making the user feel like they are being accompanied by a shopkeeper wherever they go.

Problem and Solution Overview

Problem

Through our needfinding interviews, we found that shoppers are often frustrated with an inefficient or overwhelming shopping experience. When we interviewed store workers, we found that store workers want to do more of what they love: interacting with their customers.

Solution

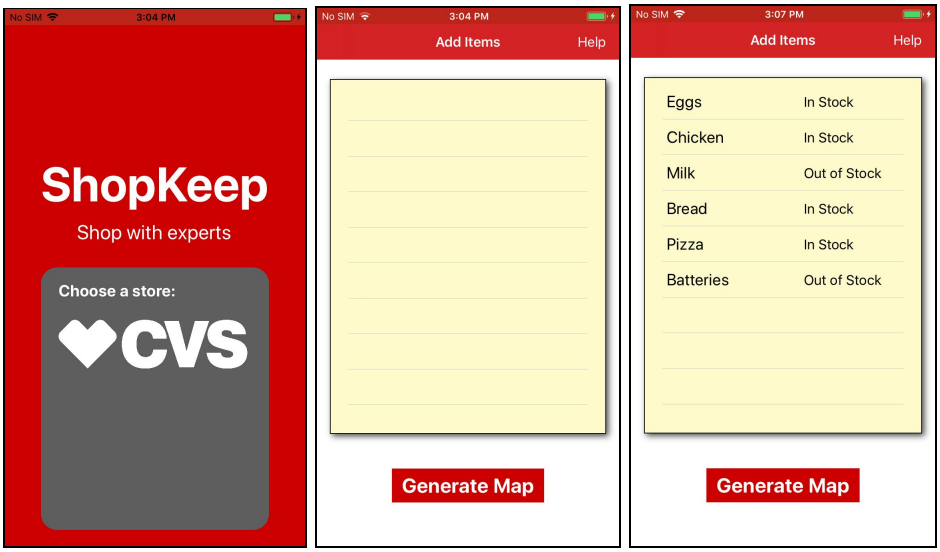
We will provide a lost or confused shopper with the means to “shop like an expert”, by checking ahead of time whether items are in stock, navigating through a store with a personalized and optimized path, and receiving personal assistance. We thought that this solution made benefited both sides of our problem - our app brings customers to

storekeepers more frequently and opens up opportunities to connect, while providing a customer with a easy, efficient store shopping experience using the storekeeper's expertise.

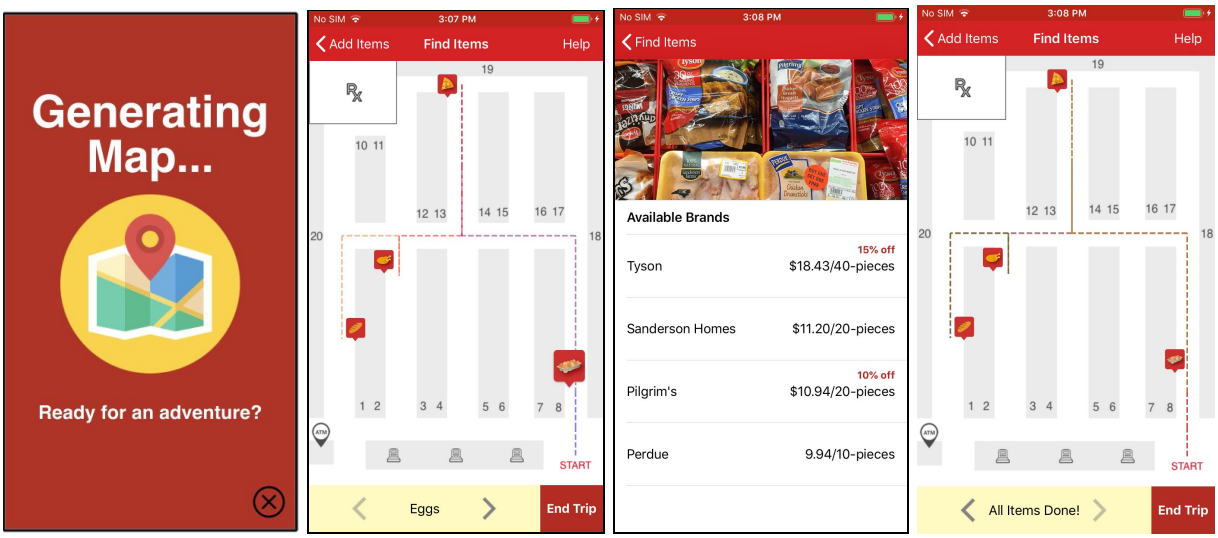
Tasks & Final Interface Scenarios

1. Simple: Check if items on grocery list are in stock at the store.
2. Medium: Navigate the store efficiently and check items off their grocery list.
3. Complex: Instantly ask a store worker for help when finding or deciding between items, and receive a first-class shopping experience.

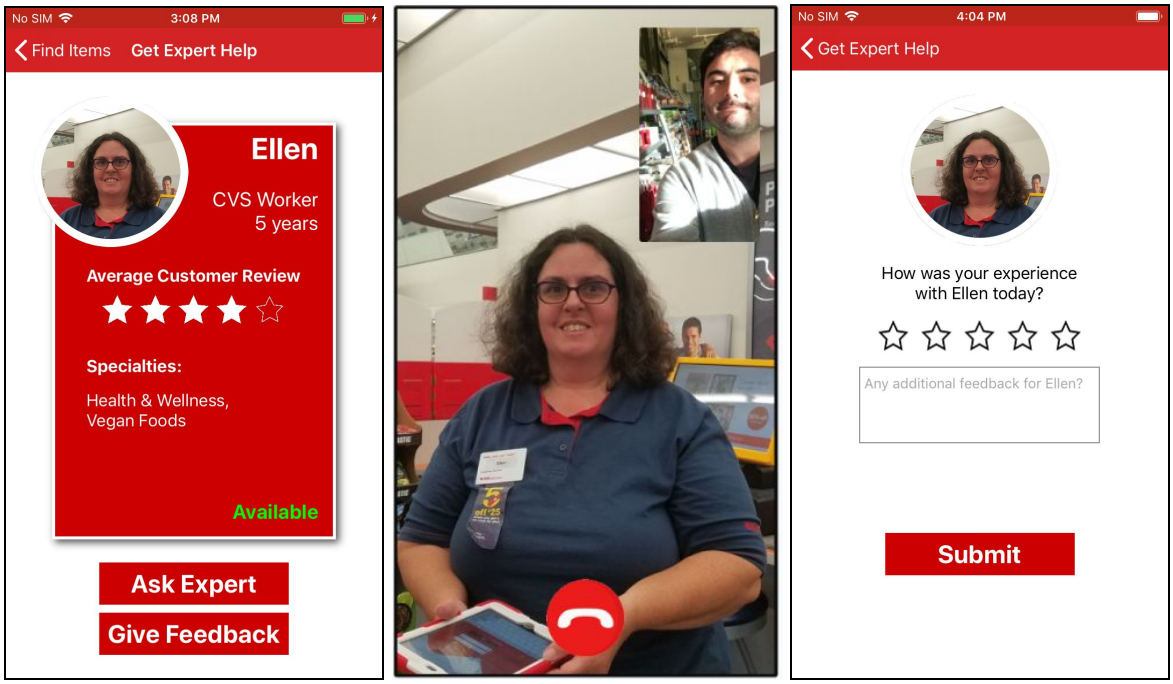
We chose the simple task because it was a common problem shoppers faced in our needfinding interviews - not being able to find the items they wanted, not being able to remember all of the items they wanted to buy, and not sure whether the item was available or not, thus wasting time searching.



Our medium task catered toward the shoppers who did not enjoy shopping, and wanted to be done with it as fast and efficiently as possible. We chose a map-based interface to cut down on the most inefficient part of an average shopper's experience - walking back and forth trying to find the right aisle. In our initial testing, we found that having a map drastically cut down the time spent looking for items, and the map UI also allowed us to display other helpful screens, like item info or weekly deals.

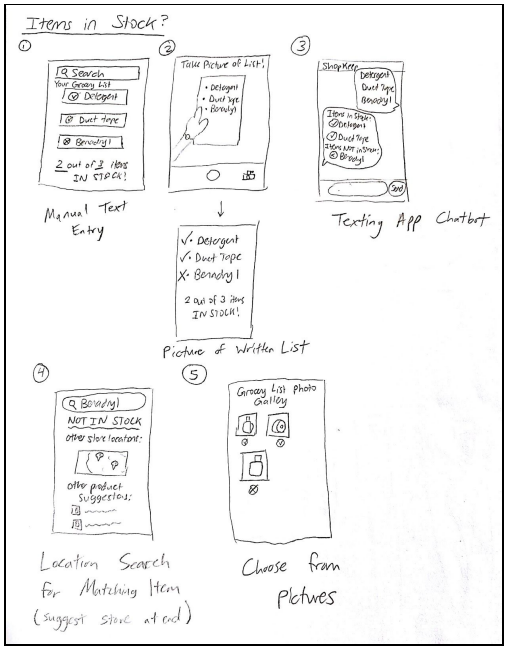


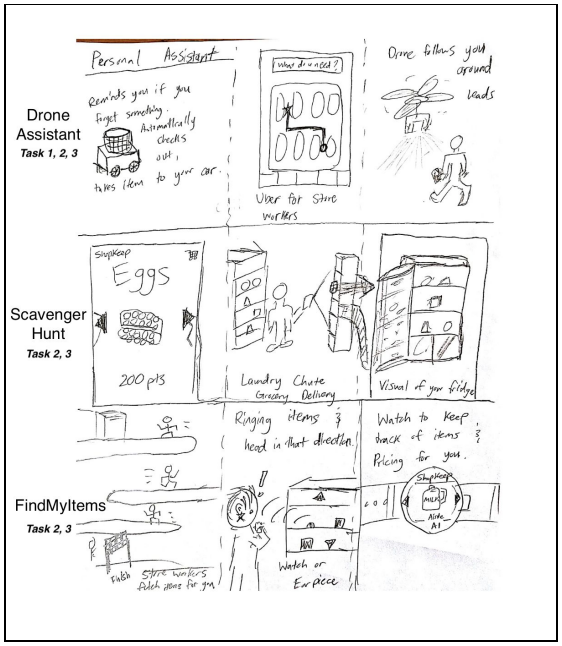
Finally, we chose the complex task to reopen a missing link between the storekeeper and the customer. We realized our app could serve as a platform for connecting the storekeeper's expertise to the customer's experience. By allowing the customer to call in-store employees directly, we allow the customers to get answers to their questions. At the same time, we allow the storekeeper to interact with the customers more, which is often the highlight of their day. Though this was the most difficult task to implement, we felt that our solution was very similar to what a real store could introduce to connect their employees to customers, thus improving employee morale and improving the customer's experience.



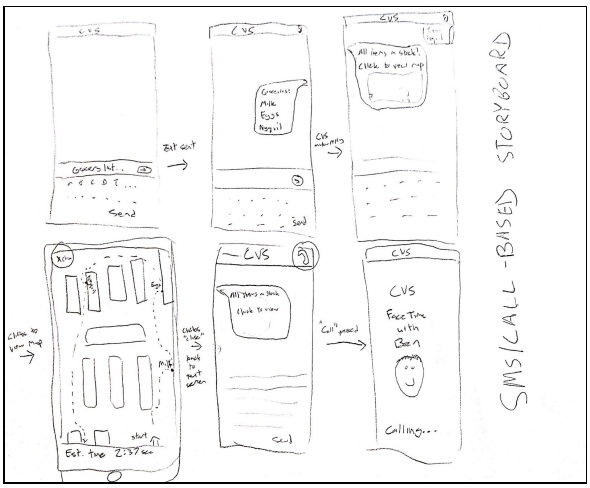
Design Evolution

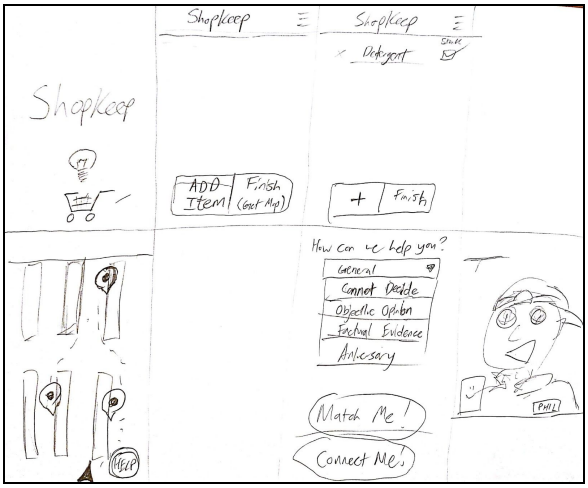
- **Brainstorming Sketches**
    - In our initial brainstorm, which we started in studio, we alternated coming up with one stretch idea and one more realistic idea from each person. We found that this process made our creative ideas more creative, and made the more realistic ideas keep the overarching experience of the product in mind.



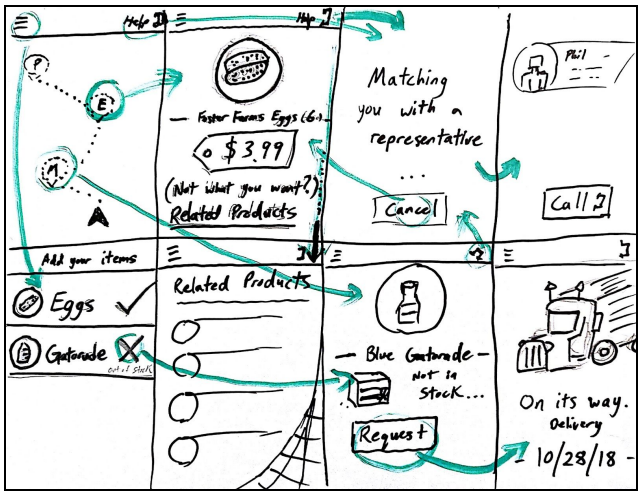


- **Storyboards**
    - Out of our initial brainstorm sketches came two potential designs for out app: SMS Chatbot or Store Catalog. Between these two, we weighed their pros and cons and eventually chose to go with the Store Catalog idea for our Lo-fi prototype.
    - SMS Chatbot, Store Catalog

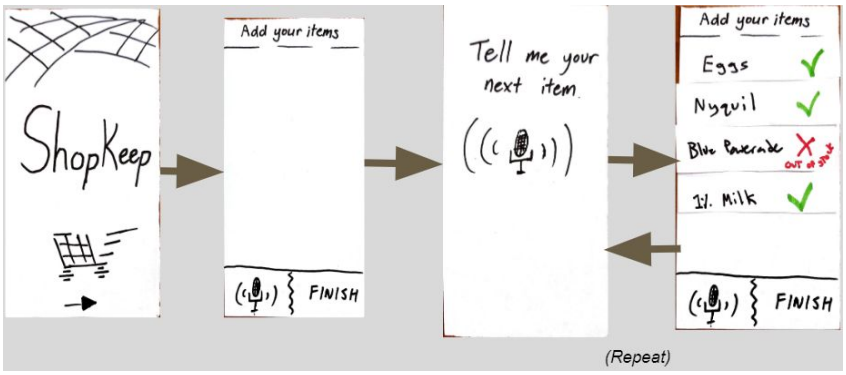




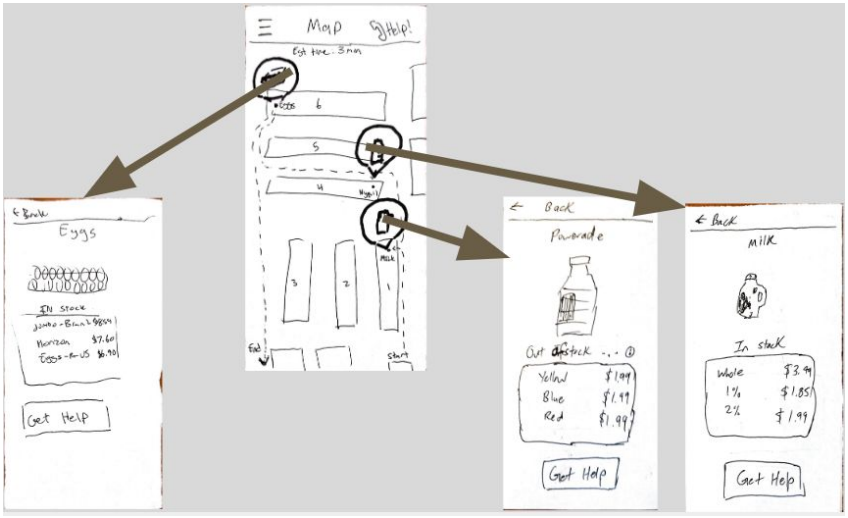
- **Selected Design Interface (Store Catalog)**
    - We chose the native app design because it can offer flexible, interactive assistance whereas the SMS-based design was limited in its capabilities. We determined that the benefit of offering in-app information and real-time navigation outweighed the added friction of downloading an app.



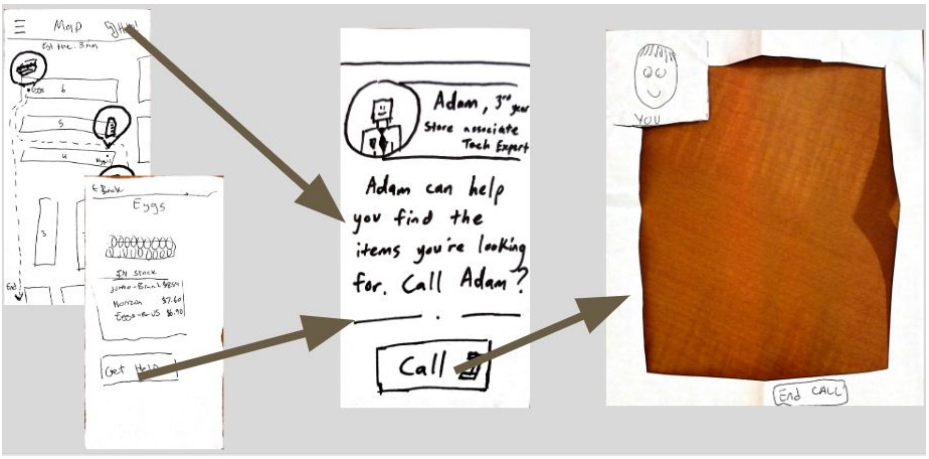
- **Low-Fidelity Prototype**
    - Task 1 - Check if items on grocery list are in stock at the store.



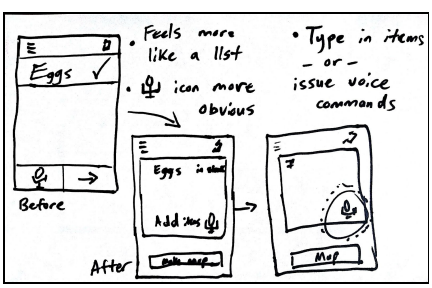
- Task 2 - Navigate the store efficiently and check items off their grocery list.

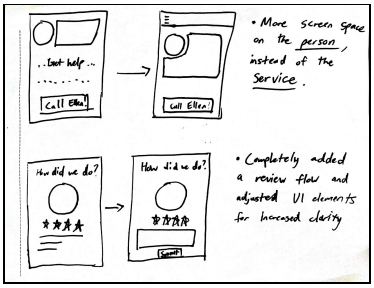


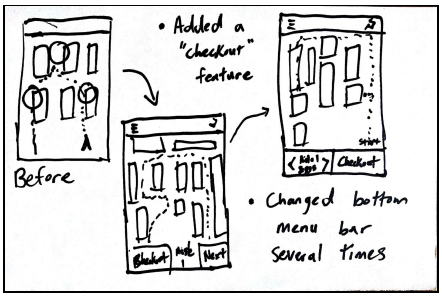
- Task 3 - Instantly ask a store worker for help when finding or deciding between items, and receive a first-class shopping experience.



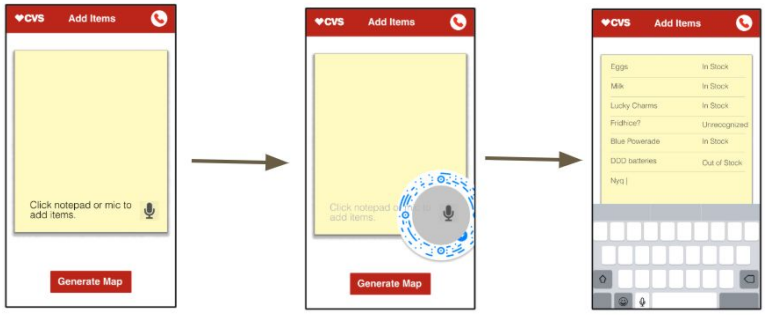
- **Medium-Fidelity Prototype**
  - Based on feedback from our Lo-fi prototype testers, we made a few design changes for our Med-fi prototype. Here are the major design changes that we made.
    - Using notepad design for list
    - Using colored dashed lines for items to make them pop
    - Moving voice command feature to the keyboard to leverage existing iOS technology
    - Emphasizing the employee profile to spotlight the employee and their experience.







- Task Flow for Task #1

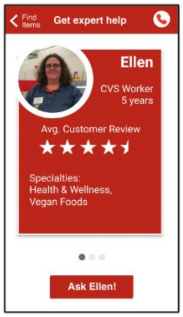


User inputs list through... ...audio or... ...typing!

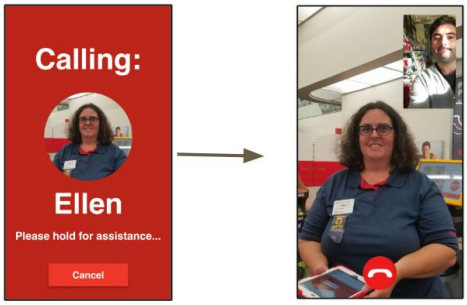
User sees which items are in stock and is ready to see their map!

- Task Flow for Task #2

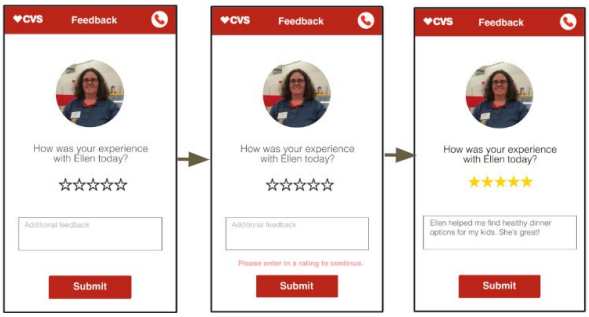
- Task Flow for Task #3



When the user presses the "Call" button, they are given profile information about the store worker that is available to help.



Personalized video call with a store worker.

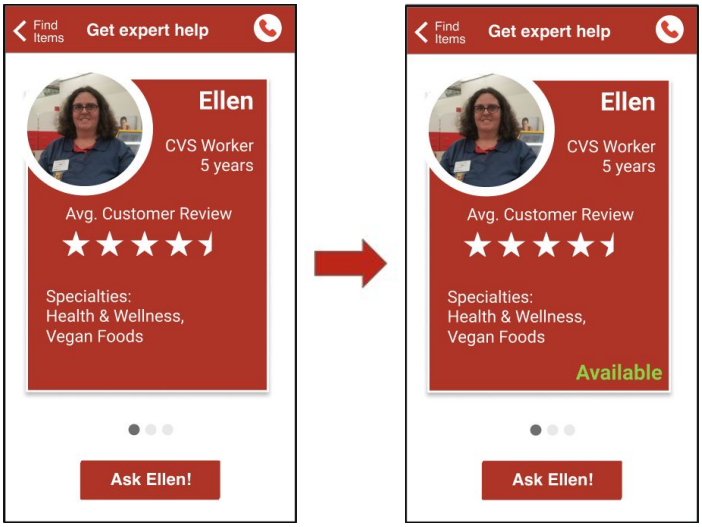


User is asked to give feedback on their personalized experience with the store worker.

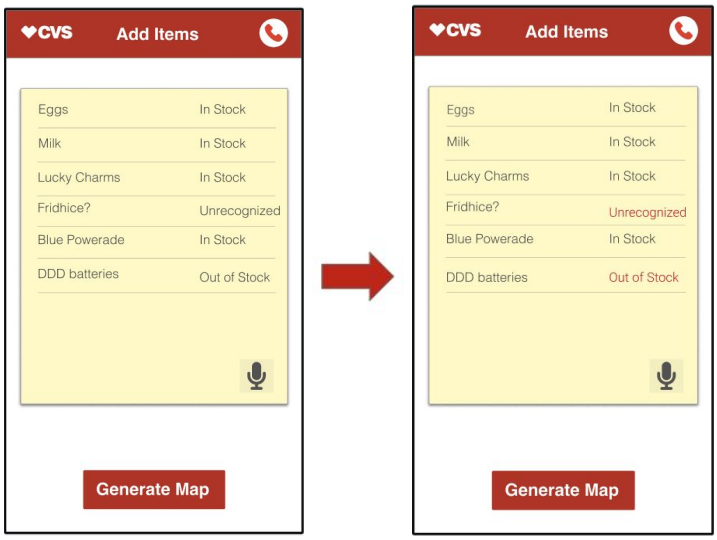
- **High-Fidelity Prototype**
    - Finally, we incorporated feedback from our Heuristic Evaluation. Some of these changes included seeing whether a store worker is available or not on their profile, having larger “Start” and “Finish” buttons on the map, differentiating more between the “Out of Stock” text and “In Stock” text on the shopping list, and changing the placement of buttons to prevent errors. These changes are described in detail below in the “Major Usability Problems Addressed” section.

Major Usability Problems Addressed

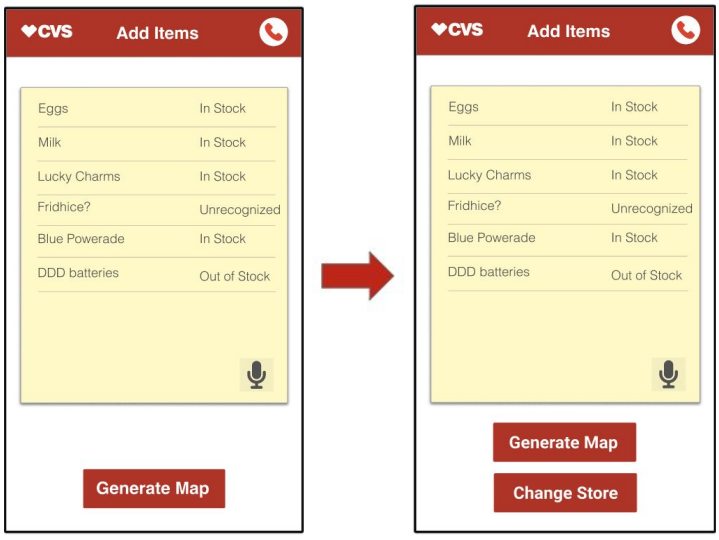
1. **Showing store worker’s availability on profile**
   H1: Visibility of System Status / Severity: 2
   Consumer should be able to tell when a shopkeeper is available or not, or whether the list of people that are given as a resource are people that are currently working at the store.
   Our Fix: We added “Available” or “Unavailable” to the employee profile page. The visual change uses green text, which draws the user to the complementary color and feels like the employee is ready to help at any time.



2. **Contrast color for “Out of Stock” text in shopping list**
   H1: Visibility of System Status / Severity: 2
   The text for an item that says “out of stock” is the the same font/color as the text that says “in stock.” A user who is not paying attention may not realize the difference.
   Our Fix: We added the colors to the shopping list to distinguish the different in-stock results, making it easy for the user to know how to fix their list.

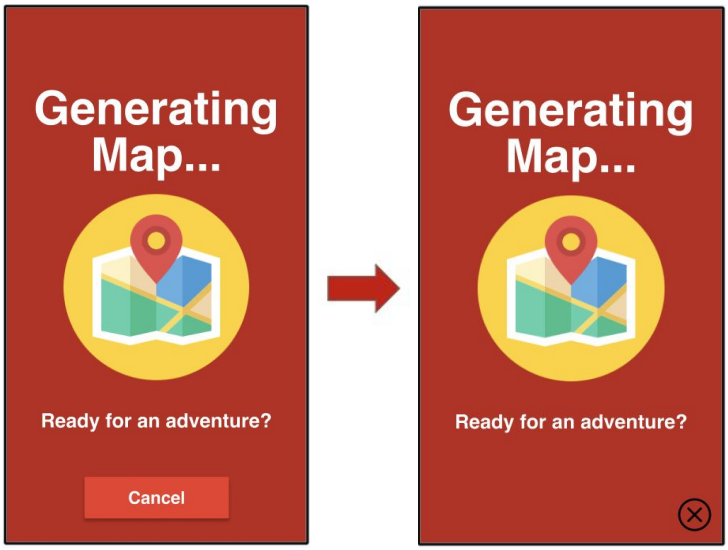


3. **Back button to "Choose Store" page if user wants to choose different store**
   H3: User Control and Freedom / Severity: 4
   Users can't choose a different store. The user may need to go back if she wants to select a different store. Similarly, clicking "Done" on the screen that says "Thank you for shopping with CVS," takes the user to the "Add Items" page, but the user may want to shop at a different store this time.
   Our Fix: We added a back button to return to the home screen. There are some small navigation issues with the back button, but on the Shopping List page the page button is fully functional.

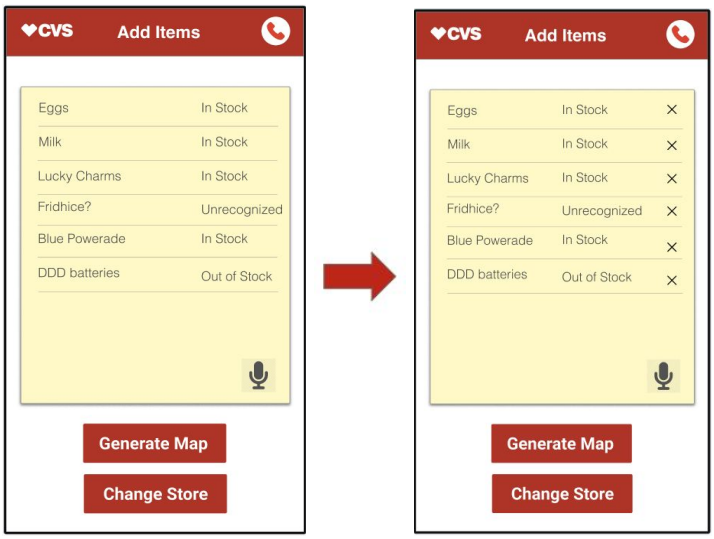


4. **Change location of "Cancel" button on Generating Map page**
   H5. Error prevention / Severity: 3

The cancel button for generating map and calling assistant is almost exactly in the same place as the buttons for making generating the map and asking the assistant for help.
Our Fix: We ended up removing the loading screen, as the map generated quickly enough in most use cases. If we did implement a loading screen, we would have used the below design:



5. **Deleting items on shopping list**
   H3: User Control and Freedom / Severity: 4
   There is no way to remove items on the “Add Items” screen. The user may want to remove items if she changes her mind about an item.
   Our Fix: We wanted to include a remove item button on each list item, but did not get to implementing it in the final prototype due to time constraints. This bug was not as high priority, since the user could remove items simply by deleting their contents.

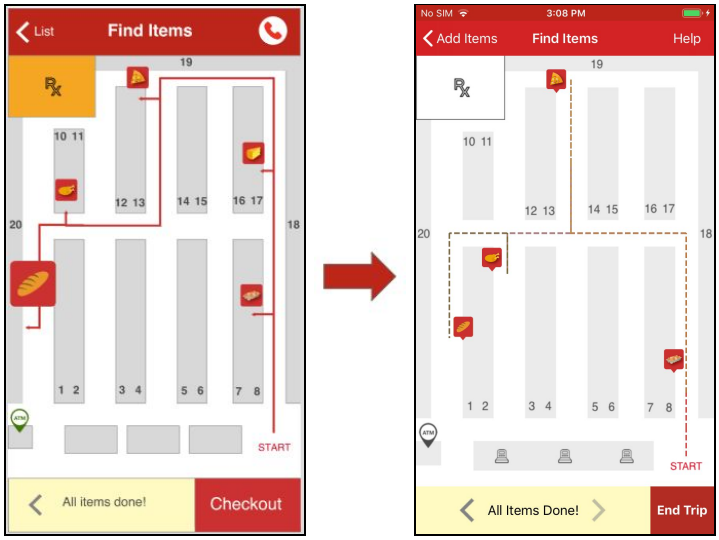


6. **Previously added items still visible even when adding items through mic**
   H1 Visibility of System Status / Severity 3
   When users are adding items, there is an option to speak the items into existence on the grocery list using the mic. However, when a user clicks on the mic to begin speaking items, the previous items they entered go missing and are replaced with the dialog box "Click notepad or mic to add items."
   Our Fix: Users are able to see the top items on the list at all times, even when speaking into the mic.

7. **Seeing shopping list screen after calling a store worker**
   H1 Visibility of System Status / Severity 4
   After clicking on the Call Employee screen, the user is presented back to the same screen where they edited their grocery list. It is unclear if the previous grocery list was deleted and also why users are sent back to their list after getting help. Shouldn't users see the previous screen (i.e. the map) again after getting help, so that they can get help with the next items?
   Our Fix: After submitting a review, the user is taken back to the map screen, and we intended for all of the items to stay on screen. However, there remain bugs in our implementation that prevent this from occurring - this navigation portion was difficult to get consistently right.

8. **Input quantity of each item to shopping list**
   H3 User Control and Freedom / Severity 4
   There is no way to specify quantity on the "Add Items" screen. Thus, if the user wants to buy two loaves of bread, they have to input "Bread" twice, which is inefficient. It's important for the user to be able to specify quantity because quantity affects the total price at checkout (and to my understanding it seems that users pay for items through the app).

Our Fix: We removed the checkout feature of our app, so the user has no need to specify a quantity per item - the item list is simply to remember the items and to map it on the next screen.



9. **Error prevention for generating map without choosing a store**
   H5 Error Prevention / Severity 4
   When the user clicks on the information icon in the top right of the screen, it shows the user how to add items to the shopping list. The only option on that screen is "Generate Map." However, the user may not have chosen a store yet (i.e., if they accessed the info button from the loading screen without selecting a store first).
   Our Fix: We did not add the tutorial, as we did not have the time and it did not seem necessary based on our testing on intended users.

10. **Error prevention for generating map without adding any items to list**
    H5 Error Prevention / Severity 4
    If the user selects "Generate Map" before adding any items, it generates a blank map, which is essentially useless for the user.
    Fix: Include error checking to make sure that the user has added at least one item, and inform the user if she did not add any items. Alternatively, the "Generate Map" button could only appear once the user adds at least one item.
    Our Fix: We ultimately decided not to address this error, as some users may still want to see a map of the store even without inputting any items. The screens are no longer spliced by a loading screen, so switching between them is much easier.

11. **Add border around CVS logo to make it more obvious that it's a button**
    H8 Aesthetic and Minimalist Design Severity 3
    It's not obvious that you can click on the CVS icon in the "Choose a store" box on the home screen, so users may get confused as to what they're supposed to do.

Our Fix: We added the black background to make it clearer to users that they must select a store.

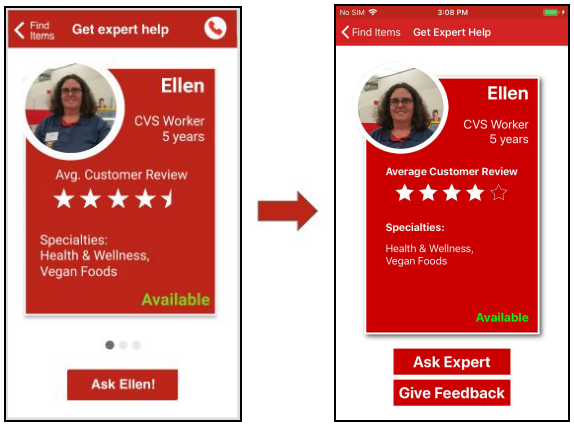
12. **Add Save and Continue option**

    H3 User control and freedom / Severity 3

    When checking out, a user may want to buy the items without clearing the list. What if a user has to cut a shopping trip short and would like to retain the list and map for the next trip? What if they want to check the total of the items they've picked up before continuing their shopping?

    Our Fix: Our app ideally would maintain its state between runs, but in our simplified model we did not include this feature.

One final change we made was adding a "Give Feedback" button to the employee profile page that would not be present in a released version of the app. This button was added to avoid needing to Facetime in order to view the Employee Rating Page.



Prototype Implementation

Our prototype was built on native iOS 12, using Xcode 10.1 on Swift 4 and some small amount of Objective-C, with a testing device iPhone 7. We chose to build directly on native iOS because we have two team members who are familiar with native iOS development. Compared to development frameworks like React Native, Swift and Objective-C implementation gives us access to the full functionality provided by iPhone and iOS.

Native iOS using Swift and Objective-C is advantage for us specifically in the following ways:

- We use the new Speech framework to enable voice dictation input (https://developer.apple.com/documentation/speech). This is a new framework included in iOS 10.0+ that in the background runs Siri services. The voice dictation is fairly accurate and easy to use. Instead of spending a huge amount of time implementing voice recognition, Speech framework provides us an easy way to implement a voice input feature.
- Native iOS has also fully integrated video call and FaceTime functionality. By using the FaceTime functionality included in the standard UIApplication, we can directly start a FaceTime session by calling a URL conferred to the stand URL scheme.
- Swift and Objective-C are both full development languages that support complex logic. Compared to JavaScript which React Native is built upon, Swift and Objective-C are much faster and more efficient for calculation. We have implemented a dynamic algorithm to search for the shortest path that connects all items of a user's grocery list. Even though this particular algorithm is an easy variation of breadth first search, if the user has a long list of items to buy or our inventory is linked a large database, the graph algorithm would need to be efficient enough to avoid any delay.

Native iOS were not helpful to us in terms of:

- Storyboarding and styling take more time to implement as there are more codes to write comparing to other lightweight implementation tools.
- Navigation and transition also take more time comparing to other mobile web app frameworks, for example, Xamarin or Angular.
- Native iOS obviously only works for iOS. We do not have cross-platform ability and would have to reimplement everything if we want to have a prototype on Android.

There are a few Wizard of Oz techniques included in the current prototype:

- We did implement a dynamic graph search algorithm on the map to find the shortest path between finding each item on the grocery list. However, in reality, the optimal path should be more complicated by taking in more parameters into consideration. For example, frozen food should probably show up last, promotion items should maybe appear first before non-promotion items, user should end up close to the counter, etc.

- After the user presses “Submit” button on the feedback page, the rating and feedback are not saved anywhere as we don’t have a real database on this prototype. The rating should be saved and calculated so that when the employee profile first shows up, the rating should be up-to-date.

The hard-coded data within this prototype:

- When pressing “Ask Expert” button, the prototype calls Adam as the current hard-coded phone number. It should be linked to each employee’s working number.
- Because we have no real business contract with CVS or any other stores, we don’t have access to a real inventory of grocery items in stock. Current implementation is limited to a short list of hard-coded grocery items.
- Current prototype only supports one store, CVS, as an example and does not change anything in-app if another store was used.
- The store map is currently hard-coded with fake locations. In reality, the map should be linked to the database of each store and show the correct locations.
- On “Get Expert Help” page, we only have one fake profile showcasing the design. It should be linked to the employee database of each store and show a list of available employees who can help customers. The average rating and specialties should also be up-to-date with the employees’ detailed information.

The missing features in this prototype:

- Currently we don’t save the user’s grocery list after he closes the app session. We should be able to save that in a database and display the list whenever the user wants to check.
- We did not implement the introductory tutorial to help user navigate through the app.
- On grocery list page, we did not implement the functionality to delete or edit items.
- On grocery list page, there are a few error handling issues that we didn’t have time to do. For example, if the user inputs an empty line, we should ignore instead of marking “unrecognized”.
- We might consider customize the FaceTime view so that the user can still see the map or his grocery list when he is asking for expert help. Currently, we are only using the default view provided by FaceTime.

Summary

Though we did not have time to implement every feature of our medium-fi product, our high-fi prototype accomplished the three main tasks to solve our needfinding problem, and demonstrates the rapid, iterative process we used throughout the quarter. The continuous feedback from our interviews and section led to many visual changes in our UI and how our end users, the store customers, interact with the final product in their day-to-day shopping experience.