# InvestorScope

\

*a seamless, intuitive way for amateur investors*
*to discover new investment opportunities.*

Aaron Sekhri, Matthew Appleby, David McLaren and David Khavari

**Contents**

**Team**

Aaron Sekhri (writing, design, management)
Matthew Appleby (design, user experience)
David McLaren (development, documentation)
David Khavari (development, management)

**Value Proposition**

InvestorScope is a seamless, intuitive way for amateur investors to discover new investment opportunities.

**Problem and Solution Overview**

We found in our contextual inquiry that investors typically only buy what they know. That is, they invest in companies they are familiar with, industries they are knowledgeable about, and in ideas they are comfortable with. Prima facie, this is not a bad thing; why shouldn't casual investors invest in what they know as opposed to what they do not? However, when considering the stakes more carefully, we realized that this behavior leads to a lack of diversification and sub-par investment decisions due to a lack of knowledge of the other options. InvestorScope seeks to improve personal investing by helping customers discover new equities using a recommendation engine powered by questions the customers answer, as well as a sortable list of equities based on both quantitative and non-quantitative metrics. The design can be seen in appendix 1.

**Task and Final Interface Scenarios**

Simple - Our first task is search. We want our customers to be able to search for any given equity, and arrive at a page that gives them the most immediately salient information about that stock, with the opportunity of digging deeper if they want to. This task is the same as for the last iteration. We chose this task because it is a fundamental way in which discovery can occur, and because it allows the customer to access salient information about a stock in a matter of seconds. The storyboard walkthrough can be found in appendix 2.1.

Medium - A slightly more complicated task we chose was using a recommendation engine. The recommendation feature asks the customer a number of questions, which it then uses to get an understanding of his or her risk profile, interests, areas of concern, and strategies. It uses this data to offer the customer a recommendation. Over time, the engine delivers better and better recommendations, as it has learned about the customer. We chose it because fundamentally the mission of this application is creating opportunities for customers to discover stocks and this task is an effective way of delivering on that mission. The storyboard walkthrough can be found in appendix 2.2.

Complex - The most complex task we have chosen is sorting equities. Our application has a feature that lets the customer sort equities on the basis of parameters as they see fit. The parameters are be quantitative and non-quantitative, in a language that is comprehensible to the casual investor. We chose this task to give the customer a guided way of discovering new investing opportunities. The investor may have an idea of what sort of company they are seeking out, but no easy way of finding and ranking them according to particular metrics. We chose to included non-quantitative metrics as well because they allow users to rank companies according to some of their values, which was something that we saw as an unfulfilled need in the need finding and interview process. We have categorized this as our most complex task because we believe it requires more thought and intent behind its execution than the other two tasks. The storyboard walkthroughs can be found in appendix 2.3.

**Major Usability Problems Addressed**

H2-1: Visibility of System Status

Problem (4): "The app logo is not displayed on any of the screens. Was this a strategic choice? There is currently no way of knowing which application is currently in use. A solution is to find a place to display the InvestorScope logo."

Solution: We have changed the UI of each page in such a way that the logo is always on the top left corner wherever you may be in the application [appendix 3.1].

Problem (4): "Screen 10 lacks indication of whether the market is open or closed."

Solution: We made no changes on the basis of this critique because the application we have created is for discovering new stocks, not actually executing trades. This critique is not relevant for our purposes.

Problem (3): "There is no connection between the "explore" button and the "Recommended" tabs. The "Explore" tab is used so that the "Recommended" tab works but there is no indication."

Solution: We removed the explore button altogether, as we realized that it does not help the user achieve any task that he or she otherwise could, and is misleading [appendix 3.6].

H2-2: Match Between System and Real World

Problem (3): "The bottom icon that prompts the questionnaire is called "Explore", but it seems more like "Find Recommendations". It seems that saying 'explore' means you are willing to move outside of normal preferences, and 'recommend' is more similar to 'give me 2 options that you think I would like most'."

Solution: We removed the explore button altogether to avoid the sort of ambiguity raised by the evaluator [appendix 3.1].

Problem (3): ""Given that the target audience for this app is likely to be unfamiliar with the majority of the stock abbreviations (with the exception of GOOG, APPL, etc…), the number of foreign stock abbreviations disorients users in a way that may discourage discovery and the actionable task of clicking on the company."

Solution: We now display the full name of the company as well as its abbreviation to avoid discouraging discovery and clicks [appendix 3.1 and 3.4].

H2-3: User Control and Freedom

Problem (3): "To be particular, I can anticipate that some answer responses in the explore module may not necessarily be the user's answers to that particular question. This is a slight violation of the freedom heuristic where the user should be given the option to freely answer in the manner that they wish."

Solution: This is not an easy problem to solve as it would require a very different approach to answering the questions (from multiple choice to free response). We do not think that this is a change we can easily make as parsing responses would be highly problematic, but we did change the wording of our question prompt (for instance, writing "which of these options most closely approximates your choice" for a multiple choice question) to alleviate this problem [appendix 3.3].

Problem (3): "Upon entering the explore phase, the user is given no accessible route to exit that task and return to the home page. Without a back page, I can anticipate some users getting frustrated with a particularly long questionnaire with no way out of the survey."

Solution: We are taking advantage of the fact that the mobile browser UI includes a back button that will be our web application's default back button [appendix 1.4 shows this].

Problem (3): "The "Explore" feature doesn't allow the user to change his/her selection for attitude, sector interest, and favorite movie unless he/she starts the process over. Fix: Add the ability to go back to previous selections"

Solution: Again, we will use the mobile browser's back button to allow users to more easily navigate the application [appendix 1.4].

Problem (3): "If system continually learns user's preferences, then it seems difficult to adjust for a sudden change in preference (e.g. typically risky-loving investor wants to find some low risk equities)."

Solution: This is an interesting critique. While we agree having a page where the user is able to customize his or her preferences, which is then incorporated into the system's understanding of the user, we feel this would be too large a change to make at this stage. Creating an account page where the user can organize her or her preferences/settings would entail a number of entirely new tasks. So we will keep this in mind for future iterations, but decided against creating this functionality for this version of the application.

<u>H2-4: Consistency and Standards</u>

Problem (3): "The icons disappear and then reappear upon searching for a company in the main page's search bar. A user might wonder why it was back on the page after the search was completed, given that the page only displays results of a small subset of

companies."

Solution: We have removed the icons altogether as we felt that they were confusing and uninformative, for the most part. While there were some that were relevant, they were not universally so, which led to our decision to simply remove them from the application [appendix 3.2 and 3.4]. The tradeoff here is that the design is more stark and slightly bland consequently.

Problem (3): "The UI experience while in the Explore stage seems to be different and a bit unexpected when entering the explore module through the main page. The inconsistent UI flow makes it hard to see how exactly the explore module is tied to the rest of the application."

Solution: We redesigned the UI flow of the recommendations (previously explore) page to be more intuitive and in line with the aesthetic of the home page. There's a common color scheme and similar button format, which eliminates this problem [appendix 3.3].

Problem (3): There are two separate sorting mechanisms on the homescreen. There are the categories and then also the "Recommended," "All," and "Best." This is confusing and it is very unclear how selecting something on one bar impacts the others.

Solution: We have eliminated the categories altogether in response to this problem as we realized that this was confusing to the user, and because the categories did not help them achieve tasks any more easily than if they did not exist [appendix 3.1 and 3.4].

Problem (3): "Some of the options on the questionnaire are ambiguous in terms (e.g. "Fire and Forget" for investment attitude), and not mutually exclusive (e.g. "Event Driven" and "Day Trader"). How should a user resolve a situation where multiple options can apply?"

Solution: We were admittedly sloppy with the questions to begin with because we were mostly concerned with the functionality of the application. We have revised our questions to be more clear, and we did our best to ensure that answers are mutually exclusive. In the case of questions which are inherently nuanced, we introduced wording in the question prompt that indicates that the user is to select the answer that is *most* applicable to them [appendix 3.3].

Problem (3): "Meaning of "chillness" on screen 7 is not clear. Fix: have a help section where each of symbols are explained."

Solution: We were slightly frivolous with our sorting categories because our main aim with the prototype was to test the functionality. We did not consider the categories with too much seriousness, and so for this iteration we have made sure that the categories are unambiguous [appendix 1.2 versus medium-fi prototype in appendix 3.1].

Problem (3): "The difference between "Recommended" and "Best" is not clear."

Solution: We simply eliminated the "Best" section because it was not particularly useful or informative to begin with, and therefore this is not a problem anymore [appendix 3.1].

H2-6: Recognition Rather Than Recall

Problem (3): "I cannot see past searches/results/viewed pages. Perhaps I liked an equity, and would like to view it at a later date, but don't remember how I got there."

Solution: We believe that this is a new task altogether, and at this late stage are not inclined towards including it in the application. While we acknowledge that this is an important feature that would definitely be useful to the user, we believe that this entails working towards a new task, and we would rather focus on honing the three that we are optimizing the application for.

H2-7: Flexibility and Efficiency of Use

Problem (3): "Very limited information for equity page, especially for someone with an intermediate (or more) amount of knowledge in finance."

Solution: We added some more, advanced, metrics for the intermediate user, such as PE ratio, market capitalization, etc. [appendix 1.4]. The tradeoff here is that we may alienate novices.

H2-8: Aesthetic and Minimalist Design

Problem (3): "The boxes for each answer on the questionnaire were really large, and it wasn't obvious initially that there were a lot of options and we had to scroll down to see them. The colors of the font and the square blend together so I didn't notice the half-concealed words at the bottom."

Solution: We have completely revamped the design of the questionnaire. We eliminated the boxes for a much simpler list format, and have made it such that the question on the bottom of the screen is only partially visible, indicating to the user that they must scroll further to see it as well as other options [appendix 3.3].

Additionally, we included a glossary page [appendix 3.5] with common investing terms to help less knowledgable investors navigate the financial landscape, the absence of which was a lower-severity concern amongst some evaluators (limitation however is that it is a standalone page and is not available at every instance the customer is sees a foreign term). Another lower-severity concern was that there was no way of exiting whatever part of the application the customer was in, which we solved by including the logo on every page which the customer can click to go back to the home page (the tradeoff being that we lose valuable real-estate for other features).

**Design Evolution**

There are a variety of motivations that have guided the design changes we made, ranging from details from the class and readings to the expertise of the more design-oriented members of our group, but first and foremost, it has been the reactions and needs of the customer that have been the driver of the changes we have brought to the design of our application.

Our initial sketches were the direct products of the contextual inquiry; we did them as soon as we realized stock discovery would be the main thrust of our application, and they were highly random and idiosyncratic. We did not consult each other on the sketches, and simply drew whatever came to mind. In this way, our original designs were effectively brainstorms. Our ideas ranged from tables to lists to graphs, and we used that opportunity to imagine as many ways of designing discovery as we could imagine.

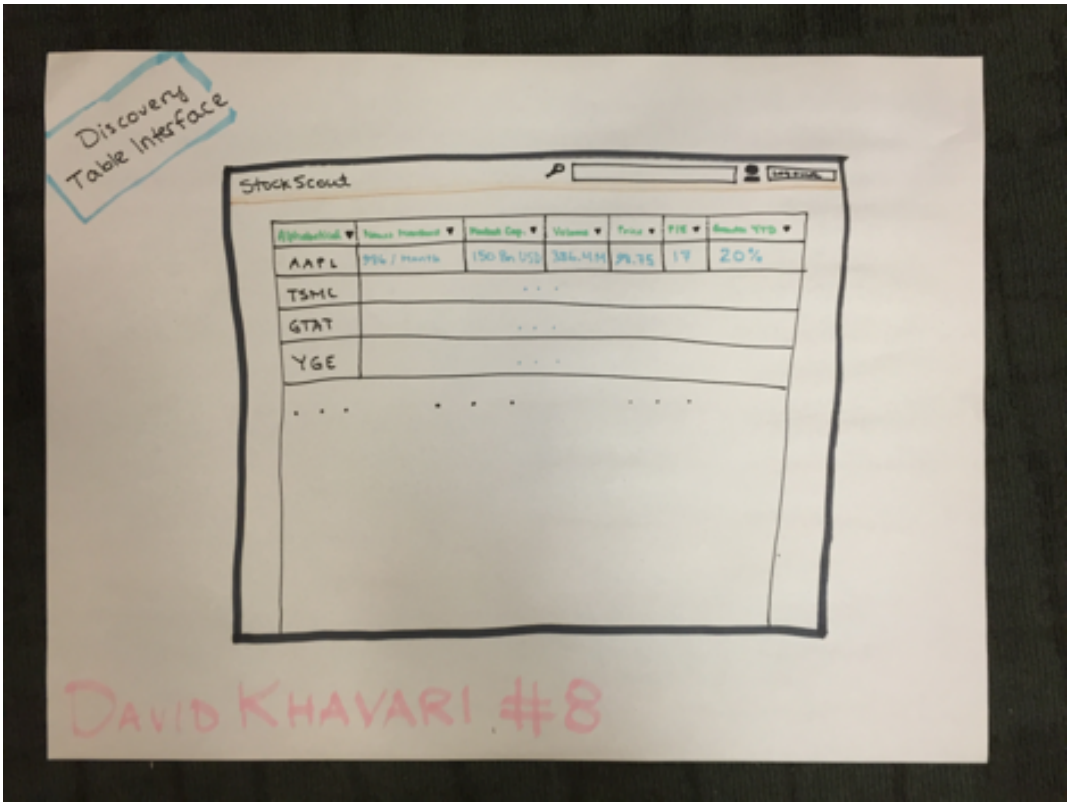
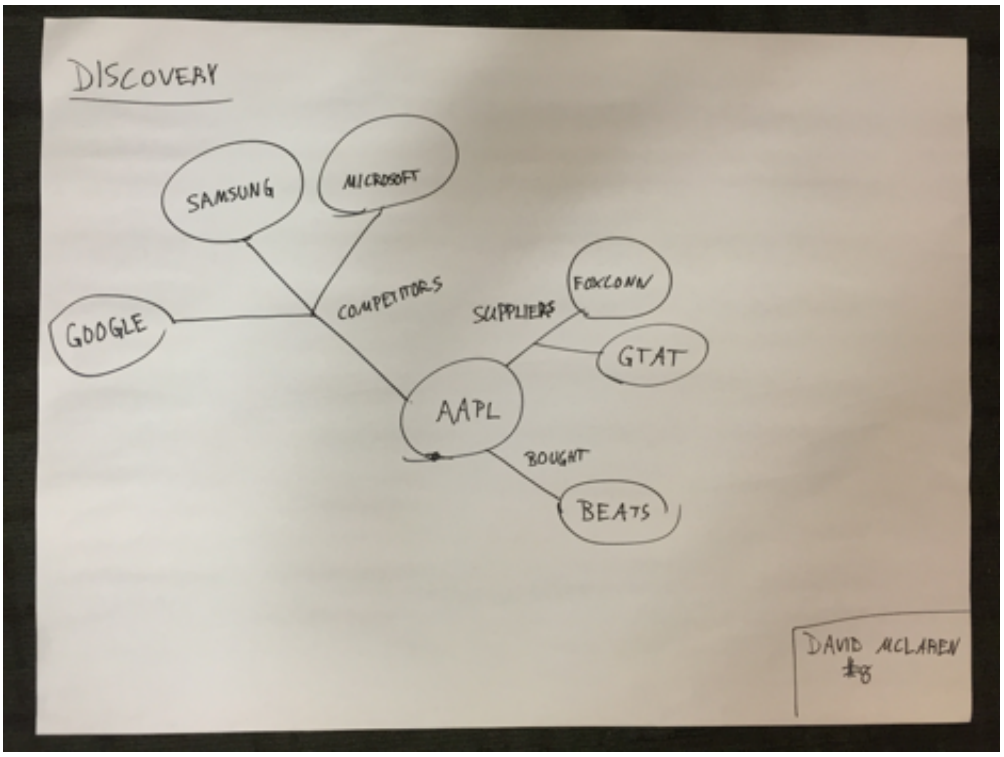
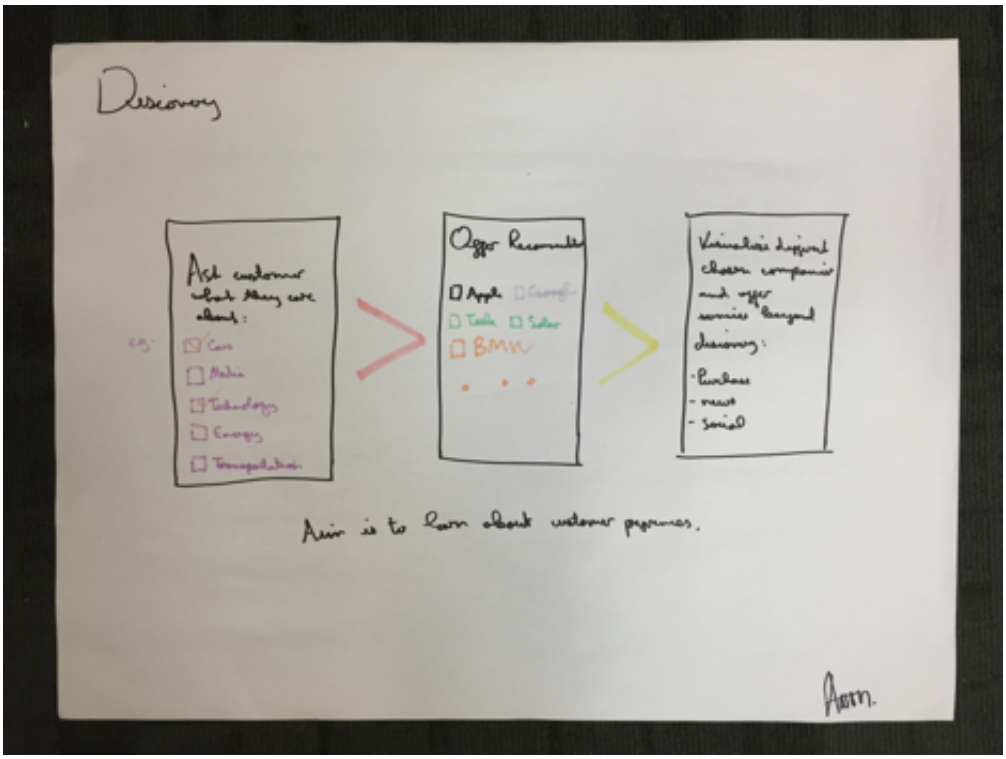



Table interface

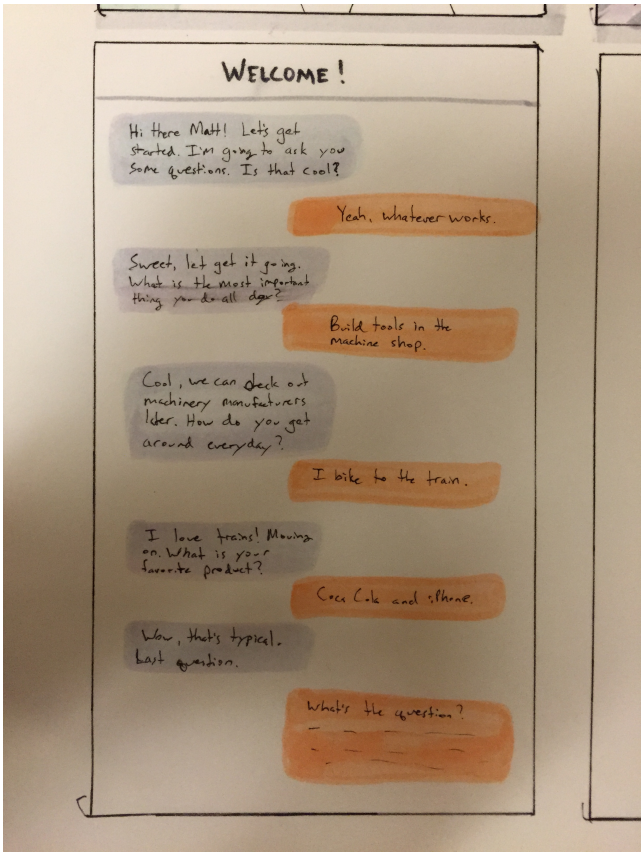



Graph interface

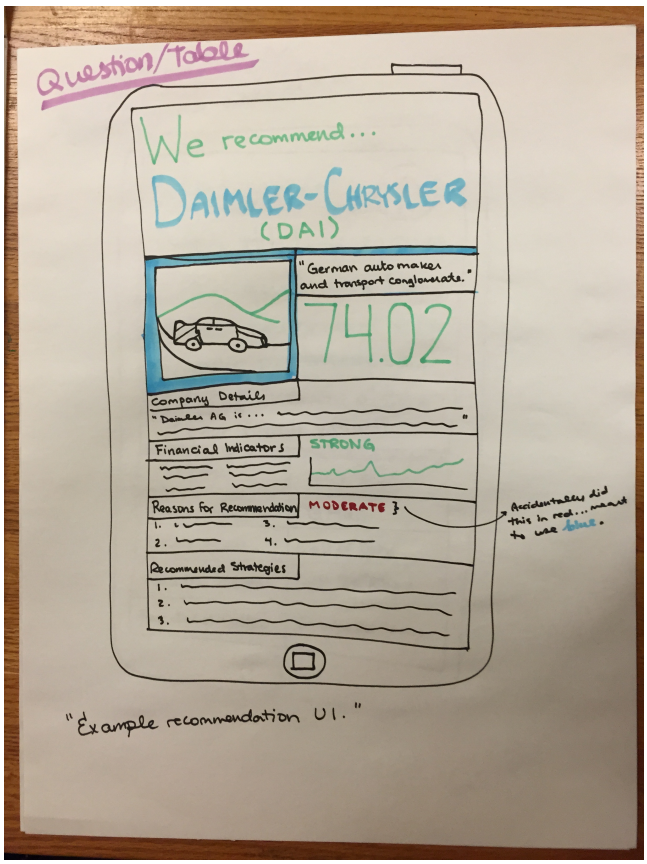



List interface

For our concept video, we debated between two possible UIs: a question and answer interface similar to most messaging applications, and a graph interface illuminating connections between companies. We made a judgement call on the basis of form and function; we believed the question and answer interface offered us more in terms of what we could build with it, as well as how it would look on the medium we were targeting (which was mobile, as that is the device most engaged by our target customers).

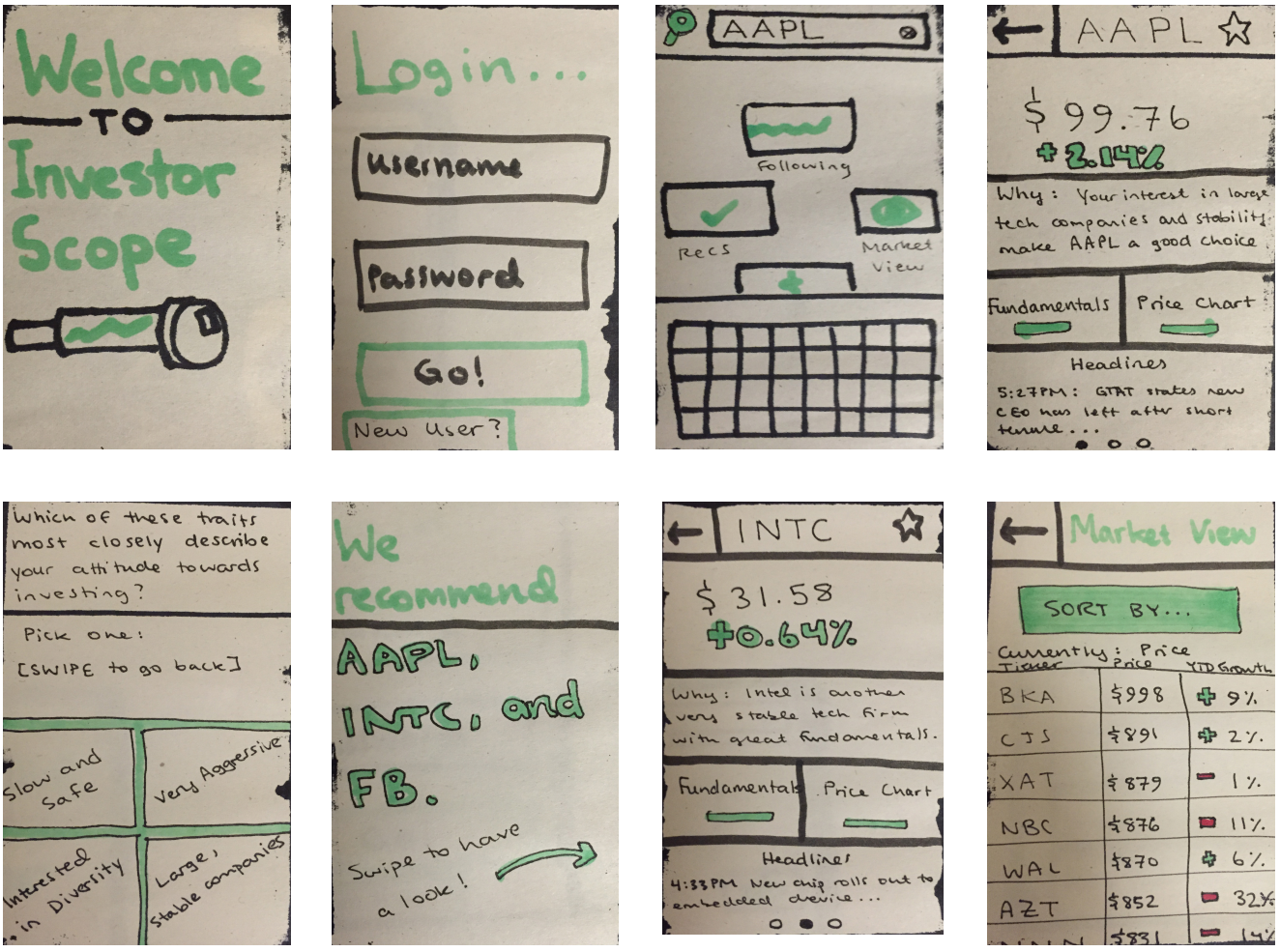



Messaging interface



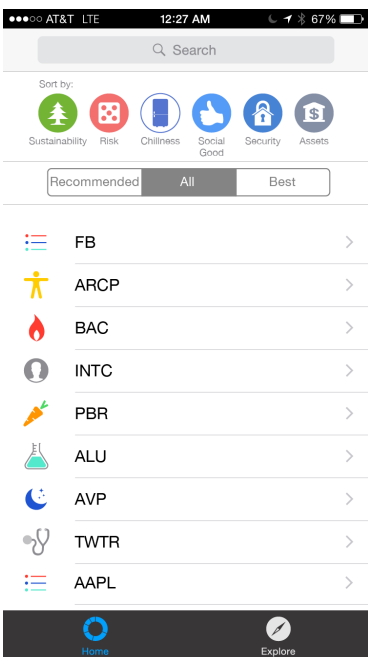

Stock page

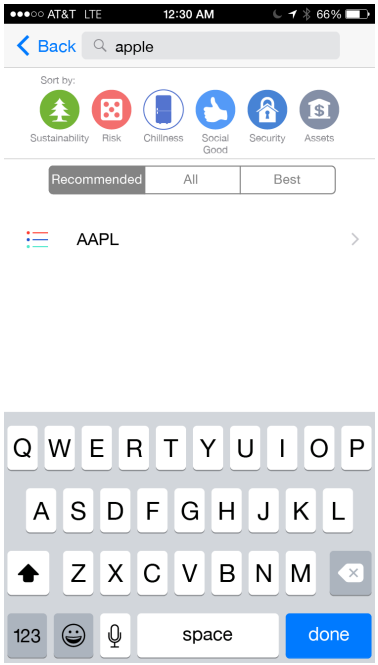
By the time we got to our low-fi prototype, we fleshed out the question and answer idea. We developed the prototype to answer a series of questions, chief amongst them, how would consumers react to our stock discovery process and paradigms (i.e. the question and answer recommendation engine and the sorting table). The actual nitty-gritty details were not important; we wanted to use the opportunity to see whether we had a concept worth using. Our results from our customer usability tests and interviews were generally positive on the function side, but we had to dramatically revise the form of the application based on customer feedback. They achieved the tasks prescribed rather painlessly but they gave us valuable critiques around the implementation of search, the nature of the homepage, and our sorting feature.

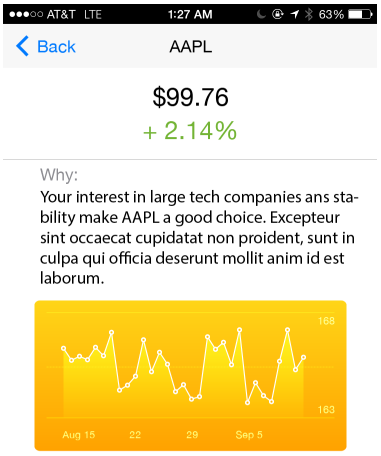


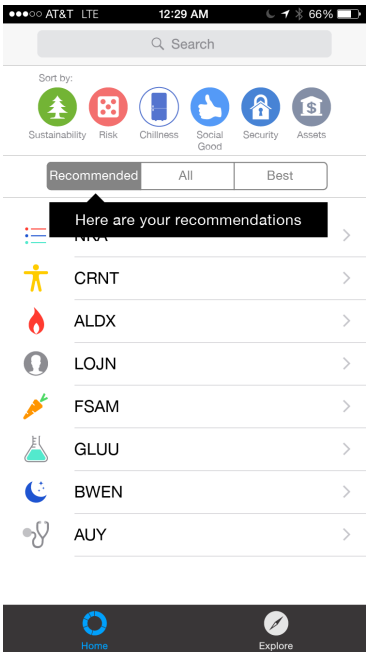

Low-fi prototype

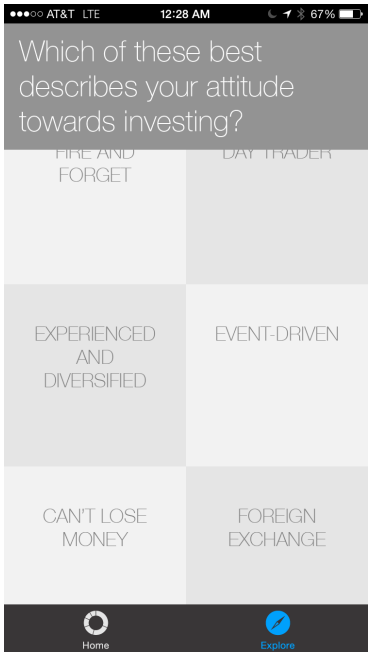
Our medium-fi prototype, which was designed with the lessons of the last prototype clearly in mind. We redesigned sorting, the home screen and the search function, and the prototype was subjected to a rigorous heuristic evaluation by four of our classmates. The results overall were focused on different aspects of user control and freedom (H2-3) and the consistency and standards of the application (H2-4) which are all described in detail above.

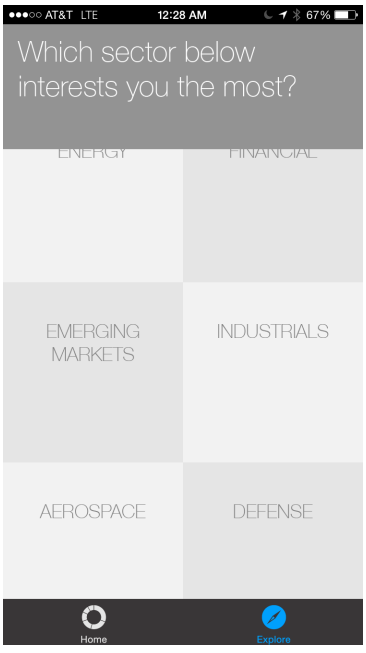


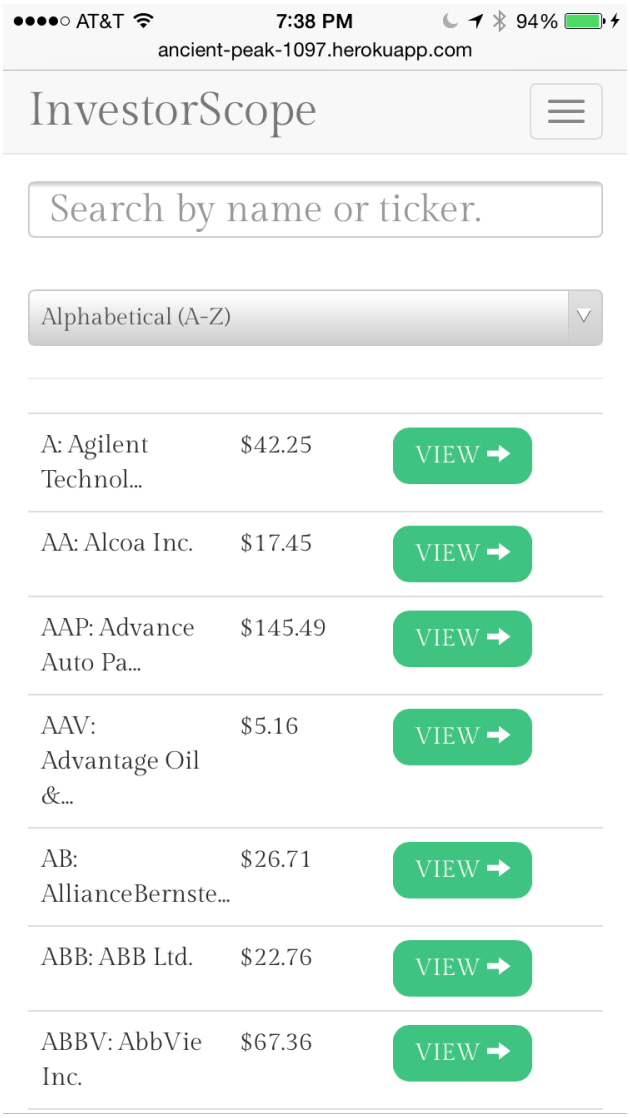



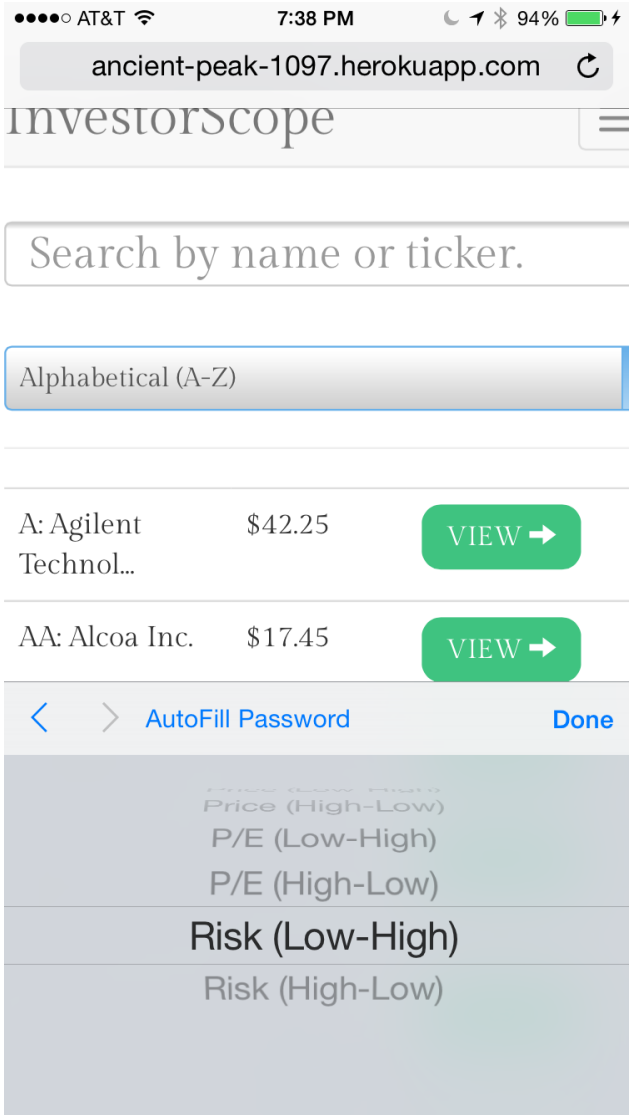




Medium-fi prototype

For the final iteration of our application, we took each and every comment we received from the evaluators above a certain severity threshold (3 and 4) and addressed each and every one of them (and then some). The outcome of this iterative, consumer-centric design approach is an application that has weathered heavy and probing criticism and has become better off for it. We designed an application that allows our customers to easily search for, discover and sort investment opportunities in a intuitive, seamless way. There are still myriad ways we can improve on it (as described in the missing features section below), but for now, we are very pleased with the core functionality and its implementation.

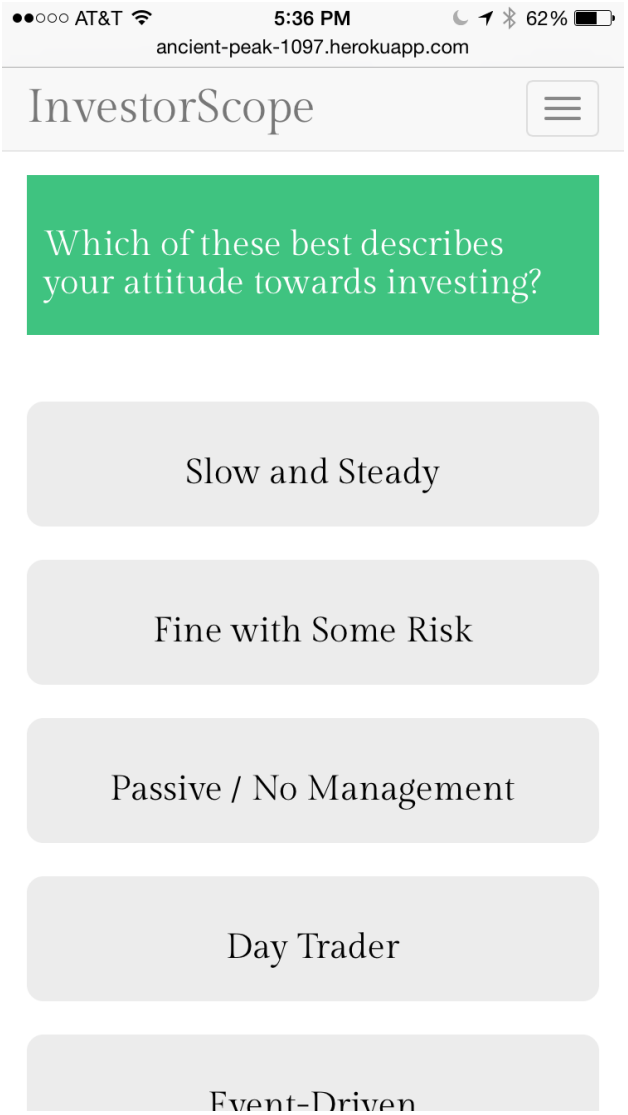



Home page




Categories


Stock page




Questions

**Prototype Implementation**

Tools

Our application is a mobile web application. The server is written in Python and uses the Flask framework to fill in HTML templates and present content to users. We scraped stock data off Yahoo! Finance and stored it in a MongoDB instance hosted on MongoLab. We used Bootstrap to format the content on the application, Chart.js to draw the stock charts on each individual company page, and JQuery to build the autocomplete functionality when the customer is searching for particular equities. Our application is deployed on Heroku.

Value of Tools and Methods

Constructing a mobile web app made it possible to write one application supported on multiple platforms (iOS, Android). Our target platform is the iPhone, but we are cross-compatible with browsers on other platforms as far as we are aware (we have not extensively tested other target platforms). The main problem we experienced deploying a mobile web app pertained to font and browser cross-compatibility. In some browsers, the font did not work when the site was live, so we were forced to improve the HTML's portability. Luckily, this was a fairly smooth process.

Flask is an intuitive, lightweight framework and both of the developers had experience using it. However, when we tried to integrate WTForms for login functionality, we encountered a lot of problems. Some of the documentation we read was out of date, and it took several hours just to get that basic integration working.

Heroku provided tools which made it easy to stage and deploy new versions of the application and run it locally using the same setup. MongoDB was helpful because it allowed us to use heterogeneous data (most of which was scraped online). It has convenient API integration with Flask (MongoEngine) and also allowed for us to easily create randomized data for some of the metrics we generated using algorithms in the Mongo shell. One problem we encountered was that MongoLab has obscure and uncommon permissions for replicating and backing up databases. For this reason, we had to make an additional experimental database on a local machine and then run our algorithm on live data after proving it in that sandbox. This was frustrating, but was really the only downside we experienced with MongoDB + MongoLab. Aside from that, it was easy, fast and free.

Bootstrap is a familiar, proven tool, and for the basic layout of the site, it worked perfectly. We considered adding a button to the navigation bar, but found that it's not readily customizable and is actually fairly rigid. After a while, we decided that using the traditional navigation bar functionality would work better, but initially this was a source of frustration. While we considered other CSS frameworks such as Foundation CSS, Pure CSS, and Skeleton CSS, we ultimately went with Bootstrap, primarily due to more easily available documentation.

Chart.js made it simple to generate a plot of the stock price, but also had some deficiencies. We wanted to show more than 7 data points for past stock prices, but Chart.js showed a label on the X axis for each point we displayed. Chart.js also showed multiple tooltips when points were spaced closely together. Correcting these problems would have required editing Chart.js source code, and we could not justify spending a lot of time on this.

We used Git and GitHub for source control. This worked well because we could also use Git to deploy our application to Heroku.

Hard-Coded Data

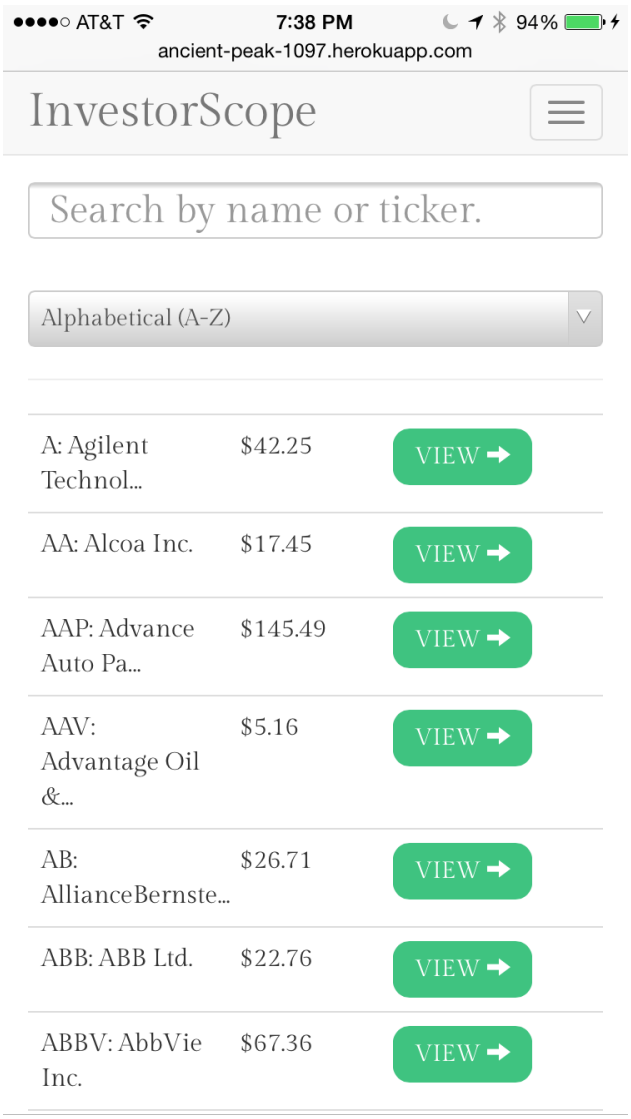
The proprietary engine for recommending stocks actually just returns a hard-coded list of the same stocks every time (please don't tell our customers!) because our desired outcomes was not to create a recommendation algorithm but instead simply deploy the feature itself. The questions and answers on the Q&A page are also hardcoded, but not randomly selected. All other data is either scraped from online sources (such as for the stocks) or generated using approximation algorithms (such as the charts). There were no Wizard of Oz techniques used.

Missing Features

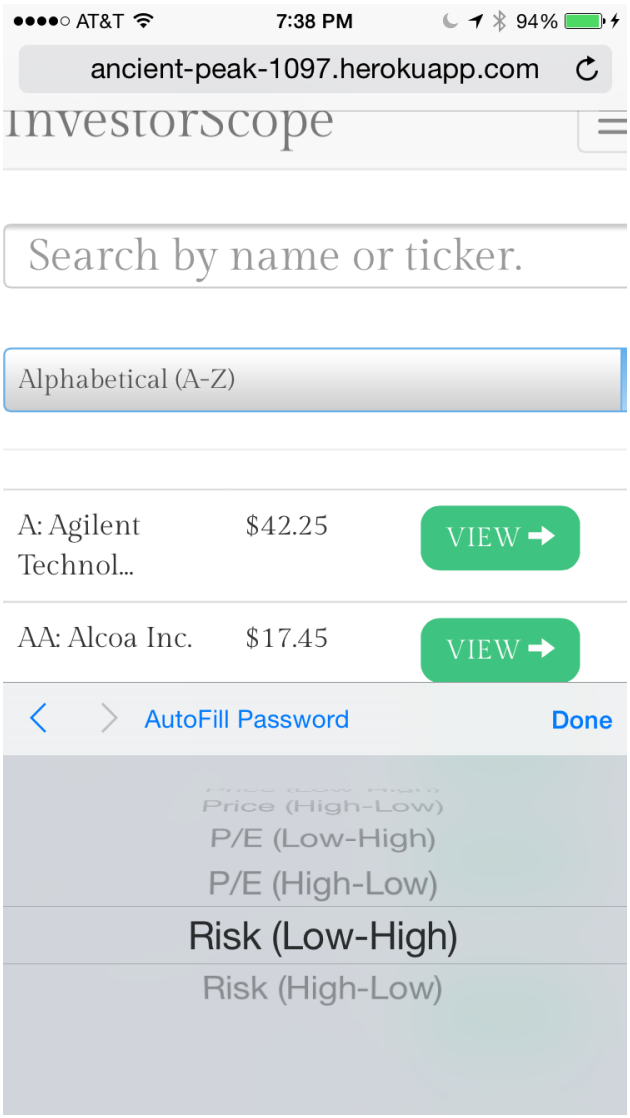
There are obviously a number of features we could add in the future. Adding user accounts, allowing customers to execute stock trades using the service, allowing customers to save particular stock picks, and building out a machine learning backend to deliver real recommendations are all ways in which we can enhance our application in the future. Additionally, we could create communities for our users so they could interact and share information with friends, and also give users more control in setting their preferences (such as for risk, frequency of questions asked, etc.). For now however, our implementation is very much functional for the tasks we focused on, and we are pleased with the features we have created.

**Appendix**

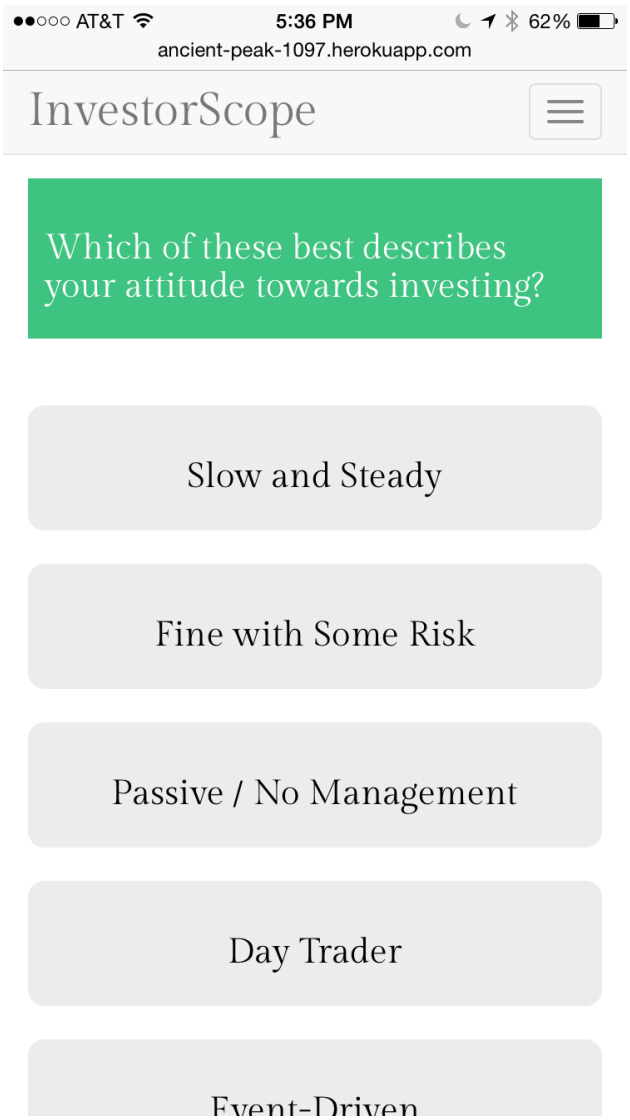
1. Current Prototype




1.1. Home




1.2. Sorting



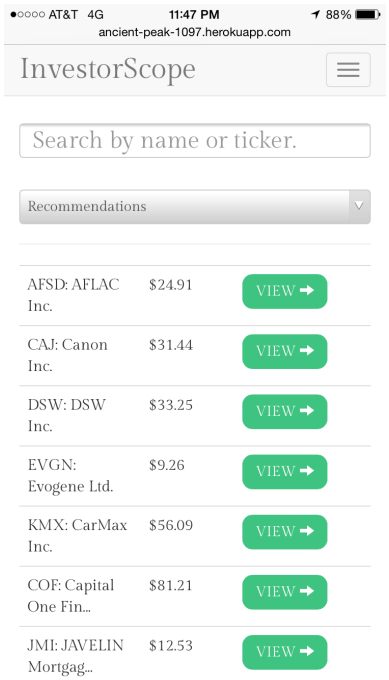

1.3. Recommendation Engine


1.4. Stock Page

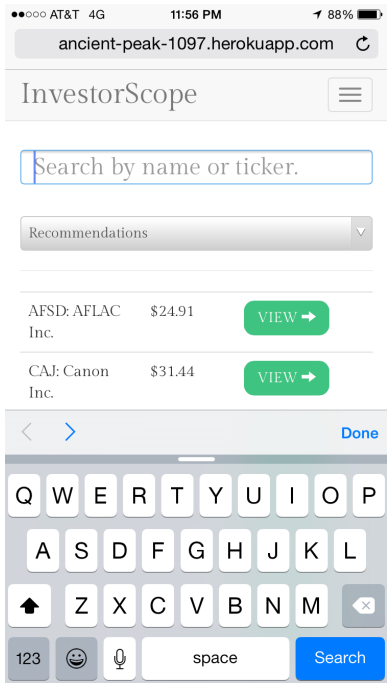
## 2. Task Storyboards

### First Task: Search

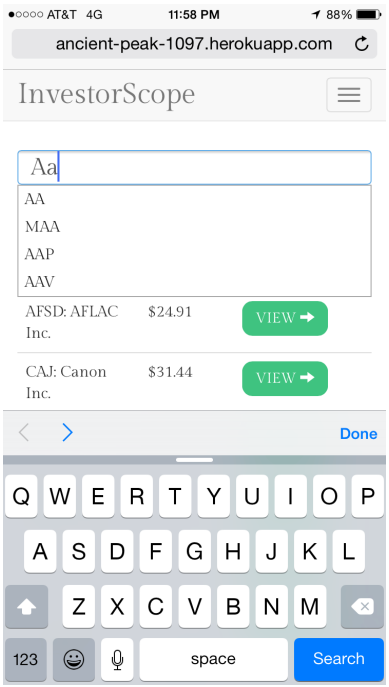
1




2



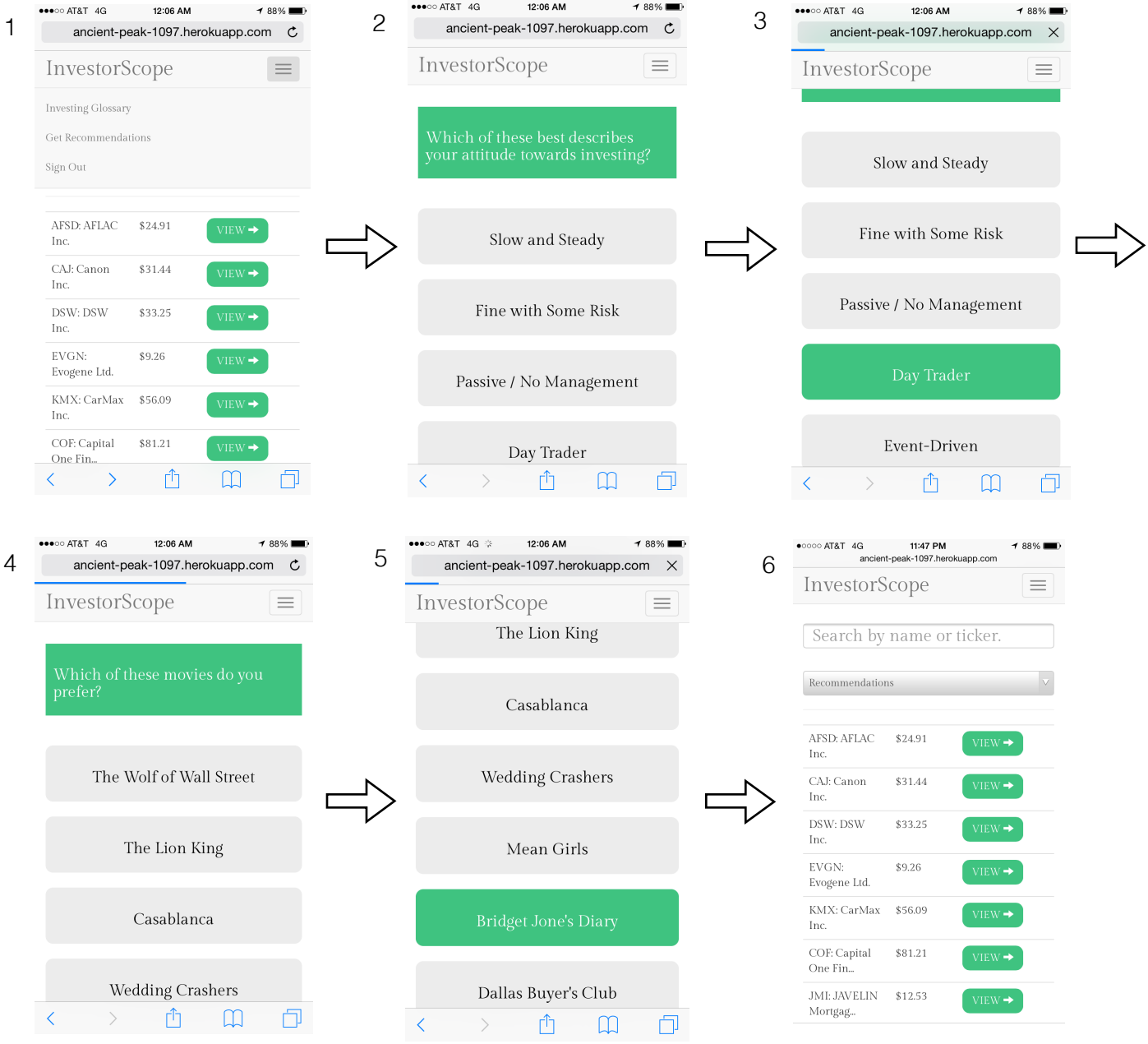

3




4


Customer clicks search bar, enters query, and arrives at relevant stock page.

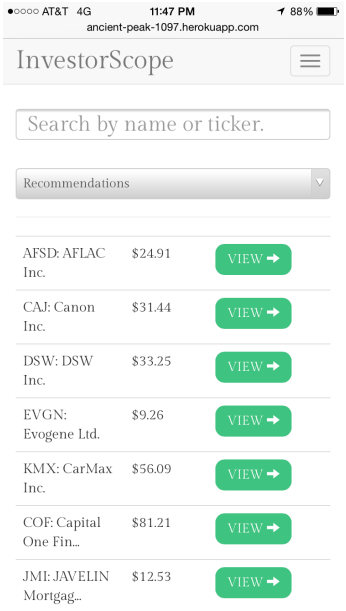
## Task Two: Recommendation Engine



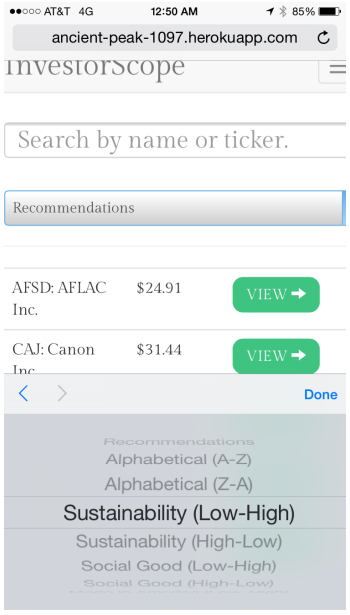

Customer clicks on “Recommendations” from menu, and answers series of questions, arriving at customized recommendations.

## Task Three: Sorting

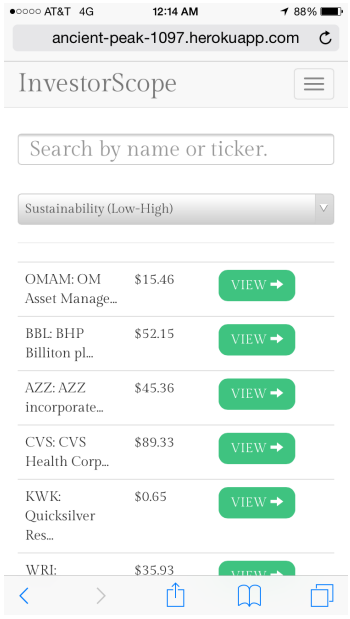
1




2




3

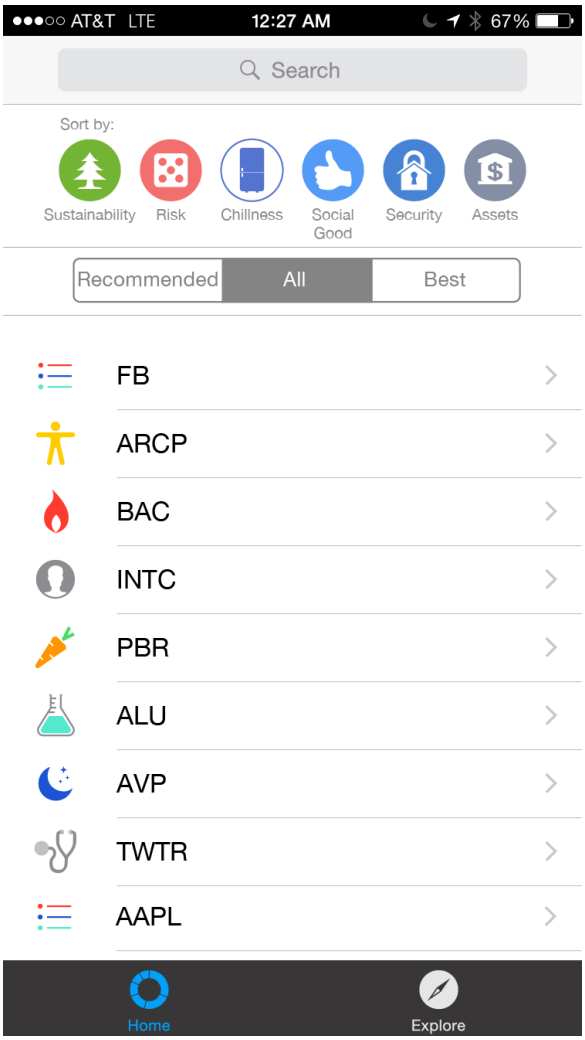



4

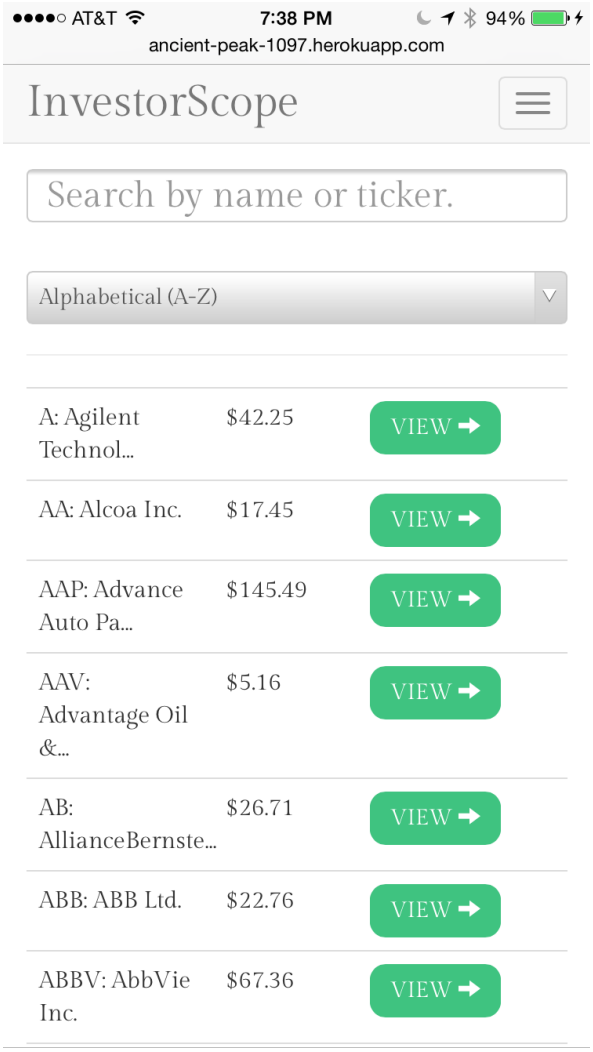

Customer clicks on menu and selects relevant category. Once customer sees rankings, he or she can select relevant stock to learn more about it.

## 3. Compare and Contrast

### 3.1 Home screen

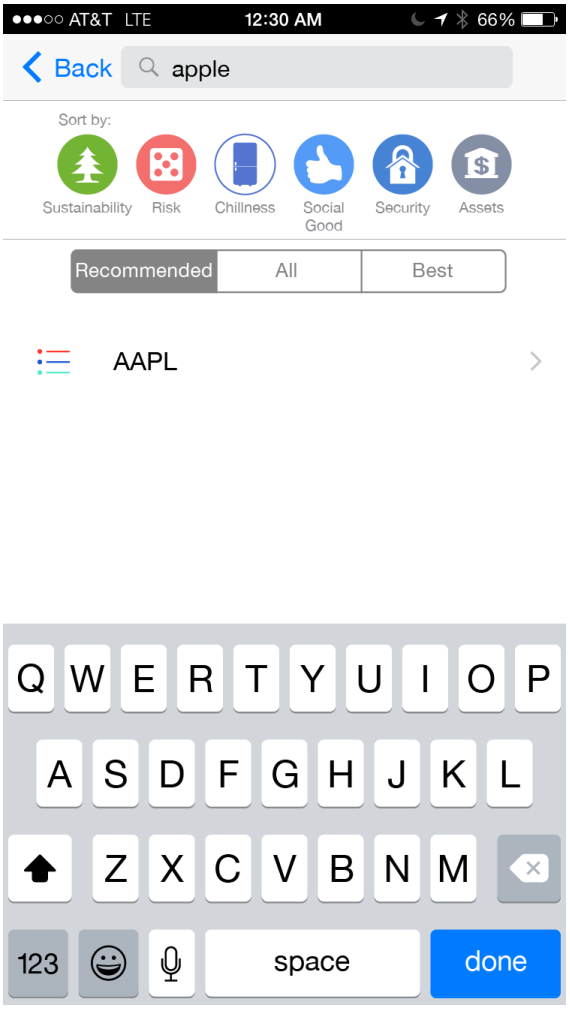



Before

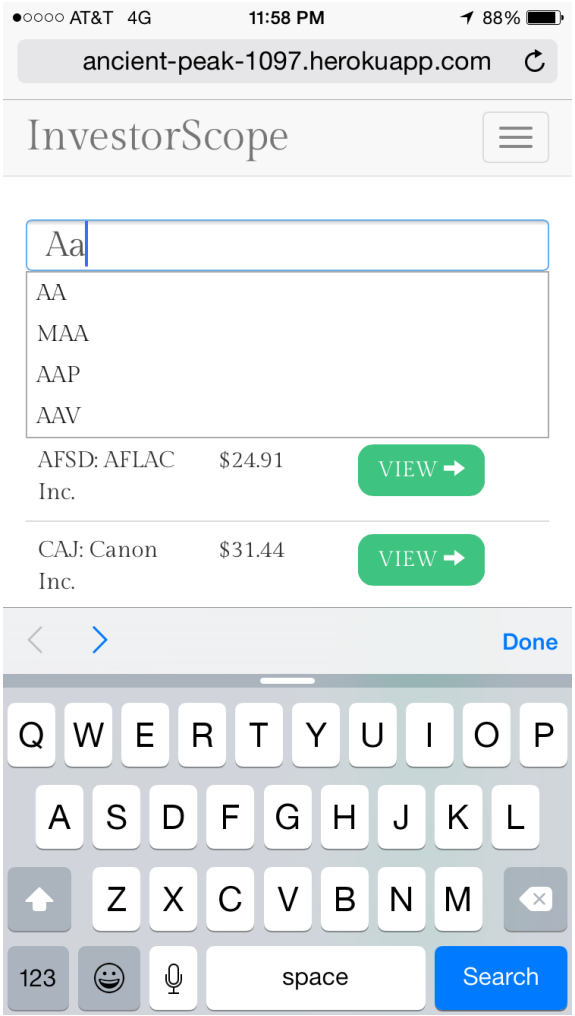



After

## 3.2 Search

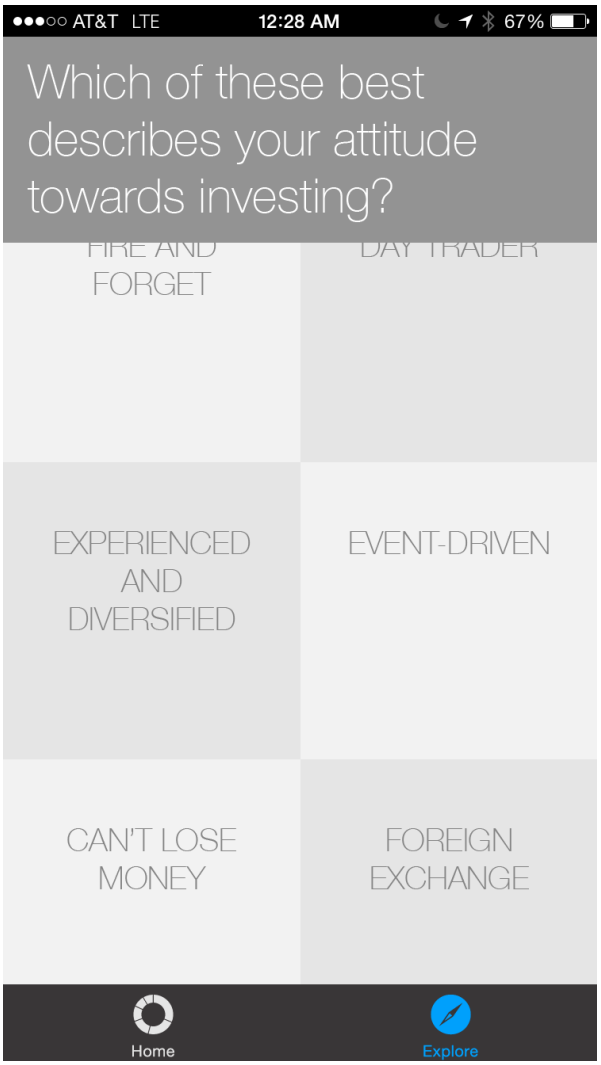



Before

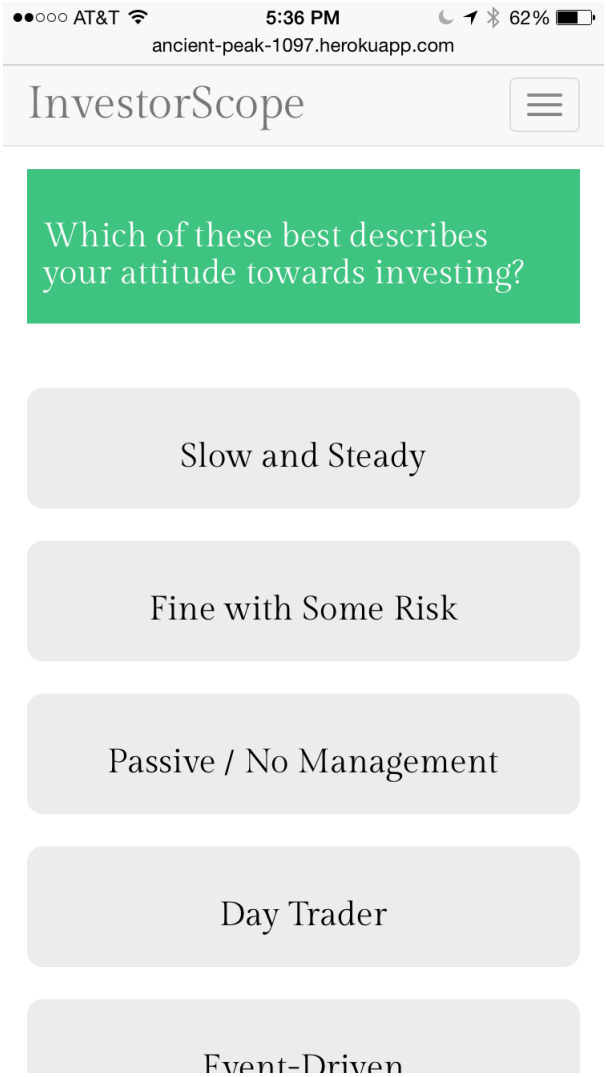



After

3.3 Questions

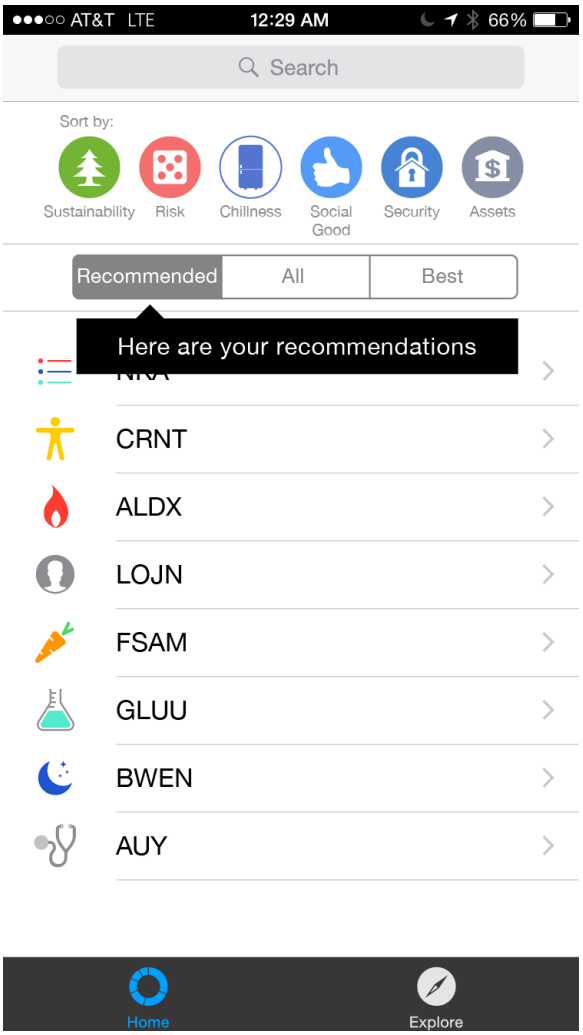



Before

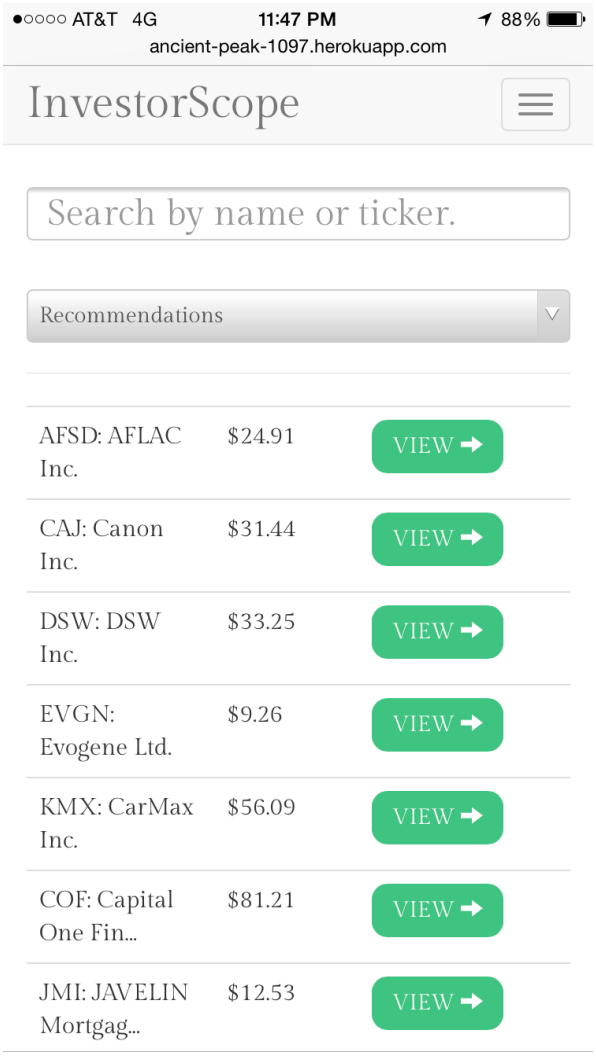



After

## 3.4 Recommendations

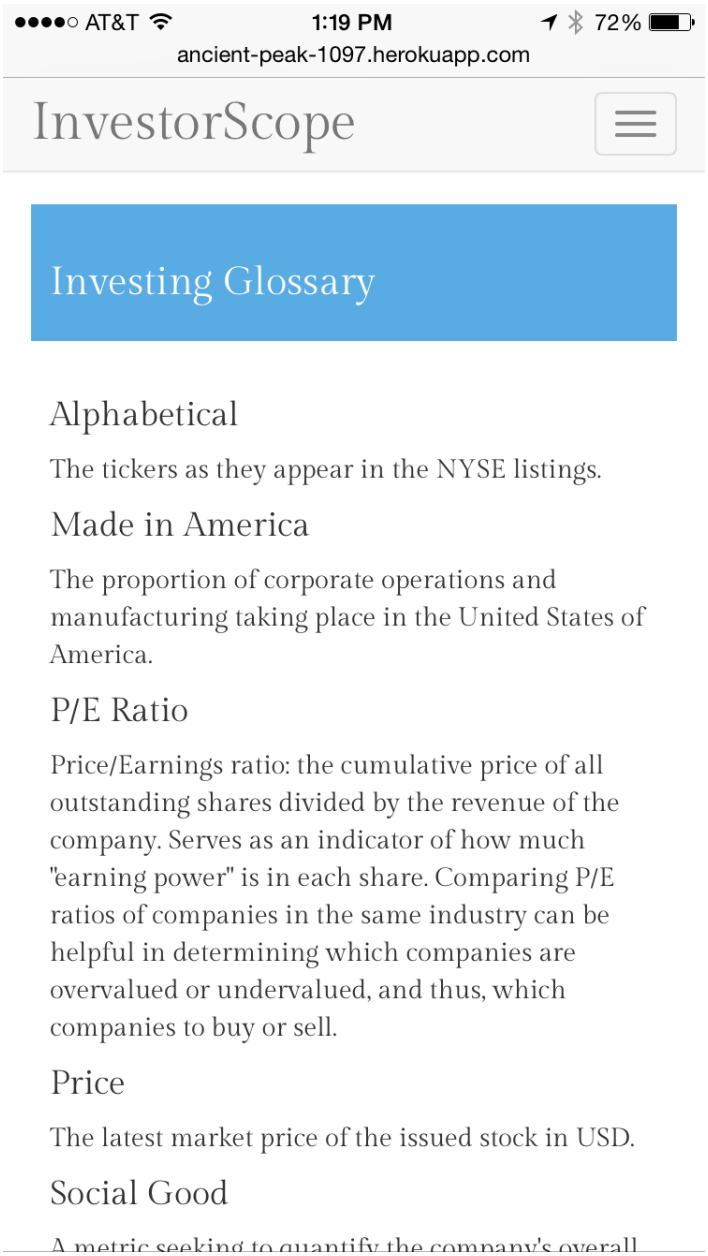



Before

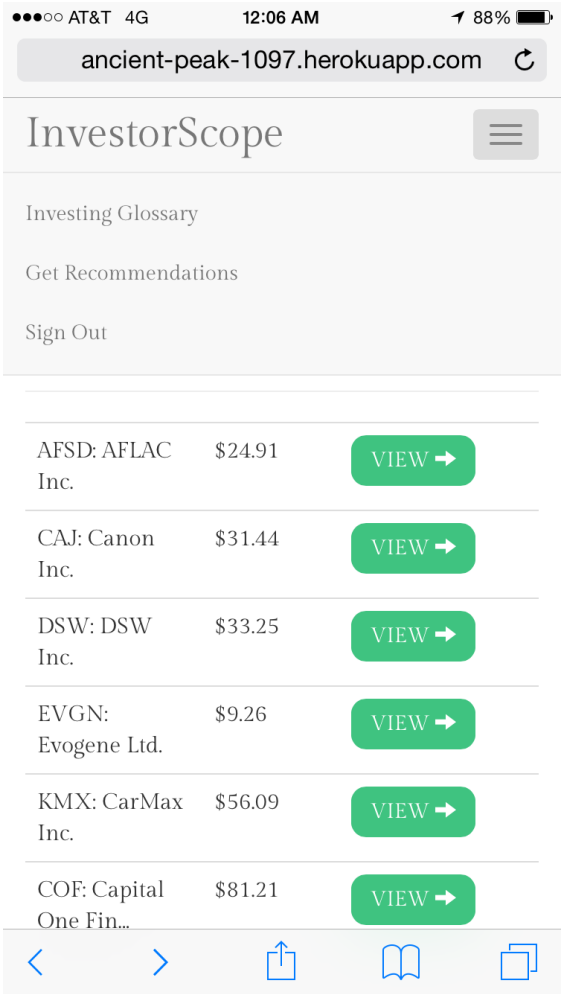



After

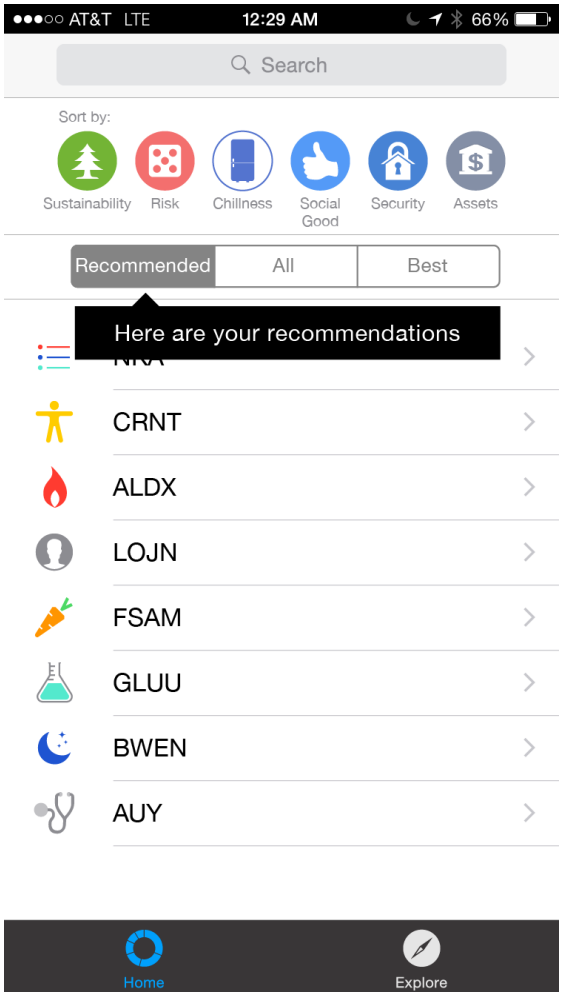
3.5. Investing Glossary



## 3.6. Menu




Before




After