USHER

shape your day *on-the-go*

High-fi Prototype & Final Report

Asad Khaliq	**Edwin Park**	**Ji Park**	**Quentin Perrot**
Team Manager	Development	Design	Documentation User Testing

Bringing travellers the perfect on-the-go companion to guide them through a day balanced with planned activities and spontaneity

Problem and Solution Overview

Through our initial contextual inquiry, our team detected an inefficiency in the way people keep track of the things they want to do in the future. People are constantly being recommended things to do and explore but existing methods of keeping track of that information do not match the way we actually behave in the real-world. For example, I write something on a sticky-note at home but can't access it when on the move. To solve this problem, we reimagined the way people keep track of the things they want to do: a platform that allows you to add things to a wish-list on-the-go and that recommends activities at the right time and place. However, delving deeper into the psychology of the matter we realized that the nature of recommendations varies too widely - from a good book to a restaurant - and as a result, users would just *snooze* on recommendations. So, we pivoted to a space where mood is less of a factor: travel.

Travelers are often - if not always - in the mood to *discover.* However, just like before, travelers feel like they're not getting the most out of their on-the-ground experience. They want to be able to do the must-do's and easily deviate from the road most travelled. Today, there is no such service that allows travelers to shape their day on-the-go. Usher situates itself in this space as a service that allows you to keep track of the things you want to do and readily shape your day's trajectory depending on your changing mood. In this way, travelers find a perfect balance between planned activities and spontaneity. Mirroring our initial platform, Usher uses its recommendation engine to take the broad list of things you want to do and recommend the next thing as you explore. *Pick up your phone in the morning, and let us usher you through an exploratory path that fits your personal preferences.*

This on-the-go experience is mirrored as much as possible in the application's design. Choose an activity, explore and repeat.

Tasks & Final Interface Scenarios

Let me first shortly describe the evolution of our tasks. Originally, our tasks existed but were not well sewn together: add, view your list, get recommendations. In our latest iteration, however, the tasks are more organically connected. For example, you are now able to affect your recommendations by manipulating your list. These organic connections were influenced by a desire to give the user more *control.* Additionally, the nature of our tasks has remained quite consistent from the beginning, but as you will notice, each task has gained in complexity. Below we will explore each task and its place within the story that is *Usher*.

Task 1: Recording the things you want to do (simple complexity)

This is the first thing a user will want to do and the foundation of our application: keeping track of the things you want to do. When someone recommends something to you, you're able to quickly and fluidly add this activity to your *list*. To make this as easy as possible, we've made it possible for users to add right from the Home page. To increase

app-flexibility, users are also able to add from their *MyList*. For these reasons, this task is *simple* for users. We think it is important to precise that adding an activity is not a typical 'browsing' experience. Users cannot browse through potential activities; instead, they must search for an activity they know they want to do. The motivation behind this is dual: (1) we consider these two things to be completely different services and thus, (2) we want to focus on actively shaping your day and not browsing beautiful activities.

A quick word on implementation. This task requires scraping possible activities off of the popular travel application TripAdvisor. This allows users to input a possible activity through search, and related queries to show up.

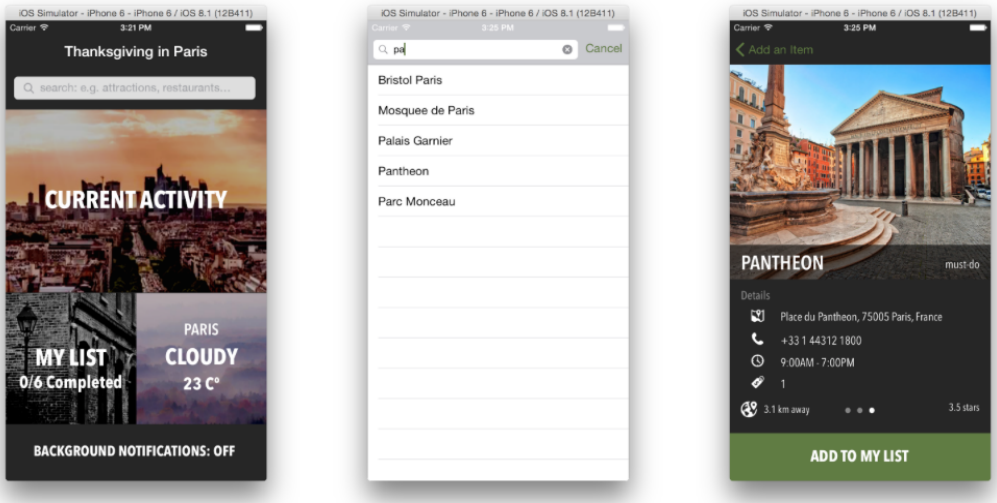

Figure 1. A snapshot of each main task. In order: add, view and organize, recommend

Task 2: Organizing and Viewing Data (moderate complexity)

In *MyList*, users are able to view the activities they will want to do in a particular destination. In past prototype iterations, this list had little function and served as a place for users to get a high-level view of possible activities without being filtered by our recommendation engine. The only existing function was the ability to search through the list. As a result of our heuristic evaluation, we have revamped this task. Users are now able to add new activities directly from the list. They are able to mark events as done. Users are able to click on entries and see a beautiful representation of the activity. Our most radical change gives users the ability to affect our recommendation engine. Directly from *MyList*, users can rank activities by easily dragging items up and down. Higher rank (position on the list) will weight this item more heavily and will be recommended more highly by our engine. As you can see, *MyList* is no longer functionless and actually acts as a vital link between recorded information and our recommendation engine - effectively unifying our tasks and giving cohesion to the overall user experience.

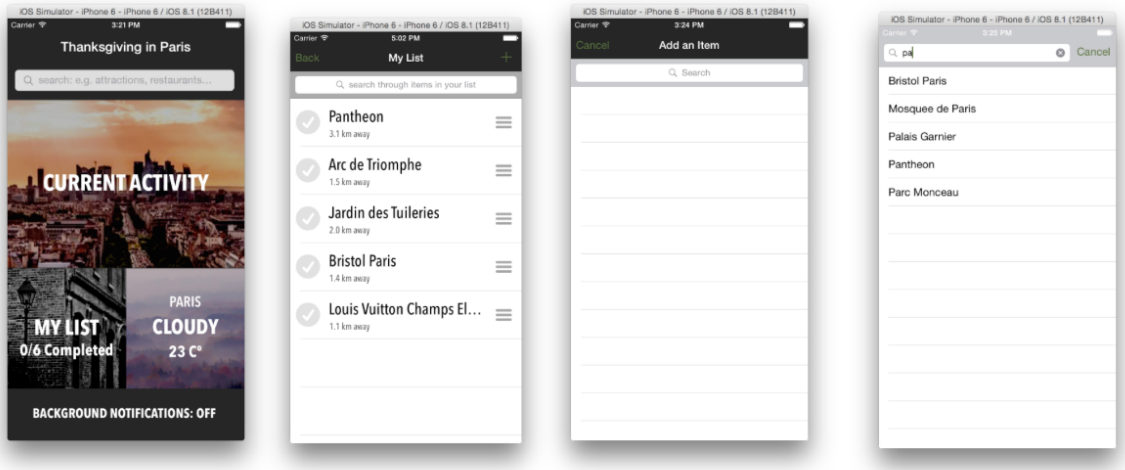

Figure 2. Task 1 or Recording: add from Home Page or My List, search and add

Task 3: Deciding what to activity to do next (complex complexity)

This task has been the most challenging for our team, as it has required us to really understand the *traveler mentality*. How do travelers decide what they want to do on a given day? It turns out that travelers have an overarching daily aim to *explore some number (around 3) of local must-do's.* On top of those things, travelers want to step out of the 'typical path' and explore spontaneously. This knowledge of traveler mindset led us to our current task design; here is a brief overview of what this task entails. When a user first opens the app at the start of their day, they choose a few must-do activities. Then, they pick their first activity based on our recommendation engine. Users are recommended activities that are (1) must-do's selected at the start of the day, (2) activities nearby or (3) activities that travellers with similar lists also want to do. Because our recommendations depend on a user's current activity, the Home Page beautifully keeps track of the current activity; this visibility of the underlying engine's status is vital.

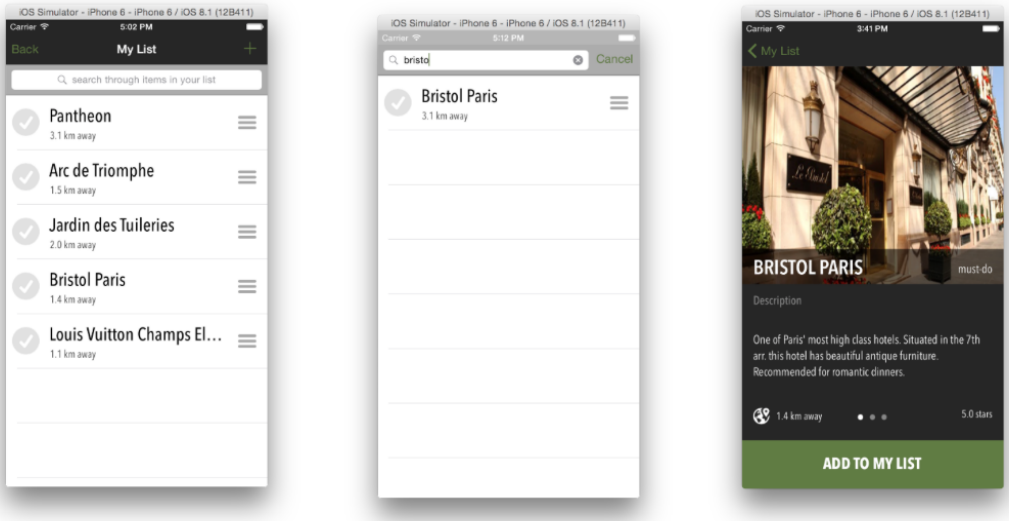

Figure 3. Task 2 or Viewing: View My List, search and set as current activity

Major Usability Problems Addressed

Our heuristic evaluation identified 33 problems with our medium-fi prototype. Although a number of these problems were related to limitations with the prototyping tool (inVision), many of the problems found were actionable. Here, we will look at each problem with severity 3 or 4 and explain if a change was made, what the change was, and what the reason was. **Bolded items** refer to problems that affected a change in our design. *Italicized items* refer to problems that did not affect a change in design.

1. **It is unclear what a user is supposed to do on *Start your Day* page (Severity: 3)**
   The user is given a list of activities, but it is not clear what to do. For a first-time user, there are no instructions telling the user to select a few must-do's for the day. To remedy this, we change the title of the page to "Select your daily must-do's" to guide the user.

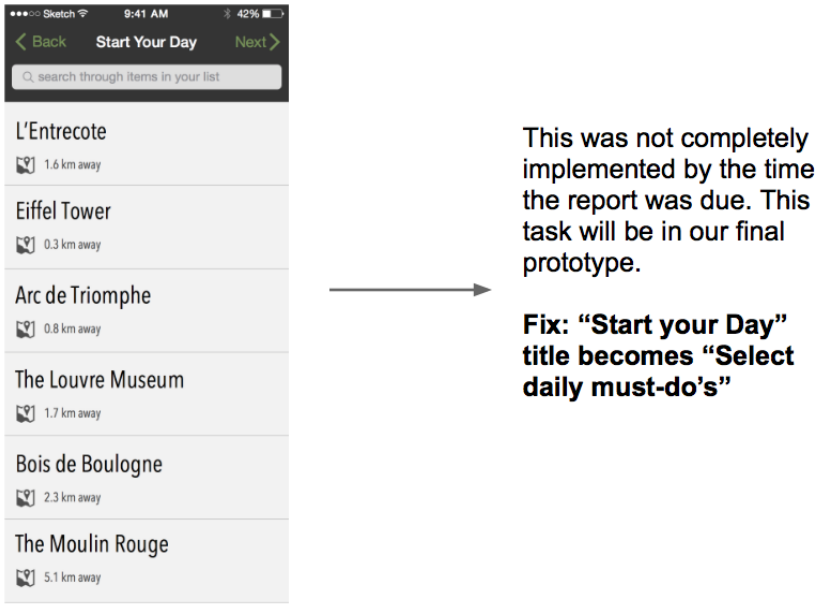

Figure 5. Start your Day page changes

2. **It is unclear what "Next" means on *Start your Day* page (Severity: 3)**
   After selecting must-do's, a first-time user will not know that the next step is choosing the day's first activity. In order to make this explicit, we change the "Next" button to "Choose next activity," so the user will know what to do on the following page.

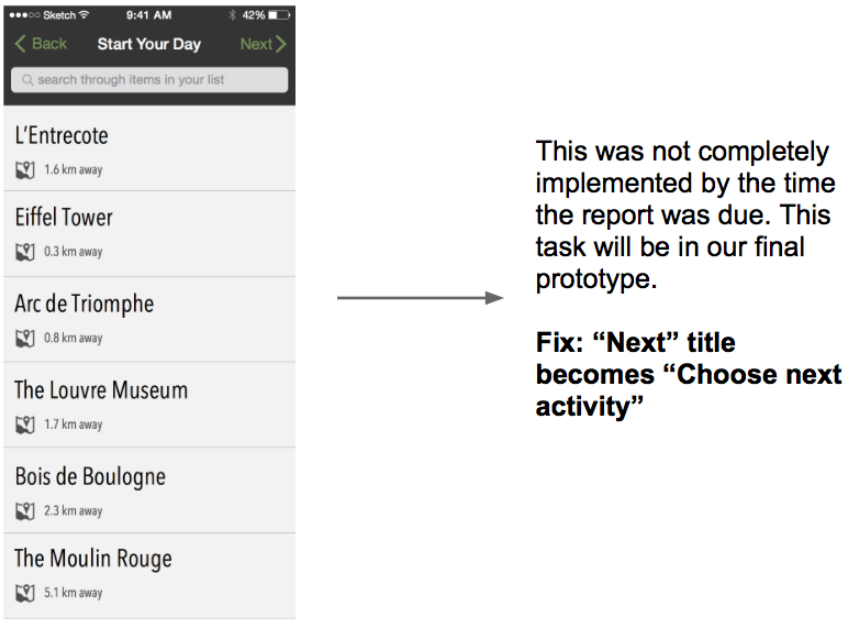

Figure 6. Start your Day transition changes

3. **Star rating on an activity's main *View Page* are poorly designed (Severity: 3)**
   Our previous iteration left it open to interpretation if a user was able to vote on how many stars to give to an activity. Our intention for this rating was purely descriptive. To remedy this, we will have numbered rating because empty stars indicates that a user is able to vote.

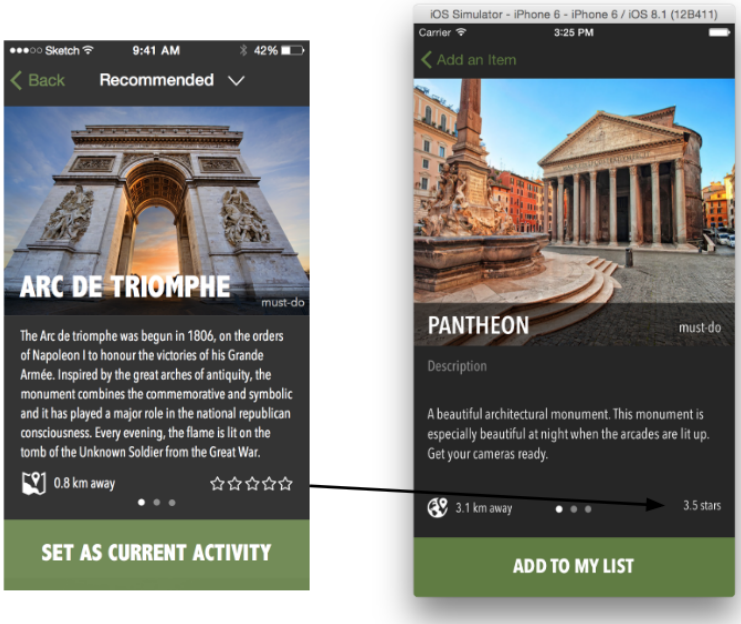

Figure 6. Change in design of star ratings

4. *On an activity's View Page, the prototype changes both image and information concurrently (Severity: 3)*
   This was a limitation of our prototyping tool. We were not able to implement multiple swiping gestures for the same page, so both picture and information had to change simultaneously. In our final prototype, image and information are actionable by swipe i independently.

5. **You cannot retroactively add recommendations made by our engine on the *Recommendations Page* to *MyList* (Severity: 4)**
   On our previous prototype, you are not able to add recommendations that arrive on *Recommendations Page* to *MyList*. This is not a problem if all recommendations come from *MyList*, as is currently the case. However, later iterations will match similar user lists and recommendations will be sourced from these 'matches.' When this is the case, an option to add to *MyList* will be needed.

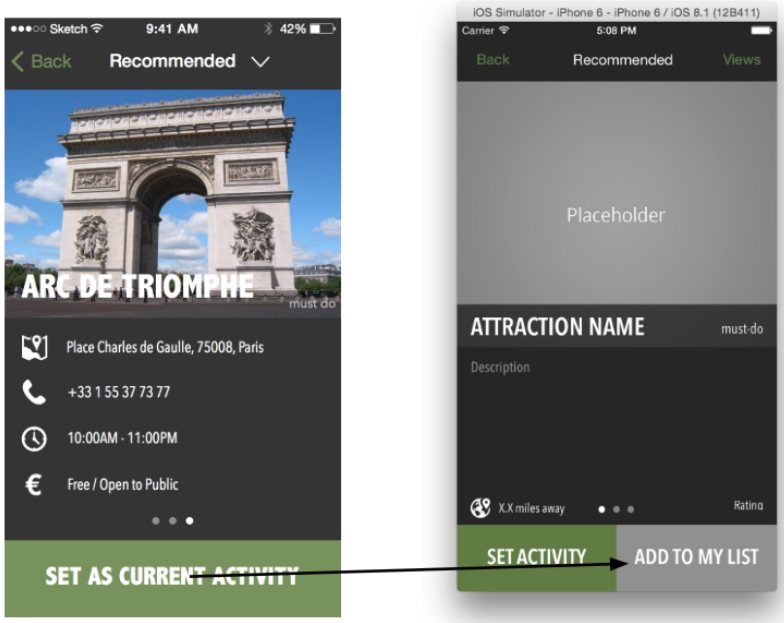

Figure 7. Option to add from Recommendation available

6. *When two recommendations items are present on the same screen, there will be confusion as to which item will be set as current activity (Severity: 3)*
   This was a limitation of our prototyping tool. InVision did not allows us to have 'snap' swipes, and so multiple activities could be present on the same page. In our next iteration, only single activities can be present at any one time. This means there will be no confusion as to which activity is being set as 'current'.

7. **Current price indications are subjective; “moderately expensive” can mean different things to different people (Severity: 3)**
   This is without doubt a problem: our system is not speaking the user’s language. To remedy this price subjectivity, we are using standardized Yelp numbered prices where applicable and available.

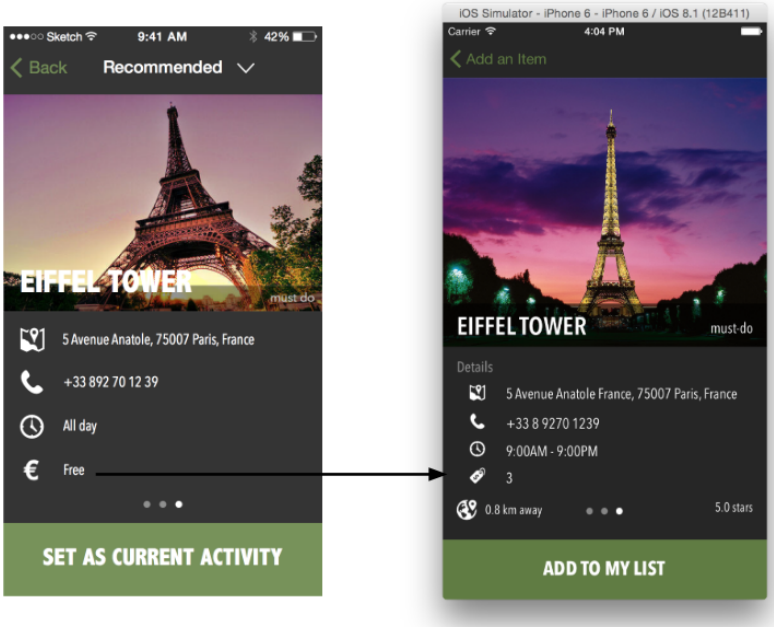

Figure 8. Price description change made

8. **Users will complete activities without using the application, and so activities will be recommended despite being completed. There is no option to complete without setting the activity as current (Severity: 3)**
   The evaluator suggests that current activity should automatically change after you leave the location. This is not a feature of our application (very advanced implementation and would drain battery life rapidly), but raises major problems with our prototype: users are not able to complete an activity without setting it as “current activity.” We are adding an option in *MyList* that allows users to complete an activity retroactively.

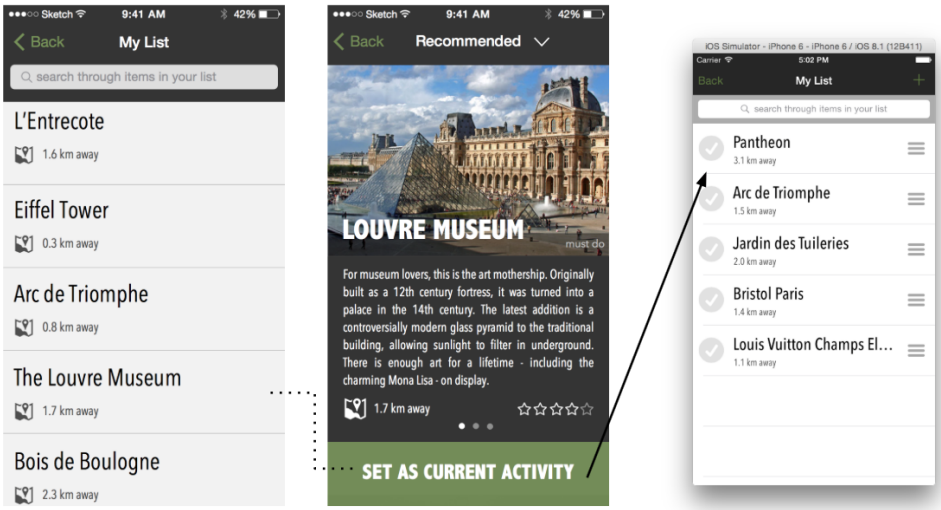

Figure 9. Complete an activity retroactively

Now we will analyze some problems found in the heuristic evaluation that have lower severity ratings of 1 or 2, but that we deem remain important to our design development.

9. **Search bars are used across several pages (*Home Page, MyList)*, and what information is being searched is unclear (Severity: 1)**
   The evaluator suggests that because we have more than one search bar, certain users will be confused by what each search does. For example, HomePage search goes through an exhaustive database of activities of a certain destination whereas *MyList* search goes through *your* activities only. We can amend this by having more informative standby messages within the search bars.

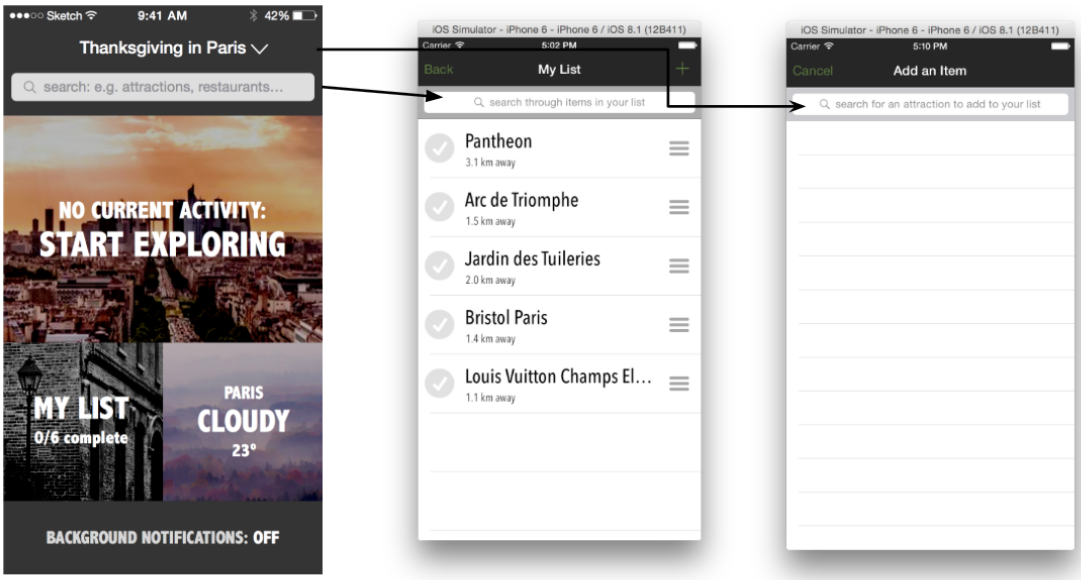

Figure 9.1. More informative search bar standby messages

10. **The *MyList* item ordering is unclear.**
    In our previous iteration, we had put no thought into ordering. In our next iteration, ordering is extremely important because it helps determine recommendations. As explained in Task 2, a user is able to rank *MyList* items. To make this abundantly clear, we believe we need a tutorial. This tutorial will be prompted by a question mark icon on the page.

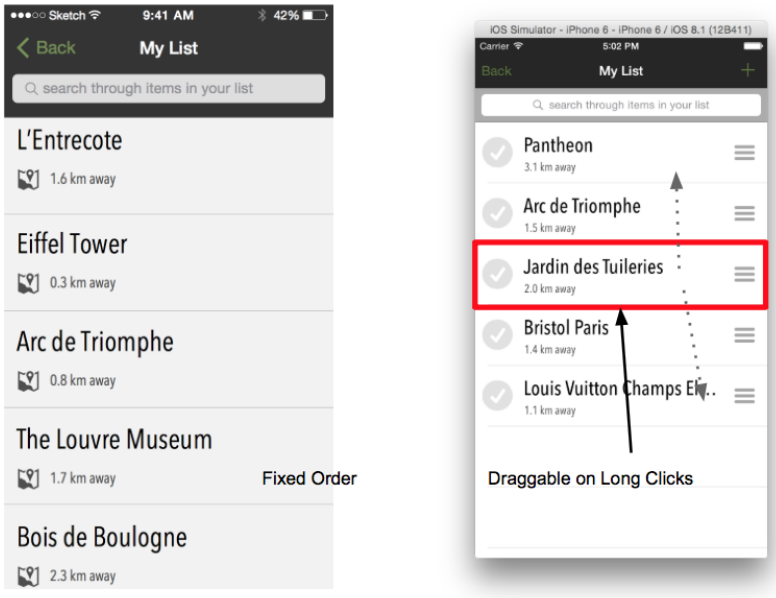

Figure 10. Move to a rankable list

11. **It is not clear where items added from the *Home Page* go. Certain users will be confused.**
    This is a very good remark. In our previous iteration, adding activities to *MyList* from the *Home Page* and from *My List* had different designs. This could lead users to question if adding from each adds to different lists. To remedy this problem, we will standardize the ‘add activity’ experience across the application.

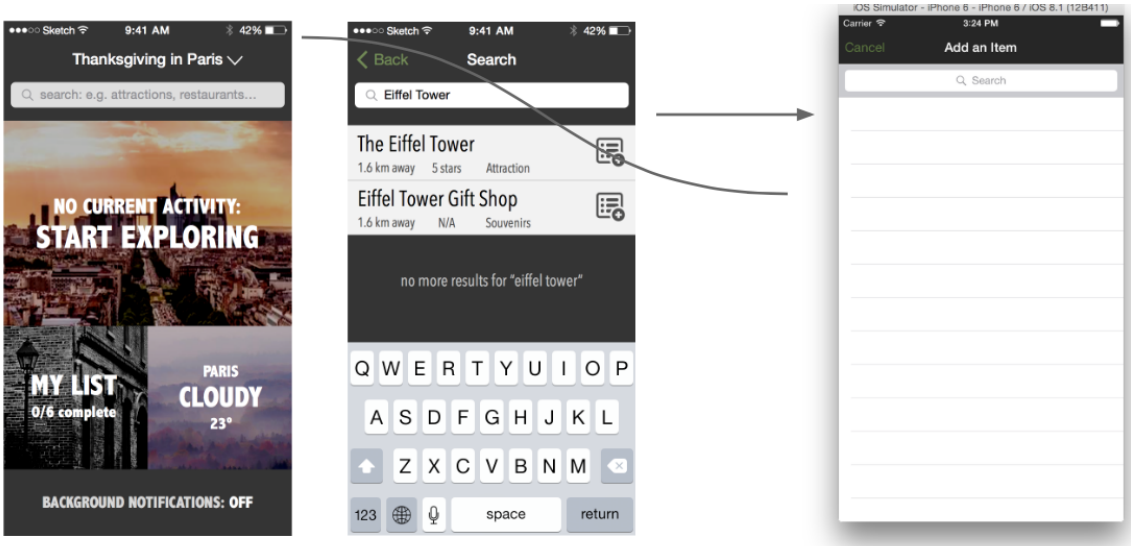

Figure 11. Standardize the adding to MyList experience

Building on the intuition received from the heuristic evaluation, our team also made some separate changes. As before, we will look at each problem with severity 3 or 4 and explain if a change was made, what the change was, and what the reason was.

12. **Although we are keeping track of how many activities have been completed, the user does not know *which activities* have been completed**
    The *Home Page* contains a fraction of *MyList* activities completed, but users are not able to see which activities are completed and must remember. In *MyList*, we add an icon that marks if the activity has been completed or not.

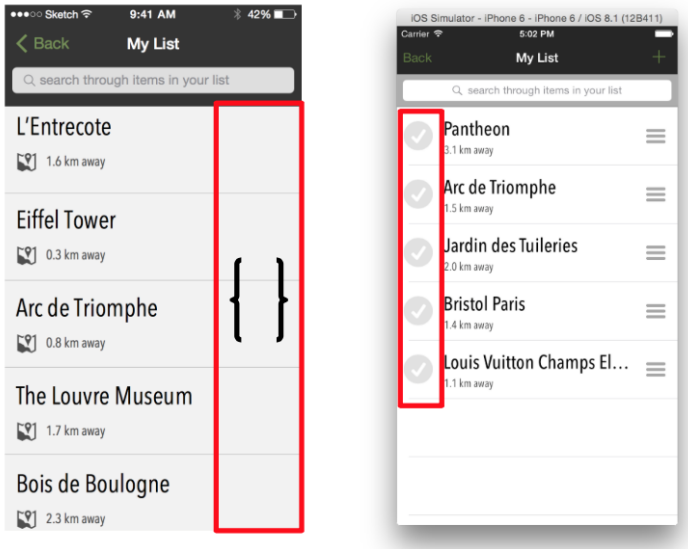

Figure 12. Changing MyList UI to reflect if activities are complete or not

13. **We had a black shade bar on each picture in Recommendation view highlighing the *type* of recommendation (must-do, list, match) but the item title was poorly higlighted and sometimes contrasted with the background picture**
    We redesigned the black highlight to incorporate both type and title of the item. The result is aesthetically pleasing and more functional.

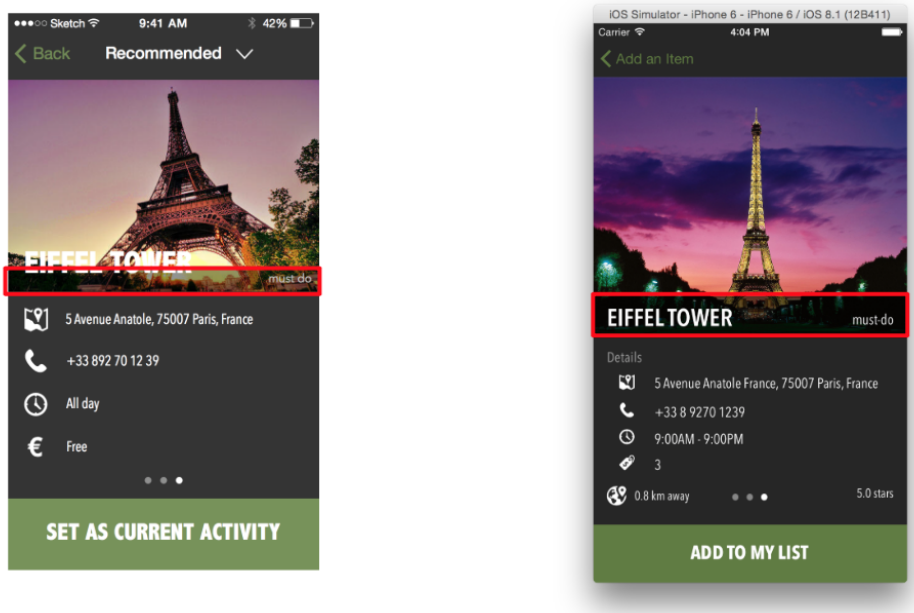

Figure 13. Extending the black-shade bar to highlight title

Design Evolution

Our high-fi prototype is the result of many design iterations. Our process followed the typical design thinking procedure. Here, we will discuss the path taken leading to our final prototype - from initial contextual inquiry to now.

Contextual Inquiry

Very early on in our design exploration, we were able to identify an unsatisfied user need by interviewing a diverse set of individuals. By determining what the user problem was - *difficulty keeping track of the things you want to do* - we decided to delve further into the problem and conducted further interviews using the Master Apprentice Model. In this round, we were able to discover various problems with the tools people use. The most memorable of these was the fact that the *recording process is scattered across multiple mediums (sticky notes, notes app, Google Docs) and disconnected with how people need to access that information during the day.*

The problem identified was multi-layered. People need a central location to record things they want to do, but they also need a process that reminds them to do these things at the right time and place. Through our task analysis, we were able to identify 3 major desired tasks:

1. Record and organize past recommendations
2. Easily remember and follow up on things, events and places that you want to discover
3. Discover these new things with friends

From this point, our team *flared out* by developing quick divergent solutions: a Google Glass picture storer, a mobile notebook and an **automatic recommender**. Quick sketches allowed us to quickly explore crazy ideas and slowly move towards more plausible alternatives, as seen in Figure 14.

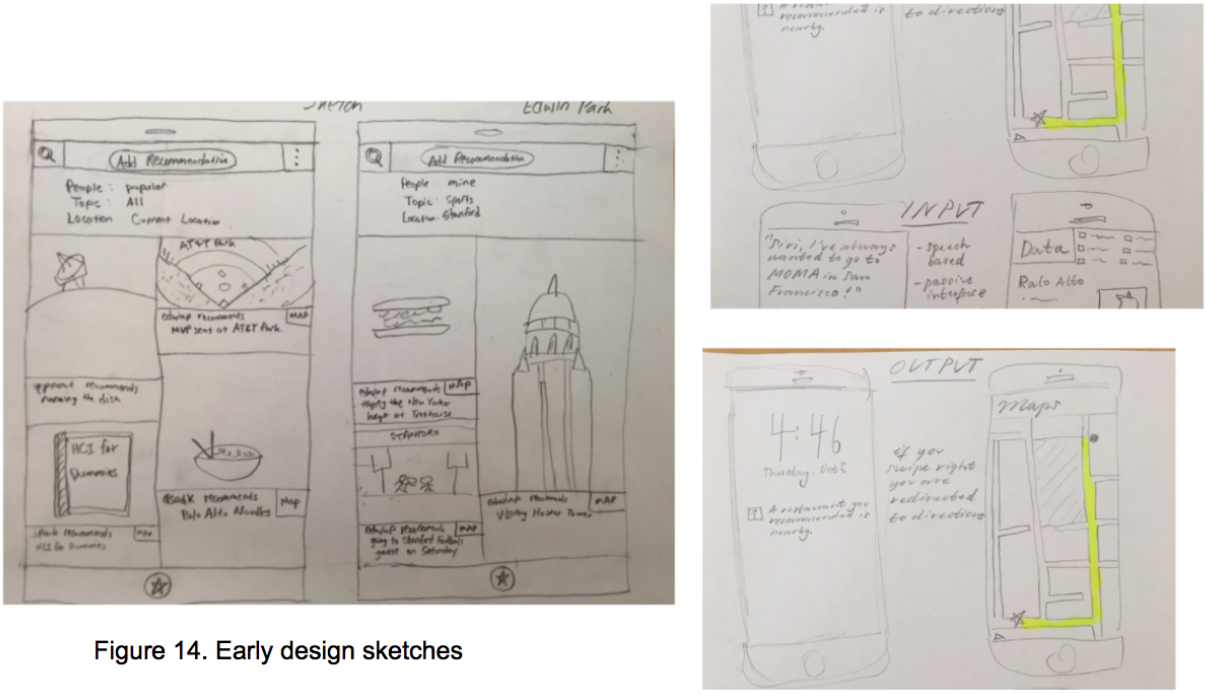

Figure 14. Early design sketches

Having identified a user problem and a platform from which to develop, we started prototypying. Here, 3 simple steps guided our design process: ***design, prototype and most importantly, evaluate****. And then repeat.*

Low-fi Prototype

From our contextual inquiry, we realized that users need a tool that is with them at all times. In our increasingly modern society, smart-phones have become an integral part of our daily lives and so we decided to target the mobile phone platform. Also from our contextual inquiry, our 3 distinct tasks identified guided the design of our first low-fidelity prototype. In fact, we separated each task into its independent section in our application:

1. *Inbox* where users quickly add a recommendation (Figure 15)

2. *Organizer* where users are able to view a list of their recommendations and add detail to each entry (Figure 16)
3. *Discover* where users are able to rapidly go through recommended activities (by time and place) in a Tinder-like way (Figure 17)

Splitting the application up into tasks was our first attempt at structuring the user experience. Although this was useful for us to explore each task in isolation, it did not produce a cohesive experience for the user. This was highlighted by our user testing where we discovered numerous problems such as confusing swiping movements, confusing vocabulary and a missing guiding force behind the application. Splitting the application into tasks so distinctly was a useful step but as a result of our user testing, this approach was revised in later iterations.

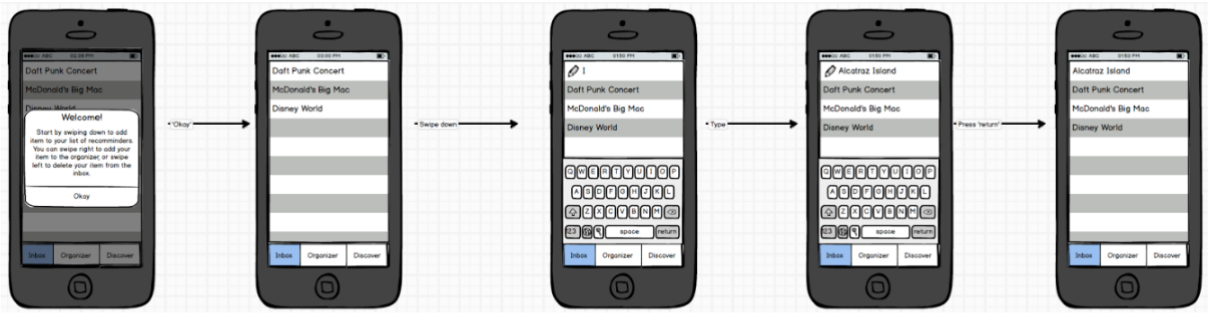

Figure 15. Low-fi Prototype Task 1: Recording Recommendations

Figure 16. Low-fi Prototype Task 2: Viewing & Organizing Recommendations

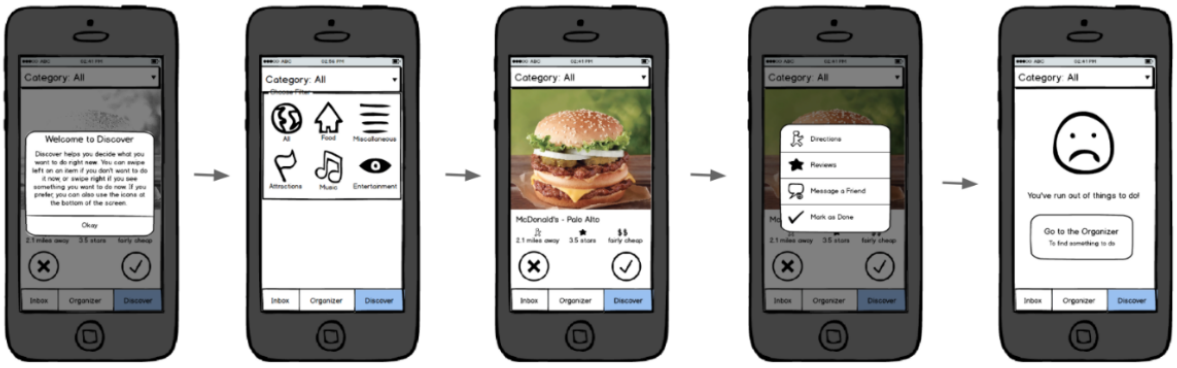

Figure 17. Low-fi Prototype Task 3: Discover in a Tinder-like UI

Medium-fi Prototype

Through our user testing, we found out two crucial points: (1) accepting recommendations depends on more than just the right time and place and (2) our overall application structure based on division of tasks was unintuitive and confusing to the first-time user. With this knowledge, we decided to pivot quite drastically into a new space where accepting recommendations doesn't depend on 1 million psychological factors (are you tired, feeling adventurous, hungry?): **travel**.

Travelers are always in the mood to explore but they sometimes complain that they can't get the most of their on-the-ground experience. To understand the user mindset, we repeated some more targeted contextual inquiry. What do travelers want? What are they not getting? We quickly realizes that it is hard for travelers to balance both planned activities and spontaneity. Existing platforms aren't made for simultaneously storing a destination's must-do's (the things a user really wants to do) and finding things to do more spontaneously. With Usher, we want to solve that. Our solution is an *on-the-go* local guide mobile application, where users can keep track of a list of recommendations and let the app guide them on a personalized daily path through a destination based on their preferences. With our eye now on travel, our design had to change with it.

Despite the pivot, the representative tasks remained largely the same: (1) recording recommendations in Figure 18, (2) organizing/viewing recommendations in Figure 19, (3) deciding what to do next in Figure 20. The only different task is (3), which changed simply as a result of our new problem definition. Travelers want a tool they can use to actively shape their day, and so (3) reflects how we've the user is now actively involved in shaping the day by asking for recommendations, whereas before the user was only a receiver. Using this redefinition, we centered our design around *shaping a traveler's day on-the-go*.

A downfall of our low-fi prototype was that most participants needed more information about each page and its purpose to understand what to do. In this prototype, we have a *Home Page*, from which users can add to their list of recommendations or start their day. This home screen gives the user more understanding of how the app functions. By starting their day, users choose a few must-do's they want to explore that day, and then they get going. After

completing an activity, they can refer back to the app and choose their next activity by browsing a list of cured recommendations.

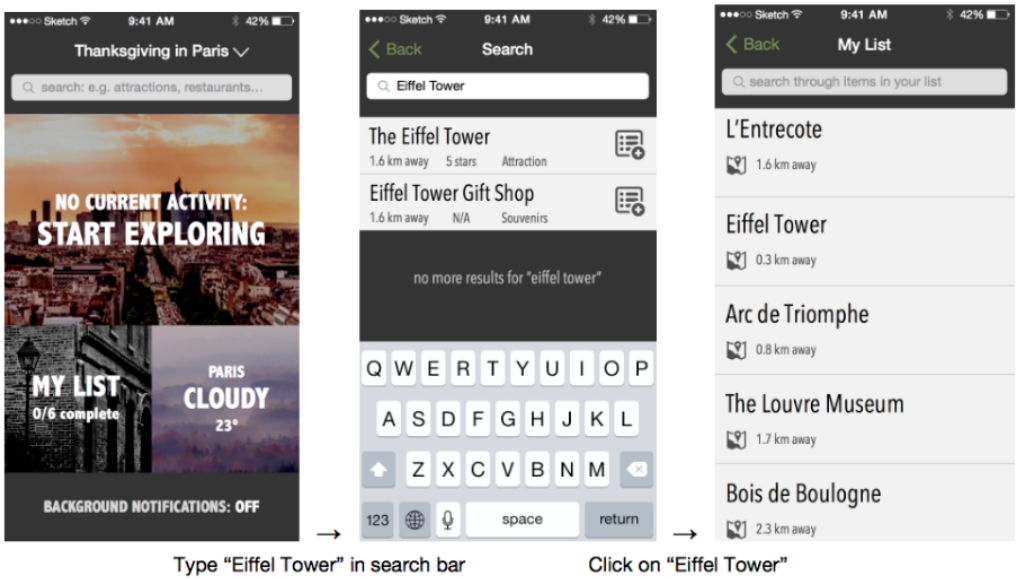

Figure 18. Med-fi Prototype Task 1: Recording Information

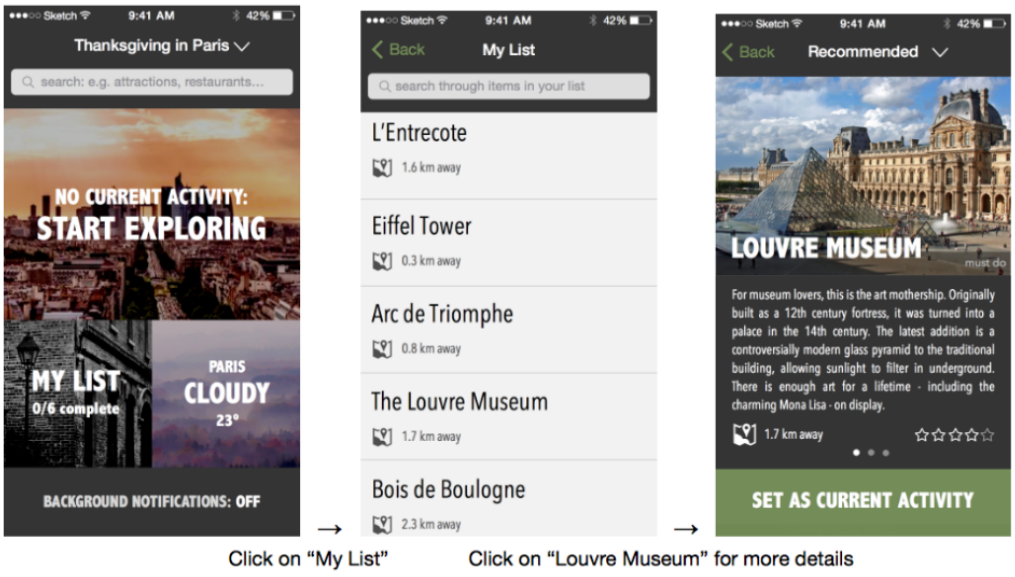

Figure 19. Med-fi Prototype Task 2: Viewing Information

Figure 20. Med-fi Prototype Task 3: Discovering new Recommendation

High-fi Prototype

As discussed previously, a heuristic evaluation found that we had **33 violations**, most of which were consistency (9 problems) and design-related issues (8 problems). For our high-fi prototype, we took all of these concerns very seriously.

Moreover, we decided to tackle an issue we had found in our low-fi prototype: a lack of connection between tasks. The first-time user, we noticed, has difficulty understanding how each task connects without understanding the higher level purpose of the app. In this iteration, we no longer treat tasks as deserving single sections. Instead, the user can complete the task from many places. Consider the following two examples: adding to *MyList* and setting a current activity. Previously, both these tasks were achievable only from their respective sections. Now, you are able to add an item from both the *Home Page* and from *MyList,* as seen in Figure 21. Similarly, you are now able to set a current activity from both *MyList* and from the *Recommended Page,* as seen in Figure 22. This gives the user much more flexibility in their app usage. Although some may think that more user freedom may dissolve and confuse this ‘higher level’ understanding of what the app does, we think the opposite. Having more paths to the same result, in our case, makes it clearer to the user how information travels across the app.

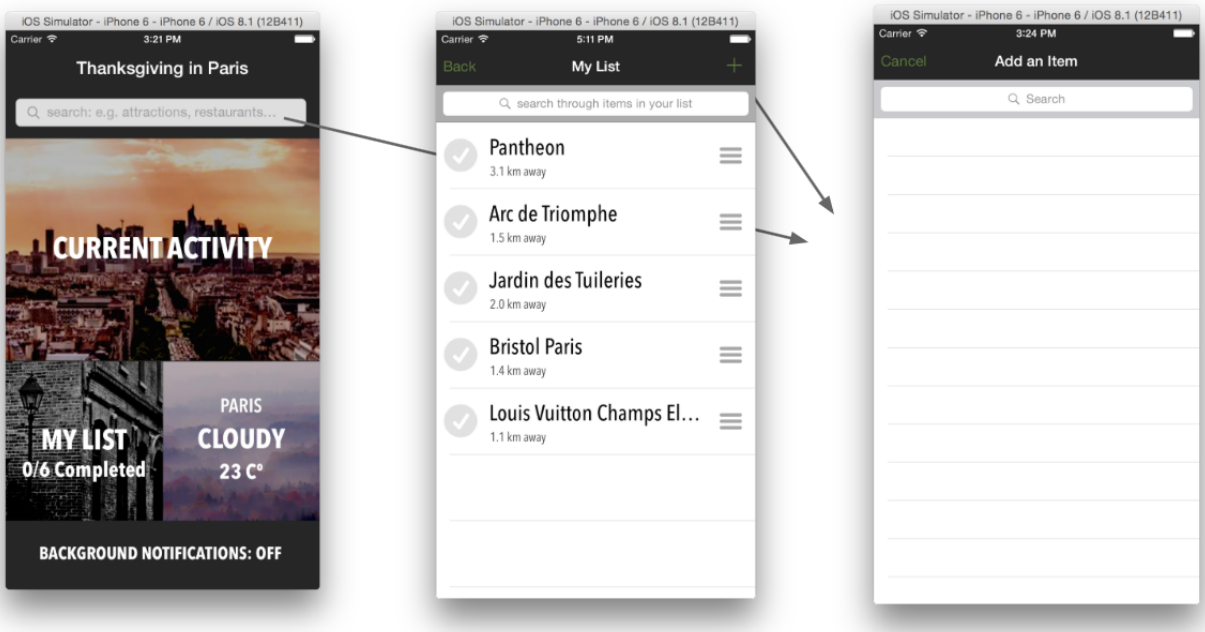

Figure 21. High-fi Prototype: easily add items to MyList from Home and from MyList

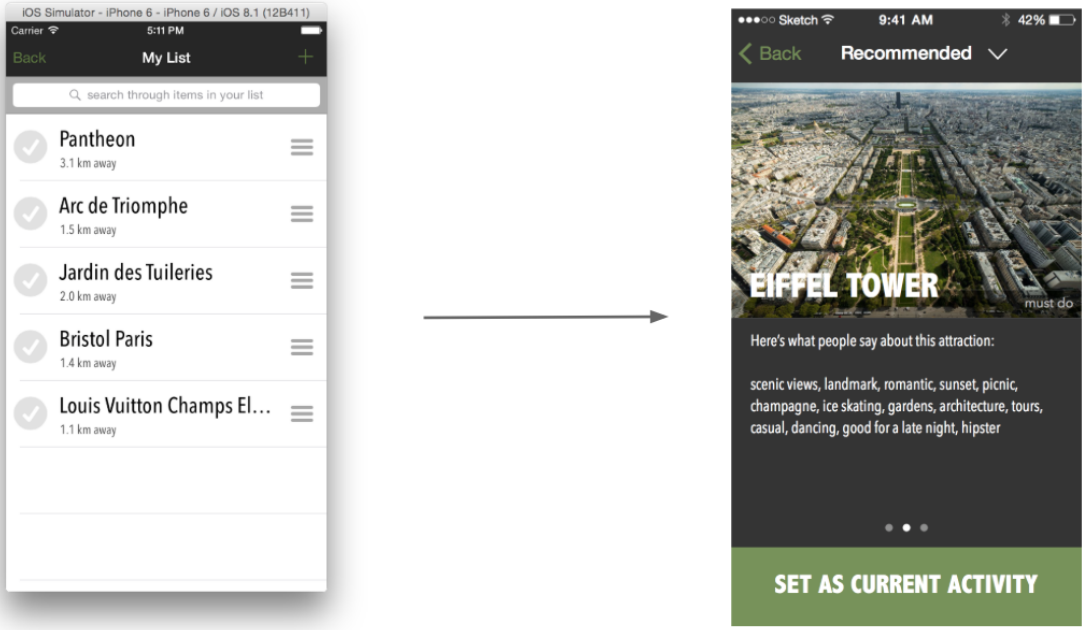

Figure 22. High-fi Prototype: easily set current activity from MyList from Home or from Recommended Page

Prototype Implementation

Tools

We used **XCode** as our coding platform. We found that although the learning curve is steep, it becomes easier after a while. **Objective-C for iOS Programming** is not an easy language to get used to but many online resources make it easier. For example, querying Google with errors and affirmations yields many **StackOverflow** responses that were invaluable to getting us past difficult bugs. We also made **Photoshop** to a large degree to make the graphical elements of the UI.

Hard-coded Data

We first developed a TripAdvisor scraper which would allowed us to get a large number of activity entries. However, the scraper missed some key information that we deemed important such as description and distance from you. Before developing a more capable scraper, we have used hard-coded data about one destination (Paris) to power our application. This data set is in JSON format with key to string value pairs.

Future Direction

- **Develop a better recommendation system**: our recommendation engine is currently very basic as it basically imports MyList without any specific reordering. We plan on using hours of operations and distance to recommend activities. We also want to make sure a user gets all their daily must-do's done, which means we must take into account closing times for all these activities and plan them accordingly.
- **Develop a proper scraper or make our own original database**: currently, our database is hardcoded. Hence, users are only able to add items that are part of our 31-item list. We plan on fixing our scraper so that it takes in the necessary information about each event.
- **Implement a first-use tutorial**: we've found that some of our features need to be taught. For example, our rankable list isn't intuitive to the first-time user and so we want to implement a first-time tutorial that holds the user's hand through the first 'day'.
- **Refine the UI**: some transitions could be smoother. For example, swiping across recommendations is very static. We could also come up with original icons to match our UI theme.