RideAlong

Eric Arendt - Aeron Bryce - Alex Jansen - Eli Williams

Interactive Prototype Report
CSE 440
Autumn 2009

Roles

Group Manager: Eli Williams
Documentation: Aeron Bryce
Design: Alex Jansen
Testing: Eric Arendt

Problem and Solution Overview

At any given moment as many as 75% of the cars on the road have only a single occupant. We know that quite a few of these drivers share the same destinations, and we can also assume that a number of individuals without consistent access to a car would like to travel between these destinations as well. The problem exists in the lack of a connection between drivers with empty seats and riders trying to reach a destination. The solution is RideAlong. RideAlong aims to promote ridesharing as a practical means of transportation by facilitating easy, trustworthy connections between drivers and passengers.

RideAlong is a mobile ridesharing application that looks to connect individuals looking for a ride with drivers who are interested in sharing. The application is centered around a rich trip sharing interface that allows users to post and find trips, determine appropriate compensation, and establish trust with other riders and drivers through personal profiles and feedback.

Tasks

Easy *Task - Confirm a rider who has requested to ride with you.*

Respond to an alert. Two riders have requested a seat on the trip you posted. Choose between the two riders and confirm one. Choose a place to pick the passenger up.

Moderate Task – Post a ride

Post a trip between your house on Capitol Hill and Electric City, leaving on Friday, 11/27 at 2:00 PM. You have space for one passenger, and you would like to get about $15 from whomever rides with you.

Difficult Task – Find a ride to somewhere specific within a specific time frame

Search for a ride between your house on Capitol Hill and Electric City, leaving on the afternoon of Friday, 11/27.

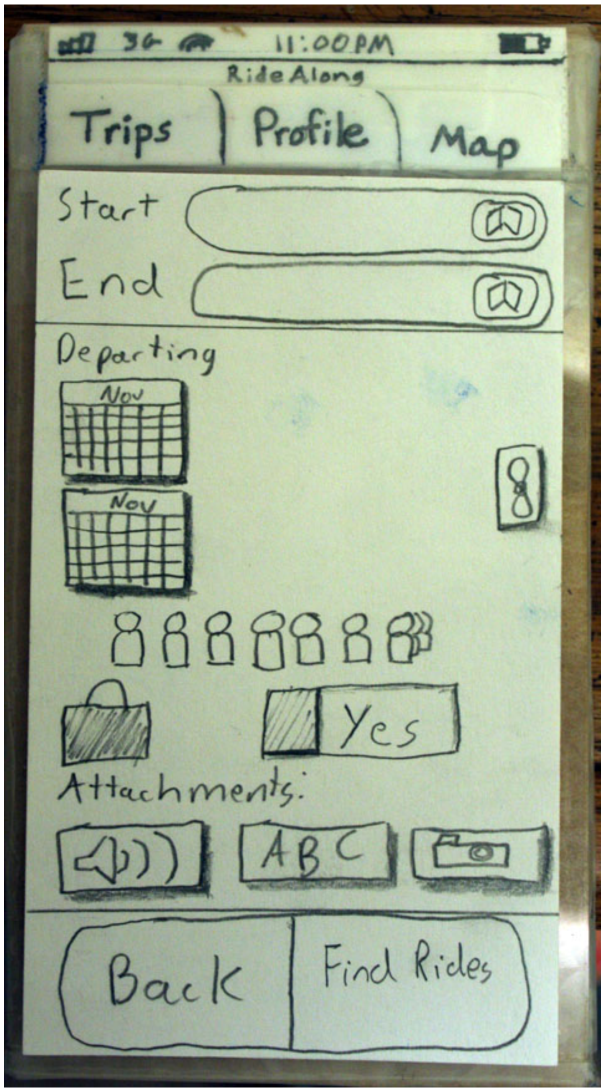
Interface Revision Sketches

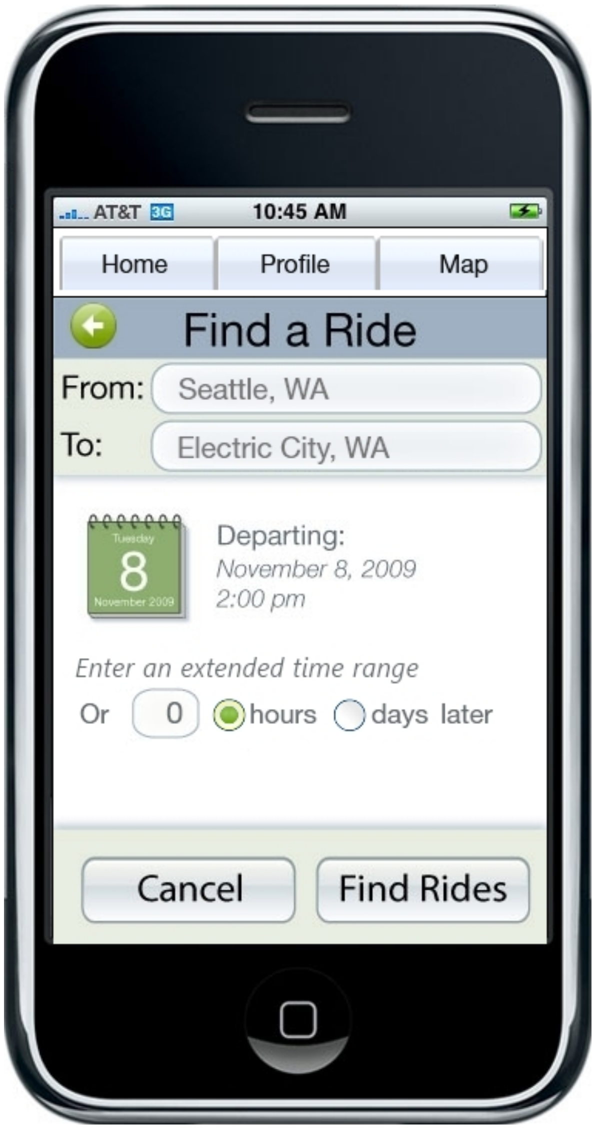
We found and corrected significant flaws in three main screens: “Find A Ride”, “Post A Ride”, and “My Trips”.

Find A Ride

The Find a Ride screen kept the overall organizational structure from the paper prototype version, however, the content was changed to better suit the needs of the user.

In testing, the two-calendar system, intended to allow the user to enter a range of date/times, was confusing. We removed the second calendar and added a range selector, enabling the user to enter a time range of hours or days following their selected departure date to search within.



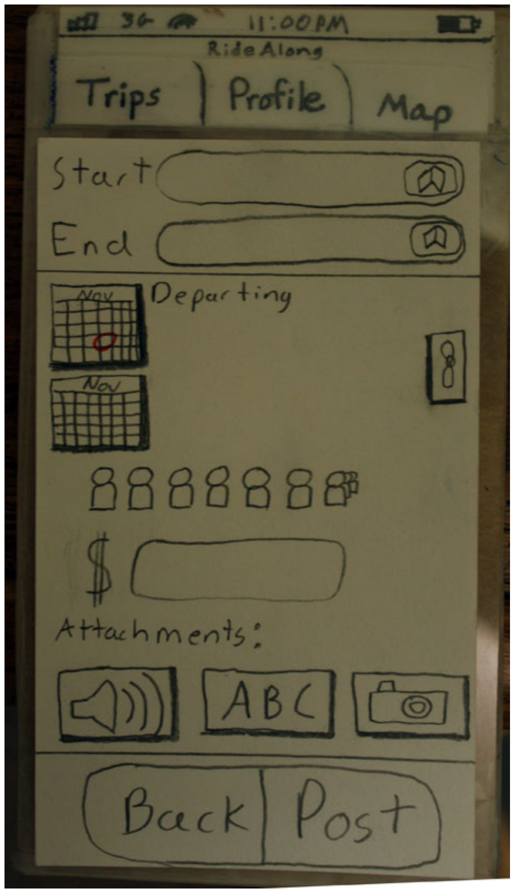


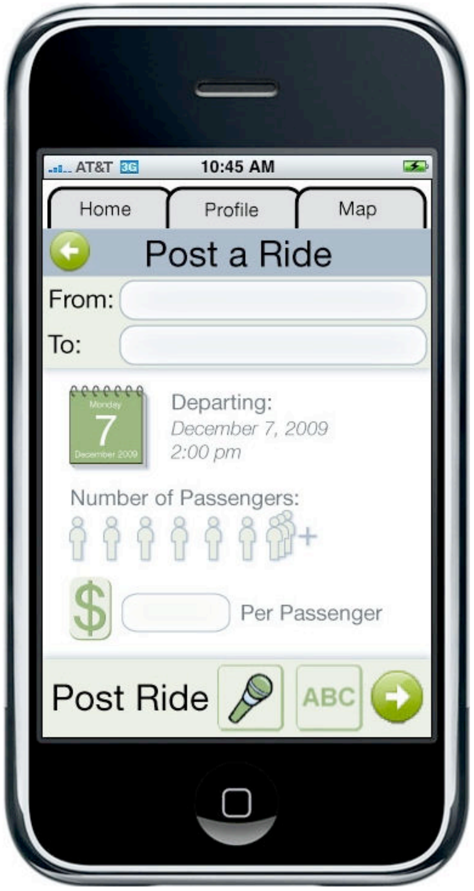
The multiple rider selector as well as the luggage options were removed. The multiple rider selector was ambiguous and easily confused with the number of passenger selector in the Post a Ride interface. The luggage was an optional feature, which we decided to remove for this version, as it received negative feedback in the paper prototype. Finally, the attachments were removed all together, as there was no logical reason to attach any form of message in this screen, and the user will be presented the option later in the task process.

Post A Ride

As the Post a Ride and Find a Ride interfaces share a similar organizational structure, many of the issues with the Find a Ride interface were seen in the Post a Ride interface as well, including the two-calendar confusion. We removed the option of entering a date range, as it didn't seem essential for someone posting a ride.

In our paper prototype, the "Post" button was disabled until the user attached either a text or voice message, but it was highly confusing as to why the button was disabled and how to enable it. Borrowing from another part of our prototype (the request portion of the My Trips screen), we were able to consolidate the attachments and clarify the need of an attachment before continuing to the next step.



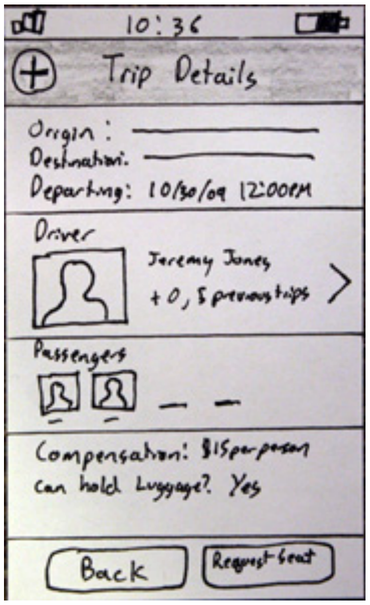


Other minor improvements include adding text to clarify the number of passengers and compensation sections.

My Trips

My Trips received negative feedback during testing with the process of confirming a rider.

First, it was unclear how to respond to a pending request. Our original design had a highlight around the open seat, hoping that the user would notice it and curiously click on it. It wasn't a clear enough cue to inform the user that by clicking on that empty seat they would be able to confirm a passenger for that seat. We changed the highlight from yellow to red and supplemented it with a popup to give the user a clear visual cue as to what to do to confirm a request.



10:36

Trip Details

Origin:

Destination:

Departing: 10/30/09 12:00PM

Driver

Jeremy Jones

+0, 5 previous trips

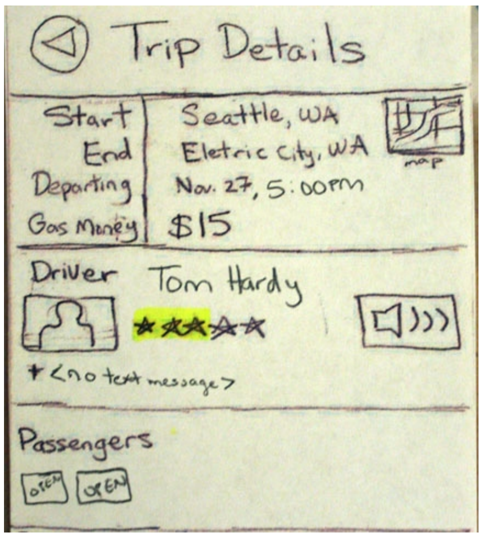
Passengers

Compensation: $15 per person

Can hold Luggage? Yes

Back

Request Seat



Trip Details

Start Seattle, WA

End Eletric City, WA

Departing Nov. 27, 5:00PM

Gas Money $15

map

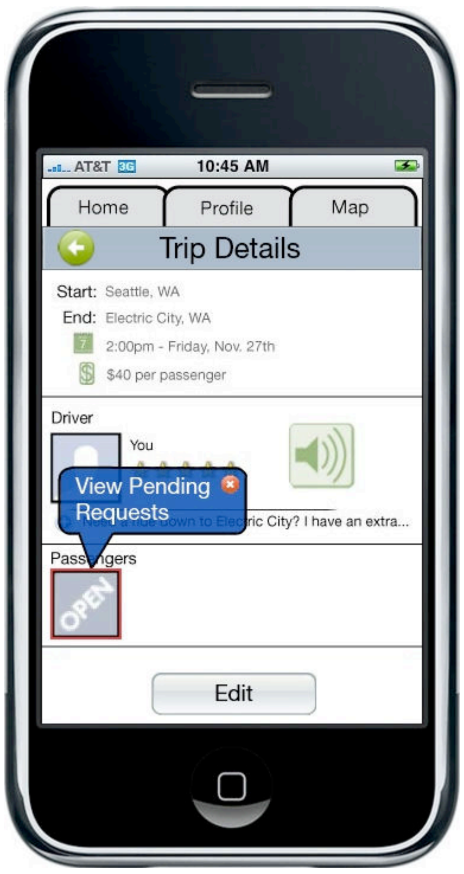
Driver Tom Hardy

+ <no text message>

Passengers

OPEN OPEN

Little visual cue above, to clear cues below.



AT&T 3G 10:45 AM

Home Profile Map

Trip Details

Start: Seattle, WA

End: Electric City, WA

2:00pm - Friday, Nov. 27th

$40 per passenger

Driver

You

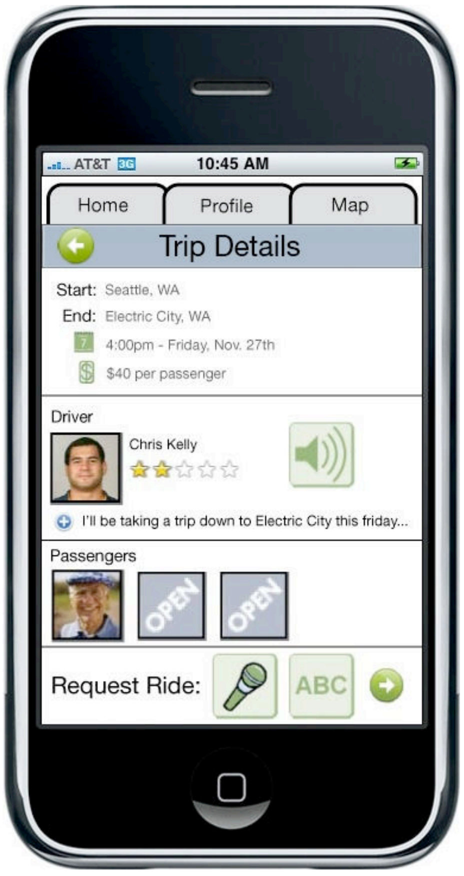
View Pending Requests

...ric City? I have an extra...

Passengers

OPEN

Edit



AT&T 3G 10:45 AM

Home Profile Map

Trip Details

Start: Seattle, WA

End: Electric City, WA

4:00pm - Friday, Nov. 27th

$40 per passenger

Driver

Chris Kelly

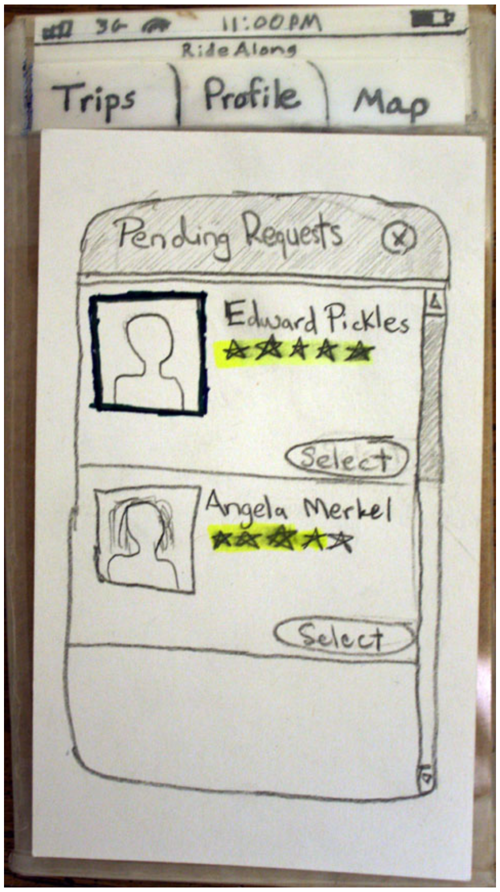
I'll be taking a trip down to Electric City this friday...

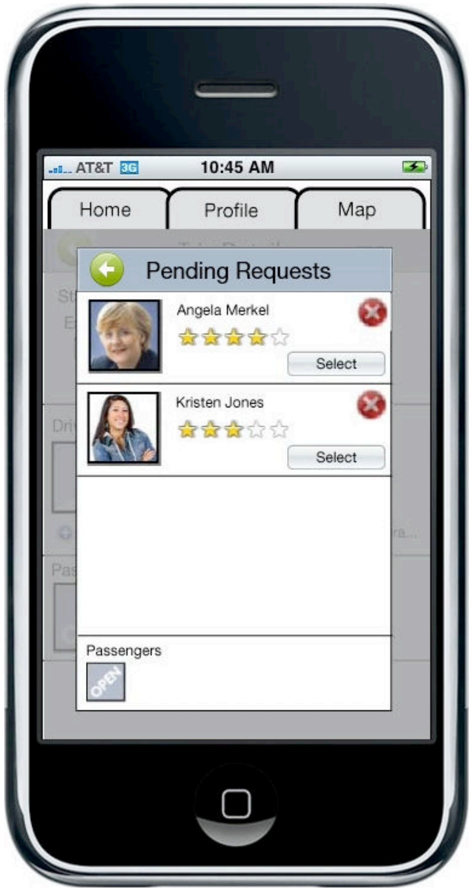
Passengers

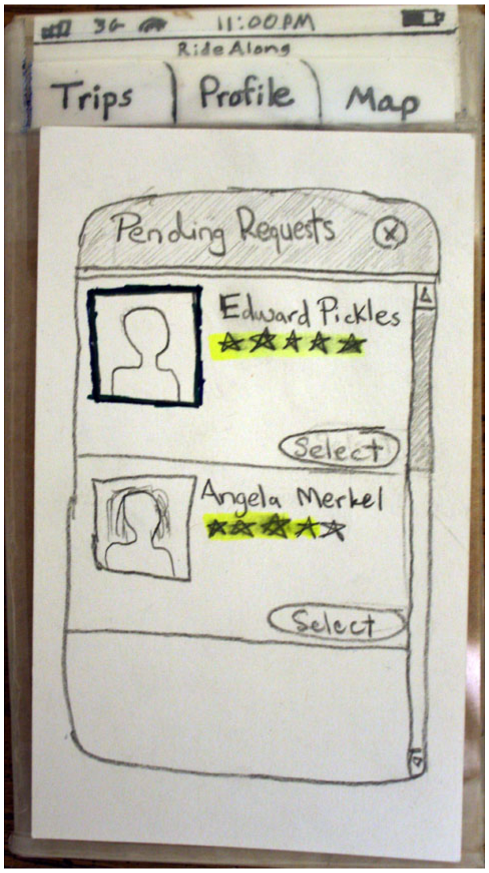
OPEN OPEN

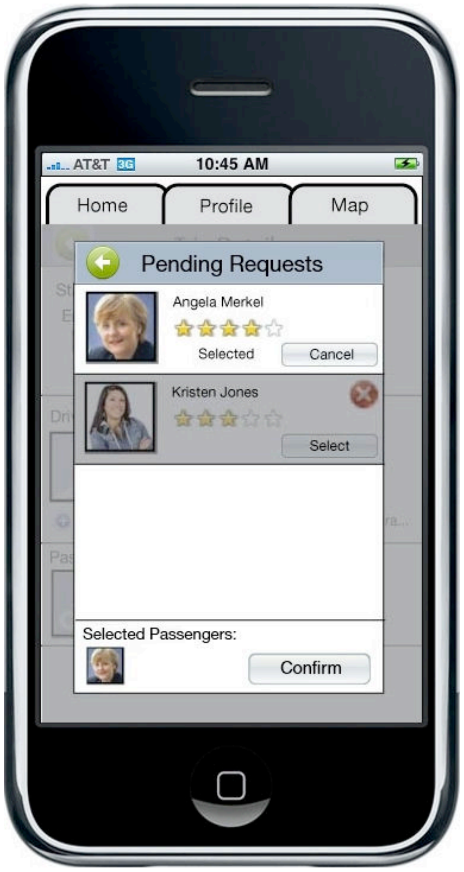
Request Ride: ABC

When the user is viewing requests, we previously would allow them to only select one rider at a time, no matter how many open seats they had or requests were pending. Each time the selected a rider, the confirm dialog would close. We reworked our pending requests popup to refine the interactions and to eliminate the need to return to Trip Details after each selection. A list of open and filled seats now appears at the bottom, showing the users open and assigned seats. The user then cycles through confirming passengers, returning to their requests after each rider selection.









Prototype Overview

Our implementation consisted of creating three separate fixed path prototypes, one per task. Each leads the user through a series of actions, guiding them to completion of the specified task. While most of the interface relies on fixed paths, allowing the user to click on only one option, some areas are more flexible, allowing the users to peruse search results and profiles as they might if the application were fully operational.

Task Descriptions

Each task is well described through step-by-step guidance next to the prototype. Below is a general sequence of actions for each task.

Easy Task - Confirm a rider who has requested to ride with you.

1. View Request
2. Select Pending Requests Bubble
3. Choose a rider
4. Select a place to pick up the rider
5. Confirm your choice and close the pending requests screen
6. Exit

Moderate Task – Post a ride

1. Open RideAlong
2. Select the Post A Ride option
3. Enter the start location
4. Enter the end location
5. Continue

6-8. Select the date and time

9. Indicate the number of available seats
10. Enter the compensation desired

11-13. Attach a text message

14. Continue
15. Exit

Difficult Task – Find a ride to somewhere specific within a specific time frame

1. Open RideAlong
2. Select the Find A Ride option
3. Enter the start location
4. Enter the end location
5. Continue

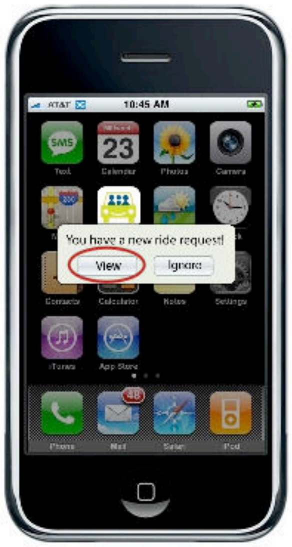
6-8. Select the date and time of departure

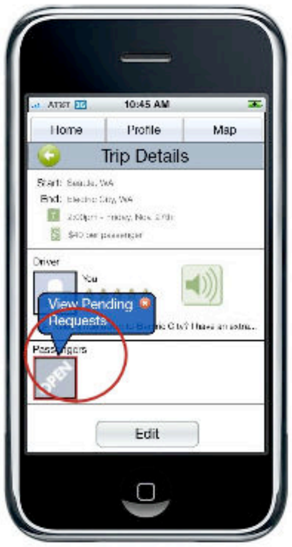
9. Select a time range
10. Continue
11. View trips and profiles

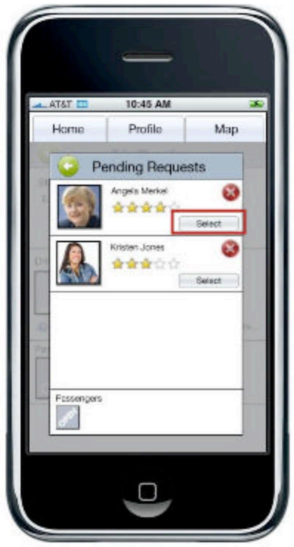
12-14. Attach a text message

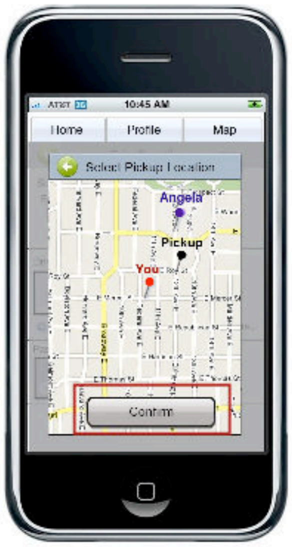
15. Request a ride
16. Exit

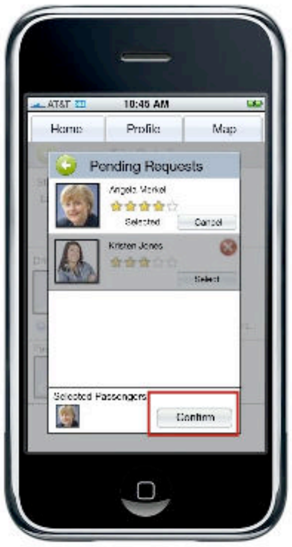
Storyboard for task #1 - Confirm a passenger. Red boxes indicate the button to click or text box to fill in to advance to the next screen.

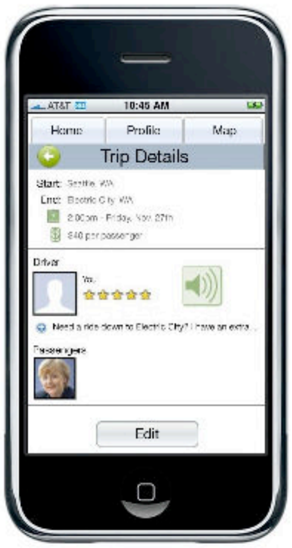

You have a new ride request!





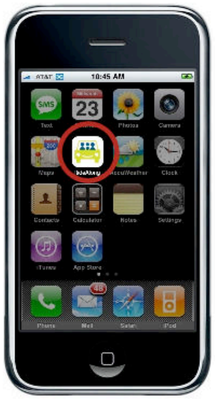


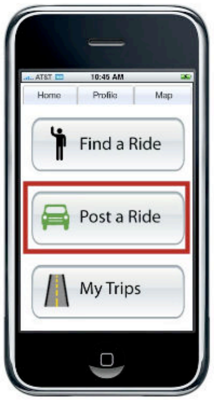


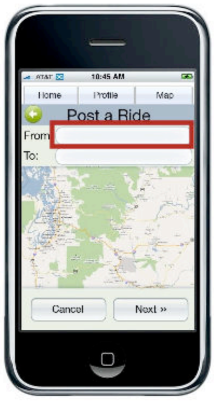


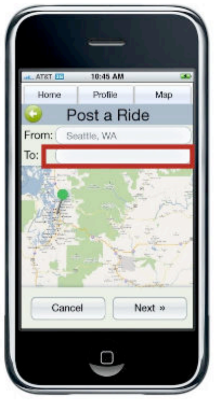
Need a ride down to Electric City? I have an extra...

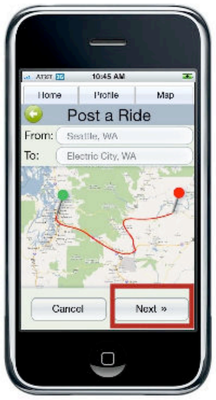
Storyboard for task #2 - Post a ride. Red boxes indicate the button to click or text box to fill in to advance to the next screen.

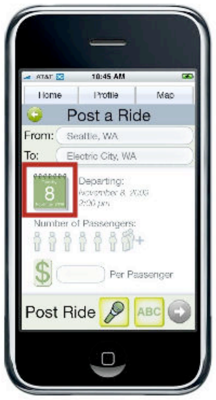


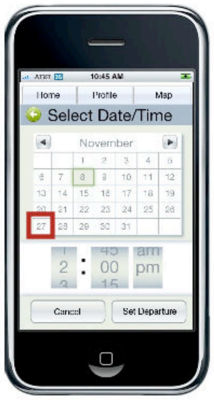


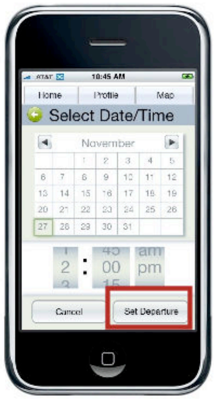


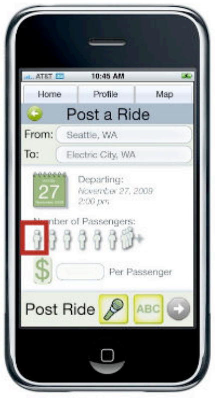


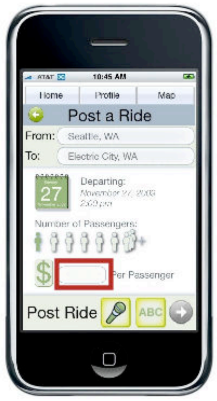


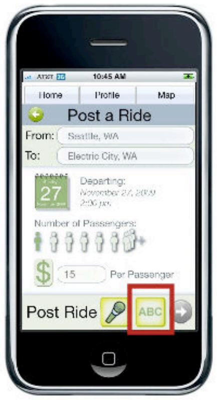


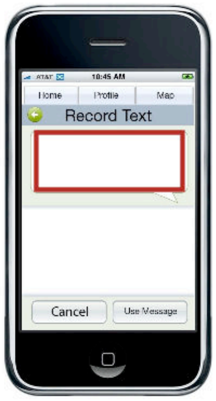


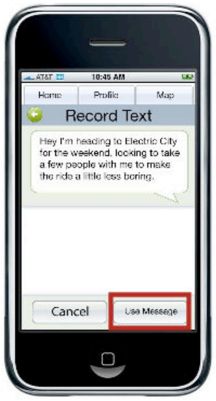




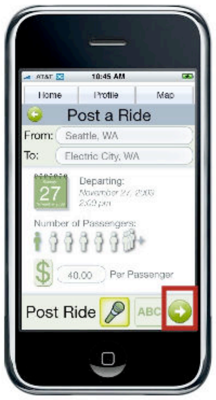


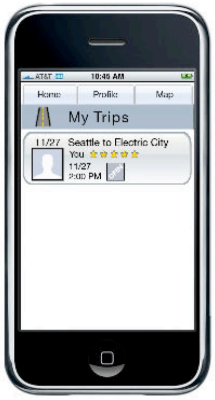




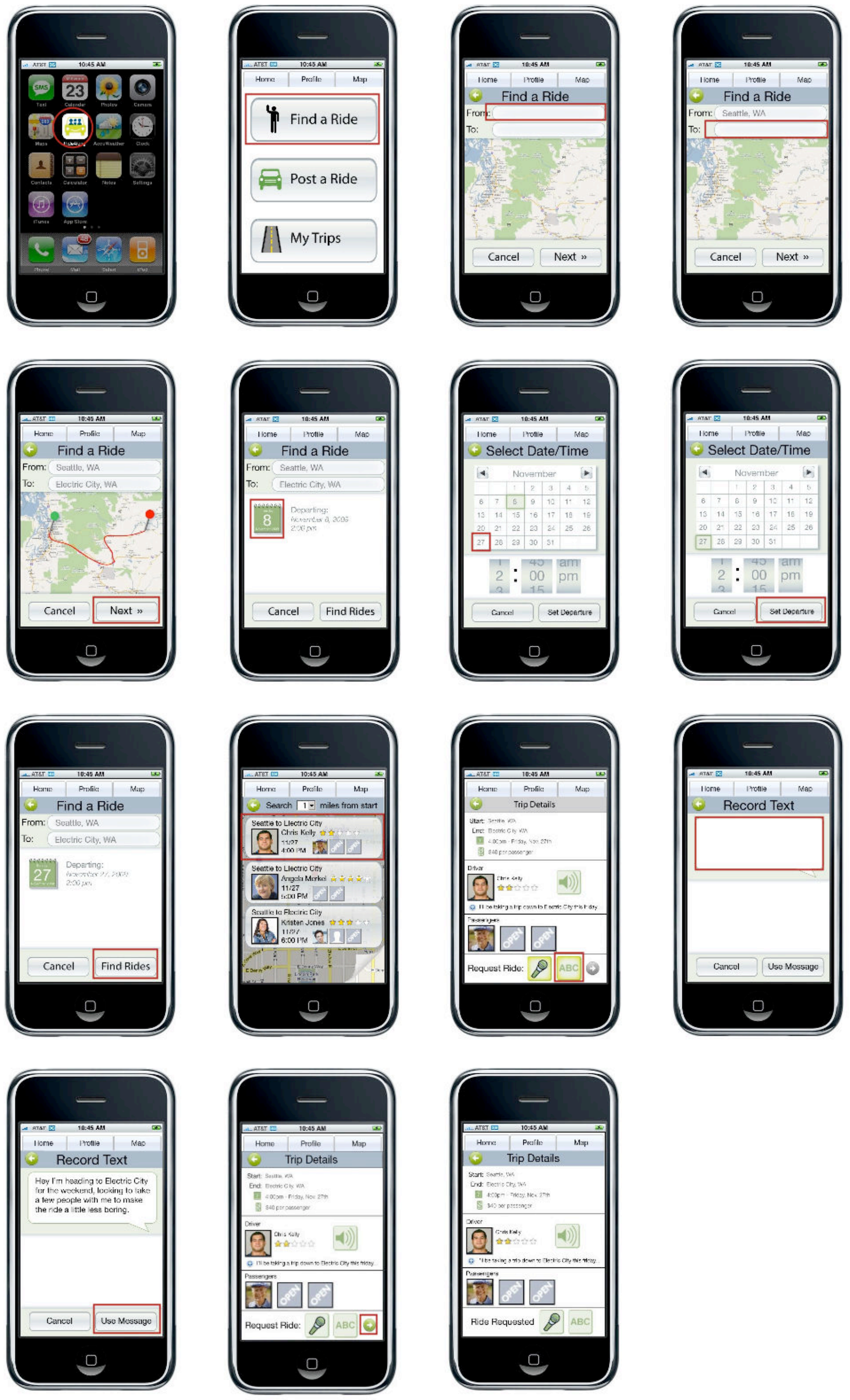


Hey I'm heading to Electric City for the weekend, looking to take a few people with me to make the ride a little less boring.





Storyboard for task #3 - Find a ride. Red boxes indicate the button to click or text box to fill in to advance to the next screen.



Tools

To generate our prototype we used images, html, and basic JavaScript. We took advantage of graphical editing software (Paint.NET, Adobe Illustrator, Paint Shop Pro) to create the images, attempting to remain consistent throughout our respective portions. JavaScript show/hide functionality was used to swap in and out different images. The clickable areas of the images were created with "map" and "area" tags, defining which regions were clickable and what page clicking on a region would swap in.

As our prototype is mostly images and links, we limited interaction so that most flows had a fixed path.

Pros

- Able to create all the screens graphically - There was no need to deal with messy CSS placement or managing many image segments.

- Div Swapping - Swapping in and out divs made the flow very flexible, allowing for many images to be switched in and out without needing to create a separate html file for each.

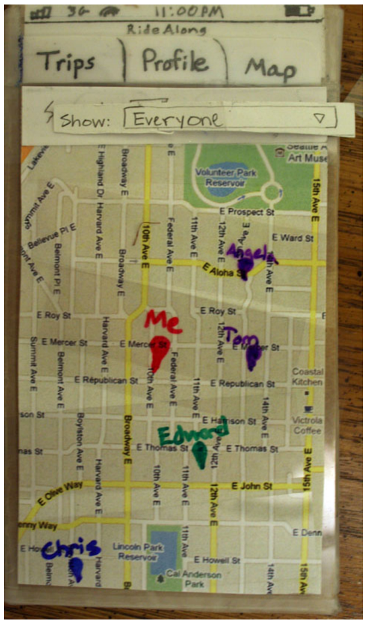
- Area tags - Using area tags let us define the precise regions we wanted to

Cons

- The only real interactions we supported were clicking. This meant we couldn't implement the interactive map part of finding and posting rides (where the user could drag the map around, and place pins, etc).

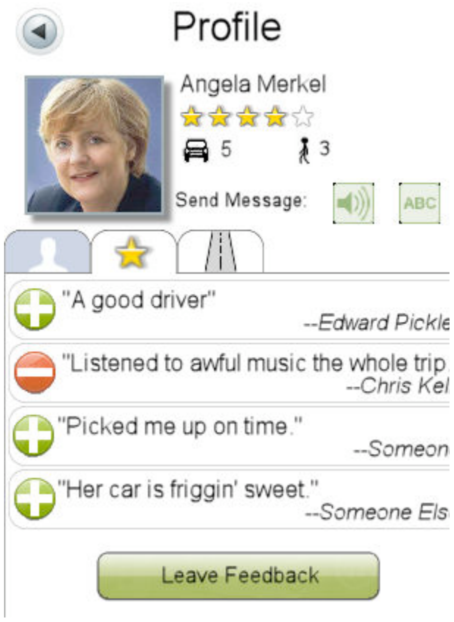
- We had trouble keeping all of our images and designs consistent. We often got bogged down in minor details: fonts, colors, icons not quite looking the same on two screens, etc.

What Was Left Out



We did not provide the full functionality of the map tab. This was intended to be a social networking feature, where users could browse for nearby people, or for trips leaving from near a given location (without specifying a destination). Our tasks cover most functionality of the system, but this feature was not included in any task so we did not implement it.

We did not implement the feedback feature, allowing users to review each other. The button exists, but the process of leaving feedback was never flushed out. It, again, was not part of any of our tasks.


Profile

Angela Merkel

5 3

Send Message:

"A good driver" --Edward Pickle

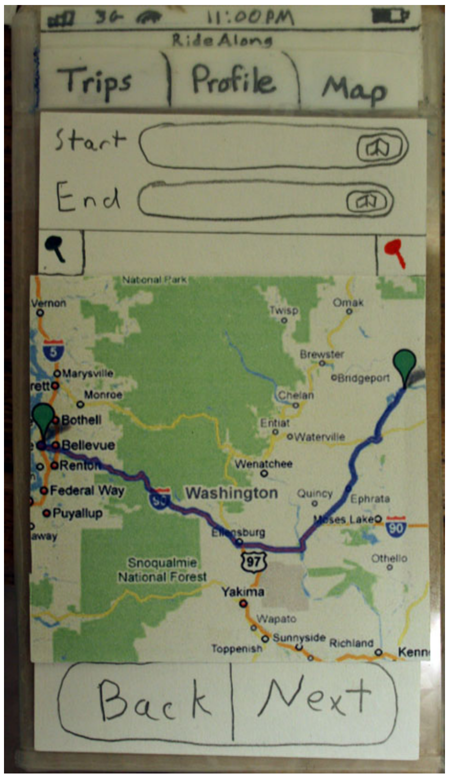
"Listened to awful music the whole trip --Chris Kel

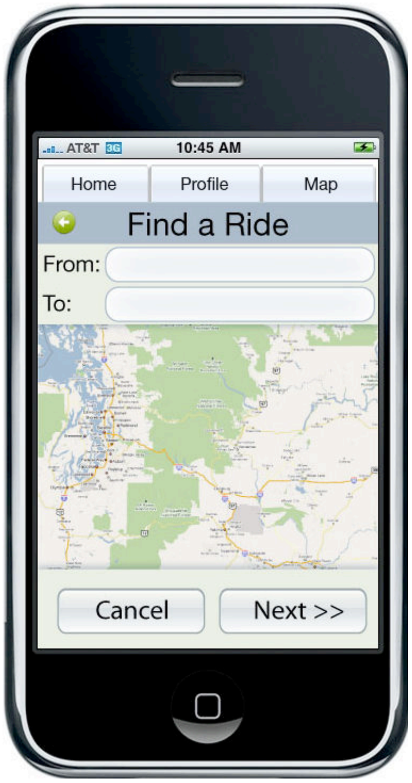
"Picked me up on time." --Someon

"Her car is friggin' sweet." --Someone Els

Leave Feedback

Finally, when searching for a ride or posting a trip, we did not make our map interactive. Although we feel it is a useful feature, for the purposes of the prototype, we felt it was not an essential component and might even distract from the task. We felt our time was better spent elsewhere.



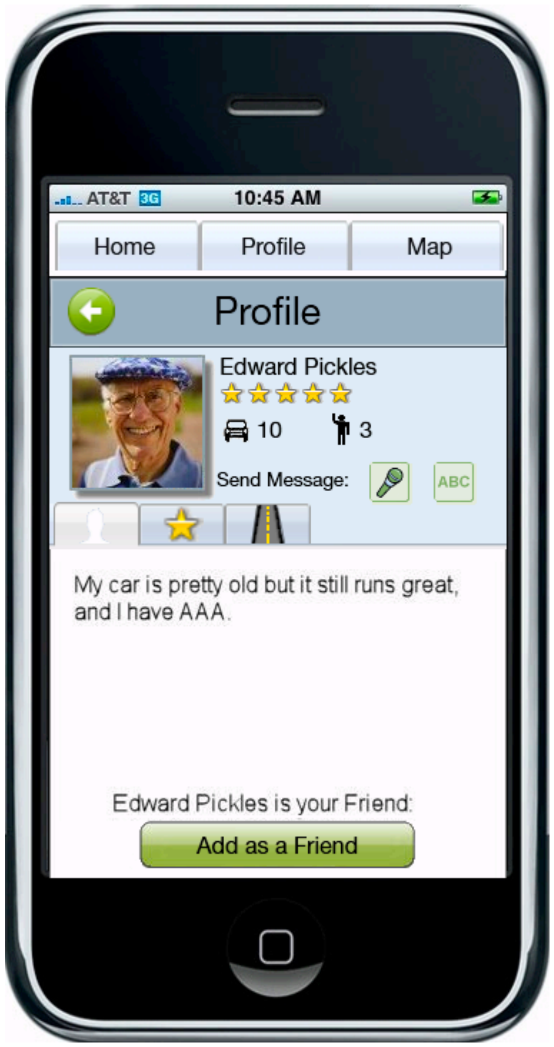


Wizard of Oz

Our back end matching service between riders and drivers was definitely magical. We didn't use any strict Wizard of Oz techniques such as magic refrigerators or flying cars in our prototype, but we did provide the user complex feedback without worrying about the difficult algorithms and back end services associated.

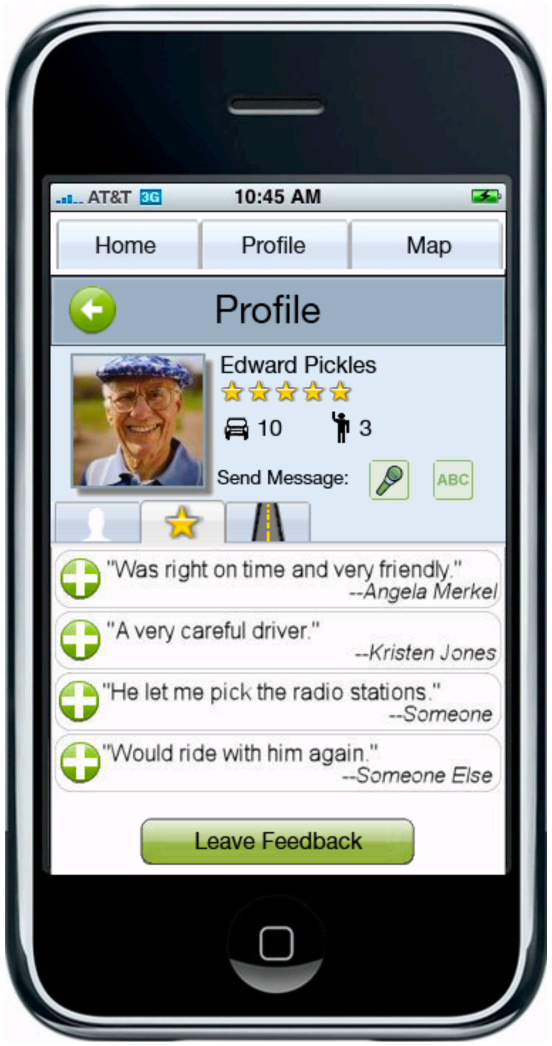
5) Prototype screenshots (as many as needed)

Most of the screens for our prototype appear in the storyboards. The only screens missing are the profile screens. Here is one profile screen. We only created the “About Me” and “Feedback” tabs for each profile, as the last tab (“My Trips”) was not important for any task.



My car is pretty old but it still runs great, and I have AAA.

Edward Pickles is your Friend:



"Was right on time and very friendly." --Angela Merkel

"A very careful driver." --Kristen Jones

"He let me pick the radio stations." --Someone

"Would ride with him again." --Someone Else