

Catch-A-Ride

By: Keith Stone, Jarret Falkner, Ashwini Varma

Problem and solution overview

Although participating in a rideshare provides social and economic benefits, the logistics of coordinating drivers and passengers create challenges that undermine widespread ridesharing. Systems like King County's Vanpool have limitations such as maintaining predictable weekday schedules, requiring a minimum of five people per trip, and upfront identification of secondary drivers and a mileage log accountant. Message board methods like Craigslist and Facebook groups have limited search functionality and leave riders with uncertainty about potentially unreliable or even unsafe drivers. Our proposed Catch-A-Ride product differentiates itself from existing rideshare systems by providing a semistructured smartphone application that includes strong user profiles to instill community trust as well as advanced search functionality to better match drivers and passengers.

Contextual inquiry participants

Steph is a grad student at the University of Washington. She lives very close to the city and does not own her own car. She has a U-Pass and subscribes to both Car2go and Zipcar for ad hoc car rental services. We met Steph through a Craigslist ad where we offered a ride to Olympia in exchange for answering our questions. We picked her up in the central district in Seattle and during the ride the three of us asked questions about how she found our post and her thoughts and experiences leading up to the ride. We dropped her off at her destination in Olympia.

Leon is the rideshare coordinator at The Summit at Snoqualmie, a ski resort near Seattle. The Summit presents unique challenges for carpooling as the majority of employees are seasonal and live quite a distance away. While visiting his office at the mountain we interviewed him on how his program works and how he uses carpooling. He is unfortunately not the model of a seasonal worker as his position is year round, but as a user of the system and with no seasonal workers hired at that point in the year he still made a great source of information. Leon is approximately 35 and represents the top end of our demographic.

Chris is another full time human resources employee at The Summit at Snoqualmie. We had not originally intended to meet him but he offered another angle of inquiry while interviewing Leon. Chris is much younger, approximately 23, and represents a lower section of our target audience. While his position is also year round he was hired recently and so provides somewhat of a perspective on what a seasonal worker could go through.

Leslie is a full time employee at Microsoft. He is approximately 35 years old, near the top end of our demographic. He has been using Vanpool since Nov 2010. His daily commute is about 50 miles round trip. We interviewed him on how the King County Vanpool program works and how he uses it. He was a good source of information, however, the biggest pain point of the

Vanpool system seems to be finding the right vanpool and Leslie was unable to provide insight here. Leslie found his Vanpool because of his neighbor - his neighbor is a VanPool driver.

Contextual inquiry results (1.5 pages)

We interviewed 3 people who used the employee vanpool for commuting to work and one person who used the Craigslist rideshare to make a one off long distance trip from Seattle to Olympia. The commonality with all 4 people was that they each did it with the intent to lower environmental impact and also as a means to save money. Since the tasks/mechanics of the employee vanpool are different from the Craigslist rideshare we will describe them separately.

Employee Vanpool

For the employee vanpool the first and more often than not, a one time task, is to find the “right” vanpool. This is usually done by registering with the King County vanpool system, where they try and match you with existing or newly forming vanpools. People are also encouraged to put up notices on bulletin boards and most companies seem to have a commute/transport coordinator who can help spread the word. The people we spoke to each had ended up in the vanpool because they knew someone who was already a part of an existing vanpool. Leslie, the Microsoft employee, got to know about the vanpool because his neighbor was the designated driver and the vanpool needed more people to join. In general though, because Microsoft has such a large number of employees, there are email distribution aliases set up for people that live in different neighborhoods e.g. one for Ravenna, another for Monroe etc. An employee can advertise to a broad set of relevant people if they wanted to join a vanpool by sending email to the relevant alias. Leon, who worked at the Snoqualmie ski resort, got involved because of an organic connection with the previous HR head. Leon took on the role of rideshare coordinator and in order to get more people involved in the rideshare/vanpool program used the following measures

- Big push at orientation. (1,500 hires per season.)
- Postings at 14 time clocks throughout the mountain
- Trickle down approach through the supervisors. (Remind them to remind the employees)

At Snoqualmie there seems to be a high churn of employees and most of the employees don't have a desk job and use smartphones as the primary means of communication. Hence there isn't as much structure in place as Microsoft e.g. there are no email alias/lists to join.

Finding the right vanpool seems to be the most difficult task of the employee vanpool system.

Another task that ties in with the initial setting up of the vanpool is to coordinate with all the other people about logistics - where to rendezvous and at what time, who the designated driver will be, the backup driver and who the bookkeeper will be, which vehicle they want to get (King County seems to have a wide selection - anything from a minivan to a fully electric Nissan Leaf). King County has a well defined structure and defined roles. There are forms and brochures that explain these in detail and you also get a designated King County employee who will be your point of contact to help answer any questions the group has. King County also provides training for the designated drivers. Most of these initial setup tasks are performed and coordinated face

to face and then followed up via email. What we learnt from Leslie is that to become a designated driver was quite difficult, King County has set a pretty high bar. King County also provides the driver with a gas card which is accepted at all gas stations (Leslie pointed out that ARCO is the only station that doesn't accept it). Leon called out that since most employees didn't have computers, they had smartphones instead, it was difficult for them to deal with the King County paperwork/forms from work.

Once the Vanpool is setup the overhead seems to be quite low. King County will send a notice/reminder to the members of the vanpool when the vehicle is due for servicing. There are specific service centers which you need to go to. King County pays for the servicing of the vehicle and will provide a loaner car while the car is being serviced. There is very little that the members of the vanpool have to do for this task. They need to decide on the exact date and time of when to take the vehicle in for servicing. They usually do this via email or when they meet for their daily commute.

One other task that needs to be performed by the bookkeeper is to log the total number of miles. Leslie, who is the bookkeeper of his Microsoft employee vanpool, says he does this once a month by updating a OneNote page that he maintains on his work PC. Leon on the other hand maintained logs daily and it was quite inconvenient for him.

The vanpool that Leon was a part of ended when other members of his vanpool left the company and Leon could no longer find people who were interested. Leslie's vanpool on the other hand seems to be running smoothly and they get pinged from people who want to join regularly but end up having to turn them away because of lack of space. The difference at the two companies can be attributed to the following -

- At Snoqualmie there is a high churn in the employees which causes the people in the vanpool to constantly change hence you need to be constantly recruiting/finding people which is not easy especially since there are no email addresses where you can advertise broadly. At Microsoft the # of people is a lot more and there isn't as much churn. Once the vanpool is formed the frequency at which people leave/join is relatively low. The distribution lists also make it a lot easier to broadcast.
- At Snoqualmie the incentives don't seem to outweigh the hassle of keeping the vanpool going. The incentives are - you don't have to pay to park up close if you're in a carpool and you can enter a drawing for gift cards (\$5 denominations usually). At Microsoft the employees are incentivised by the fact that they are saving gas money and saving miles on their personal vehicles and that seems incentive enough.

Craigslist Rideshare

The users of the craigslist/public rideshare can be categorized into 2 groups based on the intent - drivers and riders. Drivers are the folks that own the vehicle and are looking for riders to join them on the journey primarily because they want to save money on gas. Some drivers also do it because they have a sense of community and want to be able to help others out. Riders are the

folks that don't have/won't be using their own vehicle and are looking to get a ride in order to save money.

The drivers post the details of the rides that they are taking as an advertisement on Craigslist to look for interested riders. This task can be performed from anywhere but is usually performed from a laptop/pc. We didn't interview anyone who was a driver but gained significant insight by performing the role of a driver ourselves (we posted on Craigslist for a trip from Seattle to Olympia) and by looking at/studying other posts on Craigslist. The driver usually posts the information/advertisement the week of the journey - usually no more than 5 days prior to the journey. They call out their starting point, destination and a rough description of the route. They also call out the date/dates that they are planning to leave. There is no set form/wizard to fill out - it's just free flowing text; unless the drivers have performed the task several times or have spent time reading other posts, there is structured way to know everything that's useful that needs to be explicitly called out.

The riders search Craigslist to find the ride that's relevant to them. Again there are very basic filters available on Craigslist and hence in order to find the "right" driver the rider has to go through several posts before landing on the one that best matches his/her need. This task can also be performed from anywhere. Steph who signed up as a rider for the post we had put out on Craigslist mentioned that she usually does this from her laptop. The task is usually performed 2 days prior to the day of travel. Steph said that she usually checks no more than 24-48 hours before the trip. The riders glean as much information about the trip as they can from the advertisement and once they find a trip that matches their needs they contact the driver via Craigslist.

The driver and rider will then figure out if they are indeed a match or not by using the email/messaging service that Craigslist provides. If they are a match the driver will revoke the posting from Craigslist. The driver and rider exchange contact information - cell phone # or email and will then go on to coordinate the journey.

The task of coordinating the journey usually entails in figuring out the exact time of departure, the pick up location and other details such as how much gas money etc. Steph mentioned that she usually prefers to perform this task by texting the driver as opposed to email or phone call. Steph also called out that she asks to be picked up at a crossroad or public location because of safety concerns of giving her address away. The night/day before travel the driver and rider confirm that they are each still committed to taking the journey and the next day they rendezvous. Steph mentioned that safety was definitely something that concerned her (not enough to deter her from seeking these rides) and mentioned that on meeting the driver if she didn't feel "right" she would not hesitate to abort/walk away from the ride.

Answers to task analysis questions

1. Who is going to use the system?

The users of our system are people who want to give or take rides over medium distances. A medium distance is somewhere inaccessible or incredibly difficult by public transportation and less than three hours. Using Seattle as a start point, Tacoma would be the lower bound while Portland or Vancouver BC would be the upper with the exception that Spokane/Pullman would be included in our definition of medium. Our target audience is people who would be early adopters of the system so somewhere between 16-30 years of age and a smartphone user. Users must have flexible travel plans and will typically be people that care about the local community / social aspects of ridesharing and about reducing environmental impact (carbon offsets) over affordable transit.

2. What tasks do they now perform?

Since such a medium range rideshare system does not currently exist, potential users have very limited options. They either simply drive by themselves, or attempt to find carpool buddies. This can be done by searching their social network but the odds of that working are low unless it is already a planned trip. Metro Vanpools provide a way to do this but require training, bookkeeping, and four other riders. The only option for ad-hoc medium range ride matchmaking is Craigslist. Users post unstructured offers that other users search. Once a potential ride is identified, the users coordinate by text, call, or email.

3. What tasks are desired?

The key task is matchmaking. Bringing together potential riders with potential drivers. Once a match is made, allowing both driver and rider to research each other becomes key for safety and peace of mind. Communication is still necessary in the process and a method of providing reviews is desired so the community could self-police itself.

4. How are the tasks learned?

The interface is designed to be picked up using intuitions and behaviors learned from other apps and the internet. Learning the tasks is motivated by necessity and the desire to find a transit option or provide one.

5. Where are the tasks performed?

While we envision that these tasks should be able to be performed anywhere, contextual inquiry suggests that they will likely happen from one of the endpoints for a route. Of the two most likely the start point.

6. What's the relationship between customer & data?

All data in the system is user generated content except for location data. Some of the users will

post rides and provide data. Others will find rides and consume the data. All of the data is very personal, tied directly to locations and behaviors of the individual users. Privacy of the data is a very real issue in the system.

7. What other tools does the customer have?

Users could potentially use bulletin boards at work or social networks such as facebook or twitter but these forms will likely be shown to the wrong audience for finding a match.

8. How do customers communicate with each other?

Contextual inquiry showed that text messaging is likely a preferred method for craigslist. People setting up office pools will likely talk face to face or email. In our ad-hoc system they will communicate through the mask of a user name in our application.

9. How often are the tasks performed?

Since the distance in focus is of medium length and someone traveling that distance every weekday is better served by a Vanpool then these task would be performed by a user from every weekend to twice a year to maybe once in a lifetime.

10. What are the time constraints on the tasks?

Users usually performed the matchmaking tasks two days prior to the day of travel and continued the communication until the end of the rideshare experience.

11. What happens when things go wrong?

There are multiple shades of wrong in the system. Best case and most common, you won't get a ride or a partner to ride with. Worst case and least common, you have a serial killer to contend with. While this sounds silly, it is a very real fear of many users and specifically current craigslisters and must be designed around.

In existing systems, a ride that falls through at the last minute leaves the rider with a bad customer experience and they might be turned off future rideshares. Also, the customer is frustrated about the lack of any facility to provide negative feedback about the driver. In Catch-A-Ride, the rider would be able to negatively review a driver that fails to follow through with a ride.

Three tasks your application will support, one each of easy, moderate, difficult (1.5 pages)

This section outlines three main tasks supported by our prototype designs and motivated by our contextual inquiry research. A summary list of each activity and its associated difficulty is below.

The remainder of this section provides detailed paragraphs reviewing each task and discussing a variety of implementation options.

1. List a ride - medium
2. Find a ride - hard
3. Review the driver - easy

We found that listing a ride represents medium difficulty for users. Listing a ride requires input of both optional and required parameters ranging from departure date to destination location. Additionally, some user preferences like allowing pets or smoking can be merged from the user's profile in the application. Although the quantity of input fields can be large (over 10 fields in some of the prototyped designs pictured in the next section), the workflow to list a ride is very linear, so users should only expect medium difficulty while working through this long task that has few error conditions.

Catch-A-Ride's hard task is finding a ride where we like to draw a comparison to eHarmony's matching algorithm that requires pages of answered questions to generate a good companion. The wide variety of options to present available ride data is just one design consideration for this task. For example, users could be required to answer questions such as desired arrival time before seeing any of the rides available in the system. Alternatively, users could be presented with all the available rides sorted by either posting date or departure date. In the alternate example, users would access a filter panel to cull the available list down to good matches. Finally, any good matches would further be narrowed by inspecting each individual ride's details where a passenger might evaluate the driver's profile, expected route, trip cost, and eventually contact the driver. Overall, the task of finding a ride represents the most complicated workflows within our application.

Reviewing a driver and other users within Catch-A-Ride is a surprisingly easy task given its benefit to passengers hungry for assurances of safe, reliable ridesharing. Reviews can be conducted when the driver initially picks up the rider, during the trip, or even prompted by followup email or notifications pushed to the application hours following the ride. The user's GPS location data could be a valuable confirmation that the trip actually occurred. Logistically, users might be able to navigate to the review submission form from a history of completed trips if a passenger forgot to submit a review during the trip. Our interface allows users to submit zero to five star ratings on dimensions of the trip such as overall experience, driver, car, and route. The interface would also support freeform text for any additional comments. Since users are generally familiar with submitting ratings at e-commerce sites or satisfaction surveys, we feel that submitting reviews in Catch-A-Ride would be an easy task. Note that we do assume the existence of external, long-lived accounts, such as those provided by Facebook APIs or written natively within the application.

Finally, the application supports a menu system allowing users to start each of the primary tasks. We did not initially identify a menu system as a core task, but its implementation options

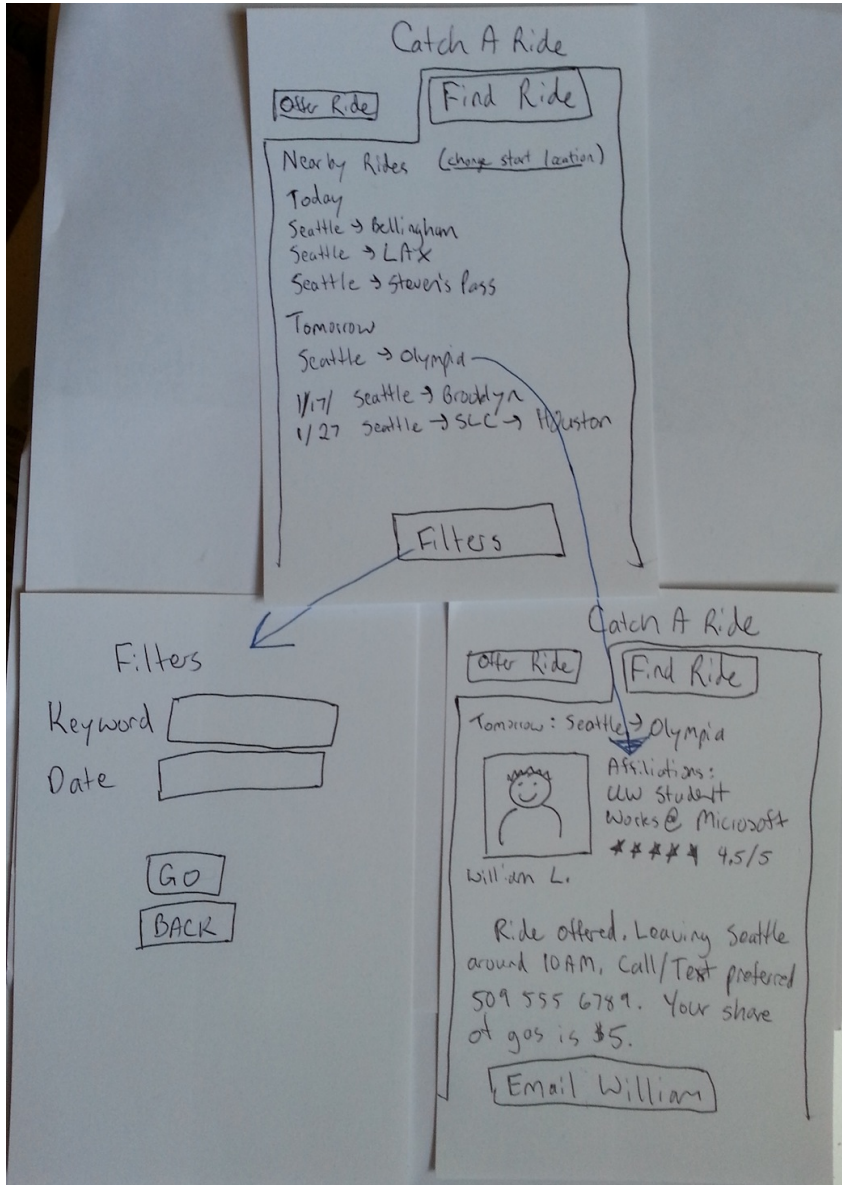
make it a notable addition to the discussion. For example, a tab navigation system might default to the ride offer form as soon as the user opens up the application. Alternatively, the user could see several buttons to start each task from a main menu. Another option is a dashboard that displays quick actions associated with a couple of the user's upcoming rides or previously finished trips that need reviews. These menu system options are explored below in our initial interface designs.

Storyboards illustrating initial 3 interface designs (mostly images)

Our team prototyped 3 initial interface designs to facilitate the core tasks of finding a ride, posting a rideshare, and reviewing a driver. This section includes photos or scans of each design's core tasks, as well as brief comments to note design choices.

Note that each design and associated commentary text is on individual pages following this page.

Design 1 - Find a Ride flow. Users see a list of available rides sorted by departure date. Clicking a ride, such as Seattle -> Olympia prototyped below, shows a detail screen that includes profile information about the driver. Clicking Filters gives the user keyword and date search.



Design 1 - Review flow. Users would reach this form by following a link emailed after the rideshare was finished.

Catch A Ride

Ride Feedback

Rate William ★ ★ ★ ★ ★

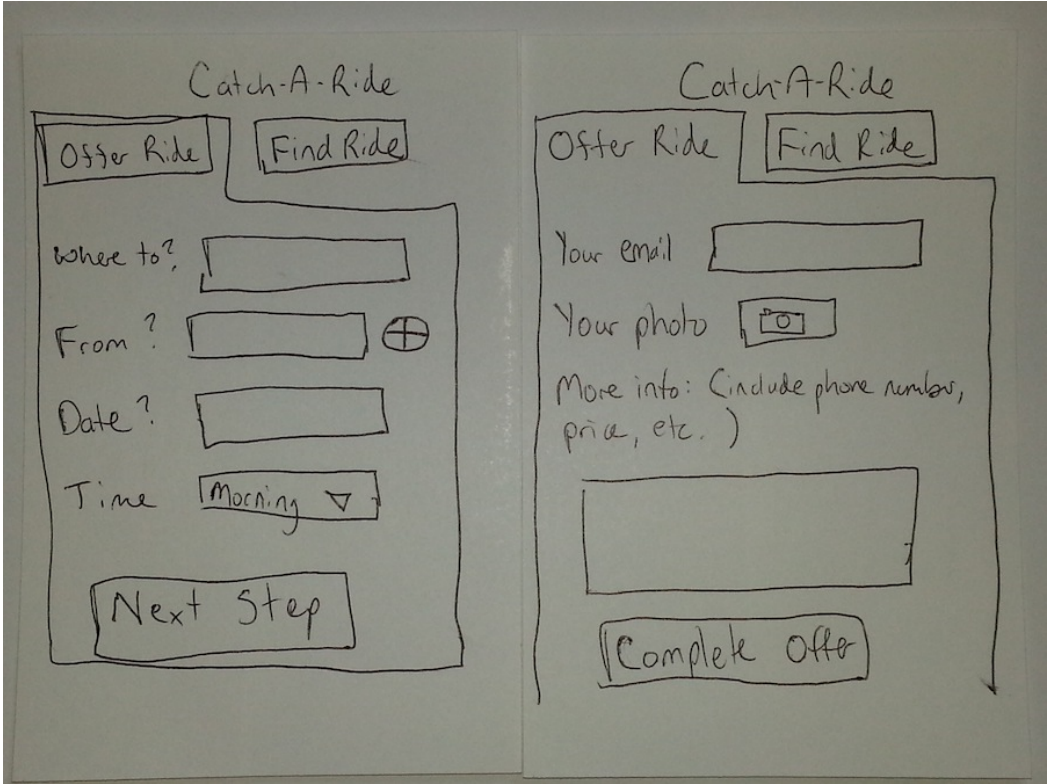
worst best

Add Comments (shared publicly):

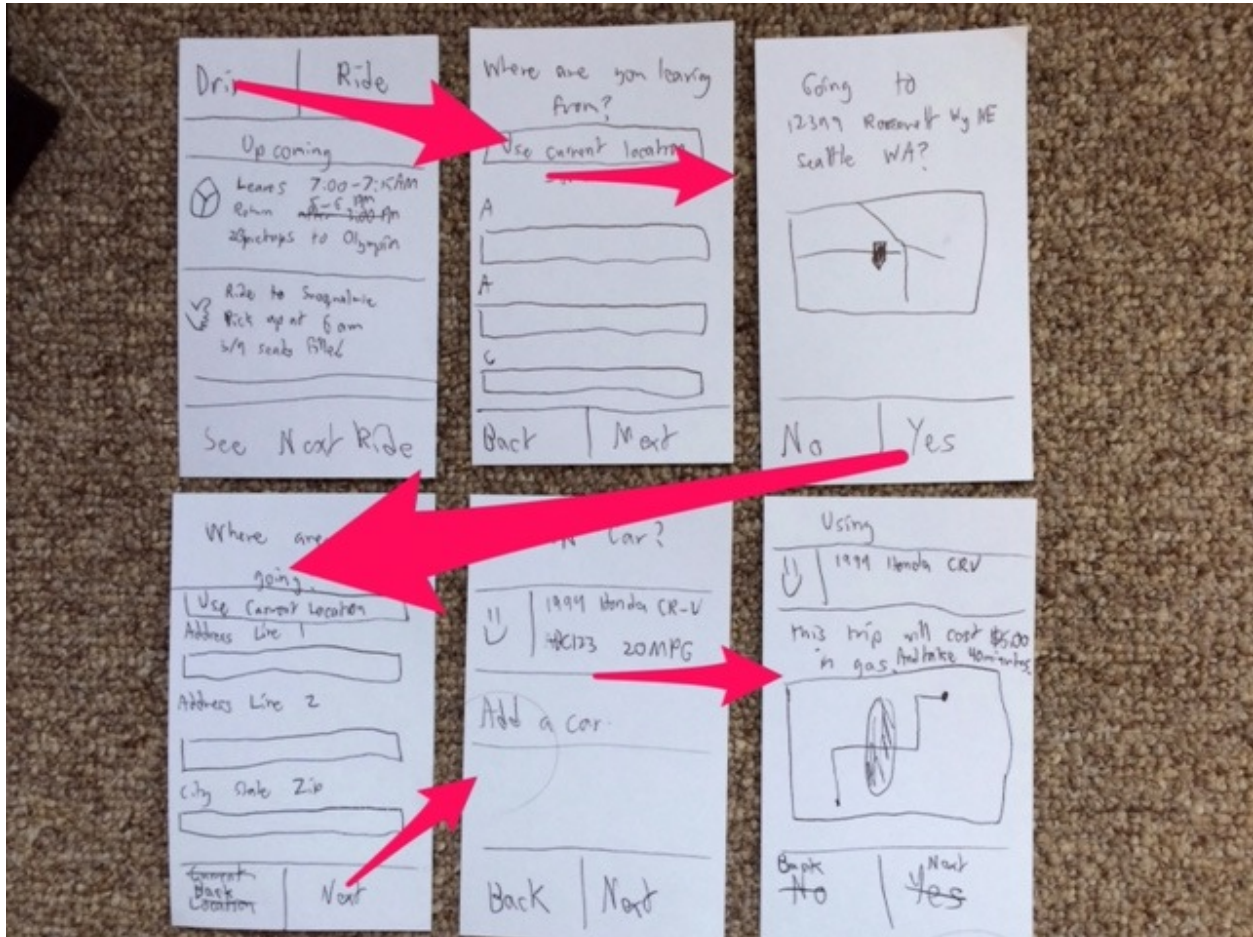
Submit

Note: Users get links from email

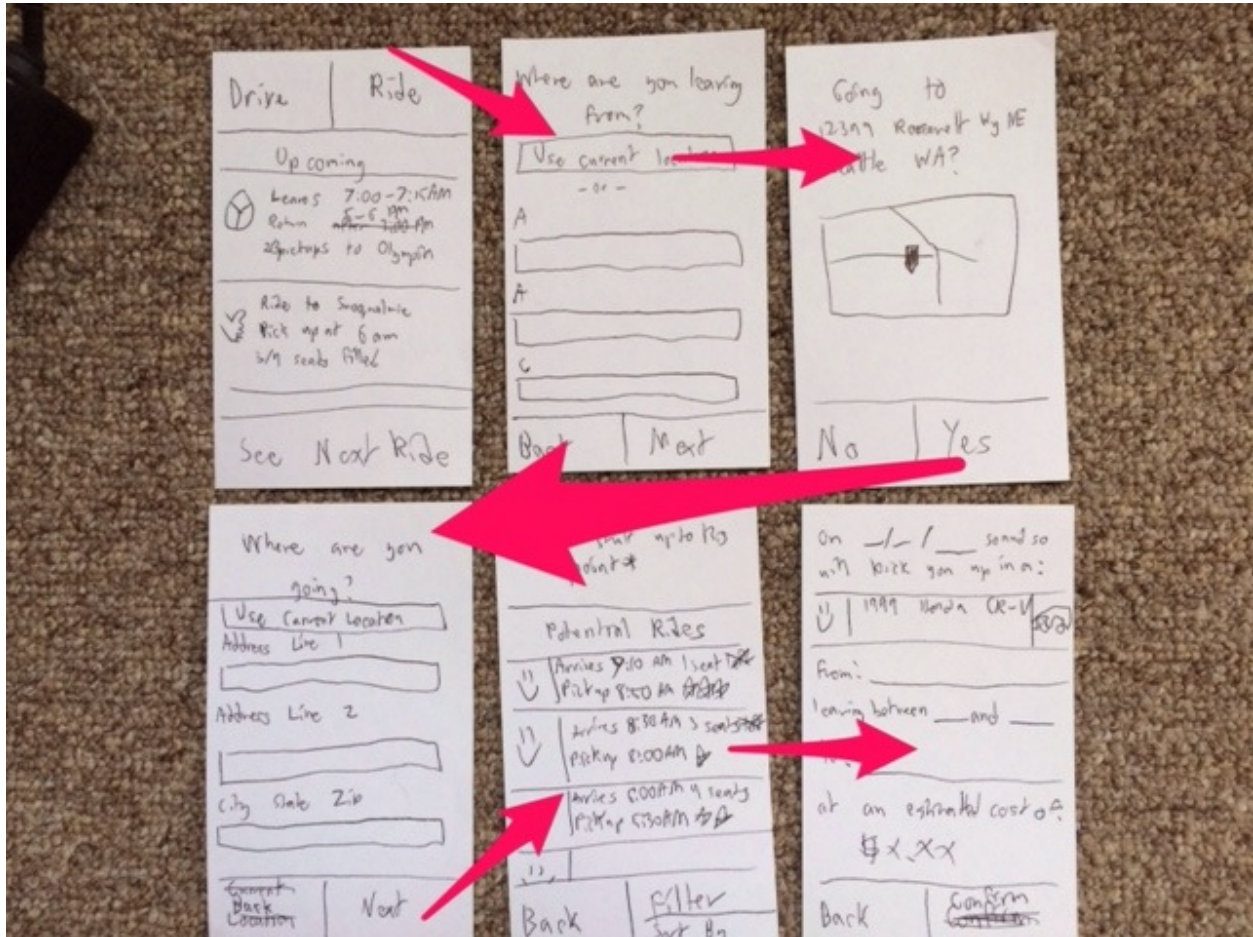
Design 1 - Offer a ride flow. The left view is the main screen when the application initially opens. Clicking next step opens the right view. Notably, this prototype assumed that users had to re-enter their email and photo on each submitted ride offer. After reviewing this prototype, we decided to assume that reusable user profiles existed and we dropped input for driver's email and photo from this workflow.



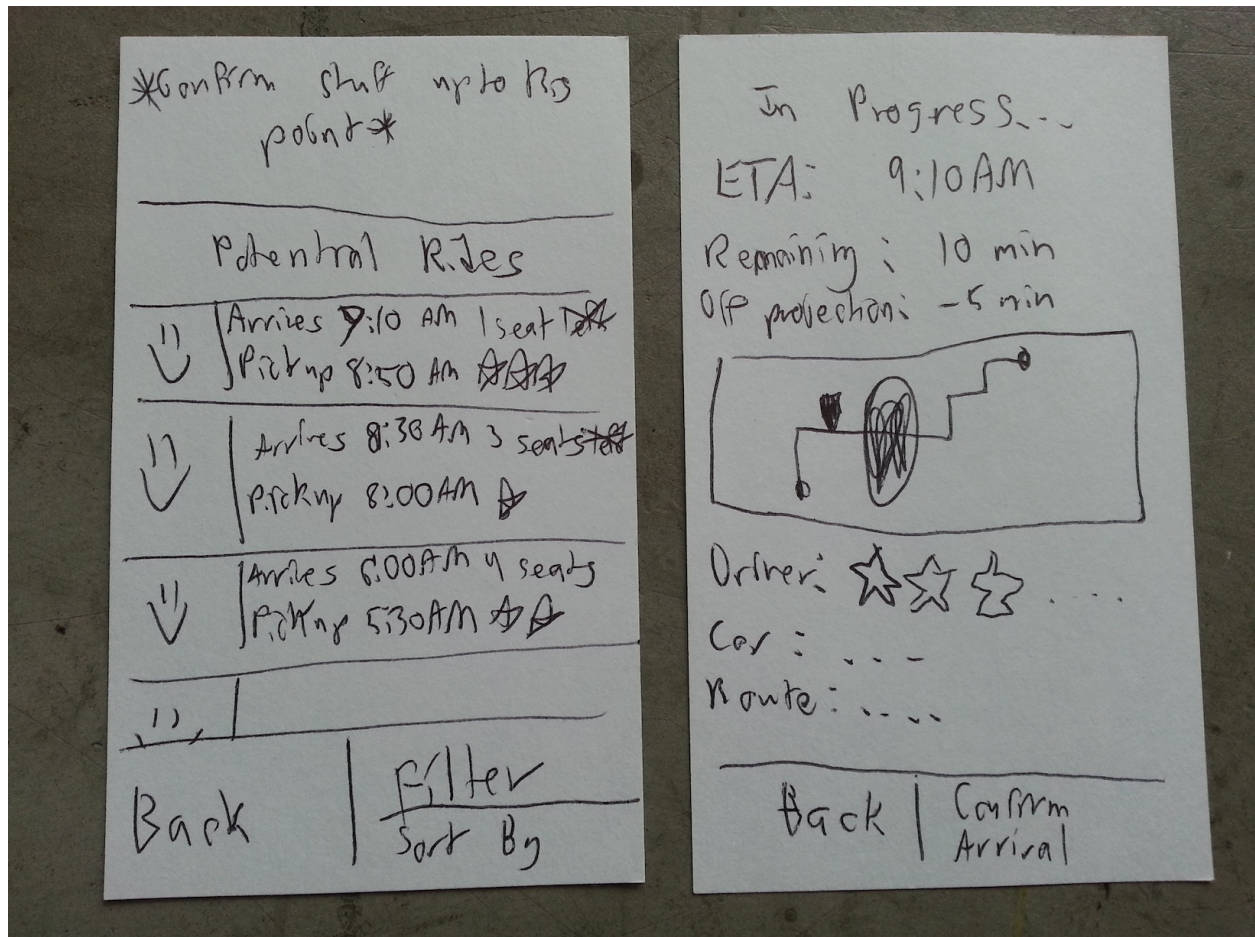
Design 2 - Ride offer flow started by selecting 'Drive' at the top left of the home screen. As incremental information is submitted, users see a map of the starting location (top right), details such as MPG of the car used for the drive (bottom middle), and a full route overview confirmation before submitting the offer (bottom right).



Design 2 - Find a ride flow. After entering departure info (top middle), users see a map confirming the starting point (top right). The matching information takes destination info (bottom left), and displays a list of appropriate available rides (bottom middle). Clicking an individual ride shows a detail screen (bottom right).

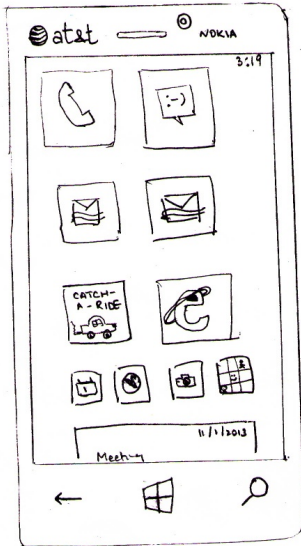


Design 2 - Review flow. The right screen shows information such as ideal route while the ride is in progress. Users could rate the driver by clicking the appropriate number of stars.

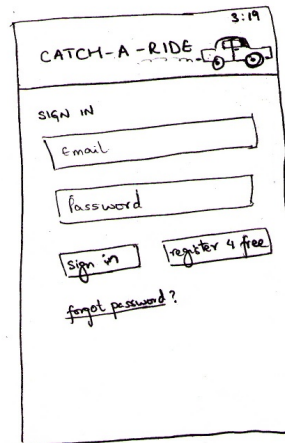


Design 3 - Ride offer flow. Login screen shows integration with long-lived user accounts.

① Give a ride - page 1/2

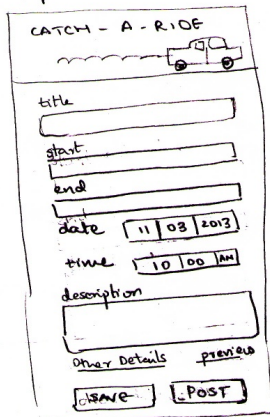


①



② Enter correct email & correct password

① Tap 'Catch-A-Ride'



④ click 'other details'



③ Give a ride

Design 3 - Ride offer / Find ride flow. Advanced filters are shared with the find a ride workflow to support preferences like smoking allowed, luggage capacity, and availability of in-route stops to pick up and drop off riders.

Give a ride - page 2/2.

②

CATCH-A-RIDE

Time: Give or take 30min

Route:

- On route pick up/drop is allowed
- Point - 2-point only

Smoking allowed:

- Yes
- No

Kids allowed:

- Yes
- No

Luggage allowed: one 25lbs bag

SAVE DONE

Not flexible
Give or Take 3hr
Give or Take 2hr
Any Time

None
2 25lbs each
unlimited

CATCH A RIDE

title: RIDE TO COLUMBIA GORGE

start: SEATTLE

end: COLUMBIA RIVER GORGE

date: 11 | 03 | 2013

time: 10 | 00 | AM

description: Going to see DMB on Sunday. Its a day trip for me.

other Details previous

SAVE POST

⑤ Hitting 'save' - saves the settings and keeps you on the page. Done takes you back to previous page.

⑥ click Post

not Done

CATCH-A-RIDE

sign out

Give a ride

find a ride

current Ride

History

Profile settings

enabled since there is a ride posted

CATCH-A-RIDE

Your post has been successfully published

click here to view post

Main Menu

NOTE - if user clicks current ride he/she should be able to 'edit' ride

⑦ not Main Menu

Selected Interface Design (2 pages)

Part a. Which design & reasoning for choice

We chose to go with Keith's design skeleton (Design #2 above) and bring in a couple features from the other 2 designs.

We chose Keith's design primarily because it was a minimalistic wizard that steps the user through each task and had the extensibility to provide the user with a boatload of extra settings (for advanced users). It also made use of the GPS functionality, which all smartphones now have inbuilt, that showed/shared the planned route and provided live tracking throughout the journey. This helps address the safety concerns that all ride-sharers have while partaking in these types of journeys.

From the other 2 designs we pulled in these features -

- Initial sorting of available rides presents a list interface with earliest departure at the top and latest departure at the bottom. This choice was made because our user research showed a preference for searching for a ride that leaves within the next 48 hours.
- Advanced filtering for both posting/listing and finding a ride allows the users to specify preferences such as Gender, Age group, Smoking preferences etc. We believe this will gain help with the safety concerns that users tend to have.

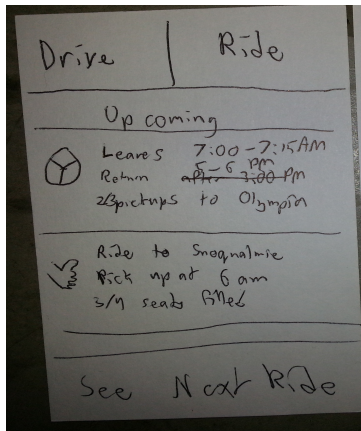
Part b. Functionality summary (what you can do with it)

- Main status screen to present the state of the system and offers start points for the separate tasks/flows.
- Ride screens to see the state of a specific ride in full detail. These screens are contextual and change based on the state of the user, usually detected by location information on the user (about to get picked up, on the way to get someone, in ride, post ride, etc).
- Ride screens must also provide a messaging mechanism for masked communication.
- Ability to list cars you own and can offer for rides.
- Ability to list a ride. This included setting start and end points, time requirements, car details and other specific preferences.
- Ability to search rides. This includes again setting start and end points, and time requirements. There must also be a faceting feature to the search to filter drivers who do match users requirements. Sorting is also important but will default to time leaving first in our designs.
- Profile and rating information for each user. The system also provides information about the number of rides, and distance travelled by each user. This helps alleviate the safety concerns most riders have about travelling with a stranger.
- Ability to provide feedback once the trip is complete

Feature that would be needed but are not represented in the mocked system:

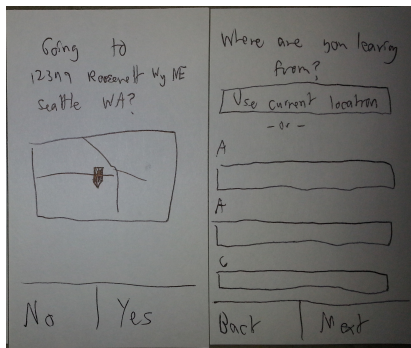
- System for automated payments or for suggesting price. While many do carpool to save money, solving the money exchange problem is only important if people are actually carpooling. The contextual inquiry showed that a system without this is viable, making this a “nice to have” but not necessary to solve the problem.
- Profile screens or profile signups. It is assumed that all users once into the mocks have a profile created and that it is unique to them individually.
- Inbox system or messaging outside the context of a single ride. While such a feature would be great to help create recurring carpools, that is not the use case in question and would be better served with a vanpool.

c. User interface description (how you use it, referencing sketches)



The main screen provides a jump off point to the three major use cases of the system. Those being find a ride, list a ride, and take a ride. These three are all larger buttons as they are intended to be used the most. The drive and ride buttons at the top will take the user into the “list a ride” and “find a ride” flows respectively. The “See Next Ride” button shows the details screen for the next ride to be taken. The middle of the screen is a scrollable list of all rides the user is involved in and clicking on any takes them to the details view for that ride.

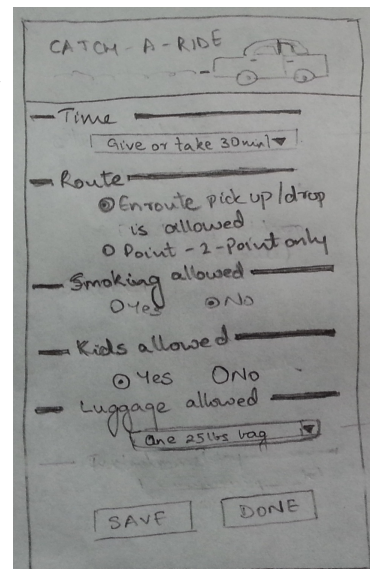
Task 1: List a Ride



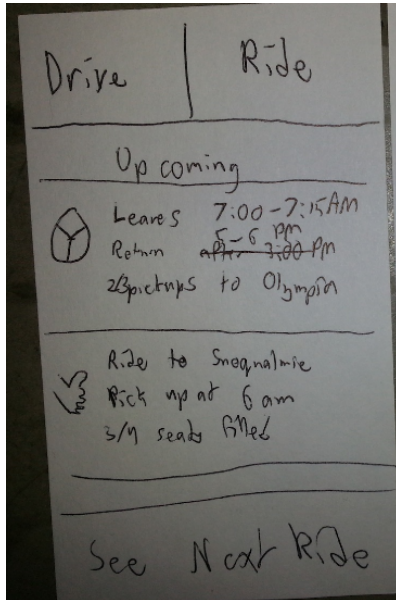
The “list a ride” flow has a set of information it needs to get from the user. The user is walked through a wizard of sorts where each required piece of information is asked and confirmed. There are back and next buttons on each screen to indicate that nothing is final yet. These switch to yes and no’s on the confirm screens. The required information to gather is where they are leaving, where they are going, which car they will take, when they need to be there, and their timing flexibility for the ride. Anytime the user inputs one of these, a status

screen confirms the input so as to reduce the need for backtracking. Once all required inputs are collected a final summary screen is presented to the user. It explains their route, their time windows, confirms the car choice and makes sure the user knows what is happening next. The next button is replaced with confirm to indicate that this is the final step.

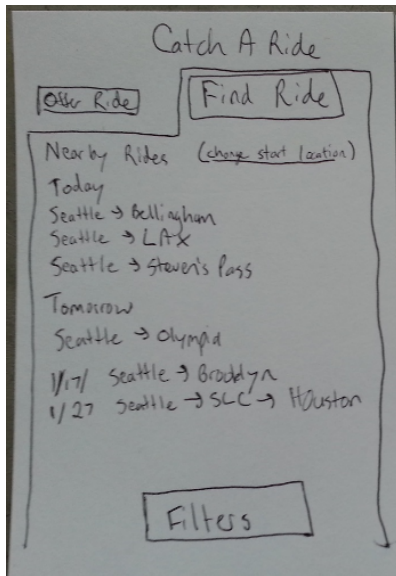
Also an advanced options button is available to allow the user to specify things optional to the system but required by the user. These are all defaulted to be very open so as to allow for the largest pool of riders but can be tightened at the users request. Once the user selects confirm they return to the main screen where their newly listed ride is there for them to see.



Task 2: Find a ride -



Start from the Main screen of the application - which shows you the list of current upcoming rides that you are participating in (blank if you have none). Click on the Ride button - top right hand side



This shows a list of the posted rides in your area (which is automatically detected from your current location) - you have the option to change the starting location. The list is sorted by earliest departure at the top and latest departure at the bottom. There is a button at the bottom for additional filters. Click the Filters button.

CATCH-A-RIDE

Departure Date

User Rating = 5★
 ≥ 4★
 ≥ 3★
 ≥ 2★
 ≥ 1★
 newbie OK
 → 0-99

Min # of trips completed

Member of my Groups
 Yes No

Smoking
 Yes No

Gender

of people travelling

SAVE APPLY

ANY ON BEFORE A-RIDE

Make Favorite other 1 only <4 dont care

The advanced filters section allows you to specify details like departure date (note it's a drop down in the paper prototype but we plan to change it to a slider control) and time you want to leave. The minimum rating of the driver. If the driver needs to be affiliated to a group that you're a part of (the user is able to associated themselves with various groups e.g. UW student, through their profile settings). Whether or not you're comfortable with the driver being a smoker/not. You can also specify your preference for the gender of the driver, age group of the driver and # of people you're willing to have in the car. Click Save on this page.

Catch A Ride

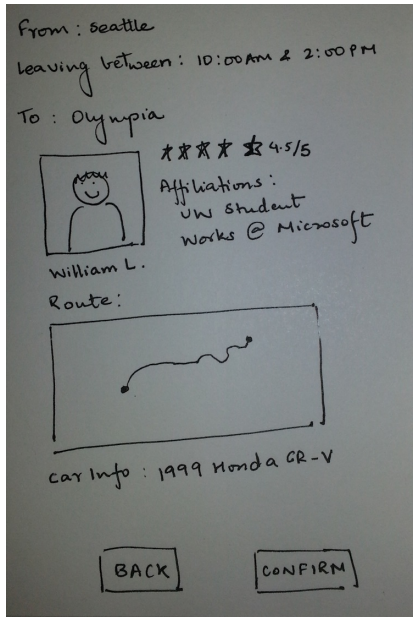
Nearby Rides (change start location)

Today
 Seattle → Bellingham
 Seattle → LAX
 Seattle → Stover's Pass

Tomorrow
 Seattle → Olympia

1/17 Seattle → Brooklyn
 1/27 Seattle → SLC → Houston

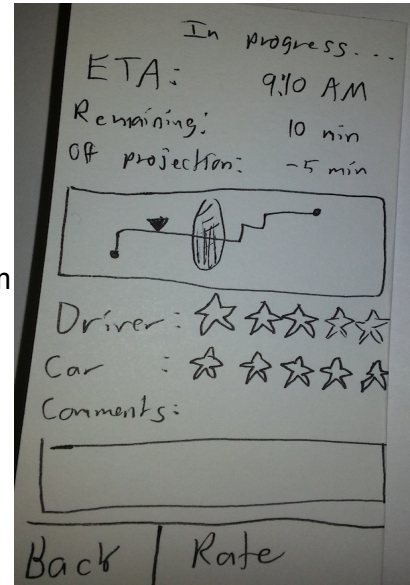
Now you're back to the postings that match your criteria. You can now select the posting that you want to get more info on and view the details of the offer e.g. driver's profile, rating, affiliations, actual route that the driver intends to use, vehicle info etc.



You can hit Confirm to select this ride or Back to view more rides. Hitting Confirm takes you back to the first screen of the app and now this ride will show up in the list of active rides.

Task 3: Submit a Review

In the details screen for a ride are several important features. It communicated the status of the trip, shows the route, and provided the feedback mechanism. The map should also include the current location and track with the user. The feedback is simply a set of empty stars next to various metrics. On clicking a star the rating is not saved. The ratings can be adjusted but do not all need to be filled out. The text is free form for the comment and is submitted on clicking rate. From then on the rate is disabled as there is logical "next" screen. The list of information presented should not be considered exhaustive as an actual implementation would no doubt reveal that much more information is available to be presented to the user. This is however a minimum feature set that at least needs to be show.



Three scenarios corresponding to your tasks

1. Sarah wants to ride to Olympia, but her friend is scared about craigslisters. Sarah finds out about Catch-A-Ride and opens the app to see what it is all about. Once signed in she see the main screen and clicks "Ride" in the top right corner to see about finding one. The initial screen shows lots of rides radiating out of seattle but all of them are for today. She clicks filter and restricts her results to those leaving on friday which is two days from today when she plans on going. She clicks ok and sees a much better list of available rides but most of the first couple are going north. She opens the filters again and fills in the address of her friends house she is trying to get to. Now there are only three matches. She clicks on the one with five stars. It all sounds reasonable for the pick up and drop off times. The route is clearly displayed. What makes her feel the best though is being able to see the drivers profile. He seems to be a University of Washington alumni and works at Microsoft. He has given many rides before and all the reviews look good. She decided she feels comfortable with the idea and clicks confirm. She is taken back to the main screen and sees the ride listed on the main page. She tells her friends about the positive reviews for the driver and her nerves are settled about the situation.

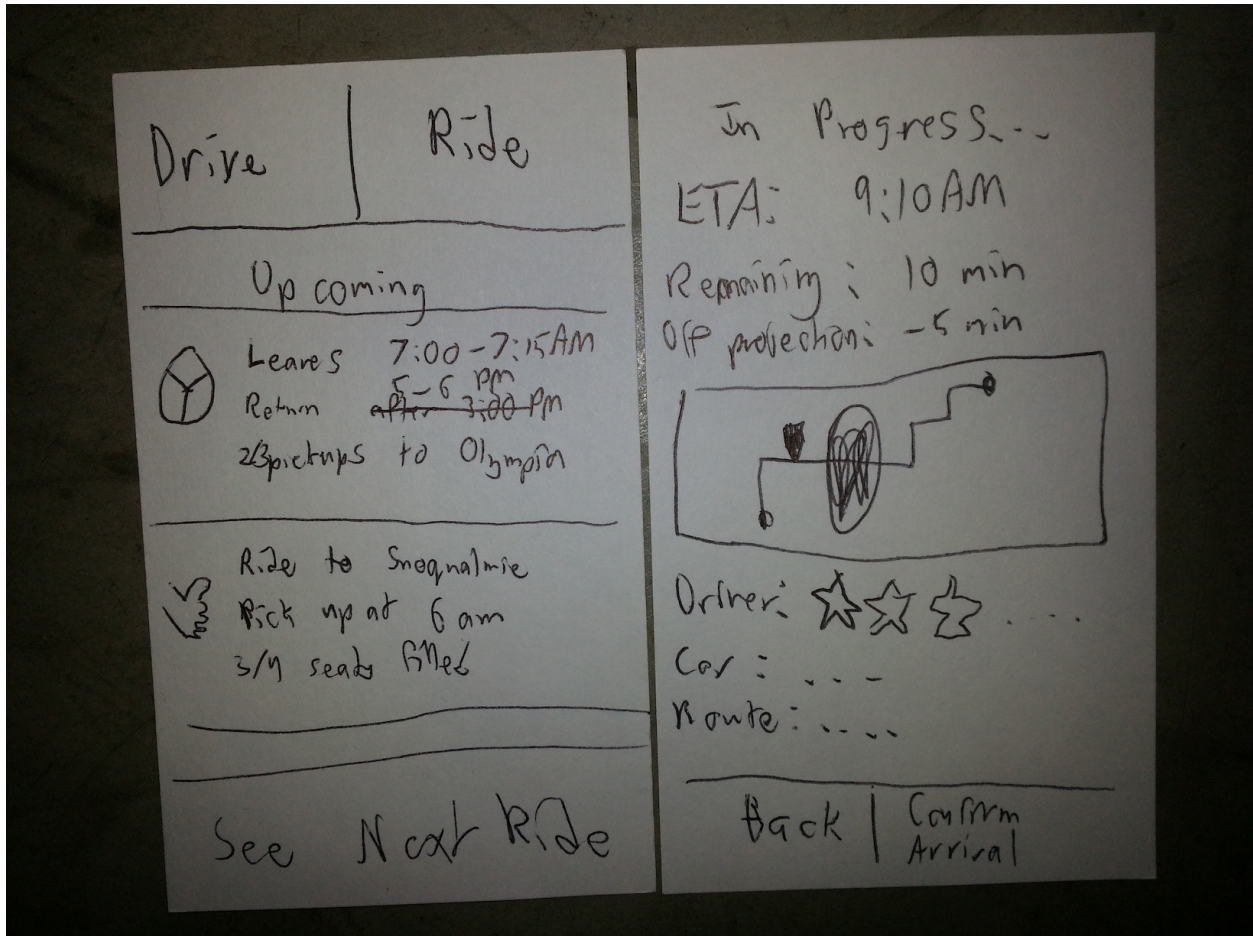
2. Gerry the ride giver likes the social aspects of ridesharing and expects to drive to Portland on Friday. He opens Catch-A-Ride because he would like having someone to talk to and getting HOV access down I-5 wouldn't hurt either. He hasn't used the system for a while so his home screen is empty as he clicks on the "Drive" button in the upper lefthand corner. He is at home so clicks "Use Current Location" since he plans to leave from home later. The next screen confirms it found his address alright and he clicks "Yes" to continue. When asked where he is going, he types in the address downtown he needs to get to for his meeting. He confirms that one as well and is then asked which car he is taking. He sees his car still listed from last time and selects and confirms it. The system is now asking when he is going and when he needs to get there. He picks the date and enters the time fifteen minutes before his meeting so he knows he has time to walk. Gerry is a morning person and is willing to get up quite early if it means HOV access. He selects 5am as the earliest he would be willing to leave. He clicks next and is presented with a summary of all his choices. He sees a final options list and notices he can restrict the number of passengers. Since he really only wants HOV access and doesn't want to drive all over town in the morning he lowers the available seats to just one. He confirms that, the interface now showing his one seat restriction rather than his 4 his car could seat. He clicks confirm and is taken back to the main screen. His ride he offered is sitting there in a summary form. Now he sits back and waits for someone to find him.

3. Sarah found Gerry's rideshare on Catch-a-Ride. It worked out awesome! He picked her up right on time, they used the carpool lane and the app tracked their progress every step of the way. She has been watching the ride status screen during the trip and notices they are actually getting into Portland ten minutes early. He drops her off and with her extra time before she meets

up with her friend she takes some time to submit a review for Gerry. Without changing windows she rates him five stars, rates his car five stars and writes in the text box, "super clean car, a++++** will catch-a-ride again!" and clicks submit review. She closes the app and goes about her day.

Appendix: Any additional sketches of important screens

The application dashboard (left) shows timely information about upcoming rides. During a ride, the status screen (right) allows users to rate the driver, car, route, as well as share a real time map with friends.



Where are you going?

Use current location

Address Line 1

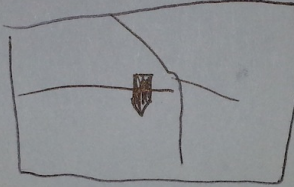
Address Line 2

City State Zip

Current Location
 Back
 Location

Next

Going to
12349 Roosevelt Wy NE
Seattle WA?



No | Yes

Where are you leaving from?

Use current location

- or -

A

A

C

Back

Next

When are you leaving?


Anywhere between

8 | ▾ AM | ▾

and

9 | ▾ PM | ▾

on

12/12/2013 

Back | Next

Is this a round trip?

No | Yes

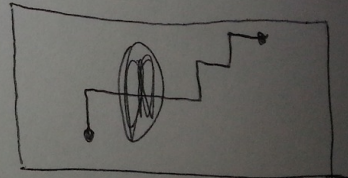
On _ / _ / _ you are driving

11 | 1999 Honda CRV
MPG

From _____

leaving between _____ and _____

to _____



Back | Confirm

Where are you leaving from?

Use current location
- or -

A

A

C

Back | Next

Which Car?

|| 1999 Honda CR-V
ABC123 20MPG

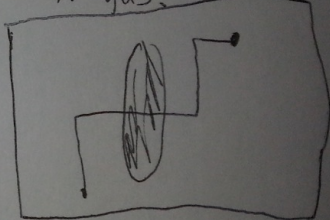
Add a car.

Back | Next

Using

↓ 1999 Honda CRV

this trip will cost \$5.00
in gas. And take 40 minutes.



Back No | Next Yes

Confirm stuff up to his point

Potential Rides

U | Arrives 7:10 AM 1 seat ~~left~~
Pickup 8:50 AM ~~AAA~~

U | Arrives 8:38 AM 3 seats ~~left~~
Pickup 8:00 AM ~~A~~

U | Arrives 6:00 AM 4 seats
Pickup 5:30 AM ~~AA~~

U |

Back | Filter
Sort By

On ___/___/___ so and so
with back you up in a:

U | 1999 Honda CR-V ~~CR-V~~


from: _____

leaving between ___ and ___

to: _____

at an estimated cost of
\$X,XX

Back | ~~Confirm~~
Confirm

Drive	Ride
Up coming	
 Leaves 7:00-7:15 AM Return after 7:00 PM ^{6 PM} 23 pickups to Olympia	
Ride to Snoqualmie Pick up at 6 am 3/4 seats filled	
See Next Ride	

Where are you going?

Use Current Location

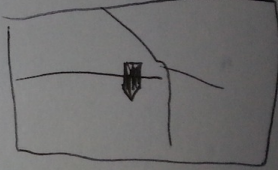
Address Line 1

Address Line 2

City State Zip

~~Current Back Location~~ | Next

Going to
 12319 Roosevelt Wy NE
 Seattle WA?



No | Yes

When?
On

12/12/13

Around

8 00 Am

Back

Next

When must you arrive?

00
8 Am 12/12/13

To leave at 8 Am ^{on 12/12/13} you
must leave by 7:35. What
is the earliest you would
leave?

7 00 Am

Back

Next