Next Ride – GR6: User Testing
6.813 Spring 2012

Chidubem Ezeaka Alex Lesman Guiseppe Zingales

Design

The final design of our interface comprises four tabs and three pop-up, with the core interaction happening on the scheduling canvas. The tabbed design is very similar our original paper prototypes but the pop-ups were added because of feedback from computer prototyping. After interviews with users from our target population, we resolved to made design decisions that maximized efficiency, safety and simplicity.

After user testing with the computer prototypes we decided that changing the request, offer and reserve pages to pop-ups would increase the efficiency of our design and help with the user understanding of the page layout. This also gave the desired behavior where the navigation on the web browser doesn't take the user to the request, offer and reserve pages. To improve efficiency, user control and consistency, we also decided that the user should be able to edit requests, offers and reservations wherever they are represented in the interface.

The general layout of the interface changed from the paper prototype. We improved the home page by removing the "View Schedules" and "My Rides" buttons and instead opted to rely exclusively on tabs for navigation. This change also let the user get back to the home page whenever they wished.

The "to House" and "to Campus" schedule pages are identical and built around the canvas based schedule representation. Both pages are identical so to help safety we had decided to make each page its own tab so that the user is aware at all points which of the schedule pages they are looking at or editing.

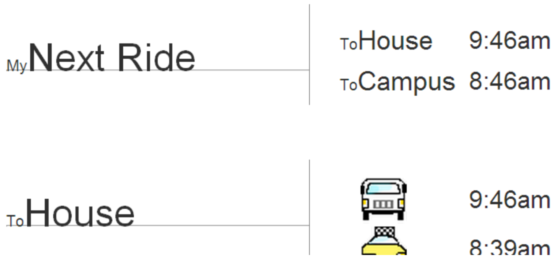
We also went through a lot of color scheme iterations and learned that the chosen colors make a much bigger impact than we anticipated. We ultimately chose a color scheme that the users found aesthetically pleasing, but also reduced visual clutter and provided good contrast and visibility.

Splash Page

This is the homepage of our project it is meant to give the user enough information to make decisions in case they don't have enough time to look through the schedule. It shows the next three rides in either direction and the user's next two rides, to facilitate double checking the next couple of rides. From the splash page the user can reserve rides, edit reservations and edit ride offers by simply tapping on the appropriate entry on the home page. This gives the users a very efficient way to use NextRide by providing immediate access to the information that is most likely to be relevant to them.

- Paper Prototype
    - As we said before this page didn't exist with the same purpose that it has now until after the paper prototype. Originally it was just a log in screen or it would immediately redirect to the schedule page.
- Heuristic evaluation
    - Not much we changed the layout of the page.

  - After much debate we dismissed concerns about the distinction between my next rides and to House. There was a discussion on if it was obvious if people created them themselves.
  - Instead we just moved the label left, instead of it being centered to improve space usage and improve clarity

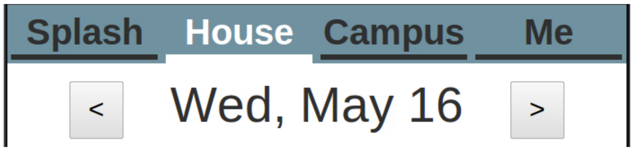


  - We added the ability to click on icons and reserve them right on the spot without going through the hasssle of going to the schedule page.

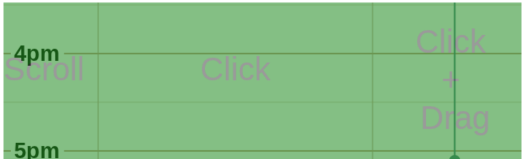
House/Campus Page.
The Schedule page as the most innovative part of our design had the most improvement over the iterations. The purpose of the schedule page is to present the user with information about available rides and requests for rides and allow them to efficiently interact with this information. The original design which had been developed after a lot of iterations amongst our group, the paper prototype, heuristic evaluation and user testing brought a lot of new information.

- Paper prototype:
  - Adding a header to the schedule to clarify what the different columns represent

  - Developed the idea of having a splash page for quick access of the information
- Heuristic evaluation:
  - The buttons to change the date, were positioned so that they did not shift position when the length of the date title changed.
  - Tabs were fixed so it was very evident what page you were on when.



  - Time was fixed so it was consistent with the local and the rest of the project (aka from 24-hr format to 12-hr format).
  - To reduce clutter on the schedule page, we decided to only write the times in hour intervals instead of 30-minute intervals. We also change the time format and de-cluttered the screen
  - We also added an element to help users who are new to the site. It is an overlay that appears the first n time you visit the schedule page that provides basic. This is meant to help with learnability issues.

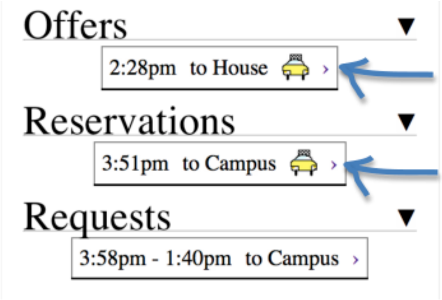


- User Testing
    - We discovered that on mobile our icons are a little small, especially when it is required to click on them directly. This is especially problematic when trying to click on a ride to reserve it because missing it will instead bring the offer popup.

Me Page

This page gives the user access to information that is relevant to them all in one place. All of the users offers, reservations and requests appear on the page and the user is able to view details and edit any of these elements by simply clicking on them in the interface.

- Paper prototyping
    - The me page and the splash page were created from scratch after paper prototyping we realized we did not have a clean way of allowing people to see what rides they had offered, requested or reserved.
    - The page originally was a settings page where you could only change you preferences like your name or phone number.
- Heuristic evaluation.
    - The widths of the everything on the me page were made constant to fix the problem below



Offers ▼

2:28pm to House ›

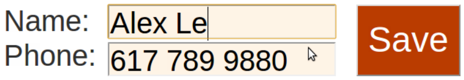
Reservations ▼

3:51pm to Campus ›

Requests ▼

3:58pm - 1:40pm to Campus ›

    - The save button was designed to re-enable if the fields were modified.



Name: Alex Le

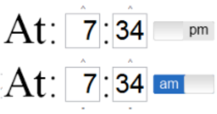
Phone: 617 789 9880

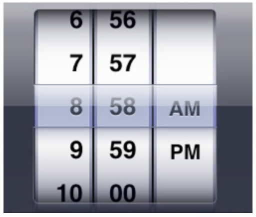
Pop-ups

In our interface we have 3 different pop-up menus: offer ride, reserve ride and make request. Each of these pop-up has a create and an edit variation. The pop-ups are the same on all of the different tabs of our application and provide a consistent way of inputting and manipulating information throughout the interface.

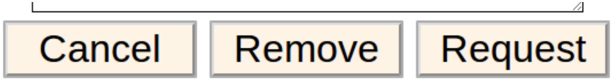
- Paper Prototyping
    - From our paper prototype the users discovered very little wrong with the request/offer/reserve pages.
- Heuristic evaluation
    - After some heuristic evaluation we decided to move to popups instead of going to a different page, because the clicking and bringing you to another page was a little too sudden and the complete change of scenery was startling to the user. This also improved safety and efficiency because the pop-ups loaded and close faster than a page reload.

- Our custom input method for time was dropped and instead became a text field which smart phones automatically interpret and provide a nice and consitent scroll wheel.





- We changed the look and feel of buttons so they actually looked and feel like buttons.



Implementation

Frontend

1. **Canvas**
   We decided to implement our schedule interface using multiple layers of HTML Canvases – one each for the time, offers and requests sections and another for the whole. This gives us better control of the whole interface and allows us to efficiently reduce the redrawing done and allow the interface to feel more responsive (especially important on an iphone). This does make it much trickier to make sure that events are getting to the correct handlers. We could still be much more efficient with this by opting translate the canvas and then redrawing only the needed portions instead of clearing and drawing a whole canvas every time.
   Because of the variance of sizes of different phone screens (and when you turn the screen sideways), we had to dynamically resize the canvas on page load to fit the screen appropriately.
   We also had to actions such as momentum scrolling and drag scrolling to the canvas schedule that users are accustomed to.
   We were making AJAX calls whenever the user made a new request, offer or reservation. Instead of giving the user instant visual feedback on the action, we decided to wait first for the confirmation from the server that the action was indeed completed and give an error if it wasn't. Depending on the network speed, this might make the interface seem unresponsive but we decided that it was better that than giving the user a false system state.
2. **Popups**
   We implemented our pop-ups as HTML divs with the display CSS attribute set to none. Whenever we wanted the popup to be visible, we set the display attribute to block.
   We later wanted the popups to be able to come up on any of the four tabs but our previous implementation as divs meant that we had to add the same html elements to all other pages (a lot of duplicate code).
   In hindsight we should have implemented these the same way as we did the tabs.
3. **Tabbing**
   We decided to add the tabs after we had already started with our implementation so we accomplished this by creating a JavaScript method that we called on leading each page. This method appends the necessary HTML to the html of each page. It felt like it was a bit messy but it accomplished the purpose.
4. **Platforms**
   We wanted our app to work on both mobile and computer web browsers. This meant that we had to do a lot more to be able to dynamically resize the pages.

We also needed to have handlers for events on each platform (such as touch vs. click handlers)

Backend

1. **Serving**
   The NextRide server is implemented using Node.JS server-side JavaScript. All of the HTML, JavaScript, CSS and other resources are all static and served statically from the server. On loading the JavaScript makes asynchronous requests to the server for data. The server replies to these requests with JSON data which the client side JavaScript parses and handles. Serving static HTML rather than generating it dynamically on the server (such as a typical PHP implementation) greatly simplifies server code and makes for easier debugging.
2. **Database**
   All of the data is stored in a MySql database. The data is normalized into 4 tables: users, rides, reservations and requests. When the client makes an AJAX request to the Node server, the Node server makes an asynchronous request to the MySQL database. The database reply is then processed and returned to the client.

Evaluation

We conducted user testing on three members of our target user population – ZBT brothers.
We briefed the users that our interface is supposed to be “a website to help brothers schedule rides to the house or to campus.” We didn’t give a demo or give too much of a briefing because we wanted them to go into the test with as little knowledge as possible so we can better observe any obvious learnability concerns.
We asked our testers to perform the following three tasks:

1. Charles is planning on going to campus today. He woke up at 11and needs to be on campus by 3pm, but besides that he doesn’t really care when he goes. He’s willing to give people a ride and leave whenever is more convenient for others. He is going to check to see when people have requested a ride. He sees that 2 people have requested a ride at 2, and then schedules a ride then. His car seats 5 (including him).
2. Alexis is on campus and he needs to get to the house, Saferide and the T have both stopped running for the night, its too cold to walk and he wants to get a ride but he would not mind getting a cab. He is flexible on when he is leaving but he knows he can’t leave in the next 30 min.
3. Jorge needs to get to campus before 1pm. It’s 11am right now and he doesn’t care when he goes as long as he gets a ride before one. Jorge looks at the available rides and reserves himself a spot on a ride in Yifan’s car at 12:30pm.

Bad

1. The schedule page presents information in a novel way and because of this it is not very learnable. The informative overlay isn’t very readable and not completely clear.
2. People tried to click and drag offers like a request. It was not clear that unlike requests, rides are at a fixed time.
3. Some users tried to drag-scroll on the rides column which was not an allowed action.
4. It wasn’t completely clear to users if “House” and “Campus” meant to or from.
5. The pop ups were made to display information to the user in ‘prose’ to make it clearer what the information was, however some users found this more confusing.

Most of our problems stemmed from the novelty of our interface making learnability bad. We have tried to fix this by adding prompts for the new users to explain how the interface works.

Good

1. The splash page gave the user almost all of the information they typically wanted to see in one place.
2. Once people understood how to use the schedule page and developed an accurate use model they found it to be convenient to use.
3. The drag-to-request and drag-to-scroll features are very intuitive and convenient on a mobile phone.
4. Understood the distinction between house and campus.
5. Once people read the tool tip they caught on pretty quick.

Most users thought that the application was good for efficiency and safety.

Reflection

Nextride was supposed to be a mobile website but we like the idea of also having it work on desktop. We all agree that it would have been helpful to make up our minds on the target audience and platform for our application before we begin design iterations. As a consequence, we didn't start early enough to test our website on a mobile platform. We assumed we could create the site for the desktop at the appropriate resolution and then make some minor adjustments and it would work similarly on mobile. This is unfortunately not the case. It would have been extremely helpful if we also had user testing on the mobile platform. We could have noticed and possibly fixed some of the performance issues.

Another problem our group had was on prioritizing our issues. For example the schedule page has momentum scrolling and animated day switching, however, we implemented this before we made sure that the pop-ups on all our pages worked properly. We would also try to allocate more time than we thought was necessary because we encountered some problem that we could not anticipate. The week before the assignment was due we thought that we were in good shape, but we got overwhelmed at the end because some of the new features we added would interact unexpectedly with other features.