Alisha A.
Rohit T.
John W.

Studio Theme: Sharing

Prototype README

Tools:

We used Sketch to create the screens for our medium-fi prototype. We applied what we learned from testing our low-fi prototype in order to refine our design and make the interface more intuitive. We also took advantage of Sketch's UI Kit to incorporate more realistic iOS elements (e.g. segmented controls to allow users to choose a time frame for each activity on their list). After creating the app screens, we used Marvel to link them together and allow users to interact with our prototype.

Operating instructions:

Visit http://web.stanford.edu/class/cs147/projects/sharing/dream_team/ to access our prototype through our team webpage (or access it directly at https://marvelapp.com/gd827d). From here, simply interact with the prototype as if it was a real app!

Note that not all of the buttons on each screen are functional (since it wasn't feasible to create an exponential number of paths through the app). To see the possible actions from a given screen, click somewhere on the screen and active buttons will light up. Alternatively, you can hover the mouse over various parts of the screen to check if those elements are clickable; the cursor will change from an arrow to a hand when it passes over active buttons, indicating that clicking on those buttons will take you to a new screen.

Limitations:

As with any prototype, there are several limitations to our implementation. Firstly, we weren't able to make the prototype fully interactive. For example, it was impossible to create screens for every possible activity that users might add to their list. As a result, we hard-coded user input on several screens. During the set-up flow, for instance, we hard-coded communities for the user to join and activities that get added to the user's initial bucket list. Similarly, we hard-coded user input on the 'Add to your List', 'Learn to Salsa', and Chat screens. In addition, elements such as date pickers and segmented controls are shown to illustrate what the interface will look like, but aren't actually functional.

Secondly, we haven't implemented a back-end using the Facebook API, so we pre-filled data that would normally be pulled from Facebook. For example, the 'Join Communities' page in the initial setup flow would normally detect communities in which you have many Facebook friends (or just nearby communities, if you decided not to log-in with Facebook). However, since we haven’t implemented the back-end, we hard-coded the communities for now. Similarly, we hard-coded the individuals and shared activities that appear on the ‘My Communities’ screen.

Thirdly, there are a number of buttons on each page that are inactive (i.e. nothing happens when you click on them). We included these buttons for completeness (and to indicate functionality that we hope to incorporate in the future); however, in order to focus on our three core tasks, we left the implementation of these additional buttons incomplete. For example, from the ‘My Communities’ page, you can click on a user (in this case, ‘Arthur Curry’) to view that user’s list of activities. Eventually, you will be able to click on one of Arthur’s activities to send him an invitation for that activity. However, in our current prototype, you can only send invitations from your *own* list page.

Finally, we couldn’t create real-time notifications, so we hard-coded a notification (‘Barry Allen wants to Skydive with you!’) to appear after you add ‘Get a tattoo’ to your bucket list. In the real app, you would receive these notifications in real-time when other users send you invitations.