SOCIAL MUSIC PLAYLIST CREATION IN REAL-TIME

Peter Washington, Gio Jacuzzi, Parker Odrich

Value Proposition
Social music playlist creation in real-time.

Problem and Solution Overview
<u>Problem</u>: Music is a dynamic force that brings communities together, but there is currently no way for all listeners to engage with each other in the creation process in real time - together, right here, right now.

<u>Solution</u>: Rubato allows everyone at an event to contribute in real time to the music playlist. The result is a constantly evolving playlist, curated by the entire community attending the social event.

Tasks and Final Interface Scenarios

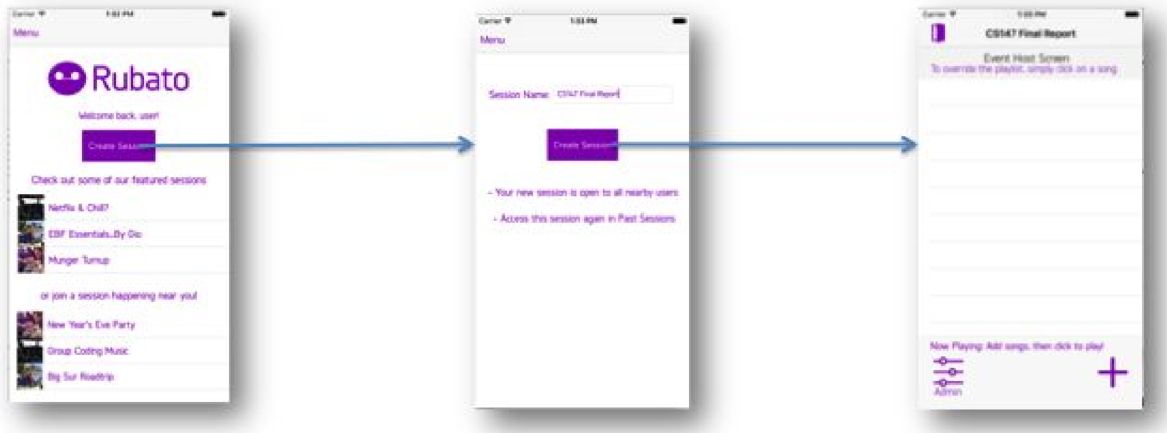
Tasks
Our tasks, in order from easiest to most difficult, are as follows:

1) <u>Easy</u>: Create a communal playlist. We chose this task first and categorized it as easy because creating a session in Rubato lies at the core of our user experience, and as such it needed to be an easy and intuitive task to carry out.
2) <u>Medium</u>: Relive the social event experience by listening to music playlists curated during past events. We chose this task because, after creating a Rubato session, joining one is the most important/frequent task users will use the app for--it is the primary activity behind the social nature of Rubato.

3) <u>Hard</u>: Update the playlist in real-time. We chose this task because it takes joining a session one step further, and gets users personally engaged with making their own individual contribution to the playlist--this requires a much more complicated set of actions to complete as a task, including typing, searching, and other typical interactions, so it became our hardest task.

Final Storyboard Walkthrough of Tasks

<u>Easy Task Storyboard: Create a New Session</u>



Menu
Rubato
Welcome back, user!
Create Session
Check out some of our featured sessions
Netflix & Chill?
EBF Essentials..By Dio
Munger Turnup
or join a session happening near you!
New Year's Eve Party
Group Coding Music
Big Sur Roadtrip
Session Name: CS147 Final Report
Create Session
- Your new session is open to all nearby users
- Access this session again in Past Sessions
CS147 Final Report
Event Host Screen
To override the playlist, simply click on a song
Now Playing: Add songs, then click to play!
Admin

Figure 1: Our easy task was the creation of a new playlist. The user simply has to click the button to create a new session and give the session a name, and then the session is created.

<u>Medium Task Storyboard: Relive The Playlist In The Future</u>

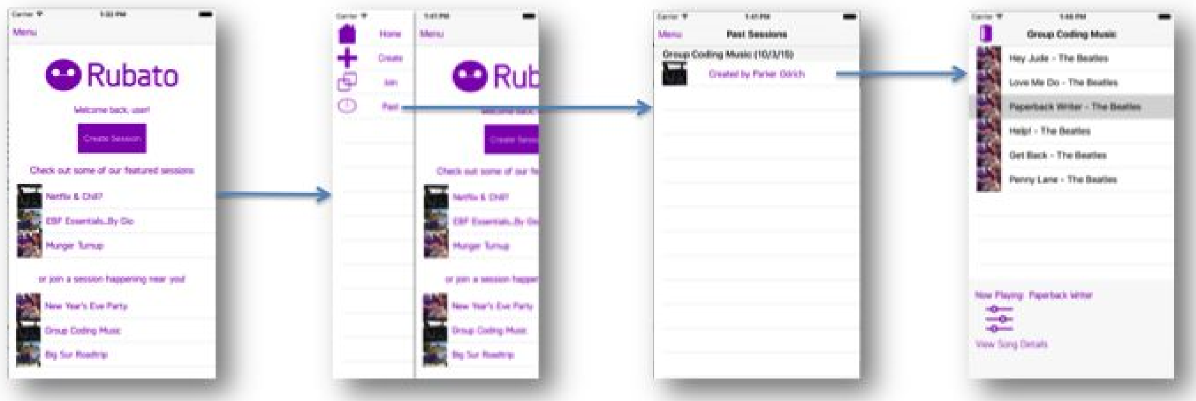
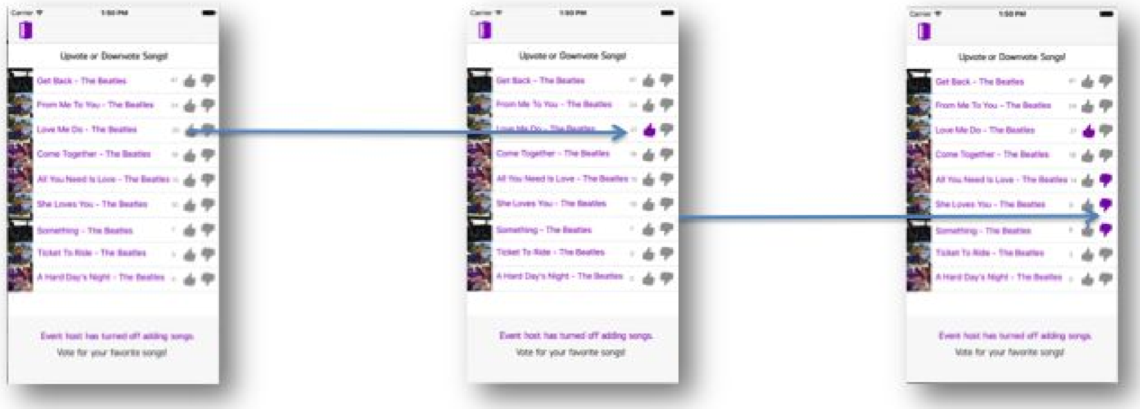


Figure 2: Our medium task is to relive past sessions. To do this, users can click the "Past" button on the sidebar, click on the session they want to listen to, and start listening.

Hard Task Storyboard: Update The Task in Real-Time

Part 1: Any user can upvote and downvote songs.

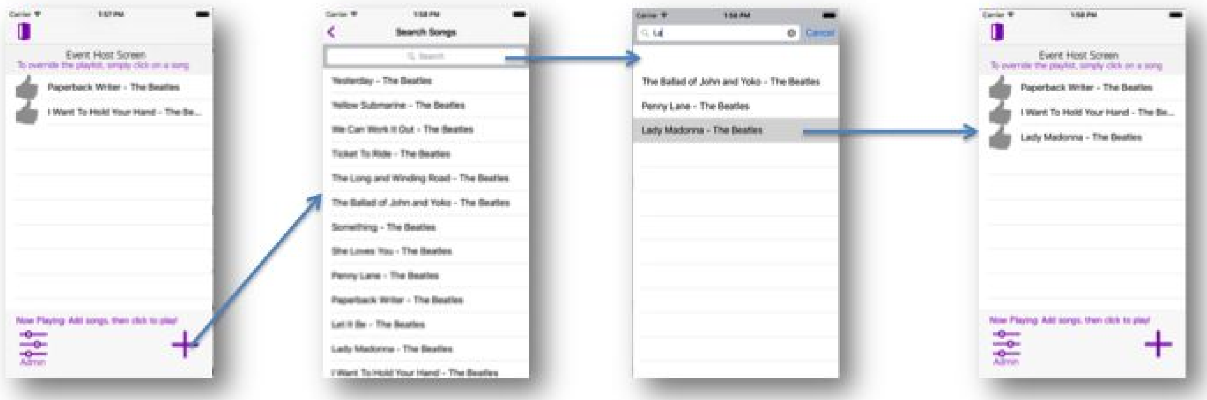


Get Back - The Beatles
From Me To You - The Beatles
Love Me Do - The Beatles
Come Together - The Beatles
All You Need Is Love - The Beatles
She Loves You - The Beatles
Something - The Beatles
Ticket To Ride - The Beatles
A Hard Day's Night - The Beatles

Event host has turned off adding songs.
Vote for your favorite songs!

Figure 3: Part of the hard task is for users to upvote and downvote songs in the session in real-time. They can instantly upvote and downvote any song that they see on the session.

Part 2: The session owner can add songs to the playlist.


Event Host Screen
To override the playlist, simply click on a song
Now Playing: Add songs, then click to play!
Admin
Search Songs
Search
Cancel

Paperback Writer - The Beatles
I Want To Hold Your Hand - The Be...

Yesterday - The Beatles
Yellow Submarine - The Beatles
We Can Work It Out - The Beatles
Ticket To Ride - The Beatles
The Long and Winding Road - The Beatles
The Ballad of John and Yoko - The Beatles
Something - The Beatles
She Loves You - The Beatles
Penny Lane - The Beatles
Paperback Writer - The Beatles
Let It Be - The Beatles
Lady Madonna - The Beatles
I Want To Hold Your Hand - The Beatles

The Ballad of John and Yoko - The Beatles
Penny Lane - The Beatles
Lady Madonna - The Beatles

Paperback Writer - The Beatles
I Want To Hold Your Hand - The Be...
Lady Madonna - The Beatles

Figure 4: The other part of the hard task takes place from the view of the session creator, who has the ability to update the playlist by adding songs to the session and having executive control of which song is to be played.

Design Evolution

The primary development in the session view screen through our research was the implementation of the voting system. In our earliest sketches, the application did not feature any kind of voting or order system, which left the platform vulnerable to “trolling” or malicious user use (i.e. with joke song queues). As seen in Figure 5, over the course of our development, we came to establish a one, and ultimately two-tiered voting system to determine song order and solve this problem, featuring both positive upvotes and negative downvotes.

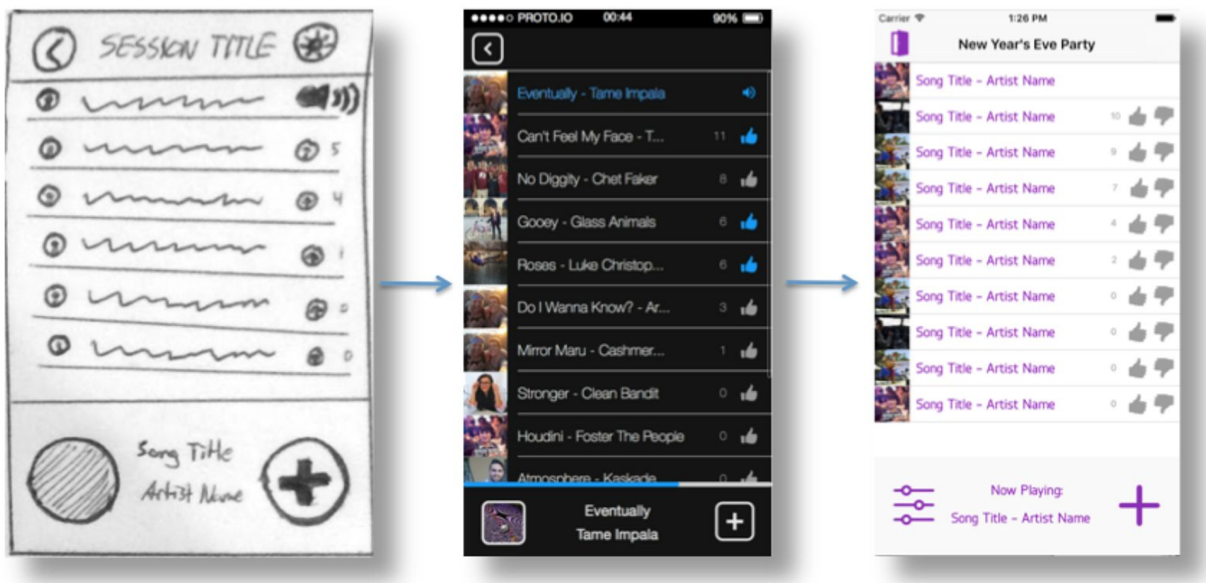


Figure 5: Evolution of the session screen.

While our create session options screen retained much of its original design through our medium fidelity prototype, after extensive heuristic evaluation results we came to the conclusion that the user experience is best when simple and direct. For our high-fidelity prototype, we removed the excessive additional options and created a simple and clear splash screen for session creation, as shown in Figure 6. This streamlined the user experience and clarified user understanding of Rubato session mechanics greatly.

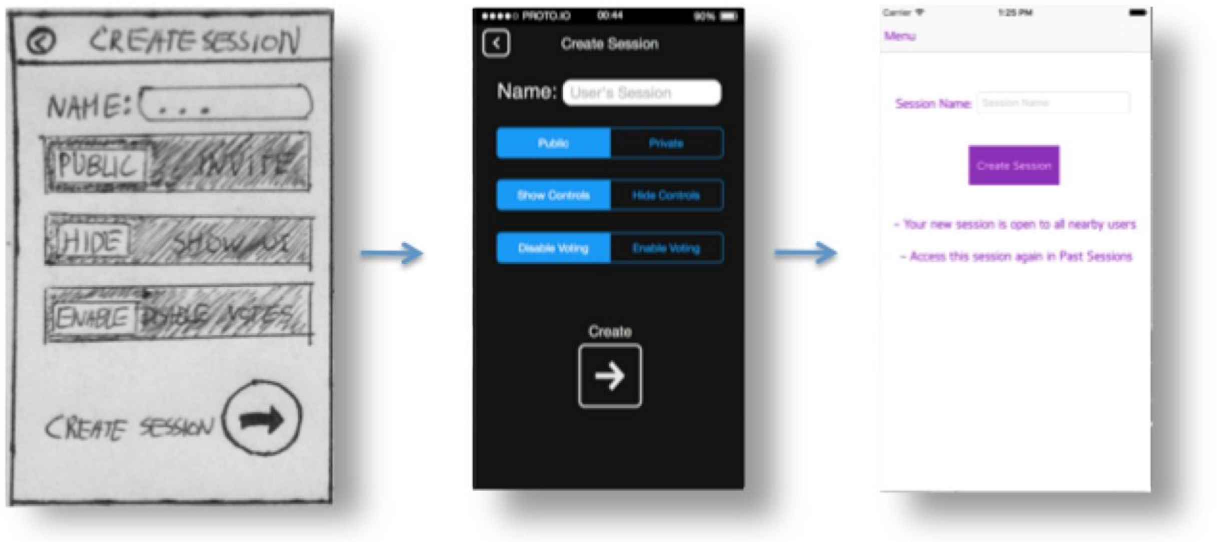


Figure 6: Evolution of the create session screen.

Our design began with a centralized home screen that provides direct access to the application's primary tasks and functions. After user testing and feedback, we removed the unnecessary settings button and replaced it with a more visually appealing profile picture. Finally, after further testing and heuristic evaluation, we found that the natural task flow of Rubato is more adapted to a smaller hideaway menu bar, and a home screen with quick to access featured sessions, local sessions, and a create session button, keeping all the most important functionality of the application easily accessible to the user's fingers.

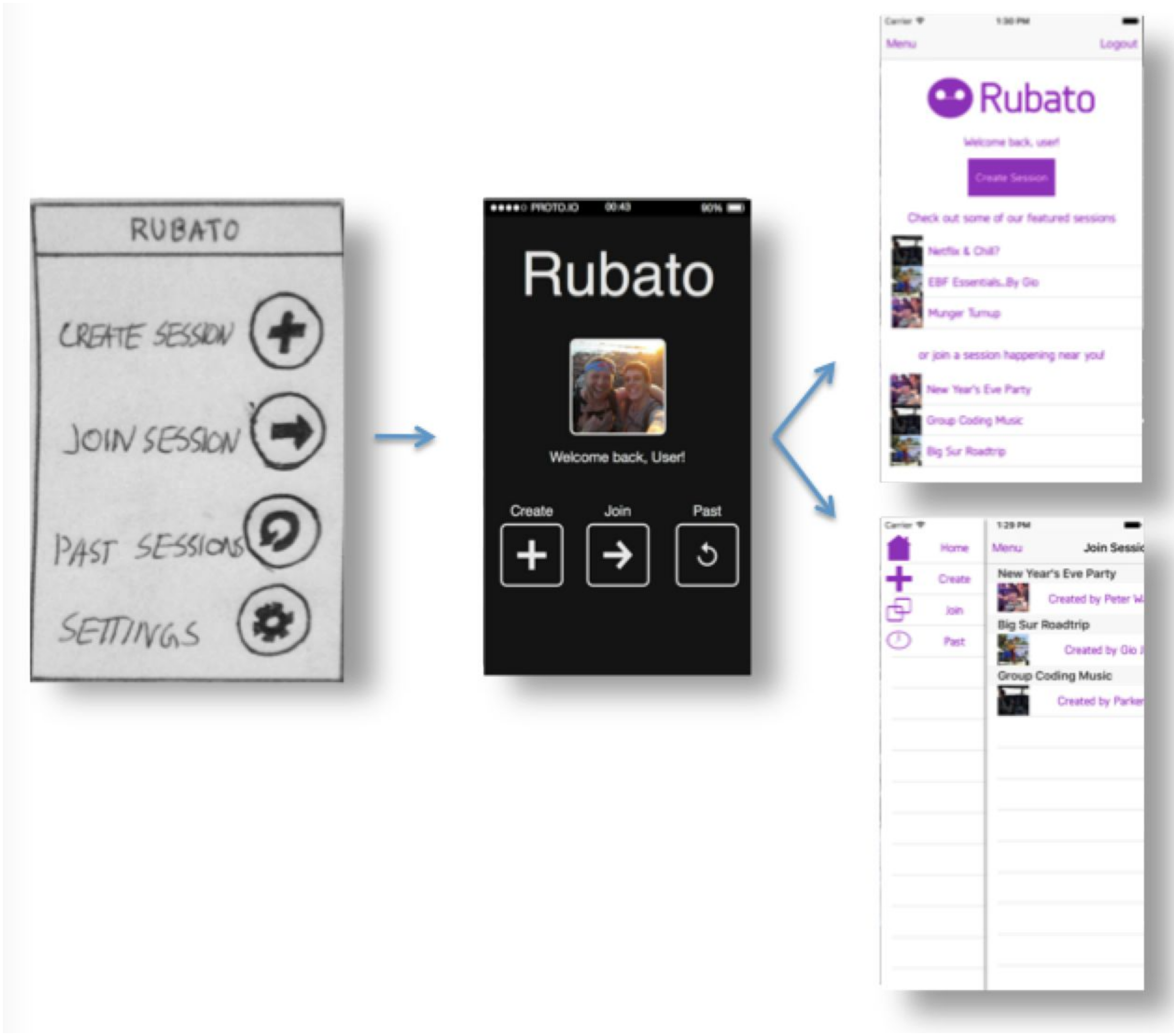
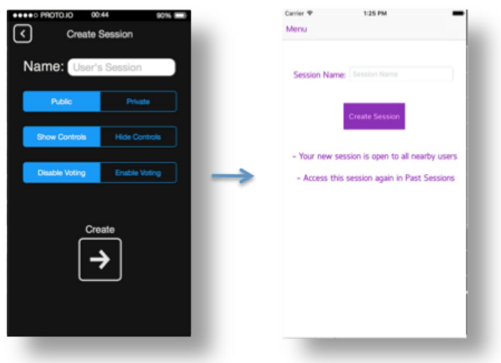


Figure 7: Evolution of the home screen.

Major Usability Problems Addressed

Heuristic Evaluation Change

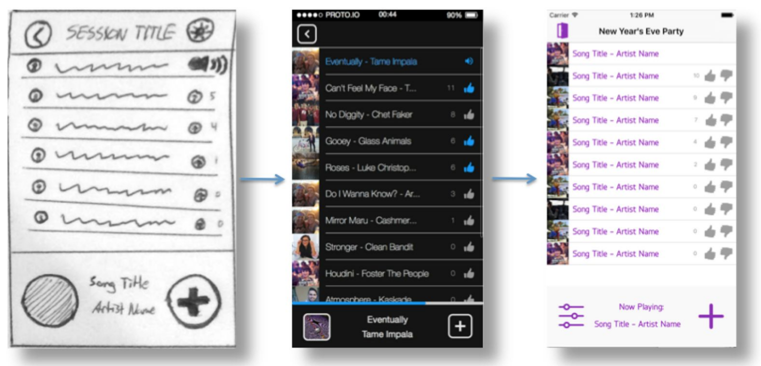
[H2-2: Match System & World] [Severity 3] For some users, it was jarring to move to the "invite friends" screen as soon as the "private" option was toggled on the create session screen. It interrupted the user's expected flow for creating a session and adding in the appropriate settings. Ultimately, we responded to this and other problems users had with the create session screen and its initial complexity by drastically simplifying it to only the session name and the create button, as shown below:



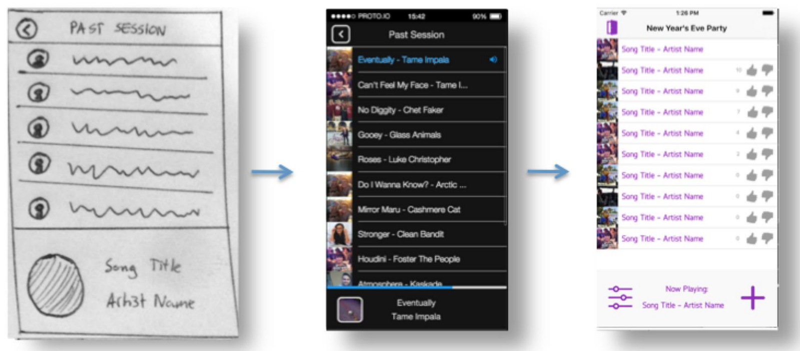
[H2-2: Match System & World] [Severity 4] Similarly, many were not sure what "show controls" and "hide controls" means. There was no immediate feedback for the user indicating that this parameter changed the presence of playback and volume controls. Again, we ultimately responded to this and other problems users had with the create session screen and its initial complexity by drastically simplifying it to only the session name and the create button, as shown above.

[H2-2: Match System & World] [Severity 3] Some users did not initially realize that the lefthand picture for songs in the session view was an image of the user, not the song's related album cover. As shown in Figure 2, we decided to retain this design decision, because while it may not be the user's first inclination, it complements the social nature and atmosphere of the application, and encourages multi-user interaction.

[H2-3: User Control & Freedom] [Severity 4] In earlier prototypes, there was no clear way for users to change settings after they create a session. While one version did feature a setting icon in the top right hand corner, next to the title of a session, we ultimately decided to remove the setting access altogether to emphasize the simple interface of the application, because after making the earlier-mentioned reductions to the create session parameters, by result there was no need for settings access once the session was made, because there were no settings to change. These changes can be seen here:

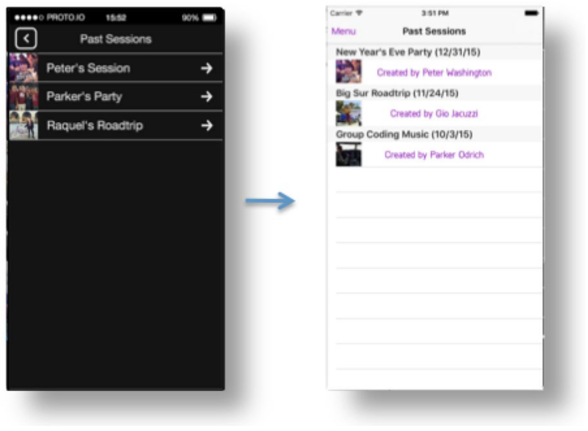


[H2-4: Consistency & Standards] [Severity 4] While exploring past sessions, users became confused because there is no way to see the song controls from earlier (such as bringing up the forward/backward button and play/pause buttons by clicking the bottom left album), which gave them conflicting information. Users didn't immediately realize their ability to replay songs from past sessions. We responded to this problem by redesigning and reintroducing the song controls from earlier, making it very apparent for the user that they have the ability to select and listen to any song from the past session they choose, as shown below:

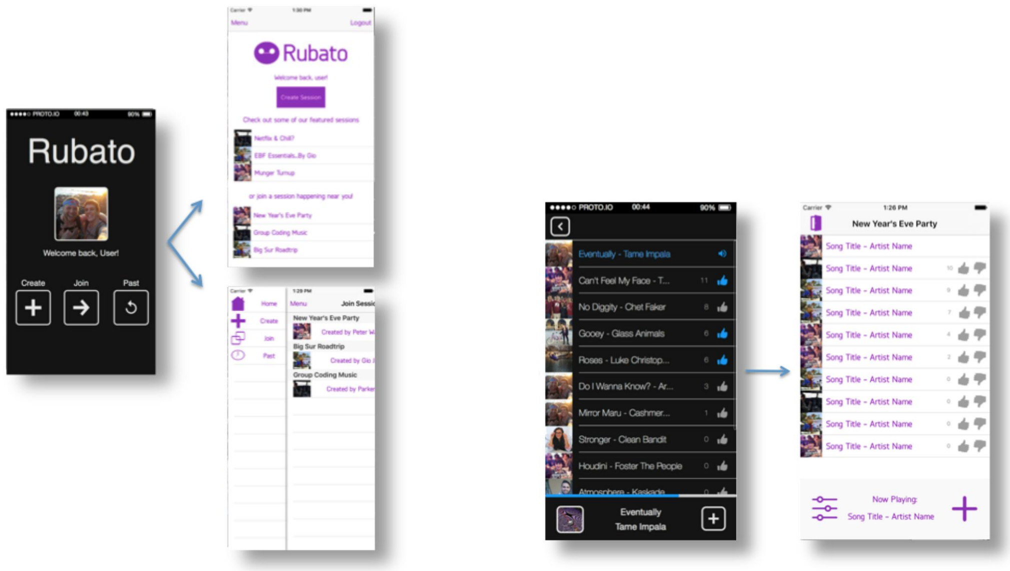


[H2-4: Consistency & Standards] [Severity 3] In our medium-fidelity prototype, all past sessions failed to feature their session title, and instead listed "Past Session". We fixed this minor cosmetic issue by featuring the session title on all past sessions (see in the above picture).

[H2-6: Recognition not Recall] [Severity 3] Users were unsure about the organization of the past sessions screen. There were originally no demarcations of time or creator identification. So, in our final prototype we included names, session titles, and dates to provide clarity for the user:



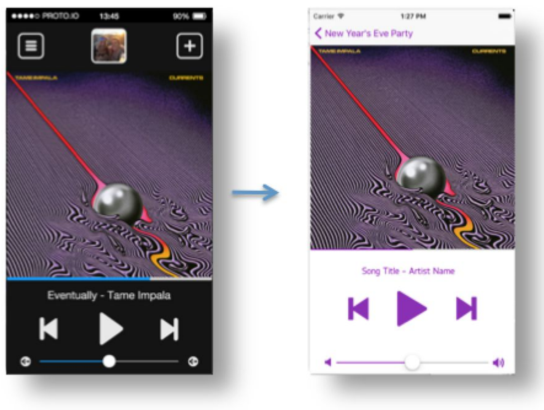
[H2-1: Visibility of Status] [Severity 4] The general flow of the application, moving from feature to feature and task to task, was compromised by our home menu design. Users were unsure how to leave a session and return home. To fix this, we added a simple exit door icon to leave a session, and revised the home screen layout to feature an easily and quickly accessible slide menu bar and separate featured home screen, shown below:



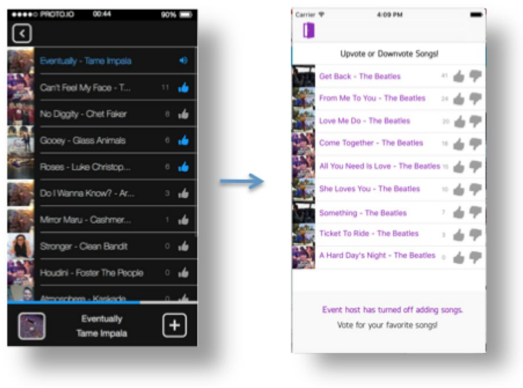
[H2-6: Recognition not Recall] [Severity 4] Originally users were unable to remove songs. We decided not to implement song removal for fear of abuse by other users, and to ensure the streamlined constant queuing usability mechanic.

[H2-1: Visibility of system status] [Severity 3] Originally invitations to other users to join a session were unclear in the way they were distributed and whether or not they had been received. We solved this problem by removing invitations and keeping the application strictly localized, reinforcing the right-here, right-now aesthetic and user experience.

[H2-4. Consistency and standards] [Severity 3] The hamburger icon, which is commonly used as an icon to bring open a menu, was instead used as a back button. We fixed this by using back button icons instead:



[H2-1: Visibility of Status] [Severity 4] Users were confused by the presence of playback and volume controls when in a joined session. We clarified this confusion by removing the controls in this case and replacing them with a clear and helpful user message indicating their abilities:


●●●●○ PROTO.IO 00:44 90%
Eventually - Tame Impala
Can't Feel My Face - T... 11
No Diggity - Chet Faker 8
Gooey - Glass Animals 6
Roses - Luke Christop... 6
Do I Wanna Know? - Ar... 3
Mirror Maru - Cashmer... 1
Stronger - Clean Bandit 0
Houdini - Foster The People 0
Atmosphere - Kaskade 0
Eventually
Tame Impala
Carrier 4:09 PM
Upvote or Downvote Songs!
Get Back - The Beatles 41
From Me To You - The Beatles 24
Love Me Do - The Beatles 20
Come Together - The Beatles 18
All You Need Is Love - The Beatles 15
She Loves You - The Beatles 10
Something - The Beatles 7
Ticket To Ride - The Beatles 3
A Hard Day's Night - The Beatles 0

Event host has turned off adding songs.
Vote for your favorite songs!

[H2-1: Visibility of Status] [Severity 4] On the join session screen, it can be unclear where the sessions are coming from--public, invite, geolocation? We made this clear in our final iteration by removing invitations and focusing the entire application around locality and geolocations, so that we could better the right-here, right-now experience.

<u>Other Changes We Made</u>
Ultimately, the three primary changes we made which most drastically fine-tuned the user experience were the following:

1) Adding downvoting to sessions. This allowed users to more easily filter out songs that they do not want to listen to.
2) Creating a centralized home screen and a simple slide out menu bar for navigation. This helped simplify the user experience and made the user flow more similar to that of existing mobile apps.
3) Simplifying the create session screen, in turn removing global settings from sessions as well. This also helped to simplify the user experience and made the user flow more similar to that of existing mobile apps.

Prototype Implementation

<u>Tools</u>
We primarily used XCode 7.1 to implement our high-fidelity prototype. We used Swift rather than Objective-C to implement the backend parts of the project. We tried using the Spotify SDK to implement the music searching of our app, but due to problems associated with the SDK still being under development, we abandoned using the SDK midway through implementation in favor of hardcoding songs for our prototype and demo.

How Tools Helped
XCode 7.1 allowed us to preview our app on an iPhone 6 simulator so that we could quickly test our app as we made changes to it. We chose to implement our project using Swift rather than Objective-C because it is an easier programming language to comprehend syntactically, which was especially important considering that all of our group members had little to no experience with either language. Fortunately, there was a lot of documentation about Swift online.

How Tools Did Not Help
XCode made it very difficult to implement the project for different versions of the iPhone device, since we would have had to hardcode the design of the app for different screen sizes, or spend extra time to make the screens adaptable to all screens (which would have been very difficult for a team of new iOS developers). We were eventually able to get this working for iPhone 6 screens.

A problem with Swift is the constantly evolving API of the language. For a lot of the things we wanted to implement, the documentation was out of date for the most recent version of Swift. After a lot of searching, though, we were able to find out how to implement what we needed.

Wizard of Oz Techniques

We didn't use Wizard of Oz techniques for the final prototype. Instead, we addressed the limitations of our prototype by focusing on using hard-coded data.

Hard-Coded Data

We hard-coded the songs that users could choose from. This resulted from the iOS Spotify SDK still being in beta mode (making it extremely difficult to get working). Users had the option to choose from any song by The Beatles from their compilation album "One".

We also hard-coded the sessions that were nearby. We did this primarily because we only had one device to demo, and Bluetooth/Wifi functionality would take a lot more experience with Swift.

What Is Missing and Future Plans

There are many parts of the Rubato app that have not been implemented yet, including:

- Having the ordering of songs in the playlist actually change based on upvotes. We spent a considerable amount of time trying to get this feature to work, but it ended up being much more complicated than originally anticipated (given our collective low level of iOS experience).
- Detecting nearby Rubato users based on geolocation.
- Having the Rubato users connect to a centralized server using Bluetooth.
- Allowing users to add songs to the current session.
- As a nice-to-have, it would be interesting to have the app change colors based on the current "mood" of the playlist.

Two out of the three group members plan to take CS194H next quarter, so it is very possible that we will continue the implementation of the Rubato app to completion next quarter, provided that we can get two very experienced iOS developers to join our team. It would not be feasible to finish the app next quarter without this, as the current team members have very little iOS development experience and the completed app would require many sophisticated features to complete.

Summary

Rubato is an iPhone 6 application that facilitates *social* music playlist creation in real-time. Our team implemented a high-fidelity prototype of Rubato throughout the quarter, focusing on the design and theme of the app rather than the complicated backend features of the app. The design of Rubato changed greatly during the quarter, but we ended up with a functional high-fidelity prototype that took into consideration the feedback we received from user testing and heuristic evaluation.