Submission 5: Dynamic UI Mockups

Outstanding Bugs:

- We can't search for tracks yet
    - We have to implement the SoundCloud API (getting the access tokens, ids, figuring how to search, streaming tracks)
    - We need to figure out where in the page (component) we want to populate the search results and how many results we want to display
- We haven't figured how to update a user profile or how to save a playlist
    - Need to use writeDocument() method to update the json in our "users" collection
- On the playlist page, we don't have an embedded SoundCloud player to play tracks
    - As of now, we have mocked up some songs in our database, but in the future these songs will come directly from the API. We will store the SoundCloud song track number in our database, so we can pull out information easily using the API
- We have only partially implemented the like function for songs in the room, due to bugs with onClick code
    - The playlist queue is based on the number of likes a certain song has
        - The song with the highest likes will be played next
        - A user can only like one song, so we need a way to check that using the database and the server
- Our CreateRoom component makes the same room every time with the same ID. One of our members has implemented a hash function that returns a unique code every time you create a room, but it has not been implemented in our component as of yet. We will implement this as soon as we have a real MongoDB database.
- After joining the room, it does not show the room_id anywhere in the room. We will implement this by passing the props to the room component.
- The room component does not have a SoundCloud embedded player at this moment. This will be implemented once we actually use the SoundCloud API to get data and the player.

Dynamic UI Responsibilities:

1. Sidebar (Aarsh Patel)
    - The Sidebar component displays the saved playlists for the user. These playlists come after the user has been in a Room session and has decided to save the songs in that session to a playlist. As of now, we have mocked up the playlists for the user in our database. In addition, each playlists links to a playlist component where the user can sees the songs in that saved playlists. The linking of the pages is done through React Router.
2. Navbar (Aarsh Patel)

    - The Navbar is essentially a static component. It contains the main text for SoundRoom and also holds links to the AccountInfo and Signout page (not implemented yet).
3. Homepage (Aarsh Patel)
    - The Homepage component is also is a static component. It contains links to the CreateRoom and the JoinRoom components. The Homepage component is what is displayed first when the user logs in into his/her account on SoundRoom. From there, the user can create a room and join a room.
4. Playlists (Bhavik Jain)
    - The playlist component is displayed after clicking on one of the user's playlists from the Sidebar component. This will take the id of the playlists from the sidebar and using a function from the server, it will display the name of the playlist and the songs contained within that playlists along with the artist and the album of the songs. At this moment we have mocked the list of songs it's artist and it's album in the database. After implementing the SoundCloud API, we will be able to use the song-track id to populate the data. The linking for this page was done through the React-Router.
5. React-Router (Bhavik Jain)
    - React-Router is a routing library for React. This library helps keep the UI of the webapp in sync with the URL and links within the webapp. This library also allows us to send queries within the links of the webapp, which makes the building of the dynamic UI very easy.
6. CreateRoom (Ronit Arora)
    - CreateRoom is a dynamic component that helps the user create a room with his user_id. This component creates a unique alpha-numeric hash code for the room and redirects the host to the room, calling the Room Component with relevant props.
7. JoinRoom (Siddarth Patel)
    - JoinRoom is a static component. It lets users join an existing room via the hash value generated upon the creation of the room. Right now, we are using the room id instead of the hash value in order to reference the room, but we plan to change this soon. Once the user has entered the correct code, he/she will be redirected to the room.
8. Room (Lynn Samson)
    - Room is a dynamic component and the main component of the SoundRoom app. It contains the implementation of a particular room, which contains the playlist for the room, the list of participants in the room, as well as the search component that lets one add songs to the playlist. We decomposed the Room component into multiple subcomponents, in order to deal with the complexity and range of functionalities within a room. At the moment, playlist and participant information is being dynamically pulled from the mock database. We have partially implemented the functionality to like a song, but there is an error with the onClick function that only updates the latest rendered song. The search functionality will

be implemented when we figure out the SoundCloud API, since we query the API to get the available list of songs. The ‘Save Playlist’ button functionality will also be postponed, until we implement the database, since it will be adding a new playlist into the particular user’s entry in the User document.

9. Account Info (Justin Martinelli)
    - Account Info is a static component in which the user’s account information and profile picture is displayed, and where the user can change their account information. At the moment, input boxes have placeholders that display the current information of the user, which is dynamically pulled from the mock database. The avatar on the top right takes the id of the user and dynamically displays the profile photo saved for that user in the mock database.