MusiQueue

Product Backlog

Coordinator: Tori Shurman

Brian Hanford, John Stimac, Vincent Maggioli

David McGary, Sairam Galla, Samuel Kuhns

**Problem Statement**

A group of people want to play music, but each person has their own music they want to play. Typically for a bluetooth connection to a speaker, in order to play your own music you must first kick the person who is currently on the speaker off of it and then connect yourself to the speaker. Our application will allow users to connect to a queue, playing on one person’s device (a.k.a. The hub) and enter music into a queue in the hub. The music will then play from the speaker after the song that was currently playing in real time, removing each song as it is played.

**Background Information**

**Audience**

Our demographic consists of groups of people looking for a convenient way to play their music in a local public setting along with all other users at an event. This app will look to mostly serve at parties, events, or small get togethers with friends. With such a large demographic range, the application should be able to reach a large audience size quickly after full functionality is implemented.

**Similar Applications**

Spotify, Chromecast and YouTube have similar versions of this implementation. Spotify has shared playlists, which, when shared with other users, can have multiple people adding songs into the playlist. Chromecast achieves a similar effect by having users connect to a Chromecast and creating a queue of YouTube videos. YouTube has a feature where you can create a public playlist and add users to the playlist. This gives the added users the ability to add their own videos to the playlist. None of these reach the niche group that our application does though. Since we are focusing on having a playlist for get togethers, this allows the users to add songs in real time, and move off the playlist once they are done playing. Our app also makes it easier for users to connect to the queue, making it convenient to use at parties and gatherings.

**Limitations**

Other projects that have implemented similar ideas have had limitations of achieving a more social atmosphere where users are interacting with the songs input by other users. A direct limitation we will be facing is allowing users to turn their phone screen off while still playing the music that was retrieved from YouTube. Additionally, the organization and layout of the user interface will be difficult to implement because we are retrieving an entire video from YouTube causing the challenge of arranging it nicely with the other elements of the interface in the hub.

**Requirements**

Functional

* As an administrator, I would like to have hubs auto delete after a few days of inactivity
* As an administrator, I would like to store and view all active hubs with their queues on a server.
* As an administrator, I would like to identify individual hubs by phone IDs.
* As a user, I would like to connect to a hub by the hub name and password.
* As a user, I would like to find a nearby hub based on my current location, **time permitting**.
* As a user, I would like to be able to sign in to MusiQueue securely.
* As a user, I would like to have an option to reconnect to a recent hub without having to type the name and password again.
* As a user, I would like to type in my name when connecting to a hub.
* As a user, I would like to see the songs currently in the queue in priority order.
* As a user, I would like to have the queue reorder as songs get voted on.
* As a user, I would like to have songs leave the queue after they get played.
* As a user, I would like to see who put each song in the hub.
* As a user, I would like to see how many votes each song in the queue has, **time permitting**.
* As a user, I would like to not allow duplicates in the queue.
* As a user, I would like to be able to queue youtube videos as songs.
* As a user, I would like to be able to search for a song by name, see results, then add one to the queue.
* As a user, I would like to be able to add youtube videos by their url.
* As a user, I would like to be able to disconnect from a hub.
* As a user, I would like to auto disconnect from a hub if I don’t view the queue for over a day.
* As a user, I would like to vote up songs to raise them in the queue.
* As a user, I would like to vote down songs to lower them in the queue, **time permitting**.
* As a hub manager, I would like to create a hub for people to join and queue their music into.
* As a hub manager, I would like to give my hub a unique name.
* As a hub manager, I would like to have an option to make my hub password protected.
* As a hub manager, I would like to receive a PIN code for my hub if I set it to be password protected.
* As a hub manager, I would like to have an unlimited amount of people to connect to my hub.
* As a hub manager, I would like to see who is connected to the hub.
* As a hub manager, I would like to remove users from my hub for any reason.
* As a hub manager, I would like to be able to close a hub, disconnecting everyone.
* As a hub manager, I would like to see who added a specific song in the queue.
* As a hub manager, I would like to delete songs from the queue.
* As a hub manager, I would like to play the song at the top of the queue.
* As a hub manager, I would like the app to autoplay the next song when the current song ends.
* As a hub manager, I would like to be able to pause the playing song.

Non-Functional

* The application will run on Android.
* The application will run on web, **time permitting**.
* The application will run on iOS**, time permitting**.
* Allow users to sign in through Google to see their active hubs.
* The application will connect to a server to implement the queue.
* There will be a central server that will maintain the different queues.
* Support many users on the same hub at one time.
* Support many active hubs at the same time.
* Avoid saving sensitive user information in our database.
* Have less than two hours of server downtime per month.