

# Screen Sketches

## JR\_2

Jack Goldsworth

Josh Edwards

Alexander Young

Nate Tucker

## SelecTunes

# Actors

Guests:

Functionality:

- Search: Allows the user to search for a song to be added to the queue, in which other users may up-vote or down-vote.
- Disconnect: Allows the user to leave the party.
- Join: Allows the user to join a party based on a one time code.

Moderators:

Functionality:

- Kick: Allows the moderator to remove other users from the party (for example: If a user is downvoting every song).
- Ban: Let's the moderator ban certain guests (by phone number) from entering the party.

Hosts:

Functionality:

- Create: Allows the user to create a game and receive a one time code.
- Spotify Login: Makes the user login to spotify so that music may be played through their device, and API calls can be sent and received.
- Destroy: Let's the host destroy the party instance and disconnect all of the users.
- Kick: Allows the host to remove other users from the party (for example: If a user is downvoting every song).
- Ban: Let's the host ban certain guests (by phone number) from entering the party.

# Non-Functional Requirements.

- Reliability.
  - First and foremost we want the party to never die. It would be a really disappointing experience to not be able to join a party, or if the party server failed and the party was left in silence
- Scalability.
  - We want to be able to support as many parties as we would need on any given day. Maybe one or two during the week? No problem. Thousands on the weekend? We can handle it.
- Maintainability.
  - Even after the application is finished, engineers and programmers should easily be able to work on the application in-case of an upgrade to android or the backend, or even to add entirely new features.

# Tables and Fields

## Table 1: Parties

Each party has a host (defined by us), has a list of banned users, has a list of guests, an integer join code, a list of banned members, and not in the db but just in memory is a queue of songs (our own song model based on the spotify api response).

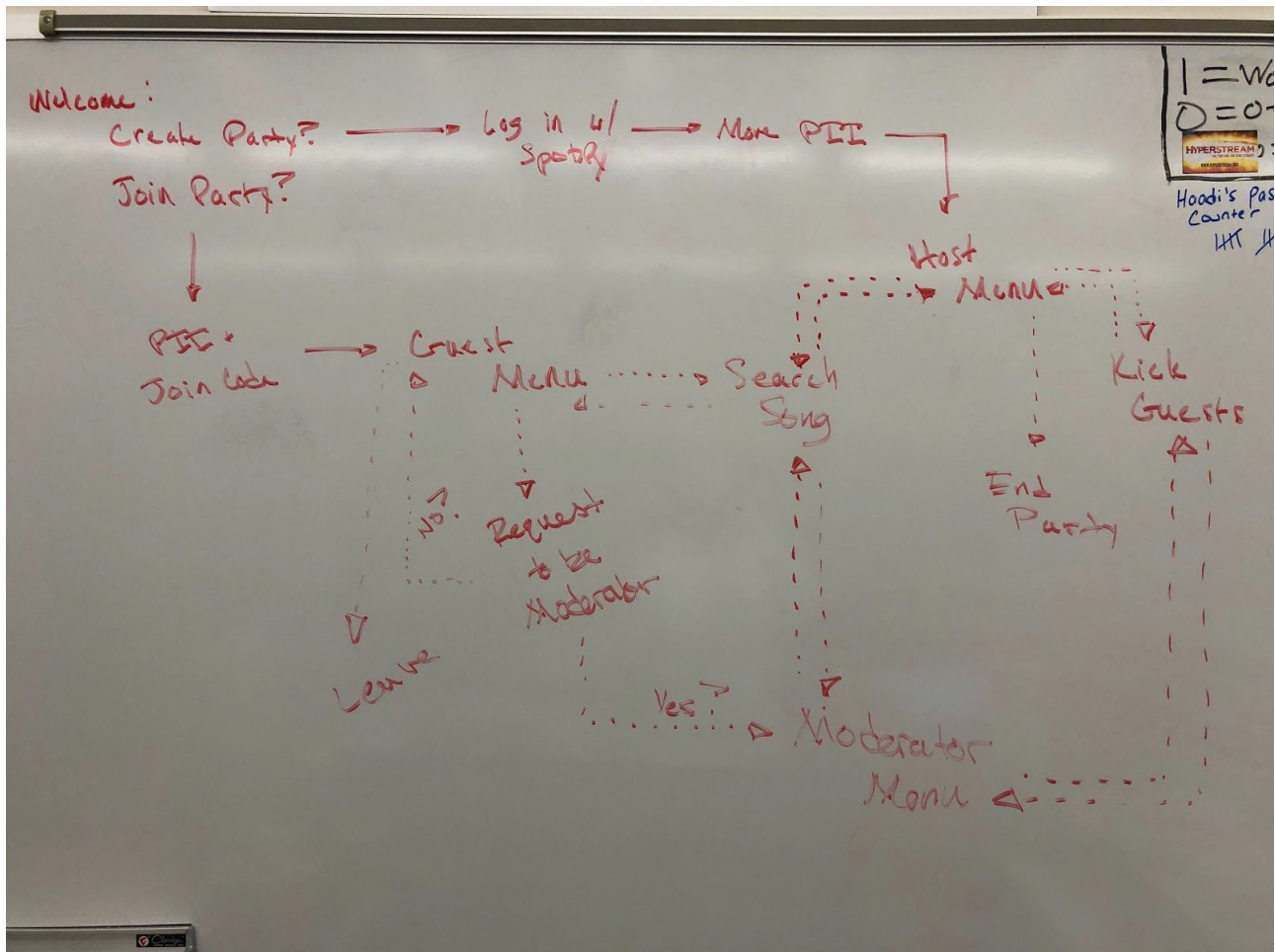
## Table 2: Host Users

The host is required to “Login with Spotify”. This will all send an account token to our backend service which will register a new user model, or update it if one already exist. It stores Emails, Spotify Tokens, Phone Number, and UIDs.

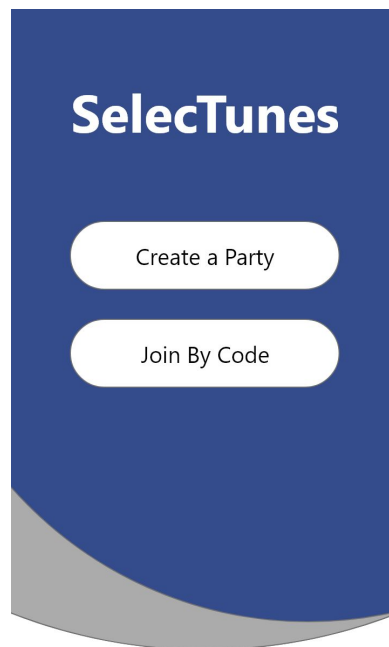
## Table 3: Banned Users

This table stores all the users banned from parties. It stores their phone number, and uid. If a user is in this table, they will be prevented from joining another party.

# Screen Flow



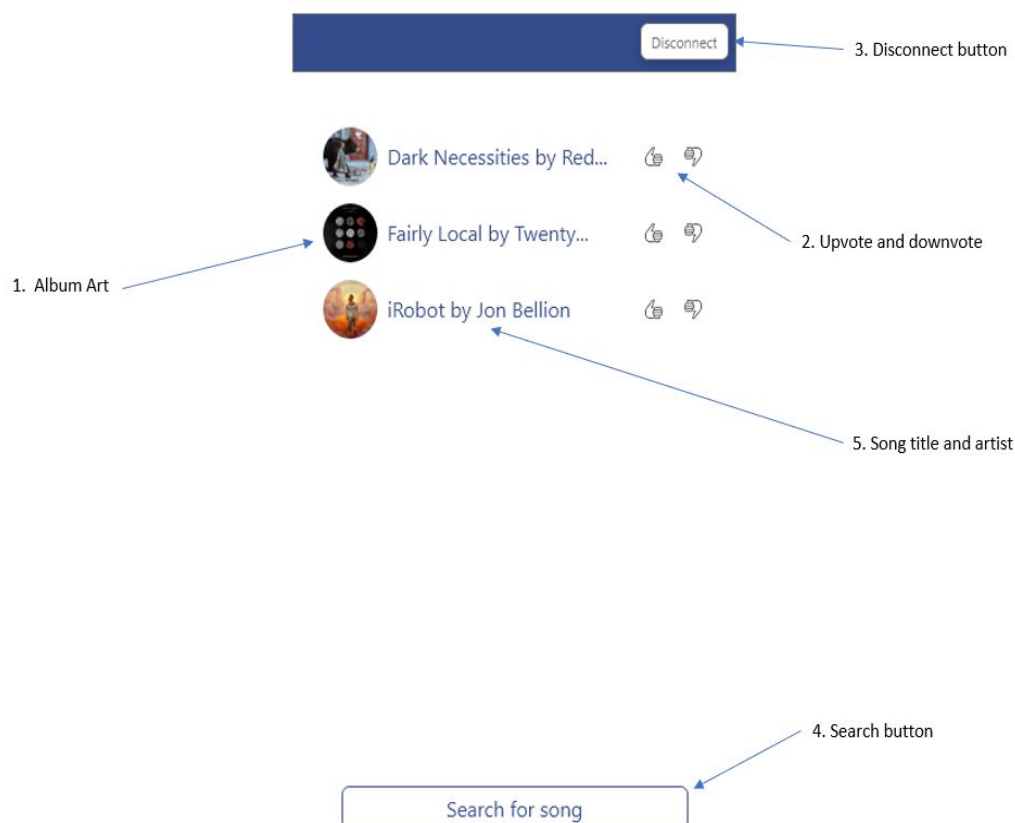
## App Landing Screen - Nate



This is where the app will open. Users have the option to either join a party by its unique join code or create a party. Upon creation of a party, the host will be prompted to log in with spotify, and then another menu to enter PII such as phone number and username.

If join is requested, then they will be prompted to enter a number code and they will join a party if that party number exists

## Guest Main Menu by Jack Goldsworth



The guest menu exists as a way for the user to interact with the current playlist of songs, either upvoting or downvoting the picked songs, so that the most popular song will be played next. This menu will be produced when a user enters into a party using the parties one time code.

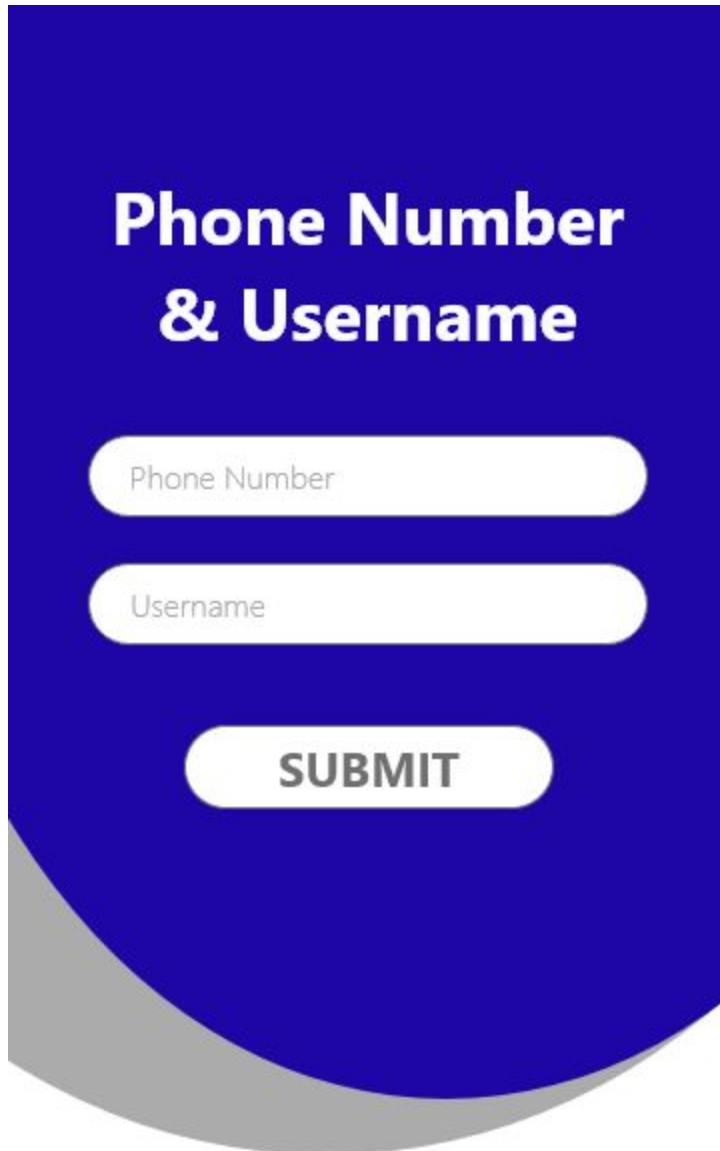
1. The album art comes from the corresponding song to be played through Spotify. This will be retrieved using API calls to Spotify.
2. The upvote and downvote buttons exists so users may show their support for a song in the playlist. When it becomes time to pick the next song, the most liked song will be played next. Most liked = upvoted - downvotes.
3. The disconnect button is there so guests may leave the party that they are currently in. This would take them back to the homepage where they could decided to join another party.
4. The search button will take you to a menu where you can search songs so that they may be entered into the queue, and be voted upon by your fellow party goers.
5. Finally, the song and title that correspond with the song that is in the queue. This exists to give users an idea of what songs could be up next.

## Song Search - Nate



This menu will be accessible to all users of the app. Users enter in a string of a song name they want to search. It does not have to be exact, as it will return a few results that could match the intended song. Search queries are sent to the server, where we will query the spotify api and return a slimmed down version of the spotify response. These results will show on the screen as a list. When a user clicks add, it will add the song to the party's queue with a post request. It will only add each song once, but any number of searches can be done and any number of unique songs can be added. Additionally, each user will be able to return to their respective menu.





The image shows a mobile application interface with a dark blue background. At the top, the title 'Phone Number & Username' is displayed in large, white, bold font. Below the title, there are two white, rounded rectangular input fields. The first field is labeled 'Phone Number' and the second is labeled 'Username'. Below these fields is a white, rounded rectangular button with the word 'SUBMIT' in bold, dark blue capital letters. The bottom of the screen features a grey, curved shadow effect.

#### Guest Phone Number and Username - Alexander Young

This screen is where the app collects the guests phone number and preferred username. Upon submit, the information is sent to the backend server. The username and phone number are displayed on the Guest Menu, and are used to uniquely identify users for administration purposes.



## MOD MENU

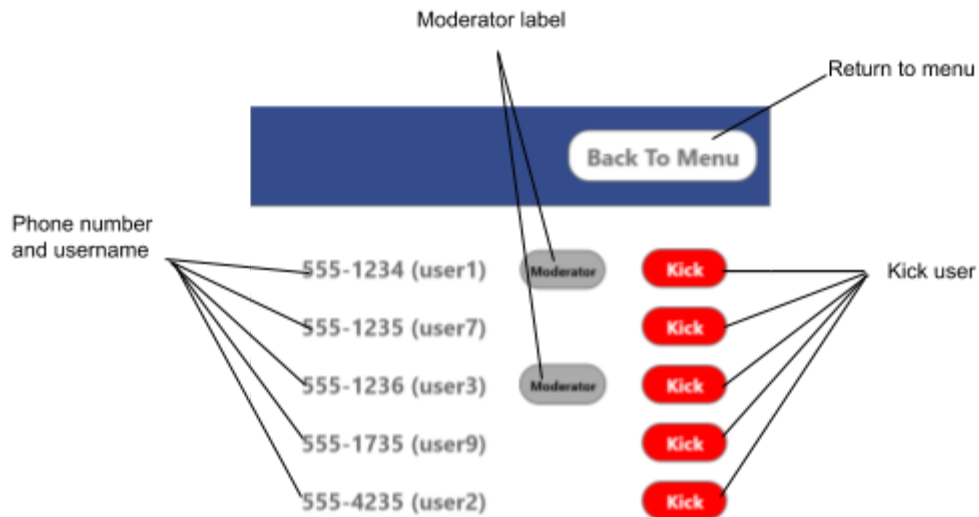
**Kick Users**

**Moderate Songs**

Moderator Menu - Alexander Young

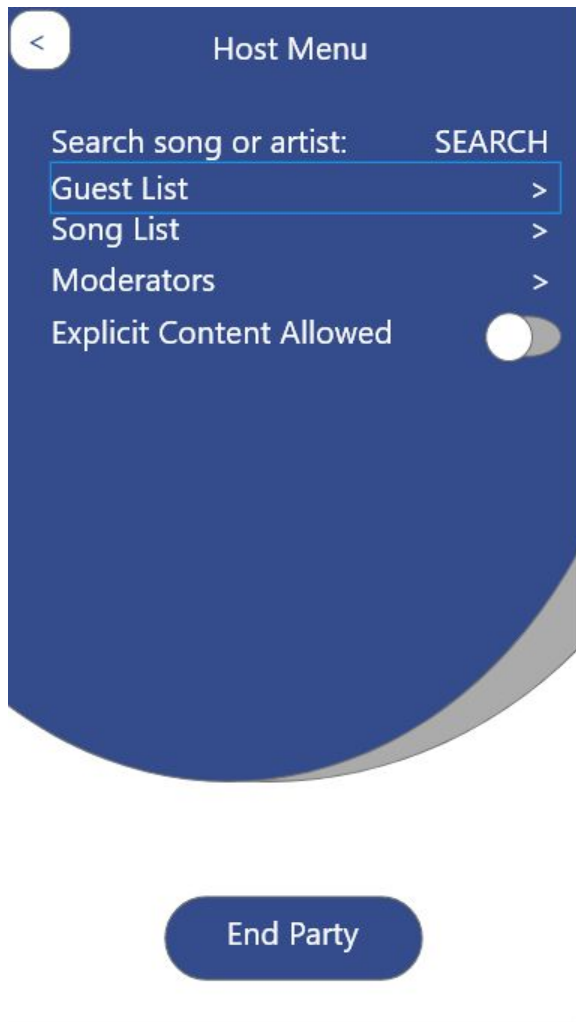
This is the Moderator Menu. From here they can go to Kick Users, Moderate the Song Queue (via removing songs, or by insta approving songs). The kick users menu looks like the next screen.

## Kick Guests Menu - Nate



This menu is only accessible by moderators and the host. It shows the phone number and username associated with a specific user in this party, as well as gives the option to kick a particular user. When kicked, a user will be added to our blacklist and will not be able to join the party again. The guest can join other parties, however, but as long as this party continues, that option will not be allowed.

## Host Menu: Joshua Edwards



Search song or artist: Editable text box allowing query through spotify to search for songs and allow them to be added to the party queue.

Guest List: Button that takes host to guest\_layout that has a dynamic scroll view of each guest logged into the party, also allows for removal of guests

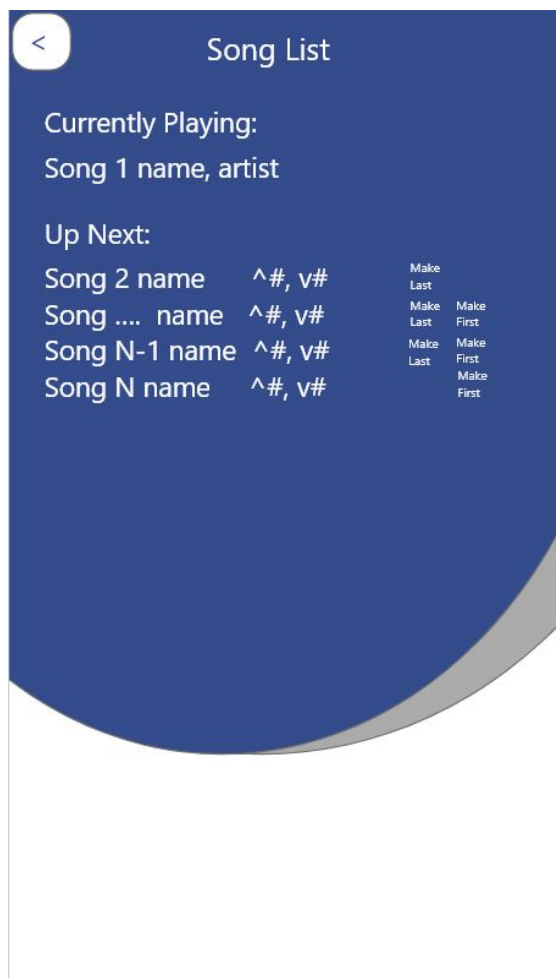
Song List: Button takes host to host\_songList\_layout that also has a dynamic scroll view of queued songs. List displays up and down votes for each song and allows host ability to remove, force queue front, and force queue last for each song.

Moderators: Button takes host to moderatorList\_layout also implementing dynamic scroll view of all the moderators, allows for adding moderator via picking from guest list, or revoking mod abilities.

Explicit content switch: Changes whether or not explicit content is allowed to be queued.

End Party: Button ends party, revoking all mod abilities, removing all guests, and removing all songs from queue.

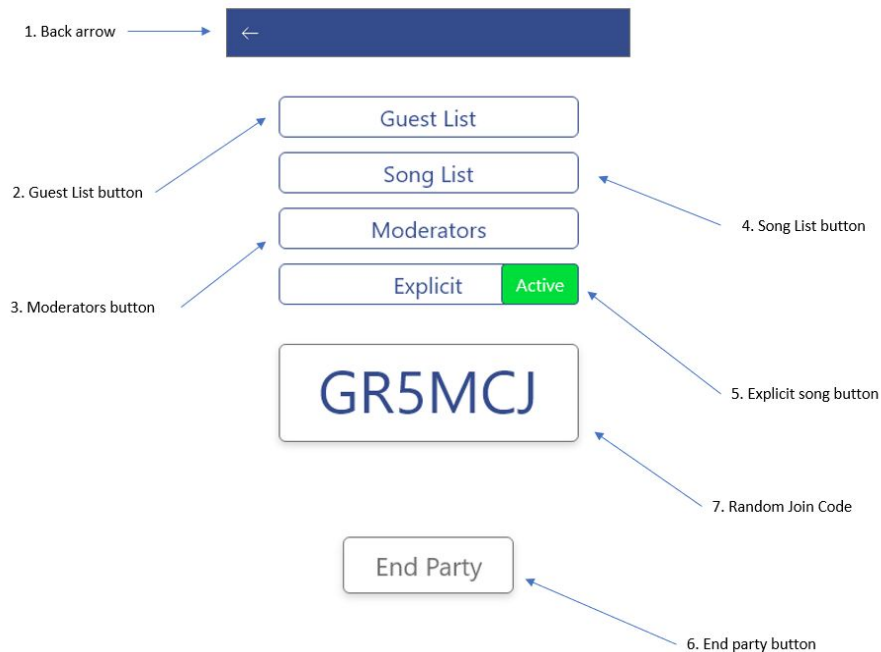
## Song List: Joshua Edwards



Quite simple dead-end layout, only button that changes the current layout is the back button that takes the user back to their respective menu(guest, mod, host).

Other than that there are text boxes that display the song currently playing, and then a scroll view of the queue along with the number of up and down votes. Mods and the host can both force songs to the front and back of the queue despite the number of votes, as well as vote. Guests can only vote for songs.

## Host Menu by Jack Goldsworth (Re-design of Josh's work)



The host menu exists as a way of letting the host entertain the users, while keeping the party clean and without interruption. This will be the first menu that the host enters into once starting a party, so it will be easy to make quick changes to create a desired setup tailored to the hosts needs.

1. The back arrow just takes you back to the home screen of the app and immediately ends the party.
2. The guest list button will take you to a list of guests in the party. From that list you may kick the guests or make them a moderator.
3. The moderator button will take you to the list of moderators in the party. From that list you may kick moderators or demote them to normal guests.
4. The Song list button takes you to the list of songs that are currently in the queue. When inside this view, you can remove songs from the queue, downvote and upvote songs, or immediately choose a song to play.
5. The explicit song button allows the host to decide whether songs that say curse words in them are allowed to be played. This is important if this app were to be used in locations such as a highschool dance.
6. The end party button immediately ends the party and removes all users. You can decide to restart the party without going back to the main menu.

7. The random join code is the code that users can input into their devices to join the party. It will be a 6 digit code with letters and numbers.