

CS495 Spring 2018

Software Requirement Document & Presentation

Juked

Francisco Roveló, Laura Phillips, John Mills, Mohammed Rahman, Noah Patton

02/06/2018

Revision History

#	Revisioner:	Comment:	Date:
1	Francisco Roveló	Added Title, Revision, Contents page, & Format	02/06/18
2	Laura Phillips	Added Introduction	02/08/18
3	Mohammed Rahman	Added function , Non function, Comparison	02/16/18
4	John Mills	Added project description	02/16/18
5	Mohammed Rahman	Updated non functional requirement	02/25/18
6	Noah Patton	Reviewed overall paper	02/16/18
7	Noah Patton	Made powerpoint presentation	02/25/18
8	John Mills	Added class diagram and Use cases	02/26/18
9	Laura Phillips	Editing formatting and TOC	2/26/18
10	Laura Phillips	Added activity diagrams	2/26/18
11	Laura Phillips	Updated definitions	2/26/18
12	John Mills	Updated class diagram	3/1/18
13	All	Final pass before first submission	3/1/18
14	Mohammed Rahman	Added observations, reviewed the paper	3/1/18
15	Laura Phillips	Activity diagram descriptions	3/1/18
16	Noah Patton	Added use case description	3/1/18
17	John Mills	Added class diagram description	3/1/18
18	Noah Patton	Reviewed function and nonfunctional	3/1/18
19	Laura Phillips	Reviewed additions to § 3 & 6	3/1/18
20	Francisco Roveló	Reviewed/edited/added information to § 4	3/1/18
21	Laura Phillips	Added prospective database information	3/1/18
22			

Table of Contents

0.0 Title Page	0
1.0 Revision Page	1
2.0 Table of Contents	2
3.0 Introduction	3
3.1 Purpose	
3.2 Goals	
3.3 Definitions	
4.0 Project Description	4-5
4.1 Feature Overview	
4.2 Generating Lobby	
4.2-A Selecting Available Music Median	
4.2-B Host Options	
4.3 Guest Menu	
4.3-A Searching and Selecting	
4.3-B Upvoting and Downvoting	
5.0 Functional & Non-Functional Requirements	5-6
5.1 Functional Requirements	
5.2 Non-Functional Requirements	
6.0 Functionality Comparison	6
7.0 Analytical Diagrams and Description	7-16
7.1 UI Mockups	
7.2 Class Diagram	
7.3 Use Case Diagram	
7.4 Activity Diagrams	
7.4-A Splash Screen Activity Diagram	
7.4-B Host Home Page Activity Diagram	
7.4-C Guest Home Page Activity Diagram	
7.5 Prospective Database Setup	
8.0 Observation	16

3.0 Introduction

3.1 Purpose

Juked is an application to give the power of music to the people. When a group of friends are together, the task of picking the music they listen to typically falls to one person. The host of a party or gathering can't relax and enjoy their company because of the stress of constantly choosing music everyone will enjoy. Juked allows the host to share this responsibility by allowing the entire group to pick their favorite songs and vote on their friends' choices so everyone gets to listen to music they like. Aside from making a party more interactive, the app also provides opportunity for people to venture out of their comfort zone. It is also possible for people to taste something new they were not aware of before, and prevents one person from monopolizing the party.

3.2 Goals

Our goals for Juked are to produce an application that is useful, user friendly, convenient, and fun. We want Juked to be intuitive and easy to understand so that any group of people can make use of it. We envision this application to be useful in everyday life, especially for young adults and teenagers that have frequent get togethers with friends. Having a clean and clear UI will help accomplish these goals, so that parties can be joined easily and songs can be found quickly and concisely. When an application is cluttered or hard to navigate, users are easily turned away. The application has potential for growth because once a party host decides to use Juked, any party guests that want to have input on the song selection will also download Juked. Going forward, all of those party guests can suggest Juked at a different party, where all of those attendees will download the application, and so forth.

3.3 Definitions

- API - Application programming interface
- Guest - Guest of a party, can join a lobby and request songs
- Host - Party host, has guest privileges and administrative controls of party
- Lobby - The host creates a lobby where the host and guests can select options and view the songs that have been chosen by the other guests.
- Soundcloud - 3rd party music listening application, hosts independent music
- Spotify - 3rd party music listening application, requires premium (paid) subscription
- Queue - List of songs chosen by the party guests and host, played in a first in, first out order
- UI - User interface, what is displayed on the screen that the user will interact with to use the application
- UX - User experience, the experience a user has while interacting with the application. The design and layout of the app should be chosen with the user experience in mind.

4.0 Project Description

Juked will allow one phone to act as a host, and let anyone connected to that host's lobby to choose the songs that will be played from the host phone. When the host creates a lobby, the app assigns it a four digit number from 0000 to 9999. The host can let the party goers know the generated lobby code. The guests then use that number to join the lobby. Every user gets to choose one song in a given turn. Each guest may upvote or downvote any song in the queue and the top voted song goes to the top of the queue, while negative voted songs get pushed to the bottom (hence it is collaborative). If a song obtains a low enough score it is removed from the queue. If the host forgets to close out the lobby at the end of the party, the app's back end automatically closes the room after a certain amount of inactive time.

4.1 Feature Overview

Guests have minimal features, of which include selecting songs, viewing the party's song history, and voting up/down songs. The host has all the perks of a user, but more administrative powers.

4.2 Generating Lobby

At the main screen, the host chooses the option to host a lobby. A random four digit number is assigned to the host to give to guest. If the app becomes popular we will increase available lobby numbers exponentially.

4.2-A Selecting Available Music Median

The host will also be given the option to limit the guest to choose from just Spotify, just Soundcloud, or both (possibly YouTube if time allows). The host will be required to have the Spotify or Soundcloud application downloaded to their phone in order to use their streaming services in Juked. Additionally, Spotify will only be available for premium Spotify users.

4.2-B Host Options

The host is allowed to skip songs or delete songs that he deems not fit in the queue. The host can also pause and resume playback as necessary.

4.3 Guest Menu

Once a guest chooses to join a lobby, they will be prompted to enter the four digit code that the host gives out. Once in the lobby, the guest will choose both a nickname and an avatar. Avatar options will include either a pre-uploaded one(default avatar) or upload one from the device

4.3-A Searching and Selecting

The guest will have a search bar on the guest screen to search for songs. The search uses either Spotify's search or Soundcloud's to pull up the closest match to the song being typed. The guest then has the opportunity to pick the song and it will automatically be added to the queue.

4.3-B Upvoting and Downvoting

The guests will also have the ability to see songs that are in the queue. They can upvote or downvote any and all songs, but may only vote on each song once. The more upvotes a song gets, the higher it will move in the queue. If a song gets 5 (or other predetermined amount) or more downvotes, it is removed from the queue entirely.

5.0 Functional & Non-Functional Requirements

5.1 Functional

- a. The app will give user the option either to create a lobby or join a lobby.
- b. User will be able to select a song from Spotify or Soundcloud. In the future, we will be open to adding more platforms such as Youtube or Google Play Music.
- c. User will be able to upvote or downvote a song. The user can change the vote at any time by either pressing the same button to cancel the vote or pressing the opposite button to do the opposite action. However, the user will only get one vote per song. A user may change their vote from up to down and vice versa, but they cannot upvote a song twice.
- d. User will have the opportunity to add another song selection to the queue after their previously chosen song has been played.
- e. Host will be able to remove a song from the queue. This will help to ensure that host maintains absolute control over the playlist.
- f. Lobby codes will be (at least) 4 digits. If the app becomes popular, this number can change to add lobby options.
- g. Host may not create more than one lobby at a time. Currently we have only 10,000 possible room. This would ensure better usage of limited resources.
- h. User may not choose more than one song in each turn. If the user has a song selection in the queue, they cannot make another selection until their first one has been played or voted out.

5.2 Non-Functional

- a. App should be able to sync user votes. The tallying of voting is done behind the scenes, without explicit input.
- b. Keep track of all the lobbies created. The lobby numbers have to be unique, otherwise user's songs may be played in a different room than they are in.
- c. The system must use resources efficiently. Given the possible number of rooms, votes and actions at once, keeping things efficient is a must.
- d. System need to be able to use Spotify and Soundcloud APIs. These companies have the largest market share of the music streaming industry and will provide a large library of songs for users to select from.
- e. If a lobby has an empty queue for a certain amount of time, the lobby will be destroyed and all users removed, including the host. Since the app only allows 10,000 lobbies at present, this will make sure we are using the number of available lobbies efficiently.
- f. Have a smooth user experience that is clear and easy to follow.

- g. Delete any system cache after the party closes ensuring data security.

6.0 Functionality Comparison

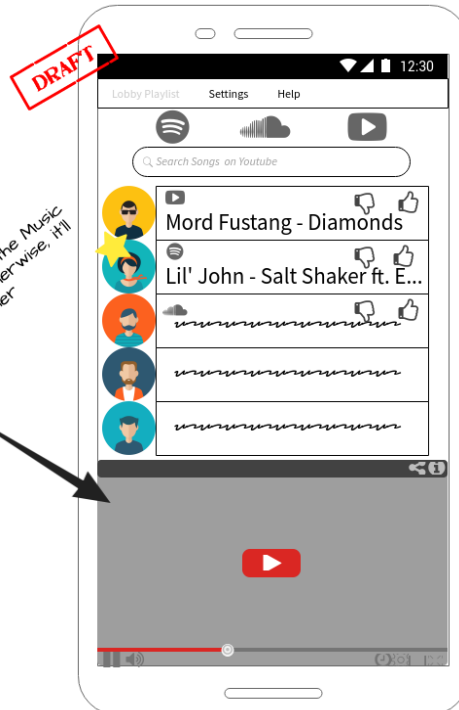
We view our biggest competitors of the market to be Spark.dj, Flo, Troppo, Festify, Jukestar. Many of these applications are only available on iOS or are web based applications. Since our application will be developed for Android, this diversifies our audience from the users of many other apps. Most of the apps other than Flo do not have spam prevention, which stops users from adding the same song multiple times. Juked prevents this by allowing each user one song in the queue at a time. While Flo also prevents this, it does not allow up or down votes to affect playback order. Troppo allows to play song from guests own phone via wifi, which can be a potential security hazard or violate file sharing guidelines. Juked is also one of few apps that will prevent guests song choices from affecting their auto generated playlists by using Spotify in a private session. Spotify has algorithms that use listening history to build playlists and song mixes based on a user's taste in music, and using Juked will not effect these algorithms. One shortcoming of Jukestar is that the application requires too separate applications, one for a host and one for a guest. The host application is only for party management, requiring a host to also download the guest application in order to add music.

Overall, Juked intends to have better functionality and user experience in comparison to other apps.

App/Req	Platform	Password Protection	Song Selection	Popular song	Spam Prevention	Affect Host's personal ranking?
Spark.dj	Apple	No	Spotify	Up/Down Vote	No	No
Flo	Apple	No	SoundCloud/Spotify	No	Host permission	Yes
Troppo	Android	Share via Wifi	User Device, online	Up/down vote	No	N/A
Festify	Web based	Party code	Spotify	Up/down vote	No	N/A
Jukestar	Apple/Android	Yes	Spotify	Up/ Down vote	No	No
Juked	Android	Random 4 digit code	Spotify	Up/Down Vote	Yes	No

7.0 Analytical Diagrams and Description

7.1 - UI Mockups



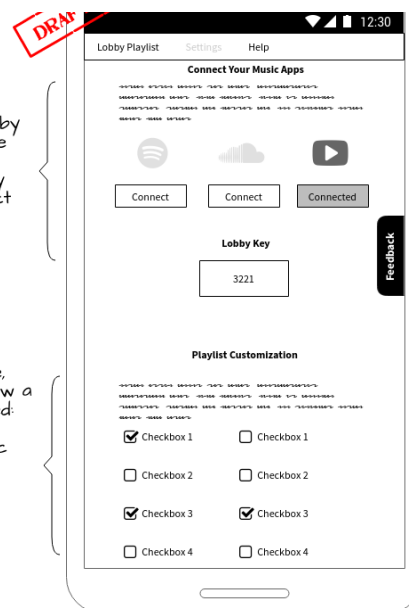
This is the Lobby menu where users will select songs and can view other users, etc

The UI/UX is designed to remain minimalist while maintaining a large amount of functionality on as little screen space as possible. One of the key points of our app vs the competition is ease of usability.

We hope to use a form of javascript for responsive UI.

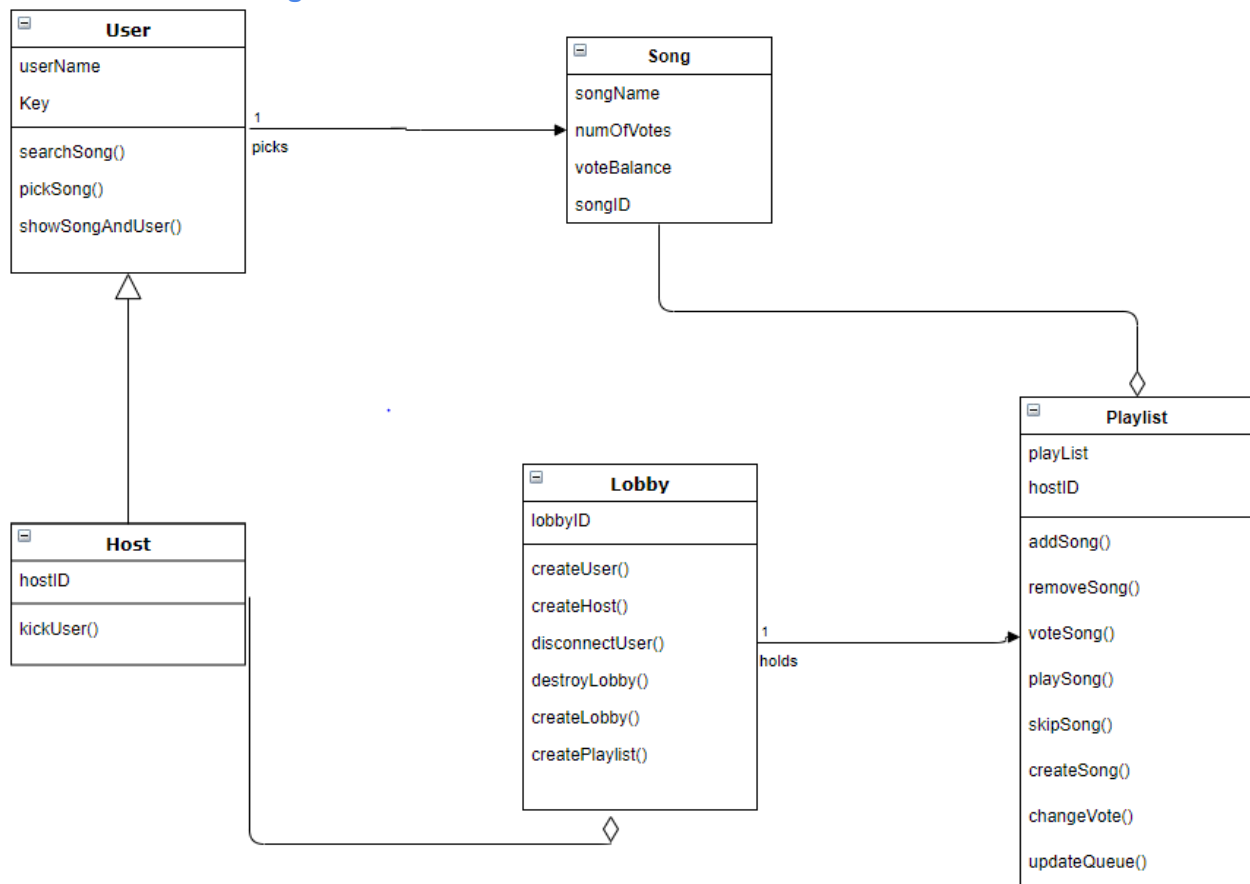
The User/Host can change lobby settings, change their avatar picture (if they please), connect apps to their account, etc

The Host, here, can change how a playlist is played: FIFO, Random, Vote Basis, etc



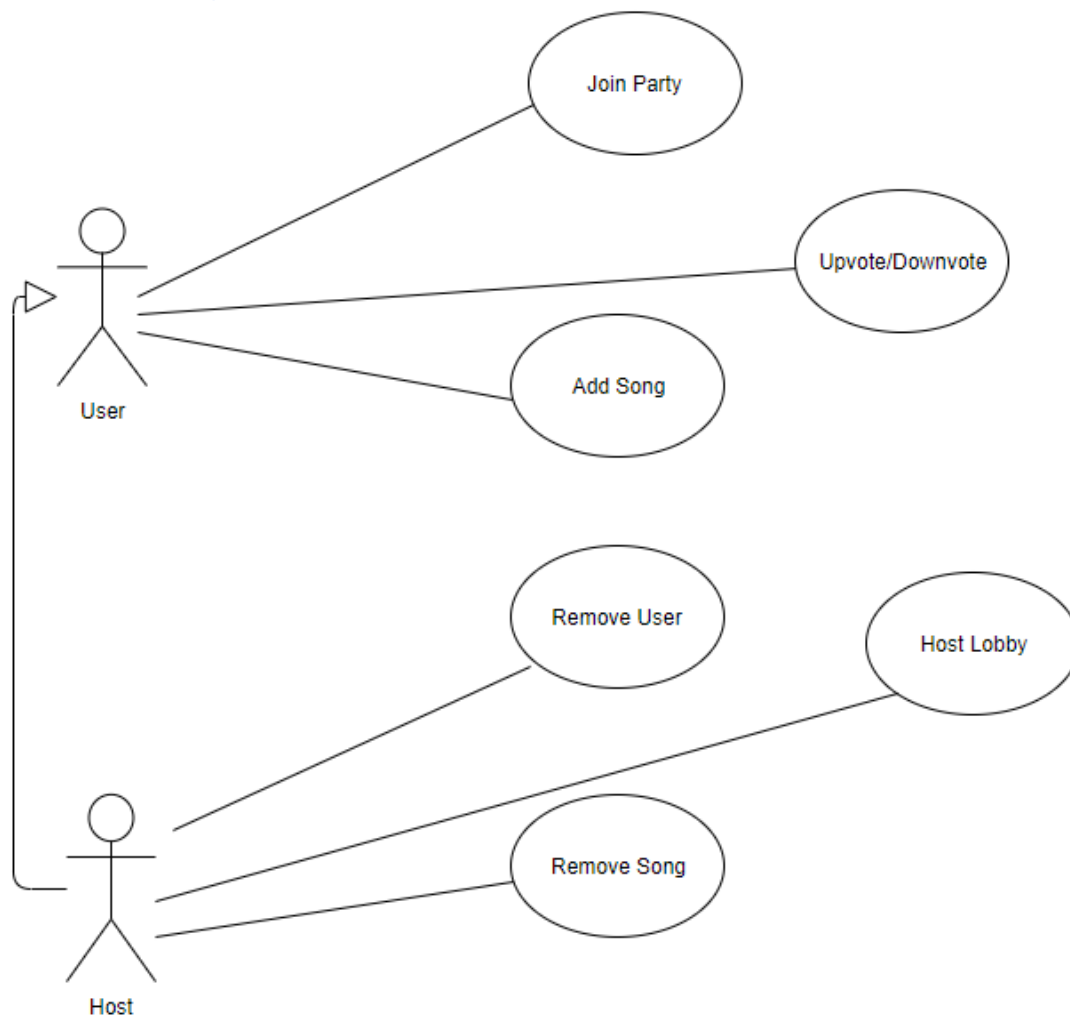
This is the setting screen usually only relevant for the Host of the lobby

7.2 Class Diagram



We currently plan on using five classes for this app. As seen, a user class will be a base class. It will hold a `userName` (or nickname) and a `key` that identifies it uniquely. The user class will currently have methods that allow them to search songs, select a song from the search results, and also show other users that are currently in the lobby and the song that they chose. A host class will be inherited from the user class. The host will have all the functionality of the user, but have an extra field called `hostID` that identifies them as the host. The host will also be able to kick a user if he has not been cooperating for any reason. A song class will create a song object consisting of only items that pertain to being played. Some of these are the song name, number of votes, the balance of the votes, and also the URI to Spotify or Soundcloud. The playlist class will hold the songs that users have chosen to be played. This is also where the methods that control how the songs are moved up, down, in, and out of the queue. The playlist will be a queue that is comprised of song objects, and has methods that allow for the changing of objects in the queue. Lastly, the lobby will hold all of the other objects. The lobby class creates the user, host, and playlist. It also holds a `lobbyID` that is stored in a database to uniquely identify that lobby.

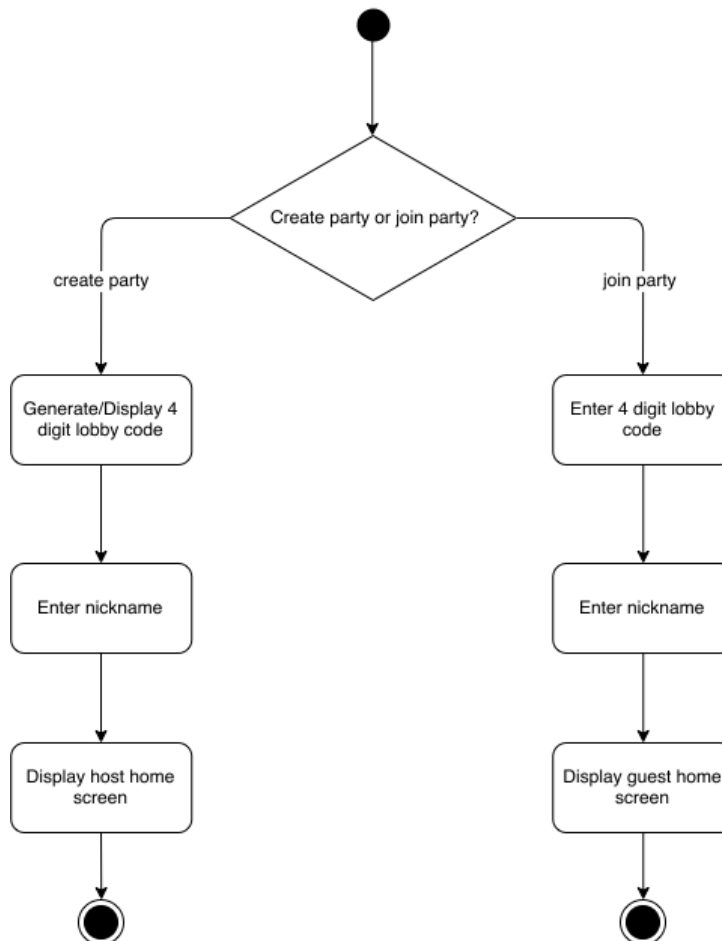
7.3 Use Case Diagram



The use case above shows the ways in which the application can be used. There is two possible actors, the host and the user. The user has the ability to join a party, upvote/downvote a song, and add a song. These are the uses that a user can do. The host can do everything the user can except for the ability to join a party is swapped with hosting or starting a lobby. The host also has the administrative ability to remove a user or remove a song from the playlist, add streaming services, and pause or play the music.

7.4 Activity Diagrams

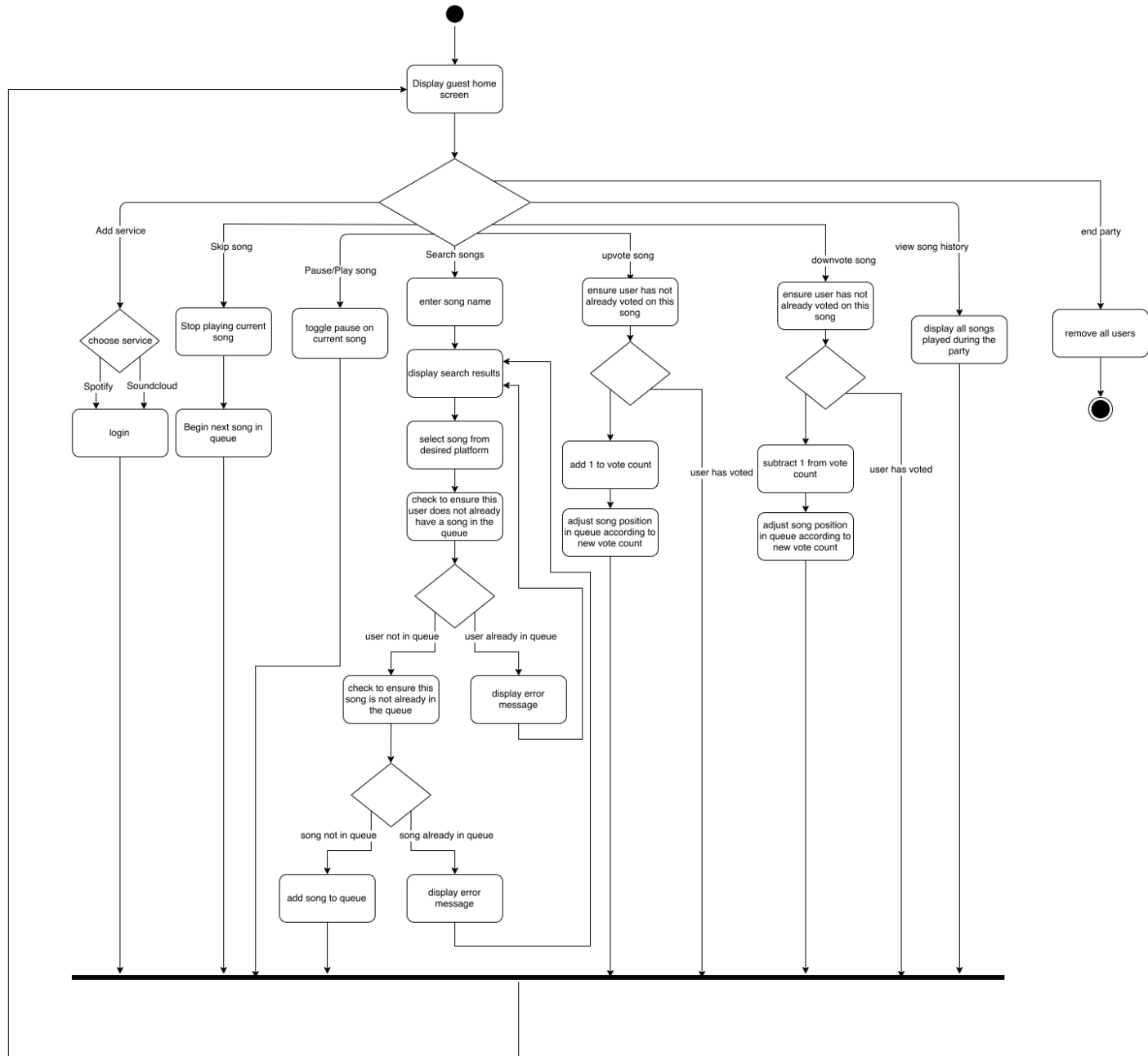
7.4-A Splash Screen Activity Diagram



The above activity diagram shows the user's options upon application startup. If the user is hosting a party, they will need to create a new party, or lobby. After selecting the host option, a four digit code will be generated for the new lobby. As covered in functional requirements, (Section 5.1 g) this will be a four digit number. The digit will be displayed to the host, so that they can share the code with their guests. From this screen, the host will enter a nickname that will serve as an identifier to all guests of the party. Once the nickname has been entered, the host will be taken to the main/home page of the Juked app.

Conversely, if a user opens the app and wants to join an existing party, they will select the guest option. The user will then be prompted to enter the four digit lobby code that was obtained by the party host. Once a valid party code has been entered, the guest will be prompted to enter a nickname, just like with a host. Once the guest has entered their chosen nickname, they will be taken to an abridged version of the home page.

7.4-B Host Home Page Activity Diagram



The diagram above shows the activities available to a host in the Juked application. All available actions will be available from the the host's 'home page'. The home page will display the current queue of songs that have been selected by the party guests, and different buttons to allow interaction with the app. The host will have a few more privileges than the guest to allow more control over the party and music selection.

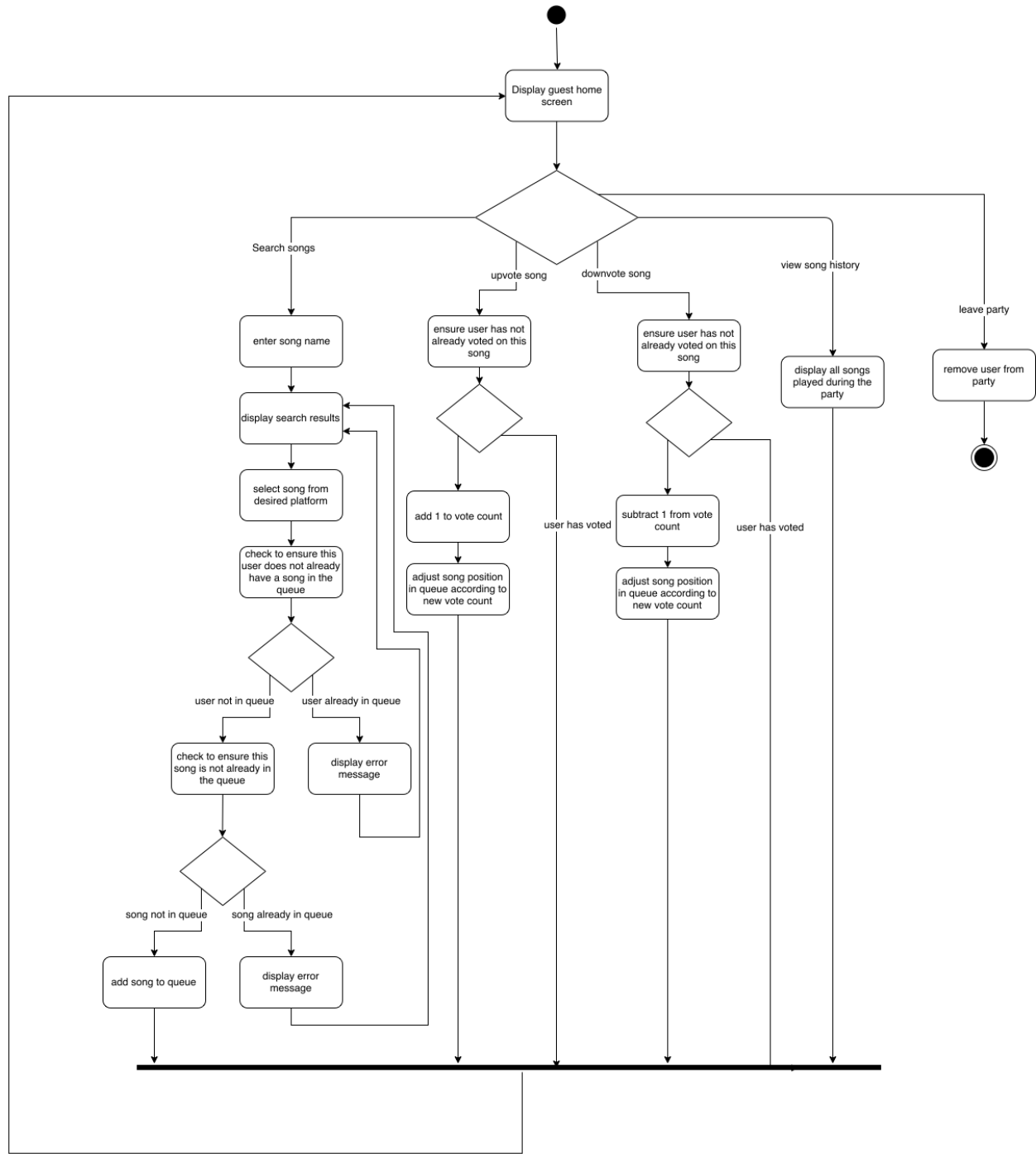
The host will have 8 different options from the home screen:

1. Add service
 - a. The host will choose to add a streaming service. The host will then choose to log in to their Spotify or Soundcloud account. Once the host has successfully logged in to a streaming service, they will be redirected back to the home screen.
2. Skip song
 - a. While any song is playing, the host has an ultimate "veto power", and can choose to skip the song. The song will stop playing immediately and the next song in the queue will begin playing.
3. Toggle play
 - a. If the host needs needs to stop the music, they can choose to pause playback. If the music is paused, the host can resume playback right where the music left off.
4. Search song
 - a. The user can type in the name of an artist, song, or album into a search bar. The Juked app will search through the libraries of the streaming services that have been added (Spotify and/or Soundcloud) and the results matching the search will be displayed. The user can then choose which exact song and which streaming service they wish to use. Once a song has been selected, the Juked app will then ensure that the user does not already have a song selection in the queue, and that the selected song has not already been placed in the queue by another user. (If a song has already been played during a party, it can be chosen again, so long as it is not currently in the queue). If either of these conditions fail, the user will receive an error notification and be redirected to the search screen so that they may search for another song, or wait until the have no selections in the queue. Once the song has successfully been selected, the user is redirected to the home screen.
5. Upvote song
 - a. The user has the option to vote once on each song. If the user likes the song choice, they can choose to upvote it. When the user selects to upvote, the Juked app will check

to ensure that the user has not already voted on the song. If the user has already voted on the song, no change will be made. Once the user has successfully voted, the song's vote count will be incremented, and the song will move up the queue until it reaches a song with a higher vote count.

6. Downvote song
 - a. The downvote option works identically to the upvote option, except that the vote count will be decremented, and the song will be moved down the queue until a song with a lower vote count is found.
7. View song history
 - a. The user can select the 'View Song History' option to view a list of all the songs that have been played since the party was created.
8. End party
 - a. The host can choose the end a party. This will remove all users from the party and all songs from the queue. All of the song and user information from the party will be lost. Once a party has been ended, the host and guests will be redirected to the splash screen.

7.4-C Guest Home Page Activity Diagram



The above diagram shows the activities available from the guest home screen. The guest has 5 options to choose from, and 4 of these are identical to the hosts options (see 7.4-B description).

1. Search songs - identical to host option 4
2. Upvote song
 - a. Identical to host option 5
3. Downvote song
 - a. Identical to host option 6
4. View song history
 - a. Identical to host option 7
5. Leave party
 - a. The guest may choose to leave a party at any time. When a guest leaves a party, their song choice is removed from the queue and the user is redirected to the splash screen.

7.5 Prospective Database Setup

Users

User_ID	User_Nickname	Host
12345	"lauraphillips"	true

Playlist

Song_URI	User_ID	VoteCount
song_12345	12345	2

Votes

User_ID	Song_URI	UpOrDown
12345	song_12345	"up"

The above is a basic overview of the SQL database setup for the Juked application. When a host creates a lobby, the above tables will be created. Each time a user joins a party, the server will be contacted and a new user will be generated. The User_ID will be auto-generated and unique. The User_nickname will correspond with the nickname entered by the user. The Host column will be a boolean value identifying whether the user is a host (true) or a guest (false). The playlist database will keep track of the queue. This table will contain an entry for each song added by party guests. The table will record the song URI (provided by the Spotify or Soundcloud API), the User_ID for the user that selected the song, and the voteCount integer. This table will allow the functionality to make sure a user does not add more than one song (by checking for the User_ID), and ensure that the same song is not added twice (by checking the Song_URI). The VoteCount column will keep a running overall score for each song, so that it can be placed in the queue accordingly. The final table keeps records of the votes. The columns here (User_ID, Song_URI, and UpOrDown) ensure that a user can only

vote on a song once. The UpOrDown column will allow the user to change their vote once it has been selected, but will not allow for multiple of the same vote; a user can not continually upvote a song, but they can switch an upvote to a downvote. With all of this information being stored on the Juked server, it can easily be retrieved to achieve the desired functional requirements.

8.0 Observations - Going Forward

App development is a process itself. Time management, team members participation, enthusiasm, communication and respect for each other makes this process smoother and enjoyable. Our first steps to actual development is securing API keys for Spotify and Soundcloud, so that the Juked app can interact with the streaming services and their song libraries. In each team meeting we are discovering potential pitfalls and technological shortcomings (knowledge and sometimes the technology itself) that the developing team will need to overcome, so getting started in the development process is our next step forward. In the end, reliability and security are higher priority than a flashy app that does not function correctly, crashes, and/or misuses user data.