

# **Complete Documentation**

Prepared by Catherine Chung, Clairisse Haines, Hunter Hurja, Ruth Libowsky, and Adam Moffitt

## Table of Contents

<b>Overall Concept</b>	2
<b>High-Level Requirements</b>	2
<b>Technical Specifications</b>	3
<b>Detailed Design</b>	5
Hardware Requirements	5
Software Requirements	6
Use Case Diagram	7
Database Scheme	8
Page Layouts	9
Algorithms	16
Class Hierarchy	16
<b>Testing</b>	17
<b>Deployment</b>	19

## Overall Concept

The desired project is to create a native “crowdsourced DJ” program named TroJams. The reason for the creation of this application is to provide a more dynamic mood for parties in which each party’s music playlist is generated based on user input.

## High-Level Requirements

### Implementation

This program will integrate Spotify’s API to fetch data and music from Spotify’s music catalog. Users can either be hosts, logged in users, or guests of the “party” - the environment in which the playlist is generated. If a user is a host, the music from the “party” playlist will play from his or her computer. The host and users can submit song suggestions which will show up in a Yik-Yak-like layout. Based off of Yik Yak’s layout and interface, the host and guests can then upvote or downvote the song suggestions. Songs will show up in the queue based on the total sum of upvotes and downvotes and will be played on the host computer in this order.

### Users

First, the user will have the option to log in to his or her account. The user can only be a host if he or she is logged in. Also, the user can only save the playlist from a party if he or she is logged in. Once the user chooses to log in or not, he or she can then choose if he or she wants to create a party. If he or she creates a party, then he or she is a host.

### Parties

The party can be broadcast as public or private. If it is public, anyone who connects to the app’s server can join and start voting. If it is private, then the host must choose a password that guests must enter in order to join. If the user chooses to be a guest, he or she can see the current parties and can select one. The user is then prompted to enter a password provided externally by the host in order to join the party if the party is private. Any user can join a public party.

### Song Selection

Once a host or a user are part of a party, he or she can submit song suggestions to the party. Hosts can only host one party at a time and guests can only participate in one party at a time. The host and other users have the ability to upvote or downvote the song suggestions which show up in a queue similar to Spotify. The songs with the highest sum of upvotes appear at the top of the queue and will be played in the order of the queue on the

host's computer. Songs will dynamically move up and down in the queue as the sum of their upvotes and downvotes change. Songs can be added back to the queue once they are played if they are suggested and have enough votes.

The host will have the option to end the party. When the host ends the party, users will be sent to the ending page and the playlist will be saved in their history. The host will also have the playlist saved in their history. The users can then decide if they want join another party or leave the application.

## Technical Specifications

### Pages/Windows of the Application:

- **TroJams Welcome page** - landing page where users can start the application
- **Login page** - where users can log in, create a new user, or be a guest
- **Create Account page** - where users can create an account
- **Selection page** - page where parties are listed and users can choose parties (and submit password if parties are private). Can also view profile account and click to create a party
- **Create a party page** - create a party name and password (if desired) for guests to join the party (host must have a Spotify premium account)
- **Party page** - guests can upvote and downvote already submitted song suggestions. Can also submit song suggestions
- **End page** - has option to leave application or join another party

### Descriptions of Each Page:

- **Login page - 10 hrs**
  - Simple window to connect it to a database (2 hrs)
  - Set up mongodb as a database with users (8 hrs)
    - Username, password (hashed) \* This will be the primary key
    - First name, Last name
    - Photo (optional)
    - Past playlists
  - Buttons at the end of page; login, create account, guest
    - Login is only enabled if all information is inputted correctly
- **Create a Party page - 6 hrs**
  - This page is for the host to create a party name, submit a password and create a party for guests to join
  - Once a party name/password have been submitted, this page will lead the host to a new page where the current song from Spotify will be displayed

- **Join a Party page - 14 hrs**
  - Displays parties that can be joined (6 hrs)
    - Must have password to join in on a party (unless the party is public and the host did not require a password to be used)
    - Uses networking
  - Potential functionalities: 8 hrs for functionalities
    - Button to host a party (Create a party)
      - Ask for the name of party and if it will be public or private
      - Initialize party object
    - Click on a party to join it
- **Party page - 18 hrs**
  - Password request if needed to join
  - Feed of songs with up vote button, down vote button associated with them - 4 hrs for single user, 4 hrs to network (8 hrs total)
    - ArrayList
    - Refresh button for user so that user isn't overwhelmed by rapidly changing order of songs
  - Way to add songs to feed — disable for guests - 7 hrs
    - Error message if the song suggested doesn't exist in Spotify's database, or is formatted incorrectly
    - Has a search bar where you can search for songs based on name
    - Doesn't allow addition of a song that is already on the party playlist
    - Functionality to leave the party
  - Logic
    - Feed is the current view of the order in which songs will be played (This can change before songs are chosen to be played next, songs are chosen to be played next 5 seconds before the current song ends)
    - Song order updates when users press refresh button
    - After a song is played, it is removed from feed
- **Backend: Spotify API - 8 hours**
  - Using RapidAPI to create a rich back end for the app, with real-time data and notifications and integration of Spotify's API
  - The SpotifyPublicAPI Package can grab any music data from Spotify's music database
  - Will take the song searches from users and grab information such as artists, album artwork
  - Create a queue based on the song suggestions

Limitation of being a guest/not logged in: Can't suggest songs , can't host a party, can't save a profile

### Classes:

- **User**
  - Username, password (hashed)
  - First Name, Last name
  - Photo (optional)
  - Playlists
- **Party**
  - Name
  - Public or private
    - Password required if party is private
  - Has a feed of all songs
    - ArrayList implementation
    - Refresh button for user so that user isn't overwhelmed by rapidly changing order of songs
- **SongQueue**
  - Arraylist of songs
  - addSong(song)
    - Adds the song in the correct location
    - Check to see if the song is already in the playlist
  - upvoteSong()
  - downvoteSong()
- GUI classes for each screen
- Networking classes: Client and Server and Messages

## Detailed Design

### Hardware Requirements

#### Windows:

Windows 8 (Desktop)

Windows 7

RAM: 128 MB

Disk Space: Recommended 512 MB

Processor: Minimum Pentium 2 266MHz processor

#### Mac OS X:

Intel-based Mac running Mac OS X 10.8.3+, 10.9+

*Linux:*

Oracle Linux 5.5+

Oracle Linux 6.x (32-bit), 6.x (64-bit)

Oracle Linux 7.x (64-bit)

Red Hat Enterprise Linux 5.5+ (32-bit), 6.x (64-bit)

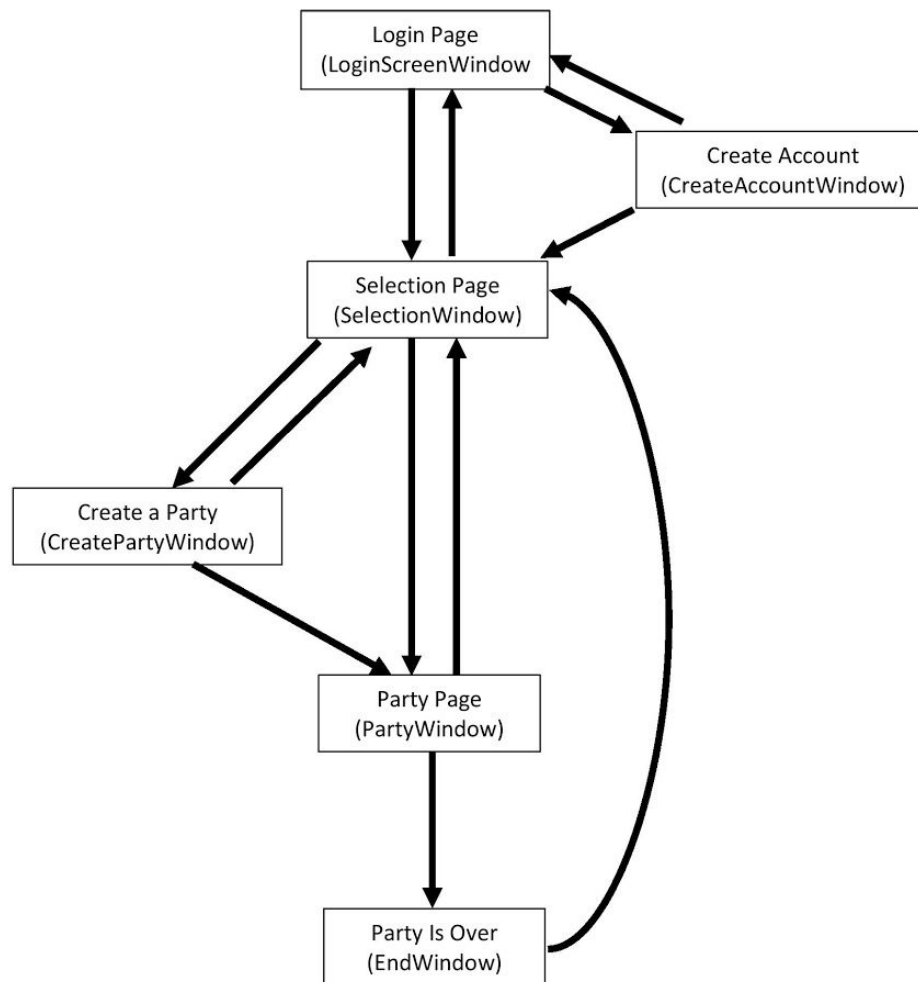
Ubuntu Linux 12.04 LTS, 13.x

**Software Requirements**

Java 8

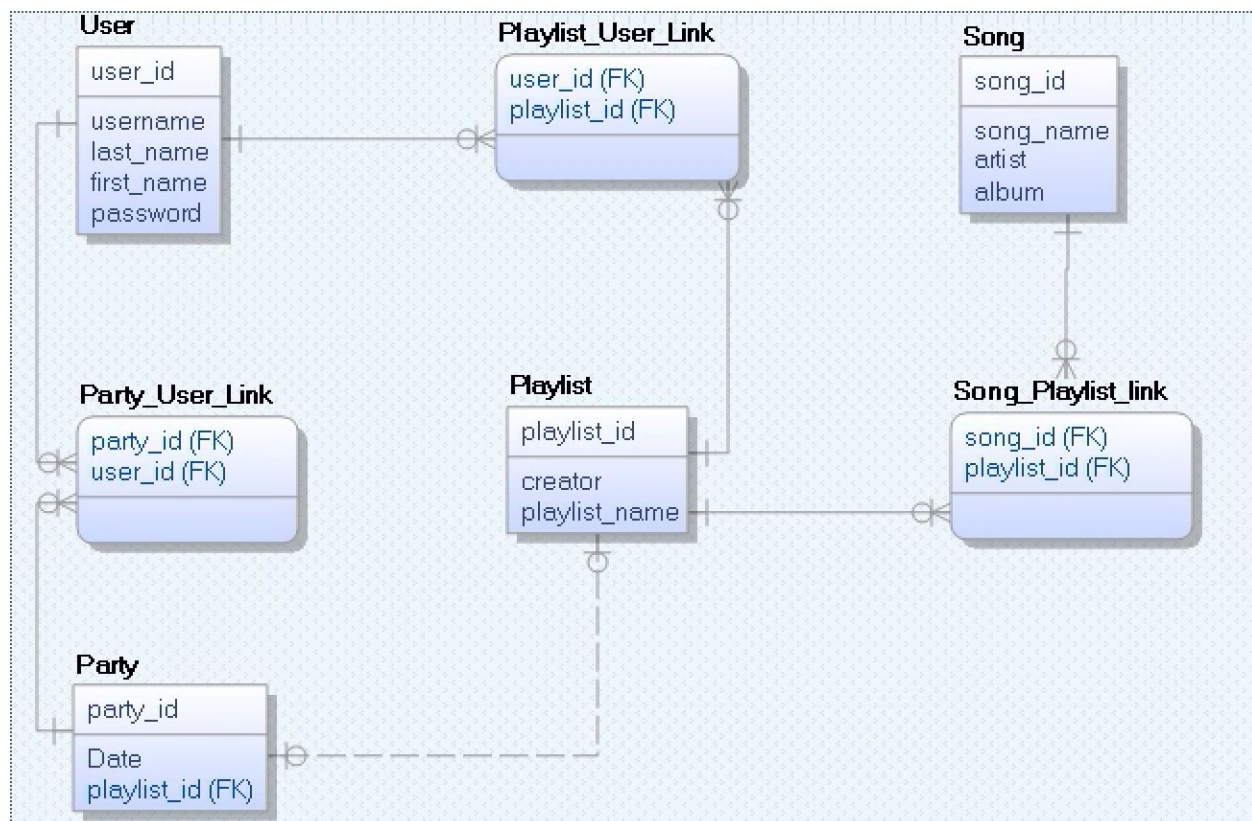
Eclipse IDE for Java EE Developers (Luna/Mars)

## User Case Diagram





## Database Scheme



## Page Layouts

### TrojamWelcomeWindow

- Very simple opening window. Contains a non-opaque button “Click to Party”. Clicking it disposes the current window and shows a LoginScreenGUI



### LoginScreenGUI:

- There are text fields for the user to enter his or her username and password. Until both of these text fields have been filled, the “Login” button will be disabled but the “Guest user” and “Create new account” button will be enabled. Once the user fills in these text fields, the “Login” button should be enabled.
- When a user presses “Create Account”, the LoginScreenGUI and CreateAccountGUI switch with a card layout.
- When a user presses “Login”, we check to make sure that the credentials are valid. If invalid credentials are given, the GUI displays an error message accordingly. Otherwise, the LoginScreenGUI is disposed and a new SelectionGUI is displayed.
- When a user presses “Guest User”, the LoginScreenGUI is disposed and a new SelectionGUI is displayed.

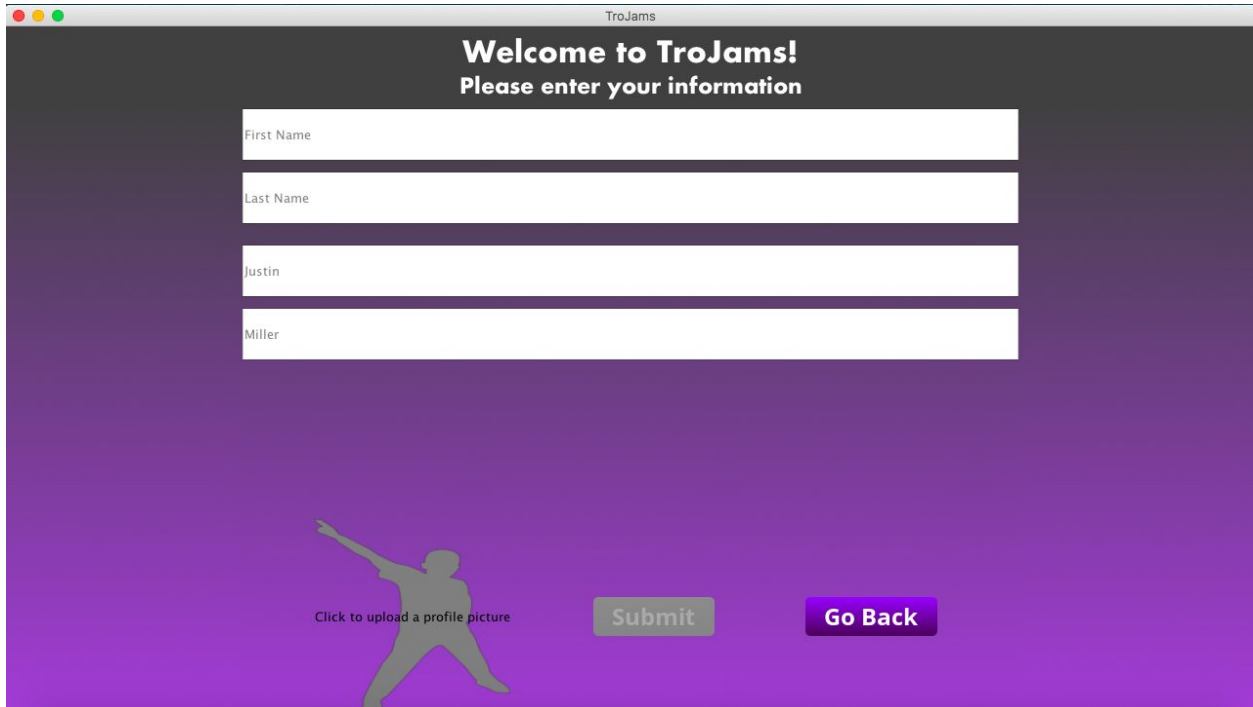
The image displays two screenshots of a web application titled "TroJams!". The interface has a dark purple gradient background. At the top, the title "TroJams!" is centered in a large, white, bold font. Below the title are two text input fields. The first field is labeled "username" and contains the text "Justin". The second field is empty. Below the input fields are three buttons: "Log In" (disabled, grey), "Join as Guest" (active, purple), and "Create Account" (active, purple). Below the buttons is a logo consisting of a stylized globe made of dots, with the word "TroJams" written in a pink, cursive font underneath it.

The second screenshot shows the same interface, but the "Log In" button is now active (purple) and the "Join as Guest" and "Create Account" buttons are disabled (grey). The "username" field now contains the text "Justin" and the second field contains the text "Miller".

### CreateAccountGUI:

- There is text fields for first name, last name, username, and password. The submit button is disabled until all of these fields are filled in.

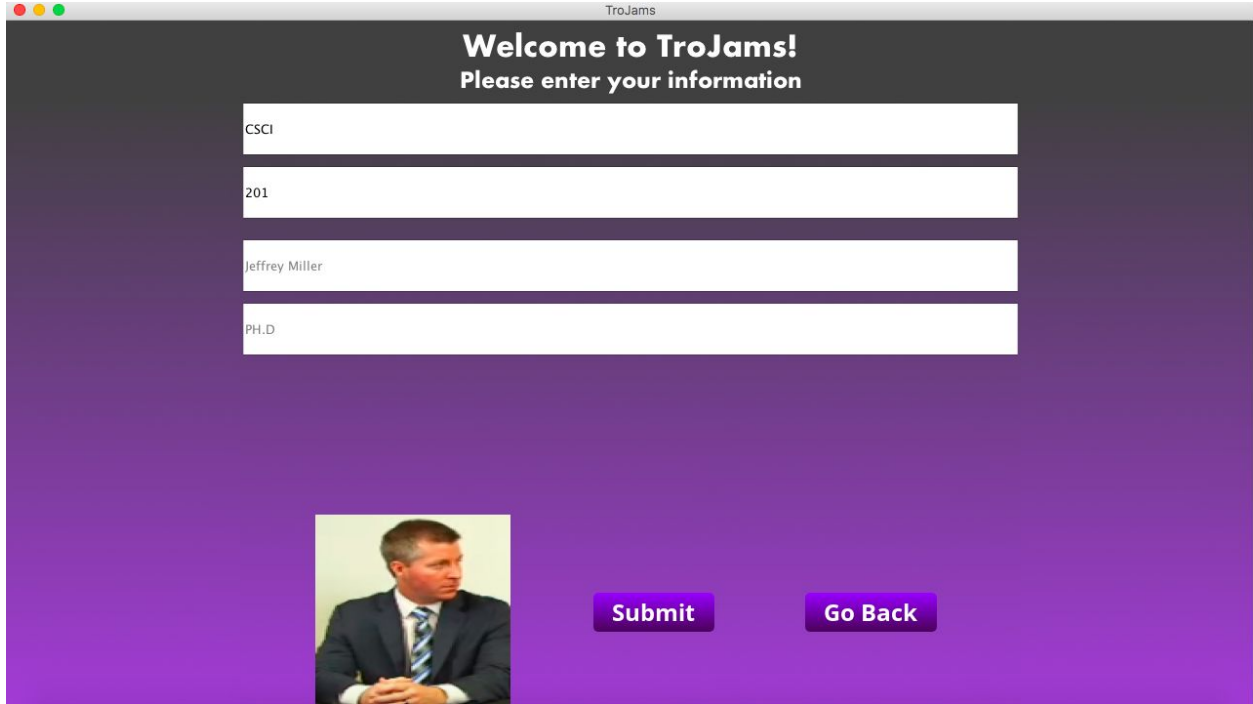
- There is a default image that the new user can choose for his or her profile image. If the user clicks on the image, a JFileChooser is displayed so the user can upload an image of his or her choice. The JFileChooser only accepts JPEG/PNG files.
- When the user clicks the “Submit” button, we ensure that the username does not already exist. If it does, an error message is displayed on the GUI. Otherwise, the CreateAccountGUI is disposed and a new SelectionGUI is displayed.
- If the username does not already exist, a user object is created with all of the inputted user info and added to the database table of registered users.



The screenshot shows a Java Swing window titled "TroJams" with a purple gradient background. The window contains the following elements:

- Title Bar:** Standard macOS-style window controls (red, yellow, green buttons) on the top left.
- Header:** "Welcome to TroJams!" in bold white text, followed by "Please enter your information" in a smaller white font.
- Form Fields:** Four white rectangular text input fields stacked vertically. The first three are empty and labeled "First Name", "Last Name", and "Justin". The fourth field contains the text "Miller".
- Footer:** A silhouette of a person in a dynamic pose. To its left is the text "Click to upload a profile picture". To its right are two buttons: a disabled "Submit" button (gray) and an active "Go Back" button (purple with white text).

Unfilled Create Account Window. Submit button is disabled until all of the fields and picture is entered. Username and Password are automatically filled in if user entered them in the Login window before clicking Create Account Button.



trojams

## Welcome to TroJams!


Please enter your information

CSCI

201

Jeffrey Miller

PH.D

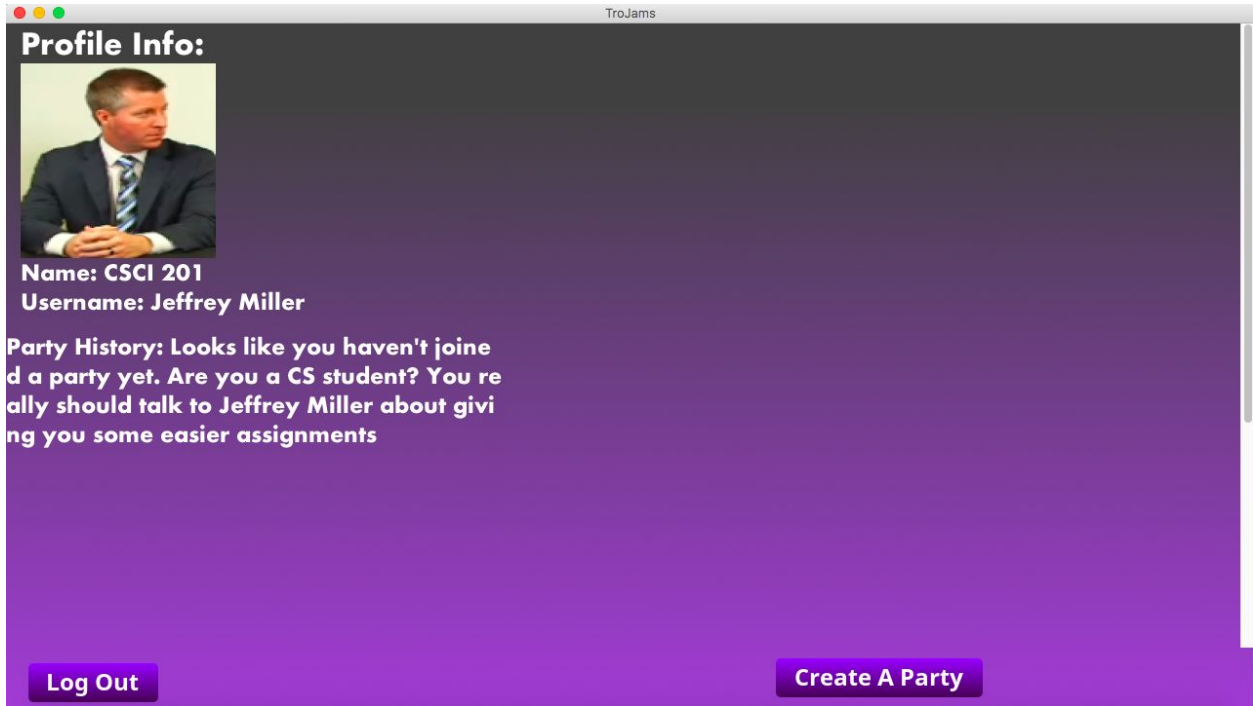


Submit Go Back

Filled in Create Account Window. Submit button is enabled, and text over image is gone.

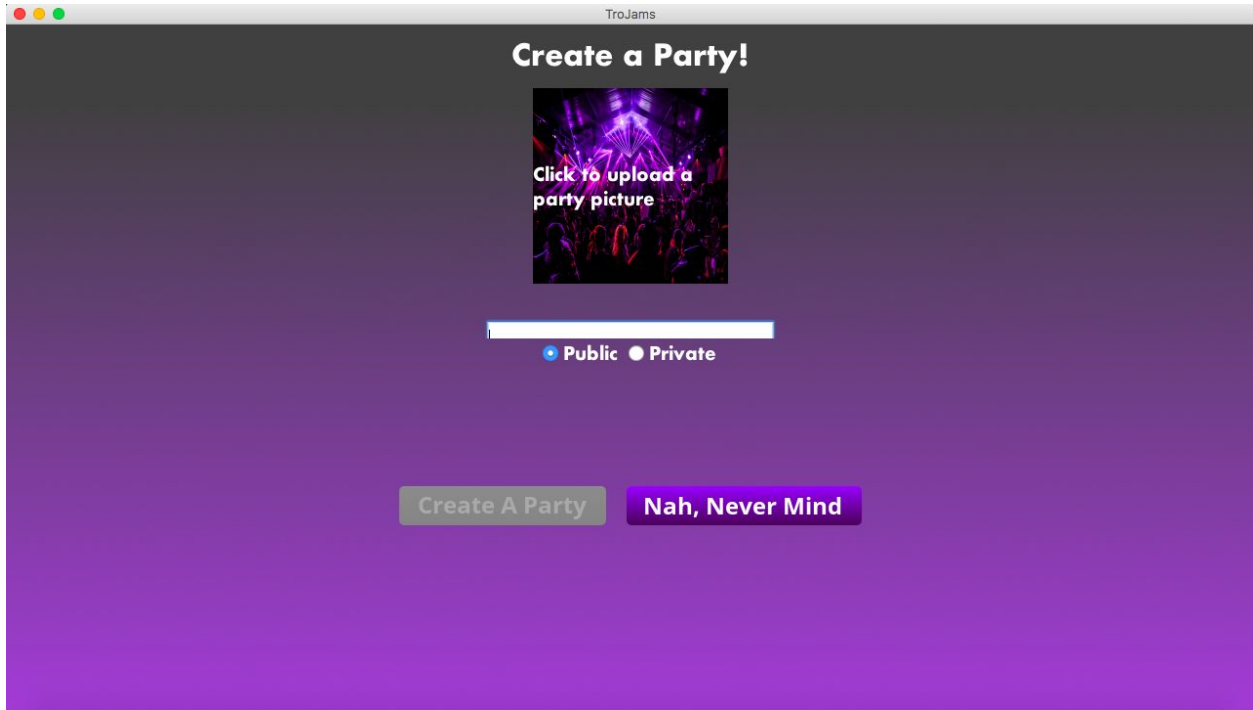
### SelectionGUI:

- Clicking the “Create Party” button shows a CreatePartyGUI instead of SelectionGUI through card layout. This button is disabled for guest users.
- Clicking the “Join Party” button will take the user to a new PartyGUI. If the party is private, the user will enter the password via a JDialogBox before he or she is taken to the PartyGUI.



### CreatePartyGUI:

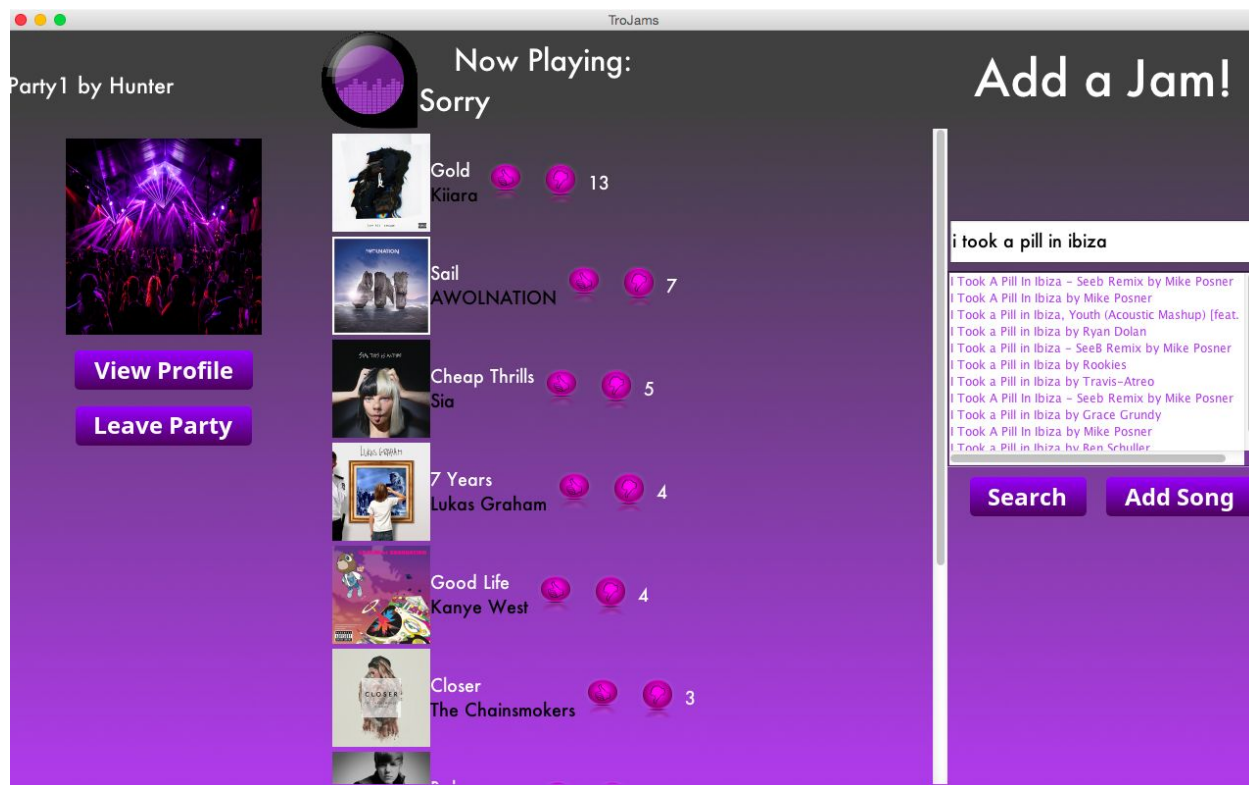
- The CreatePartyGUI gives the user (now a host) the ability to create a party.
- The host must put in a party name in a text field.
- There are two radio buttons for public or private. If the host desires the party to be private, he or she must also create a password as well. The text field for the password will only be visible if the user has toggled the "Private" radio button.
- The "Create" button disposes of the CreatePartyGUI and create a new PartyGUI. This button will only be enabled if the party name has been given and for private parties, if the password has been given.



### PartyGUI:

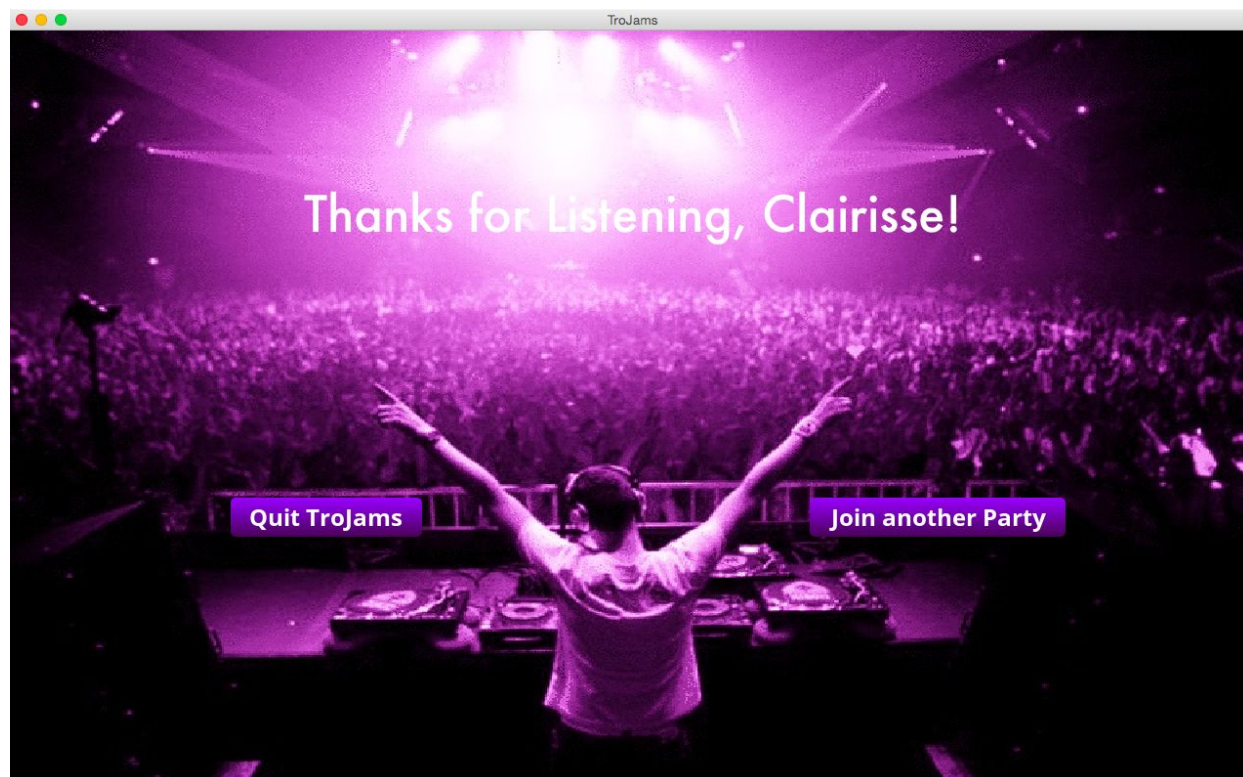
- The “Add Song” button will create a new AddSongGUI.
- The name of the song that is currently playing will be displayed.
- The songs to be played (in the order that they will be played) will be displayed, along with their upvotes and downvotes. The user can click the up and down arrows to upvote and downvote specific songs.
- There is a panel on the side that gives the guest/user/host the ability to suggest songs.
- There is a search box where users can type songs and add them to the party.
- After pressing enter, the song suggestion will be submitted. The user can submit as many song suggestions as possible, as long as they are valid and have not been submitted before.





### EndGUI:

- EndGUI will have a closing message and the option to join another party





## **Algorithms**

### **Search: Parties**

If the user is logged in, all of the parties will be displayed. If the user is not logged in, the user will only be able to see parties that are public. The user can then select which party he or she wants to join.

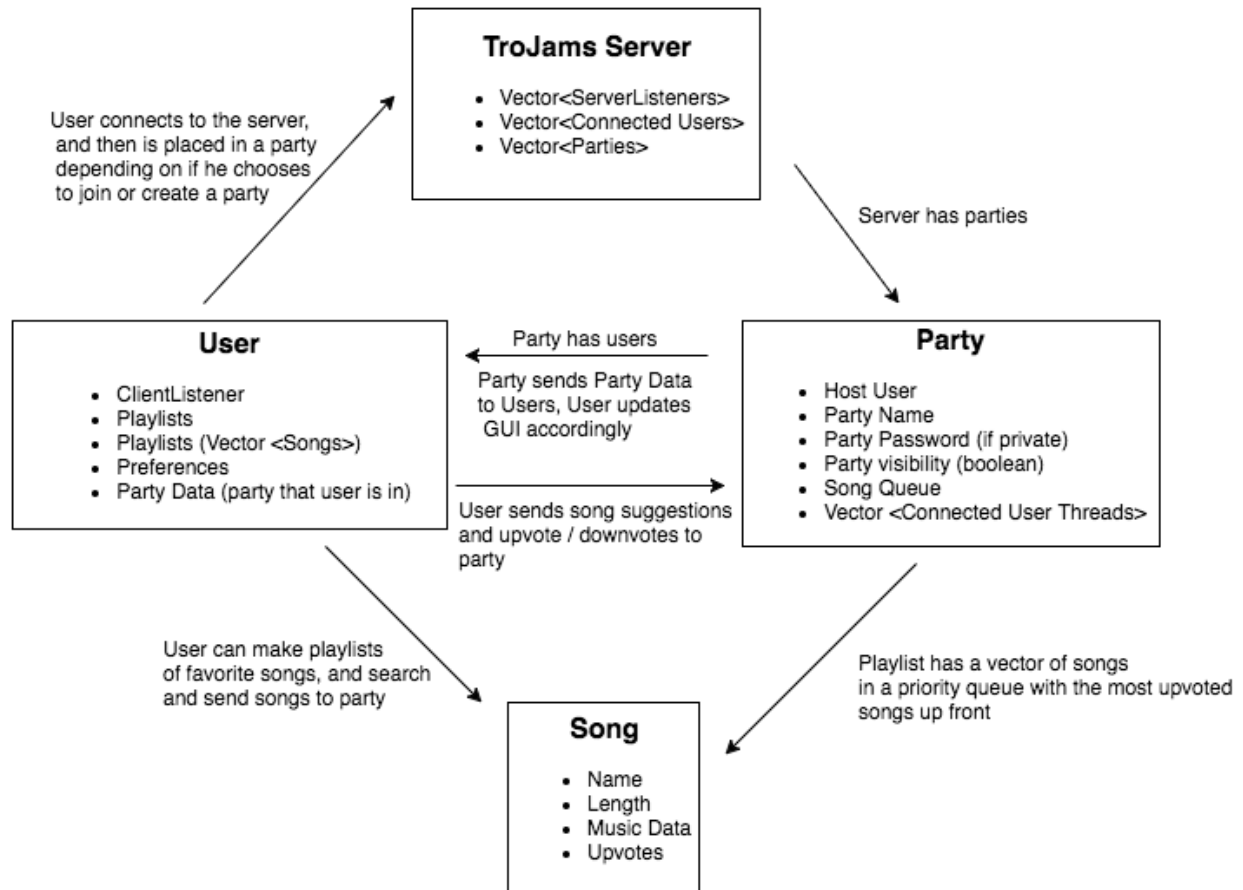
### **Search: Songs**

There is a search bar in the PartyGUI where the user can search for songs based on name in Spotify's database. The user will have to search part of a song name and we will implement a search algorithm to display matching songs. The user can then select a song and add it to a party. An error message will be displayed if the phrase that the user searches doesn't exist in Spotify's database. Additionally, a user cannot add a song that is already in the party.

### **Song Queue**

The songs are listed in the PartyGUI based on the number of upvotes they have. The rankings of songs will be refreshed when the user presses the "refresh" button on the PartyGUI. We will be using an array of pairs to store the songs and their upvotes. We will have logic to order the songs by number of upvotes when the user hits refresh. A user can only vote on a song once (either upvote or downvote) to avoid vote manipulation.

## **Class Hierarchy**



## Testing

### General Tests (will be used for every screen)

- Check logout functionality from every screen
- Access profile screen from any screen and press back button to return to previous
- Able to quit the overall program from any screen
- When exiting any screen or pressing back/cancel button, program gracefully disposes the current screen

### Login Screen screen

- Both fields empty
- User field empty, password filled
- User field filled, password field empty
- Both fields filled, username is not in the database
- Username is in database but password does not match
- Username matches one instance in db, but pass matches another
- Guest User button works
- Create new account button leads to the correct gui

- Login button is disabled until both text fields have been filled

### **Create Account screen**

- Check that only JPEG and PNG files can be uploaded as profile picture
- Check that username must be unique - case insensitive
- Create account button disabled until all fields are filled (not including photo - optional)
- Check to make sure that each new user is added to the database correctly

### **Profile screen**

- Correct information/picture is displayed for each unique user
- Able to get back to the main screen from the profile screen via a back button

### **Selection screen**

- Check that public parties can be entered without a password
- Check that scrollbar works once enough parties are added
- Check that private parties require a password
- Create a party navigates to the correct gui
- Test that the search functionality works correctly (ad hoc tests)

### **Create Party screen**

- Check that parties can't have duplicate names
- Password text field toggles with the check box for private
- Make sure that the password is stored correctly with party
- Check for blank password field
- Check for blank name field
- "Create Party" button is disabled until valid name (and password if required) have been filled out
- Create a public party
- Create a private party

### **Party screen**

- User can only upvote a song once (then upvote arrow is highlighted)
- If user has upvoted a song, can downvote it (but only once)
- User can only downvote a song once (then downvote arrow is highlighted)
- If user has downvoted a song, can upvote it (but only once)
- Songs are displayed based on their overall rating (in decreasing order)
- Songs will refresh correctly based on updated votes when refresh button is pressed
- If two people try to vote on a song at the same time (make sure that synchronized/locks work correctly)

### **Add Song screen**

- Check that a song already in the queue can't be re-added

- Check that song search correctly finds songs, returns “song not found” message if song doesn’t exist
- Test that the search functionality works correctly (more ad hoc tests)

**End screen**

- Correctly saves playlist if user selects save playlist option
- Clicking the quit button gracefully exits the program

**Deployment**

1. To deploy this application, you must download the most recent Java SDK as well as Eclipse Neon.
2. To deploy this application within Eclipse, import the TroJams.zip file into Eclipse. This will create a project called TroJams with src and images directories.
3. Download JLayerME 0.1.3 and add it to the project as an external JAR (Project -> Properties -> Java Build Path -> Libraries -> Add External JARs).
4. To execute the program, run TroJams/src/main/Main.java.