**JUKE Technical Specifications**

Chris Cognetta, ccognett@usc.edu

Rama Gosula, gosula@usc.edu

Nayha Kamboj, nkamboj@usc.edu

Meghan Mehta, mehtam@usc.edu

Katie Park, katiepar@usc.edu

Jason Roodman, roodman@usc.edu

**Opening Page (6 hours):**

**-** Will contain the login with username and password, or can sign in as a guest

-login implemented by checking username and password against the database. Has textfields for both. Guest sign in just a button

- If the user doesn’t have an account, then they can create one. A new screen will show

  up, and ask the user for their email address, username, and password. This

  information will then be saved in the database.

-Once user is logged in, the main page will open to a simple column of four different buttons

- Start Playlist

- Join Playlist

- Manage Playlists

- Manage Friends

If the user is logged in, they will be brought to this window already.

If the user has joined a playlist, their opening page will be the playlist main menu.

**Playlist Main Menu (8 hours):**

* This window will be broken down into a left and right pane, using andoid(8 hrs)
  + The left pane will show the list of songs. The currently played one will be highlighted in yellow. The user can scroll up to see songs previously played (grayed out) and scroll down to see songs to be played. Next to each song to be played in the slider, there will be an up and down arrow to vote for or against a song, and a number for the total number of votes that songs has.
  + The right pane will contain a column of buttons. The first will be a refresh button. Then an add songs button. The following button will be an “add friend” button (will not be present if already friends). Finally there will be a leave party button.
  + The host will have thin pane on the bottom of the screen with a pause/play button, stop button and a volume control slider. There will also be the ability to keep a playlist private, share with all followers, share with all users of Juke, or share with a select few people. If they click a select few people, then they will be prompted to enter their usernames.
* The playlist will be stored in the backend via a Playlist object with an arraylist for allsongs, playedsongs, and upcoming songs.(2 hrs)

**Implement Playlist (9 hours)**

* Create music node class with music metadata (1 hour)
* Implement Client update thread (3 hours)
* Connect database to playlist thread (1 hour)
* Implement refresh (2 hour)
  + The order of the songs updates automatically after 5 minutes of non-usage, once a user reloads the Playlist Main Menu, or clicks the refresh button. The information in the Playlist Main Menu will be repopulated by the information from the thread.
* Implement thread to receive actions from users and update playlist (3 hours)
* Implement music playback using existing android functions (2 hours)

**Join Playlist (12 hours):**

* This window will have 3 buttons
  + **(4 hours)** Join by username where the user will be prompted to type in the username of the host (if the host is not hosting a playlist, nothing will happen). Connect to socket
  + **(4 hours)** Join by location where the app will find hosts within a certain radius to join, using location-based service to check within radius.
  + **(4 hours)** Join by QR code where the app will open to the camera and the user can use a QR code generated by the host to join their playlist

**Manage Playlists (8 hours):**

* Playlists are stored, loaded, and updated in the database with a playlist table
* This window will have 2 panes again
  + The left pane will have a scrollable list of all the playlists that the user has liked or has generated on their own. Clicking on one will bring up a new window that takes up the whole screen, showing a list of all the songs in the order they were played as well as their final score and song data
  + The right side will have the button to delete playlists, and add tags to a playlist.

**DataBase (6 hours)**

* The user table will have a column for name, email address, randomly generated unique ID number, number of playlists, and playlist’s ID number.
* The playlist table will be for playlists and contain columns for playlist ID Number, number of songs, all the song names, and IDs that had access to the playlist (could be all or a select few).
* The database will also have helper functions to access user id, playlist id, and email address.

**Track Changes:**

* Database:
  + Added one extra table
  + Table for playlistcontent which consists of the playlist song ID, the playlist ID, song index, song name, and song artist
* Webserver:
  + We have added a webserver, in which the client connects directly to the server through jdbc and queries the database through the container class
  + We have a Database connector class for the client to connect to the webserver