CS105 Final Project Report
Manav Kohli and Tyra He
5/5/15

For our project we built a web application, Jameez, that allows users to collectively listen to a group playlist and rank songs in their favorite order. Furthermore, users can view how their friends ranked the songs and generally how popular each song is. Any visitor can view the login page. However, only users may visit the ranking page and view the songs in their favorite order.

We implemented our project using a variety of languages, including PHP for our backend, Javascript, JQuery, HTML, and CSS for our frontend, and MySQL for our database. In addition, we used the Soundcloud SDK to add the Soundcloud widget to play music within our application. In addition, we covered key security concepts, such as password checking with the databases. Finally, we used session handling to gather user information and determine access to different pages.

The goals of the project were to build an interactive web application that allows users to collaboratively listen to music, use key programming concepts covered in class, and learn to work with APIs. While we were able to develop a program that allowed users to collaboratively listen to music, we were limited in our ability to change the playlist since we did not have time to implement the OAuth login process. However, we were able to integrate almost all the languages covered in class, and made extensive use of PHP to both interact with our backend as well generate HTML content on our pages. Finally, we were successful in working with APIs since we worked with the Soundcloud API to view playlists and tracks.

If we were to spend another month on our project, we would implement the OAuth login process to allow users to login with their social media accounts to share the playlists they created with their friends and followers on Facebook and Twitter. In addition, we would add more design factors to the user page and ranking page so that there would be an improved user experience. Finally, we would have liked to make the user page more robust so that users could interact with each other by ‘posting’ on each others’ walls, commenting on their rankings, etc. Time permitting, we would add all these features to our web applications

Manav:
Files created: global.js, logout.php, userpage.php, default.css

1. Soundcloud API implementation
    a. Authorized webapp with Soundcloud, researched documentation and handled all widget integration

   b. Parsed song/playlist URIs to display using JS
   c. Embedded widgets in page with combination of PHP, HTML, and JS
2. CSS styling/Design
   a. Completed all styling from scratch
   b. Responsive design to fit width/height of browser instead of static heights and sizes
   c. Used Google fonts to improve look and feel of text
   d. Designed header, footer, and content of each page to complement user experience and focus on key information

3. Display the playlist
   a. Built page (aggregate.php) to display the playlist and allows users to listen seamlessly (i.e. songs automatically play after one another)
   b. Used HTML and PHP to build page
   c. Used Javascript to incorporate Soundcloud playlist

4. Javascript integration to access database and send requests to Soundcloud
   a. Contributed to Javascript function that displays songs in the correct order
   b. Contributed to PHP code that retrieves user information from the database and passes that into Javascript call
   c. Display songs using Javascript once the URI's had been parsed

5. Userflow/application architecture
   a. Identified broad goals of app and purpose of using Soundcloud API
   b. Determined which pages were necessary (i.e. Login, ranking, 'home', and logout page) to optimize user experience
   c. Styled pages accordingly

Tyra:
Files created: aggregate.php, db.sql, login.php, rank.php, utils.php, scwidget.js

1. Building the Database and its interactions with other parts of the application
   a. SQL Database Construction (using mysql)
   b. Interactions between Registered User and the Password in the Databases
   c. Interactions between Rank.php and the Database
   d. Storing the data after the user ranks the app

2. Building the Login Page (not including CSS)
   a. HTML/Javascript for a basic login page that includes registered user and new user, and password matching

   b. Enables the functionality that only the registered user can access into the system (security concern)
   c. Enables the functionality that when a new user is created, a new row with the specific user password is created as a new row in the database (database)
   d. Pass all user information variables to other PHP sessions

3. Building the Ranking and Aggregation system (in rank.php)
   a. Programed in Javascript to dynamically handle newly submitted user input
   b. Also use another Javascript function to calculate the sum(Manav later changed to average) of the total ranking numbers
   c. Found a way to update the result dynamically, so there is no need to refresh the page

4. Implementing a Draggable Ranking Form
   a. The draggable event is evolved from just having the users put in numerals from 1 to 5, but later in order to limit users to enter numbers that are not in the range, or not a number, we decide to create more standard output by using the draggable form.
   b. The format can be found from the StackOverFlow, but the actual implementation contains much modifications of the code.
   c. It's an element in the HTML form.

5. Sorting the Ranked Songs in the Userpage.php
   a. Programed in Javascript to realize the functionality of sorting the ranked songs
   b. The function requires interactions with the databases
   c. I finished the sorting, and Manav put the actual soundtracks below the sorted song lists.

Tyra and Manav:

1. Designed presentation