/\/\USIC

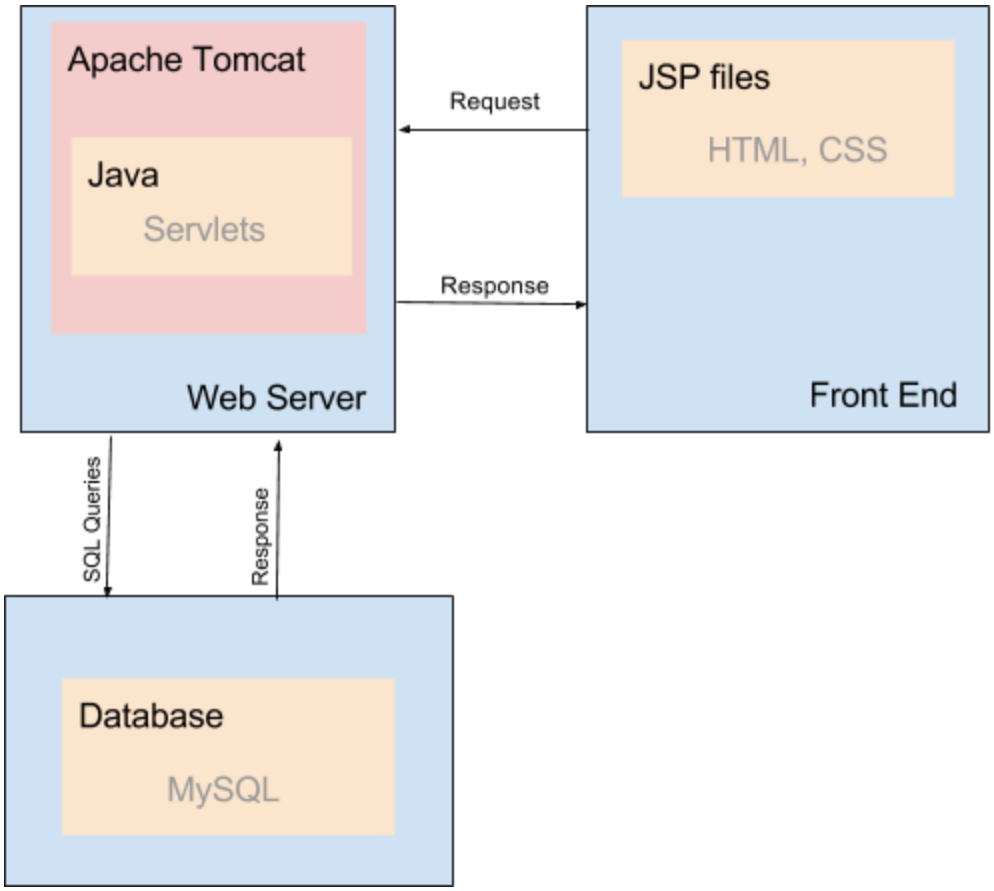
Vishaal Prasad, Elton Vinh, Austin Yam
Team 6
CS 157A Section 3

Project Description

- This project will be a database application that is based in the music domain. The goal of this application is to provide a convenient way to listen to, discover, and share music. Users will be able to easily organize songs into playlists and make them public if they so choose to. This application will also provide a user-friendly interface that is clear and intuitive. The constant struggle of issues with other similar applications, provided the motivation to create another such application with those problems addressed.

System Environment

Structure of the System



Hardware and Software used

- AWS EC2 instance
- Ubuntu
- Apache Tomcat

RDBMS

- MySQL Community Server 5.7.19

Application Languages

- Java 8, Java EE 7, HTML, CSS, Javascript, XML, SQL

Functional Requirements

How users will access the system

The system provides functionality for multiple types and subtypes of users: admins and consumers. Consumers shall be able to access the system for read and limited write capabilities. Consumers are the core target audience and the system shall be designed with them primarily in mind. Consumers shall be able to request music data, view the data and make decisions based on the their particular interests. Consumers will NOT be able to make modifications to this data. Consumers shall be able to make limited write capabilities through functions such as playlist creation. Consumers shall be be categorized as registered or unregistered users. This distinction shall determine what they will be able to do. In addition, admins will be able to add music to the system along with deletion and updating. Users/stakeholders will access the system through the internet. The user shall be able to input the url to the website in any popular search engine or directly input a url and be provided with our system.

Describe each individual function, functional process and I/O

Functions:

Request song

- The user shall be able to search the database for songs by typing in a characteristic of the song such as name, artist, or year in a search box.
- The system shall provide a song or songs that satisfy the search parameters. In the event that no song matches the search parameters, a message will be shown to the user to indicate that no song was found.

Browse songs

- The user shall be able to browse the database or a subset from the database for songs. The user shall be able to browse all songs or click on a subset to view the subset.
- The system shall provide a list of songs from the database or subset. The system will optionally also allow consumer to specify subsets such as genre or artist for browsing. The system shall display the browse results in different pages. The system shall allow users to click on a song to show relevant data.

Pages in browse results

- The user shall be able to browse songs split up into pages.
- The system shall provide the user with the ability to set the number of songs displayed per page. When the user clicks on next page, the system shall go the next page of songs.

Order songs by

- The user shall be able to view browse results in a certain order by selecting from a drop down menu.
- The website shall order the songs by the selected criteria and shall show the list of songs in that order.

Display song information

- The user shall be able to click on a song from the browse results or search results.

- In the event that a user clicks on a song from the browse results or search results a new page will display showing all relevant information from the database about that song along with additional song specific functionality. The page will show users information queried from the database such as song title, song artist, and year. The page will also have additional functionality such as add to playlist and play song.

Add to playlist

- A registered user shall be able to add a song to their playlist by clicking on a button.
- The system shall add the song to a playlist associated with that user that can browsed or searched through. Additional playlist only functionality will also be allowed. Playlist will be saved to the user's account that can be viewed later.

Share playlist

- Registered user shall be able to share a playlist with another registered user by clicking on a button and filling out target user.
- System shall provide target user with a link to the user's playlist that they can browse, search through, or clone.

Cloning a playlist

- A playlist can be cloned, or duplicated. A user can do this to a playlist that they own, or to a playlist shared with them. Cloning will result in a second playlist that contains all the data as the cloned playlist at the time of the clone. The user cloning the playlist will always be the owner of the clone.

Creating a playlist

- A user shall be able to create a playlist. A playlist will contain a list of songs, where duplicates are allowed.

Add/Delete/Reorder playlist

- A user shall be able to add songs, remove songs, and reorder songs in a playlist that they own.

Song Likes

- A user shall be able to "Like" a song. This will allow users to be able to quickly access all liked their favorite songs and this will influence the Popular Songs feature.
- The system shall provide a button a user can click to like a song and the data model will be updated to reflect this like.

Popular Songs

- Song popularity will be calculated based on the number of times the song has been played, the number of users that like the song, and the number of playlists that contain the song.

Creating a user

- Different user types will be supported. Namely, an admin account can be created, or a normal end-user account can be created. An admin account will have the privilege of using admin-only features, such as creating and removing songs.

- The system shall provide a form from which a user can use to register a user account. Total completion of necessary forms will be required for account creation.

Non-functional Issues

- Our Graphical User interface(GUI) will be designed using principles of Human-Computer Interaction(HCI) to create an intuitive user-friendly environment. An example of this is using Hick's law to minimize stimuli at the homepage of our website. This should attract users to the important portions of our music application such as "most popular" songs and a browse function to discover songs in the database. The search bar will be placed in top center of homepage to invite users to search the database for songs. This law minimizes users time reacting to the features on the application pushing them to popular functions. Once the user is logged in, they will have access to playlists and will be able to create their own. Using Gestalt's principle of proximity, premade playlists and user made playlists will be displayed on the left-side section. The playlists will be mapped using a vertical alignment starting with any user created playlists at the top to placing a focus on those playlists before any pre made ones. Important buttons that users will frequent such as the play button for songs along with the previous and skip button will be placed in bottom center of the application. These buttons will be larger in size relative to the other buttons in the application, this size and edge placement utilizes Fitt's law which would increase user speed and accuracy as they interact with the application. Options that interact with listening to songs will be grouped with the play button on the bottom center such as volume control. Once a feature such as a playlist is pressed, its contents will be shown on a main display in the center of the application. The results from the search bar will also be laid out on the main display for users to browse through.
- The security of each user account will be protected by a username and password created by the user. The username for each account will be the user's email. The user account information will be securely stored in the server. In order to login to the system the user must provide an existing email account along with the associated password. The website will be implemented using https providing a encrypted connection between user and server.
- Access Control a user will to only be able to access playlists and items that they own, or that are shared with them from their friends. For example, a user will not be able to modify a playlist that is shared with them. Different user accounts will have different level of permissions. Admins accounts will have total read and write access. Consumer users will only have select read and write access through exposed functions.