CS2300 Database Project Phase 2

Group Name: Member 1, Member 2, Doug McGeehan

Problem Statement

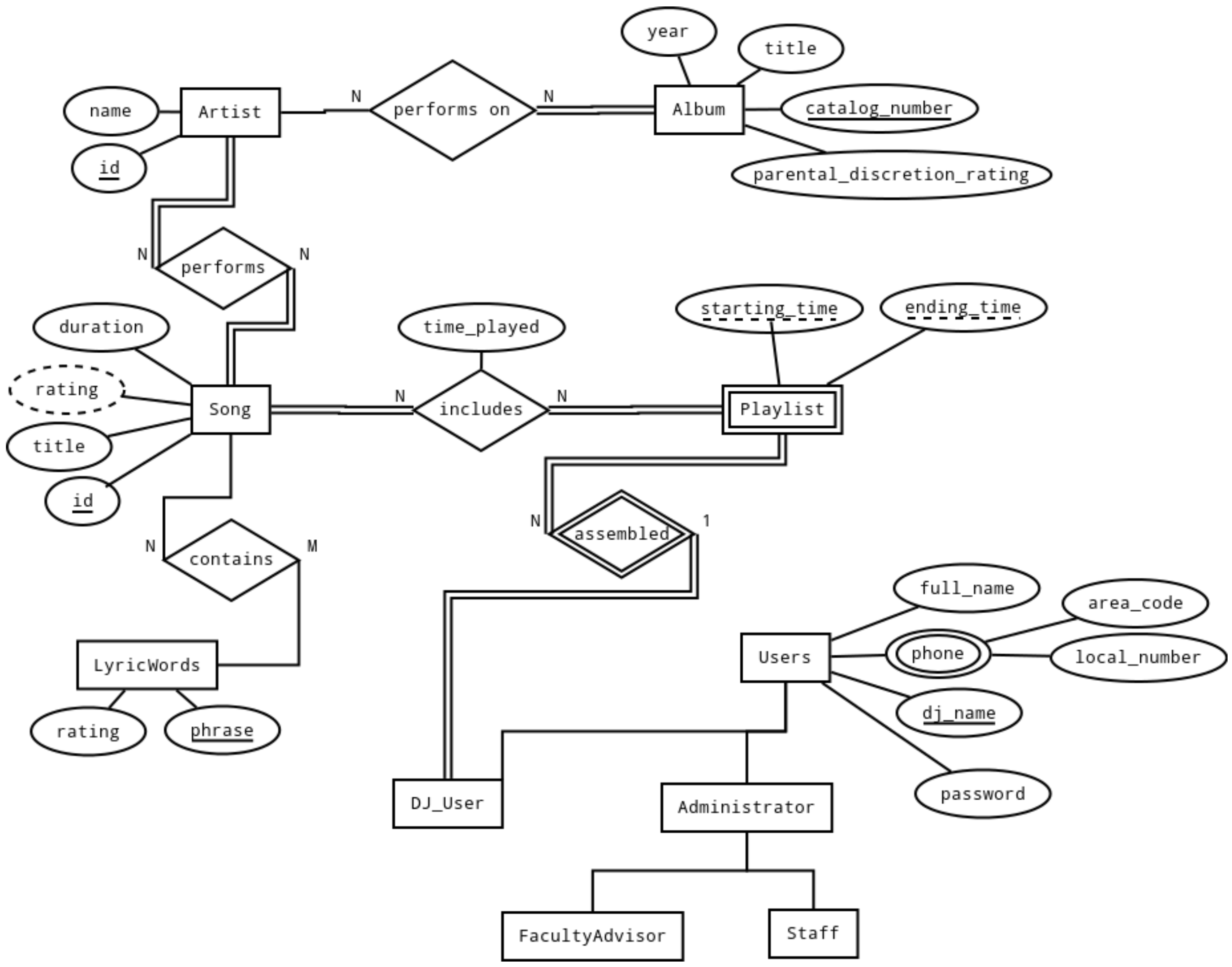
By law, a radio station cannot broadcast offensive material. The Federal Communications Commission has been granted the authority by Congress “to issue civil monetary penalties, revoke a license, and deny a renewal application” to any publicly broadcasting entity that violates these laws. When it comes to playing music on-air, a DJ at a radio station must know the content of a song’s lyrics before allowing that song to be broadcast on the radio. This can be a difficult task for some DJs who have become accustomed to the derogatory content of such songs, or for DJs who enjoy adventuring into new music.

Conceptual Database Design

Our database will consist of two portions: data that represents inappropriate material, and data representing the music and the individuals in control of the radio station. Although these two groups are disjoint in the EER diagram below, the inappropriate material will be connected to the rest of the database at the application level.

The ‘Album’ entity is a music album, with attributes year, title, catalog number, and parental discretion rating. The ‘Artist’ entity is a musician, with attributes name and ID. The ‘Song’ entity is a song, with attributes duration, rating, title, and ID. The ‘Playlist’ weak entity is a list of songs to be played, with attributes starting time and ending time, with an identifying relationship with a DJ (which is defined below).

The ‘User’ entity is a superclass entity that describes a given user of our system, with attributes full name, phone (a complex, multivalued attribute with sub-attributes area code and local number), a DJ name (which acts the the primary key of the User entity set), and password. The ‘DJ_User’ entity is a subclass of User, and is an active DJ who must have at least one Playlist. The ‘Administrator’ entity is a subclass of User. Both the Administrator subclass can overlap with the DJ_User subclass. The ‘FacultyAdvisor’ entity is a subclass of Administrator, as is the ‘Staff’ entity. These two subclasses are disjoint from one another.



The ‘LyricWords’ entity is a pool of the English words and phrases with a specific rating. Only users denoted as administrators have the ability to change the contents of this table. A ‘Song’ can contain many ‘LyricWords’, and a given ‘LyricWord’ can be contained by many songs.

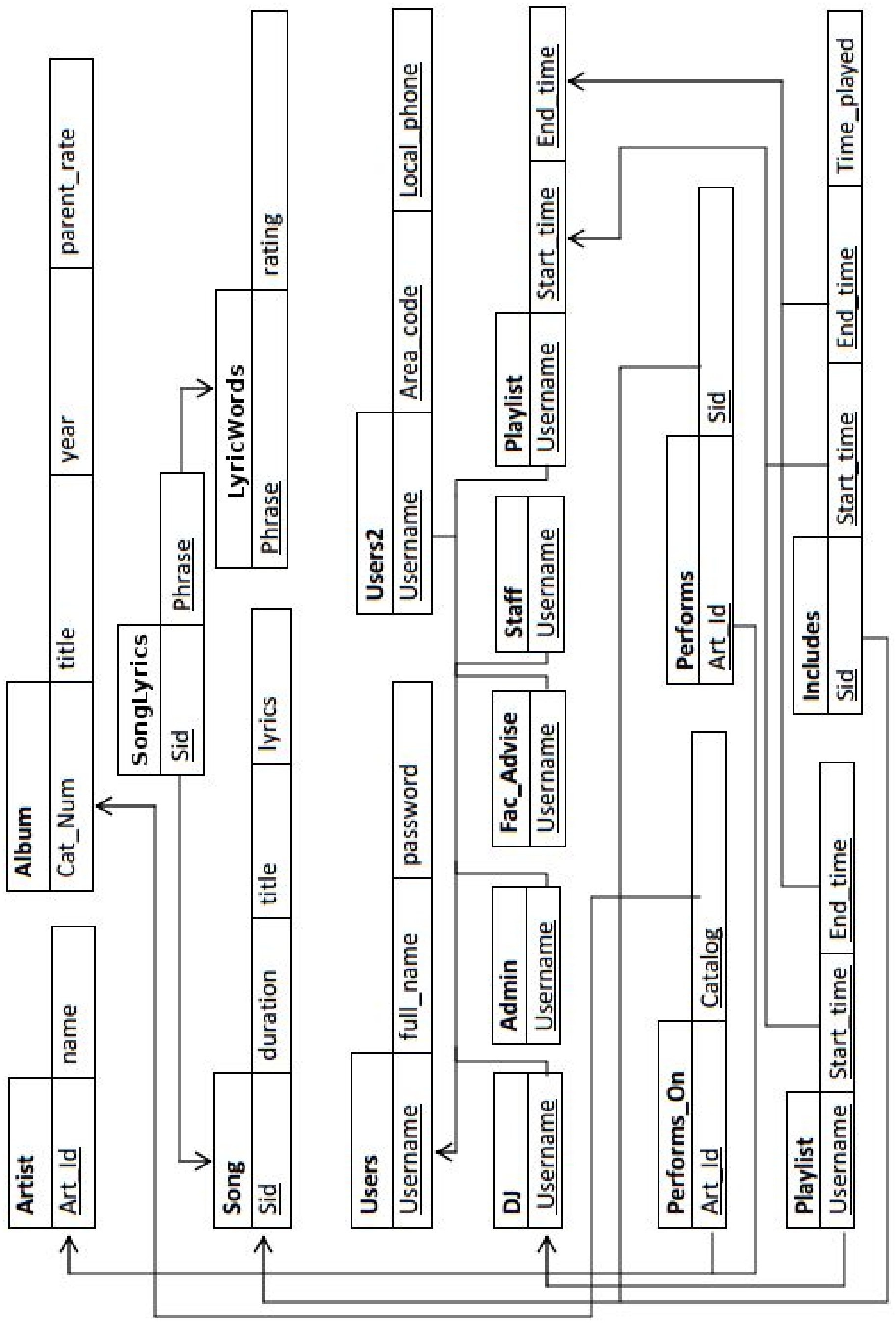
An Artist performs on an Album. Many artists can perform on an album and many albums can be performed by an artist. All albums must have a performing artist though.

An Artist performs a Song. Many artists can perform one song and many songs can be performed by one artist. All songs must have a performing artist and all artists must perform a song.

A Song is included on a Playlist. Many songs can be on a playlist and many playlists can include the same song. All songs must be included on at least one playlist and all playlists must include one or more songs. The playlist can also be described by how much time is needed to play each of its songs.

A DJ assembles a Playlist. A DJ can create many playlists, but a given playlist must be owned by only one DJ.

Logical Database Design



Summary Table of Data Types

Table	Attribute	Type	Constraint
Artist	Art_ID	CHAR(80)	Primary Key
Artist	name	CHAR(80)	NOT NULL
Album	Cat_Num	CHAR(80)	Primary Key
Album	title	CHAR(80)	
Album	year	INTEGER	NOT NULL
Album	parent_rate	BOOLEAN	
Song	SID	CHAR(80)	Primary Key
Song	duration	INTEGER	
Song	title	CHAR(80)	NOT NULL
Song	lyrics	CHAR(80)	
LyricWords	phrase	CHAR(80)	Primary Key
LyricWords	rating	INTEGER[1]	NOT NULL
Performs_On	Art_ID	CHAR(80)	Foreign Key
Performs_On	Catalog	CHAR(80)	Foreign Key
Performs	Art_ID	CHAR(80)	Foreign Key
Performs	SID	CHAR(80)	Foreign Key
SongLyrics	SID	CHAR(80)	Foreign Key
SongLyrics	Phrase	CHAR(80)	Foreign Key

[1] There are four possible ratings to choose from: clean (represented as 0), profane (represented as 1), indecent (represented as 2), and obscene (represented as 3). By using integers, we can easily use an aggregate function to determine what the highest level of offensive a song is based on the maximum rating over every LyricWord spoken/sung in the song.

Summary Table of Data Types (cont.)

Table	Attribute	Type	Constraint
Users	Username	CHAR(80)	Primary Key
Users	full_name	CHAR(80)	
Users	password	CHAR(80)	NOT NULL
PhoneNumbers	Username	CHAR(80)	Foreign Key
PhoneNumbers	Area_Code	INTEGER	
PhoneNumbers	Local_Phone	INTEGER	
DJ	Username	CHAR(80)	Foreign Key
Admin	Username	CHAR(80)	Foreign Key
Fac_Advise	Username	CHAR(80)	Foreign Key
Staff	Username	CHAR(80)	Foreign Key
Playlist	Username	CHAR(80)	Foreign Key
Playlist	Start_Time	INTEGER[2]	Unique
Playlist	End_Time	INTEGER	Unique
Song_On_Playlist	SID	CHAR(80)	Foreign Key
Song_On_Playlist	Start_Time	INTEGER	Foreign Key
Song_On_Playlist	End_Time	INTEGER	Foreign Key
Song_On_Playlist	Time_Played	INTEGER	>= Start_Time and <= End_Time

[2] Times will be stored as an integer, representing the number of seconds since January 1st, 1970, also known as UNIX time.

Application Program Design

```
Login()
      name = prompt for username
      password = prompt for password
      if(username= query administrator table for match)
            Admin = true
            logged_in = true
      else if (username =query dj table for match)
            logged_in = true
      else
            display error message

Add_playlist()
      //read in file, parse for sid, songs, etc.
      if(query song joined with Lyric_Words to get rating == acceptable)
      OR
      if (query song joined with album to get parental advisory == false)
      //parental advisory is a boolean
      OR
      if (query playlist start time > some time
            AND query playlist end time < some time)
      //Add song to playlist

Add_User()
      if (Admin == true)
            specify what type of user (Admin or DJ)
            Execute query to add user attributes to table

Delete_User()
      if(Admin == true)
            specify user
            Execute delete query starting at topmost Users table and
            cascading downwards

Override_exclusion
      if(Admin == true)
            Execute query to modify the rating of a given song.


Modify_User()
      if(Admin == true)
            //modify various user attributes.
```

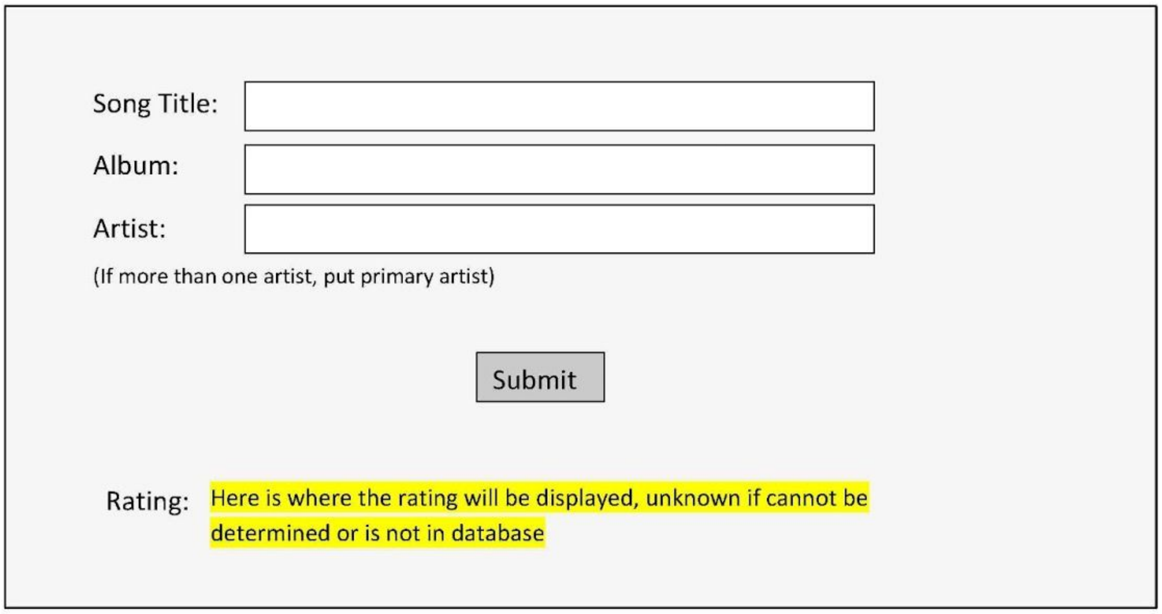
Aggregation functions:

Get Max Rating of Song's Phrases. By iterating over all of the words and phrases that are spoken/sung in a given song, the system can get an idea as to what the rating of the song is by finding the maximum rating value, where a rating of 0 is deemed a clean word and a rating of 3 is deemed obscene.

Get Average rating. This would be useful as a kind of benchmark as to what the average rating of a song that is being submitted and would in general tell a user in a general sense the type of content being submitted.

Count Songs, Max duration and Min duration all perform aggregation operations on the Songs relation.

User Interface Design

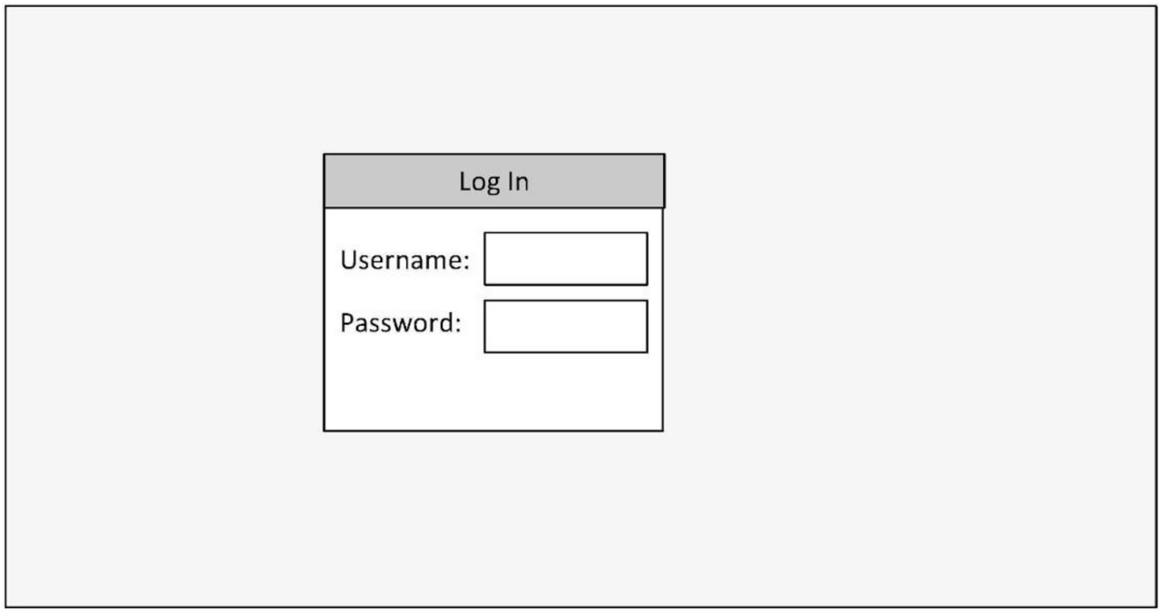

Song Title:

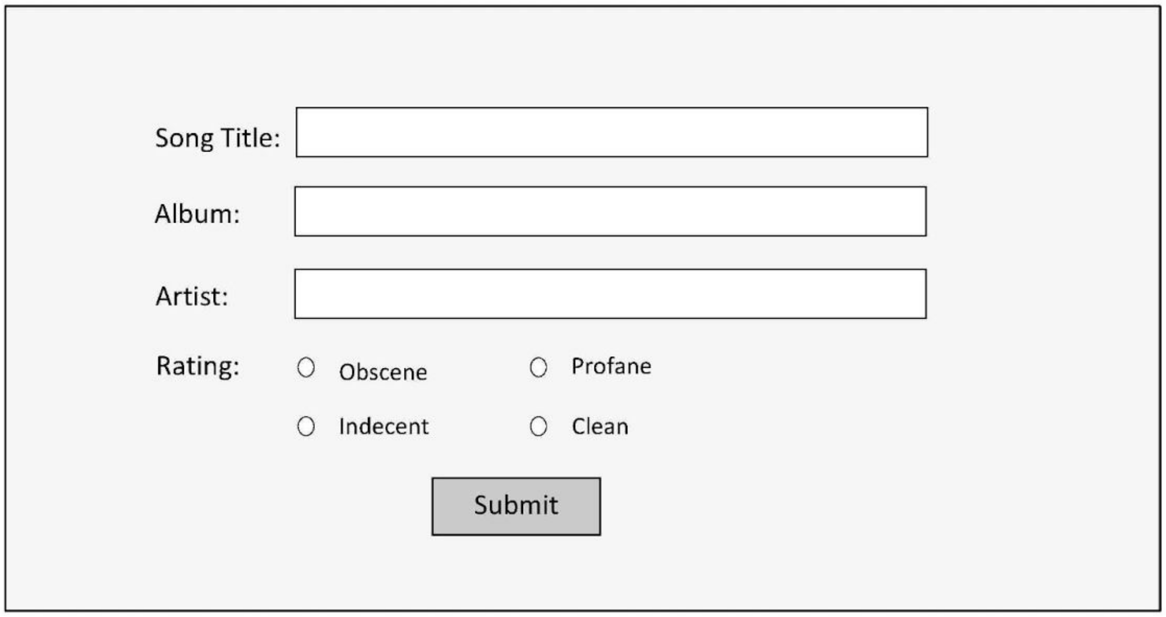
Album:

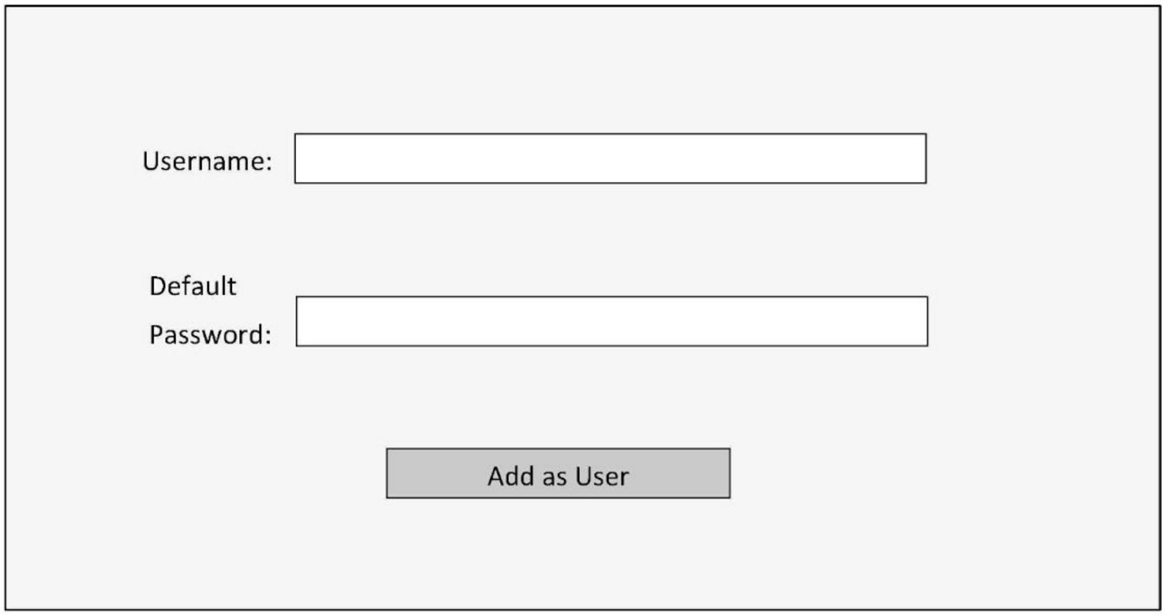
Artist:

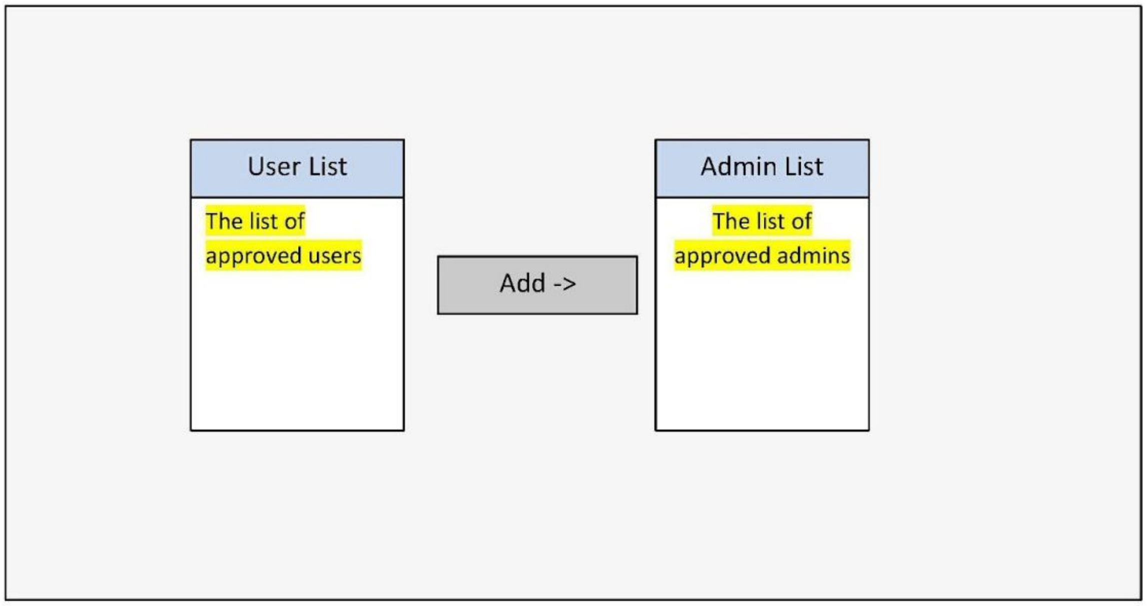
(If more than one artist, put primary artist)

Rating: Here is where the rating will be displayed, unknown if cannot be determined or is not in database




Song Title:
Album:
Artist:
Rating:
Obscene
Profane
Indecent
Clean
Submit


Username:
Default Password:
Add as User


User List
The list of approved users
Add ->
Admin List
The list of approved admins