CPSC 471 - Spoofy

Alex Stevenson - 30073617
Eric Gantz - 30031518
Ryan Fowler - 30061742

Link to GitHub Repository: https://github.com/alexs2112/CPSC471_Spoofy
This repository contains:

- Source code for our project
- Setup instructions to link the project to an Apache web server and MySQL database
- An initializer written in python to set up the MySQL database with default data
- Copies of our diagrams

Abstract:

Our project is Spoofy, a website which allows users to view songs or advertisements, as well as organize those songs into playlists. It is inspired by other well-known music services like Spotify or SoundCloud. The back end of the website is a database management system which stores information about user accounts, songs, albums, artists, and advertisements. Users are either on the free track, in which case they only have access to advertisements, or they are on the premium track, in which case they only have access to songs, artists, and albums, as well as the ability to organize those songs into playlists or a queue to customize their music-listening experience. Users are able to upgrade to premium to be able to listen to songs, or downgrade to free to view the advertisements. Administrators of the system are given tools to manage the music in the system, including adding and removing songs, adding, editing, and removing albums, adding, editing, and removing artists, and being able to specify which artist wrote which songs and albums, as well as which songs are contained by an album. A lot of this is done on the behalf of Distributors, who exist in our system as a way for artists to be represented and communicate with the system, but these Distributors only act by communicating to Administrators and thus don't have a separate type of account. Administrators are able to add and remove advertisements for the free users. Administrators are also able to delete user profiles, as well as make any other user an administrator or revoke administrator privileges from an administrator.

Introduction:

The "information superhighway" enabled by the internet has led to the mass sharing of music between people around the world. This can be a complicated problem to solve, as it involves keeping track of the metadata for songs, artists, and albums. Additionally, it requires being able to break something as human and intuitive as music into quantifiable information that can be stored online. This problem is a common one, and there have been many websites and companies that have offered a solution in one way or another. Past services include Napster and iTunes, and some present-day streaming services involve Spotify, Tidal, and Apple Music. Spotify and Apple Music are the most similar to our proposed solution, which allow the streaming of music directly from the site along with the ability to download songs directly. These services allow users to create and listen to playlists out of the music on the site, or listen to playlists other users have created. One main improvement to this type of service could be to have music stored as separate tracks known as stems, which include things such as vocals, guitar, bass, and more. This would be helpful for musicians wanting to learn to play a song on a certain instrument, or for people who want to personalize their listening experience.

System Description:

The system we have designed to solve this problem consists of a webpage linked to a database that stores info about the various bits of metadata of individual songs, albums, and artists. Additionally, we keep track of our user-base, noting subscriptions, account details, and user statistics for each person using our service. We also keep track of the Artists that have submitted music, to be paid a fixed rate for each individual play of any song they have written at the end of each month. This database of music and users is linked to a webpage hosted by an Apache server, programmed primarily in PHP. This service allows users to create accounts, organize playlists and listen to their music, and view additional information about the different songs, albums, and artists present in our database, and also allows administrators to curate the database by adding, modifying, or deleting songs, albums, artist profiles, and user accounts.

Project Design:

Our system has 4 types of users: unregistered users, free users, premium users who pay a monthly subscription, and admins.

Unregistered users can see the list of songs, albums, and artists, and can search through them using the search functionality, they can also view any of these individually. Unregistered users can login if they have an account, or can register an account.

Free users can see the list of advertisements, can search through advertisements, can add advertisements to their queue, and can play advertisements. Free users can also view their profile where they can upgrade to a premium account by subscribing, and they can log out. Free users cannot see or play songs, albums, or artists.

Premium users can see the list of songs, albums, and artists and can search through them, and they can view songs, albums, or artists individually. Premium users can add songs and albums to their queue, they can create playlists and add songs to them, and they can play songs that are in their queue. Premium users can also view their profile, view their created playlists, cancel their premium membership to become a free user, and they can logout.

Admins are ‘normal’ users that have admin privileges, and are not an account type.

Admins can manage users, this means they can view a list of all users, view an individual user’s profile, delete a user’s account, and grant a user account admin privileges.

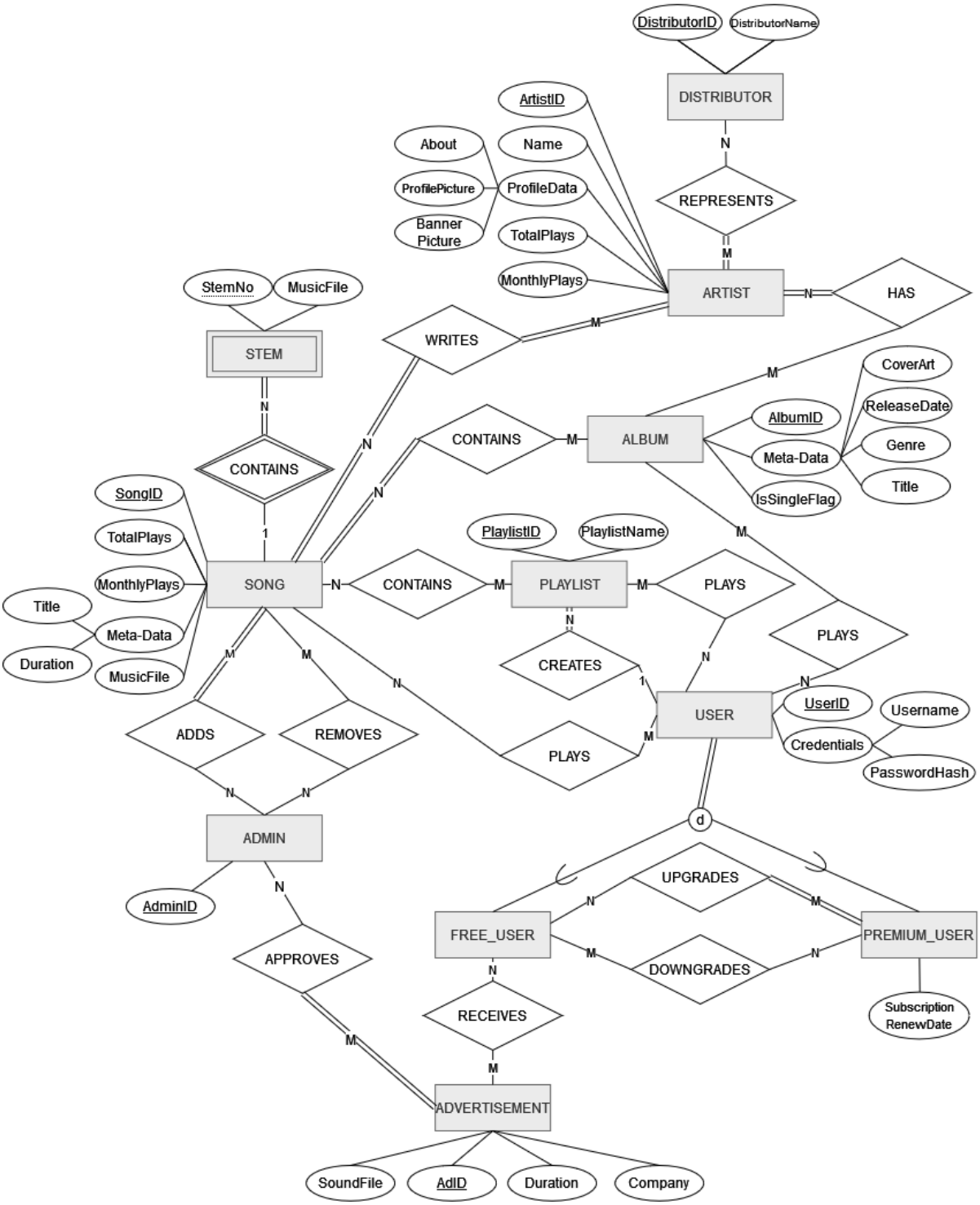
Admins can add or delete songs, reset the monthly plays of a song, add songs to albums, and add artist credits to songs.

Admins can add, edit, or delete albums, remove songs from albums, and can add artist credits to albums.

Admins can add, edit, or delete artists, and can remove albums and songs from artists, removing their credits.

Admins can view the list of advertisements, and can add and delete advertisements.

Extended Entity Relationship Diagram:



Assumptions:

- All Songs should be assigned an Album, all Albums should be assigned an Artist
- Singles will be handled as an Album with a single song

Changes:

- Removal of NumSongs and TotalLength from ALBUM
- TotalPlays and MonthlyPlays no longer derived attributes in ARTIST

Implementation:

Relational Model:

There was a slight challenge with creating these diagrams due to the ability for songs to have multiple albums and artists, and for albums to also have multiple artists. This led to the creation of the ALBUM_CONTAINS, HAS, and WRITES tables.

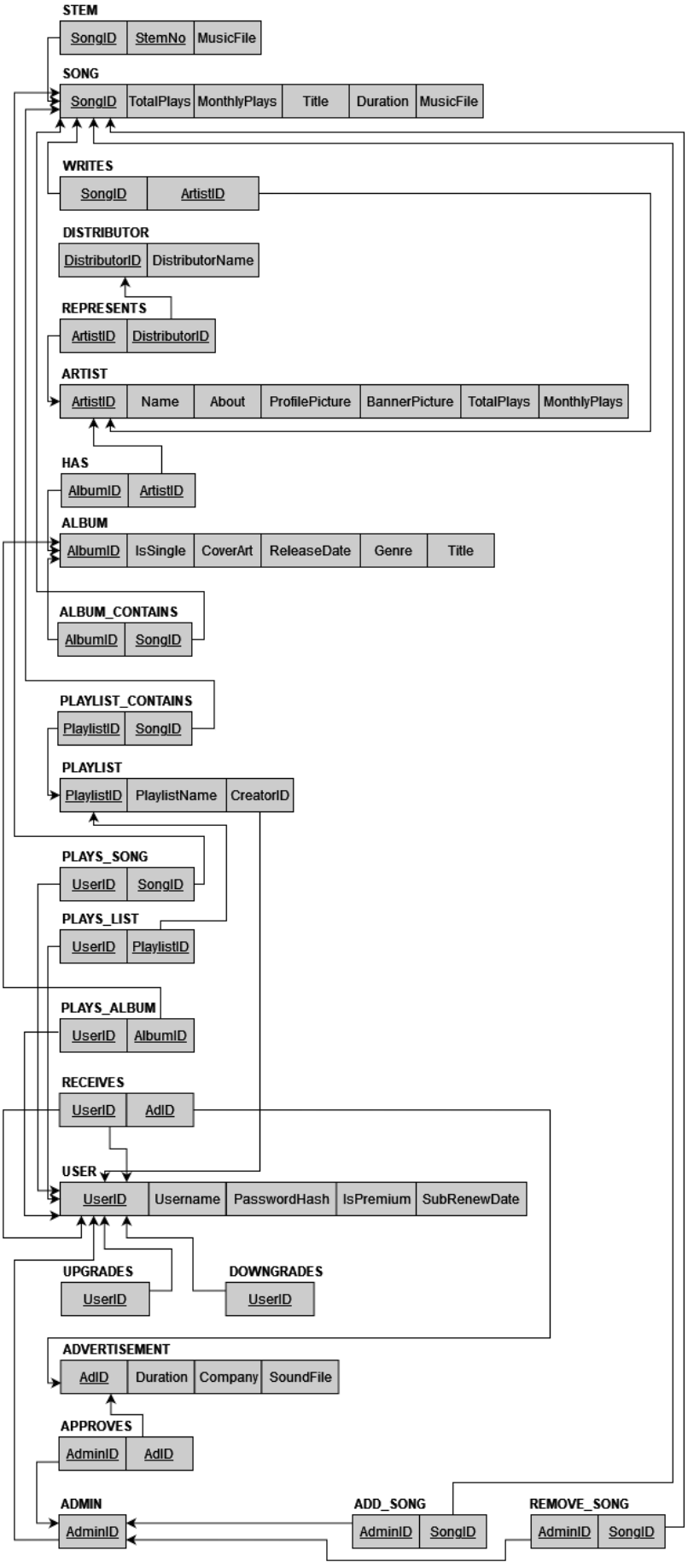
This led to unforeseen consequences for implementing the database in PHP, as these intermediary tables added an extra step for SQL queries that required pairing songs, albums, and artists.

Assumptions:

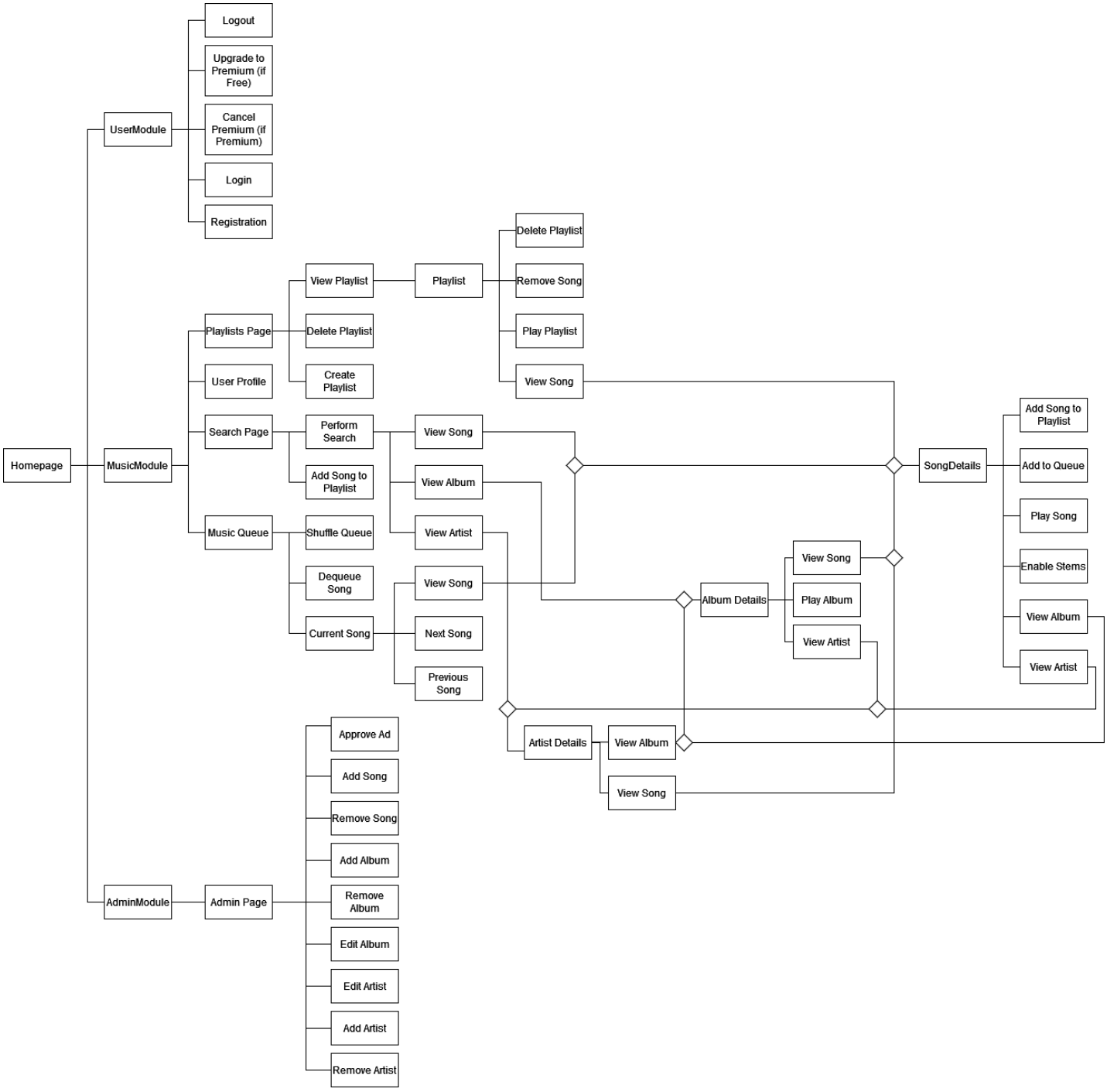
- The RECEIVES relation between USER and ADVERTISEMENT only happens if USER.IsPremium is FALSE.
- The UPGRADES and DOWNGRADES relations for USER will only happen if USER.IsPremium is TRUE or FALSE, respectively.

Changes:

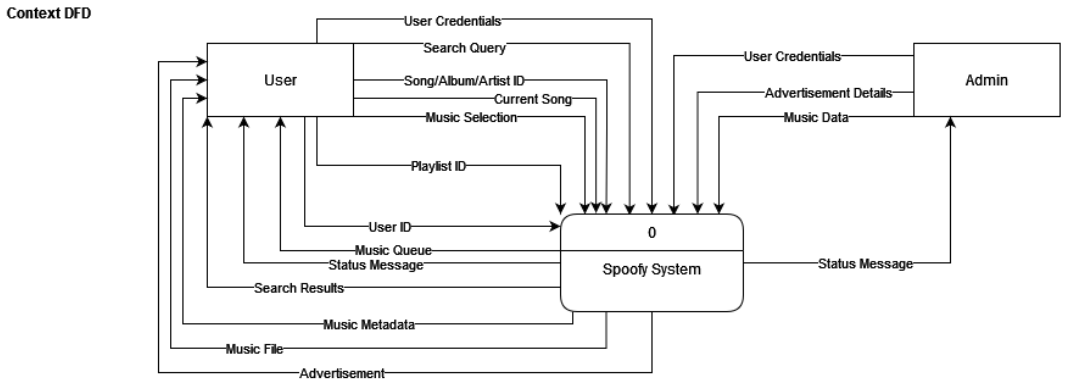
- Same changes as with the EERD:
- Removal of NumSongs and TotalLength from ALBUM
- TotalPlays and MonthlyPlays no longer derived attributes in ARTIST

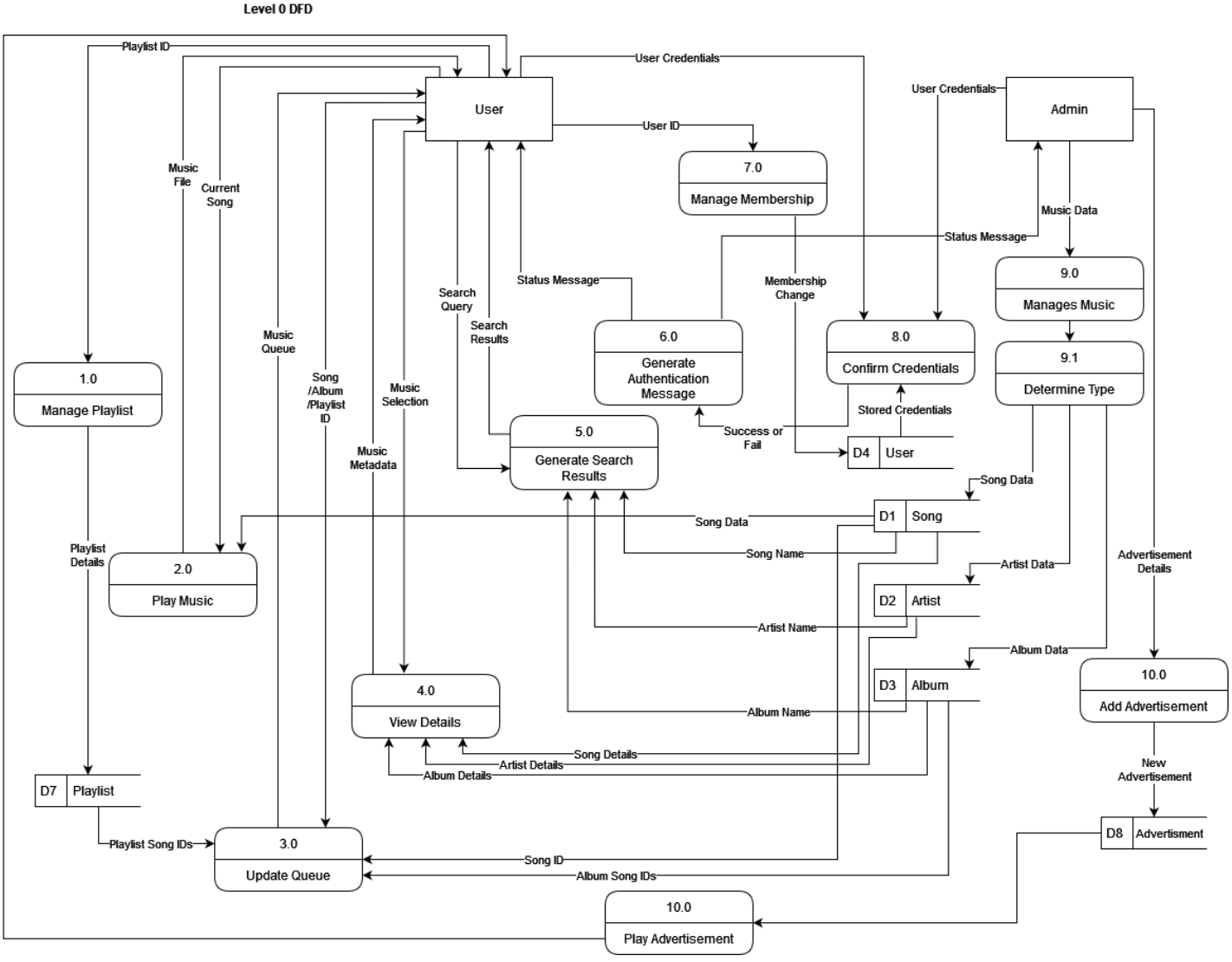


HIPO Diagram:



Data Flow Diagram:





Assumptions for both the HIPO and Data Flow diagrams:

- The UserModule, MusicModule, and AdminModule blocks are non-functional, used to group the processes in the system.
- Playing a song clears the current queue and sets it as the first queue element.
- Playing an album or playlist clears the current queue and sets the queue as that album or playlist.
- The music queue is a list of songs locally available to the user, it should not be in the database and should cease to exist when the webpage is closed.
- Functions that return lists of things are denoted with square brackets. (Example: [SongID] is a list of SongIDs).
- When editing certain database tuples, such as Edit Artist, some inputs can be null. Those null values are simply ignored and do not change the corresponding attribute.

DBMS:

The database management system we selected was MySQL 8.0.

We selected it mostly due to how widespread MySQL is as a DBMS, as well as because a large portion of the course was dedicated to teaching the SQL programming language which MySQL implements to structure its queries. 8.0 is simply the most recent full release of the DMBS.

We were also careful to prevent SQL injection. This was achieved by using a PHP library called MySQLi, which handles creating and handing off the SQL queries to the DBMS. In this library, there are many ways to call an SQL query, but we chose to use a method which involved “preparing” the user-entered variables before they were inserted into the query, and this preparation step parses input strings and escapes out of special SQL characters and commands, preventing typical SQL injection attacks which involve inputting SQL code into text fields.

SQL Statements

CREATE TABLES

Creating the database tables, this can be done automatically in the python initialize.py script in the git repository.

```
CREATE TABLE SONG
    (SongID INT NOT NULL AUTO_INCREMENT,
    TotalPlays INT DEFAULT 0,
    MonthlyPlays INT DEFAULT 0,
    Title VARCHAR(32),
    Duration VARCHAR(32),
    MusicFile VARCHAR(255),
    PRIMARY KEY (SongID));

CREATE TABLE ARTIST
    (ArtistID INT NOT NULL AUTO_INCREMENT,
    Name VARCHAR(32),
    About VARCHAR(1500),
    ProfilePicture VARCHAR(255),
    BannerPicture VARCHAR(255),
    TotalPlays INT DEFAULT 0,
    MonthlyPlays INT DEFAULT 0,
    PRIMARY KEY (ArtistID));

CREATE TABLE ALBUM
    (AlbumID INT NOT NULL AUTO_INCREMENT,
    IsSingle BOOLEAN NOT NULL,
    CoverArt VARCHAR(255),
    ReleaseDate DATE,
    Genre VARCHAR(255),
    Title VARCHAR(32),
    PRIMARY KEY (AlbumID));

CREATE TABLE USER
    (UserID INT NOT NULL AUTO_INCREMENT,
    Username VARCHAR(32) NOT NULL,
    PasswordHash CHAR(255) NOT NULL,
    IsPremium BOOLEAN NOT NULL,
    SubRenewDate DATE,
    PRIMARY KEY (UserID),
    UNIQUE (Username));

CREATE TABLE ADVERTISEMENT
    (AdID INT NOT NULL AUTO_INCREMENT,
    Duration VARCHAR(32),
    Company VARCHAR(32),
    SoundFile VARCHAR(255),
    PRIMARY KEY (AdID));

CREATE TABLE ADMIN
    (AdminID INT NOT NULL,
    PRIMARY KEY (AdminID),
    FOREIGN KEY (AdminID) REFERENCES USER(UserID) ON DELETE CASCADE ON
    UPDATE CASCADE);
```

```
CREATE TABLE PLAYLIST
    (PlaylistID INT NOT NULL AUTO_INCREMENT,
    PlaylistName VARCHAR(32),
    CreatorID INT,
    PRIMARY KEY (PlaylistID),
    FOREIGN KEY (CreatorID) REFERENCES USER(UserID) ON DELETE SET NULL
    ON UPDATE CASCADE);

CREATE TABLE PLAYLIST_CONTAINS
    (PlaylistID INT NOT NULL,
    SongID INT NOT NULL,
    PRIMARY KEY (PlaylistID, SongID),
    FOREIGN KEY (PlaylistID) REFERENCES PLAYLIST(PlaylistID) ON DELETE
    CASCADE ON UPDATE CASCADE,
    FOREIGN KEY (SongID) REFERENCES SONG(SongID)ON DELETE CASCADE ON
    UPDATE CASCADE);

CREATE TABLE ALBUM_CONTAINS
    (AlbumID INT NOT NULL,
    SongID INT NOT NULL,
    PRIMARY KEY (AlbumID, SongID),
    FOREIGN KEY (AlbumID) REFERENCES ALBUM(AlbumID) ON DELETE CASCADE
    ON UPDATE CASCADE,
    FOREIGN KEY (SongID) REFERENCES SONG(SongID) ON DELETE CASCADE ON
    UPDATE CASCADE);

CREATE TABLE HAS
    (AlbumID INT NOT NULL,
    ArtistID INT NOT NULL,
    PRIMARY KEY (AlbumID, ArtistID),
    FOREIGN KEY (AlbumID) REFERENCES ALBUM(AlbumID) ON DELETE CASCADE
    ON UPDATE CASCADE,
    FOREIGN KEY (ArtistID) REFERENCES ARTIST(ArtistID) ON DELETE CASCADE ON
    UPDATE CASCADE);

CREATE TABLE DISTRIBUTOR
    (DistributorID INT NOT NULL AUTO_INCREMENT,
    DistributorName VARCHAR(32),
    PRIMARY KEY (DistributorID));

CREATE TABLE REPRESENTS
    (ArtistID INT NOT NULL,
    DistributorID INT NOT NULL,
    PRIMARY KEY (ArtistID, DistributorID),
    FOREIGN KEY (ArtistID) REFERENCES ARTIST(ArtistID) ON DELETE CASCADE ON
    UPDATE CASCADE,
    FOREIGN KEY (DistributorID) REFERENCES DISTRIBUTOR(DistributorID)ON
    DELETE CASCADE ON UPDATE CASCADE);

CREATE TABLE WRITES
    (SongID INT NOT NULL,
    ArtistID INT NOT NULL,
    PRIMARY KEY (SongID, ArtistID),
    FOREIGN KEY (SongID) REFERENCES SONG(SongID) ON DELETE CASCADE ON UPDATE CASCADE,
    FOREIGN KEY (ArtistID) REFERENCES ARTIST(ArtistID) ON DELETE CASCADE ON UPDATE CASCADE);
```

```
CREATE TABLE STEM
    (SongID INT NOT NULL,
    StemNo INT NOT NULL,
    MusicFile VARCHAR(255),
    PRIMARY KEY (SongID, StemNo),
    FOREIGN KEY (SongID) REFERENCES SONG(SongID) ON DELETE CASCADE ON UPDATE CASCADE);
```

INSERT

Inserting data into the database. This happens at a few points:
- Registering a new user.
- Premium users creating playlists and adding songs to them.
- Administrator functions, such as adding new admins, advertisements, songs, albums, artists, and linking the three together.

```
INSERT INTO USER (Username, PasswordHash, IsPremium) VALUES (@Username, @PasswordHash, FALSE);

INSERT INTO PLAYLIST (PlaylistName, CreatorID) VALUES (@PlaylistID, @UserID);
INSERT INTO PLAYLIST_CONTAINS (PlaylistID, SongID) VALUES (@PlaylistID, @SongID);

INSERT INTO ADMIN (AdminID) VALUES (@UserID)
INSERT INTO ADVERTISEMENT (Duration, Company, SoundFile) VALUES (@Duration, @Company, @SoundFile);
INSERT INTO SONG (Title, Duration, MusicFile, TotalPlays, MonthlyPlays)
    VALUES (@Title, @Duration, @MusicFile, 0, 0);
INSERT INTO ALBUM (Title, IsSingle, CoverArt, ReleaseDate, Genre)
    VALUES (@Title, @IsSingle, @CoverArt, @ReleaseDate, @Genre);
INSERT INTO ARTIST (Name, About, ProfilePicture, BannerPicture, TotalPlays, MonthlyPlays)
    VALUES (@Name, @About, @ProfilePicture, @BannerPicture, 0, 0);

INSERT INTO WRITES VALUES(@SongID, @ArtistID);
INSERT INTO HAS VALUES(@AlbumID, @ArtistID);
INSERT INTO ALBUM_CONTAINS VALUES(@AlbumID, @SongID);
```

DELETE

Removing data from the database. There are several ways to do this:
- Premium users can delete their playlists, and remove songs from their playlists.
- Administrators have permissions to delete users along with anything that they can insert, listed in the INSERT subsection.

```
DELETE FROM PLAYLIST WHERE PlaylistID=@PlaylistID;
DELETE FROM PLAYLIST_CONTAINS WHERE PlaylistID=@PlaylistID AND SongID=@SongID;

DELETE FROM USER WHERE UserID=@UserID;
DELETE FROM ADMIN WHERE AdminID=@UserID;
DELETE FROM ADVERTISEMENT WHERE AdID=@AdID;
DELETE FROM SONG WHERE SongID=@SongID;
DELETE FROM ALBUM WHERE AlbumID=@AlbumID;
DELETE FROM ARTIST WHERE ArtistID=@ArtistID;
DELETE FROM WRITES WHERE SongID=@SongID AND ArtistID=@ArtistID;
DELETE FROM HAS WHERE ArtistID=@ArtistID AND AlbumID=@AlbumID;
DELETE FROM ALBUM_CONTAINS WHERE SongID=@SongID AND AlbumID=@AlbumID;
```

UPDATE

Updating data in the database. This does not happen as often, but is important to keep track of the following functionality:
- Users can update their IsPremium status by either subscribing or cancelling premium.
- Whenever a song is played it needs to increment the number of MonthlyPlays and TotalPlays associated with that song and all artists of that song.
- Administrators can update Albums and Artists.
- Administrators can also clear the MonthlyPlays associated with Songs and Artists.

```
UPDATE USER SET IsPremium=TRUE, SubRenewDate=@SubRenewDate WHERE UserID=@UserID;
UPDATE USER SET IsPremium=FALSE, SubRenewDate=NULL WHERE UserID=@UserID;

UPDATE SONG SET TotalPlays = TotalPlays + 1, MonthlyPlays = MonthlyPlays + 1 WHERE SongID=@SongID;
UPDATE ARTIST SET TotalPlays = TotalPlays + 1, MonthlyPlays = MonthlyPlays + 1
    WHERE ArtistID IN (
        SELECT ArtistID FROM WRITES WHERE SongID=@SongID
    );

UPDATE ARTIST
    SET Name=@Name, About=@About, ProfilePicture=@ProfilePicture, BannerPicture=@BannerPicture
    WHERE ArtistID=@ArtistID;
UPDATE ALBUM
    SET Title=@Title, IsSingle=@IsSingle, CoverArt=@CoverArt, ReleaseDate=@ReleaseDate, Genre=@Genre
    WHERE AlbumID=@AlbumID;

UPDATE SONG SET MonthlyPlays=0 WHERE SongID=@SongID;
UPDATE ARTIST SET MonthlyPlays=0 WHERE ArtistID=@ArtistID;
```

SELECT

This is our most prominent SQL query type. It happens on nearly every page to fetch information from the database to display to the user. These functionalities include:
- Selecting data belonging to a single ID, such as a Song or User.
- Selecting data belonging to a User by Username, as Username is a unique field in the database.
- Selecting information on a song, artist, or album and pairing it to another artist or album that may or may not be present in the database.
- Selecting all artists that wrote an album or song, and all albums that contain a song.
- Determining if data already exists in the database, by calling SELECT and ensuring the result is empty.
- Finding the cover art of an album that contains a song by SongID.

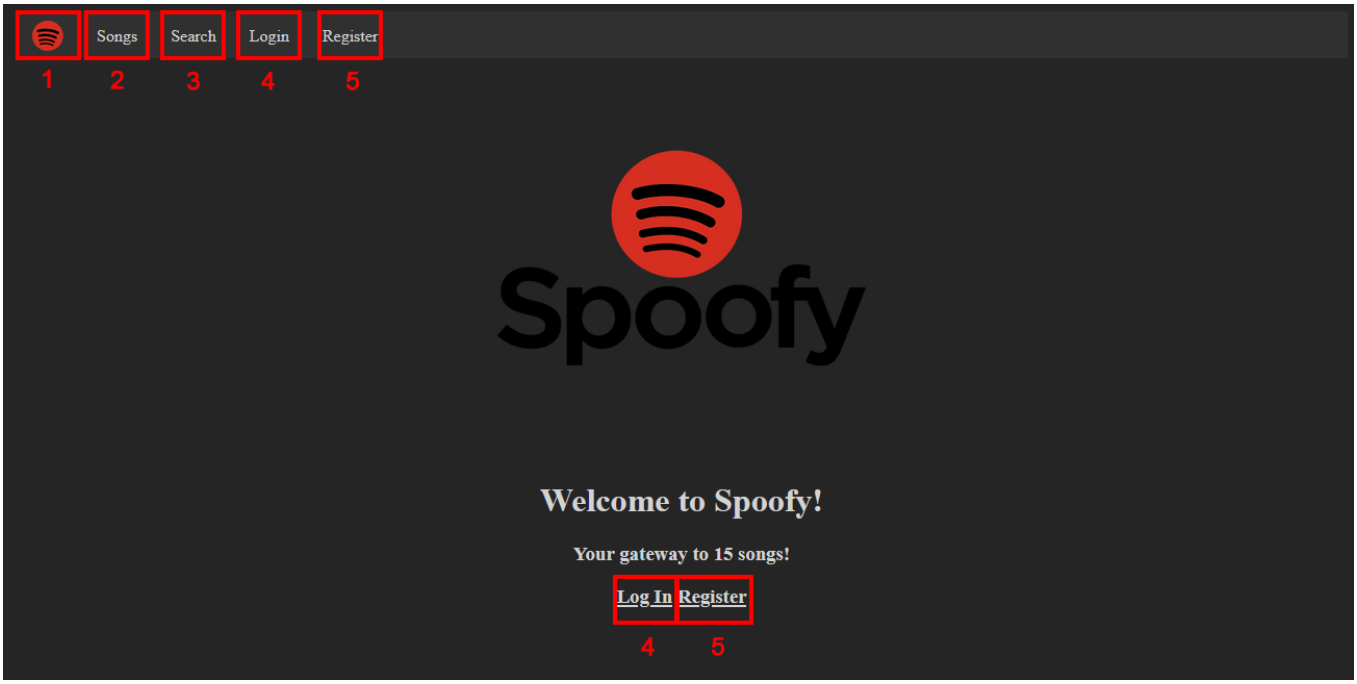
```
SELECT <* or a specific set of attributes> FROM <table> WHERE <ID>=@ID;
SELECT * FROM SONG WHERE SongID=@SongID;
SELECT CoverArt FROM ALBUM WHERE AlbumID=@AlbumID;

SELECT * FROM USER WHERE Username=@Username;

SELECT Name FROM ARTIST, WRITES WHERE ARTIST.ArtistID = WRITES.ArtistID AND WRITES.SongID=@SongID;
SELECT Name FROM ARTIST, HAS WHERE ARTIST.ArtistID = HAS.ArtistID AND HAS.AlbumID=@AlbumID;
SELECT Title FROM ALBUM, ALBUM_CONTAINS
    WHERE ALBUM.AlbumID = ALBUM_CONTAINS.AlbumID AND ALBUM_CONTAINS.SongID=@SongID;

SELECT * FROM WRITES, ARTIST WHERE SongID=@SongID AND WRITES.ArtistID=ARTIST.ArtistID;
SELECT * FROM ALBUM_CONTAINS, ALBUM WHERE SongID=@SongID AND ALBUM_CONTAINS.AlbumID=ALBUM.AlbumID;

SELECT * FROM PLAYLIST_CONTAINS WHERE PlaylistID=@PlaylistID AND SongID=@SongID;

SELECT CoverArt FROM ALBUM WHERE AlbumID IN (SELECT AlbumID FROM ALBUM_CONTAINS WHERE SongID=@SongID);
```

Spoofy: User Manual

Alex Stevenson - 30073617, Eric Gantz - 30031518, Ryan Fowler - 30061742

Unregistered User

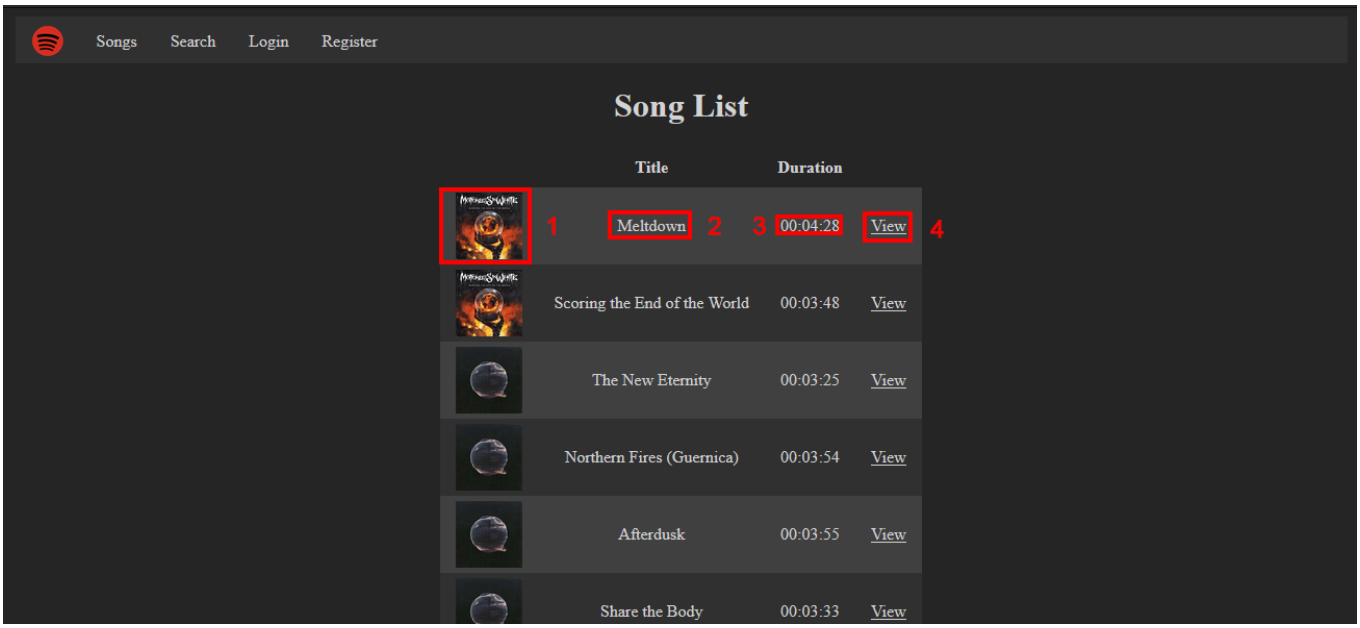
Home Page



This is the main page that a user will see when accessing Spoofy for the first time. As an unregistered user, you are prompted to log in or register while also being able to access the list of songs and search functionalities.

1. Navigate to Home.
2. Navigate to Songs.
3. Navigate to Search.
4. Navigate to Login.
5. Navigate to Register.

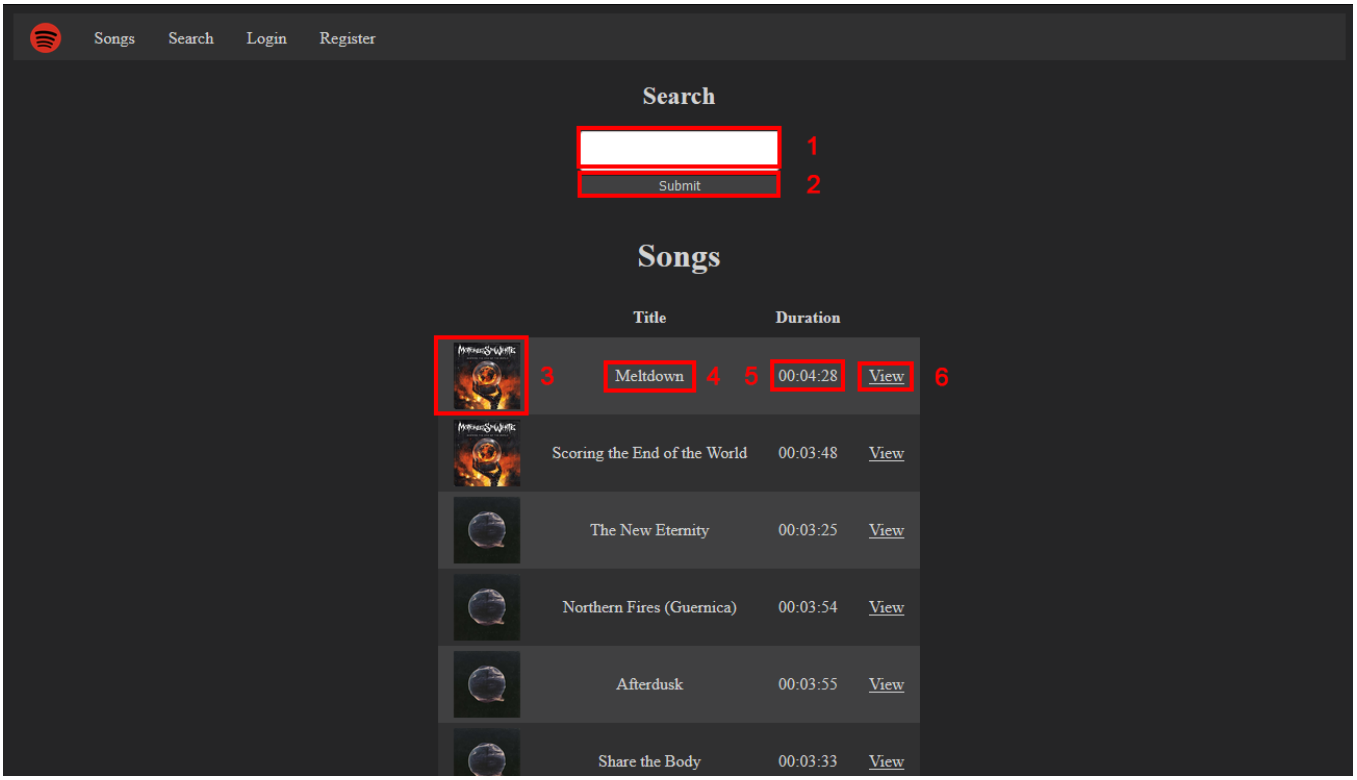
Song List



The song list that an unregistered user can see. Note the lack of "Play" and "Add to Queue" buttons, as playing music is a functionality reserved for premium users.

1. Song image.
2. Song title.
3. Song duration.
4. Click to view song details.

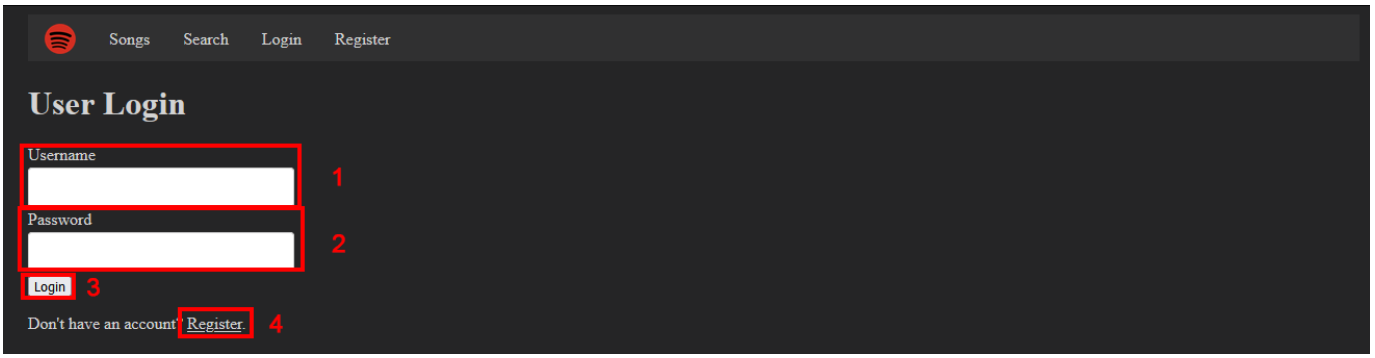
Search Page



The search page takes a query string and uses it to search the music database, returning all songs, albums, and artists that contain the query as a substring. Not pictured in the screenshot is the search results of Albums and Artists underneath

the results of Songs.
1. Click to type in a string to search for.
2. Click to submit your search. The list of songs, albums, and artists below will update to show only ones where some portion of their name matches your search query.
3. Song image.
4. Song title.
5. Song duration.
6. Click to view song details.

User Login



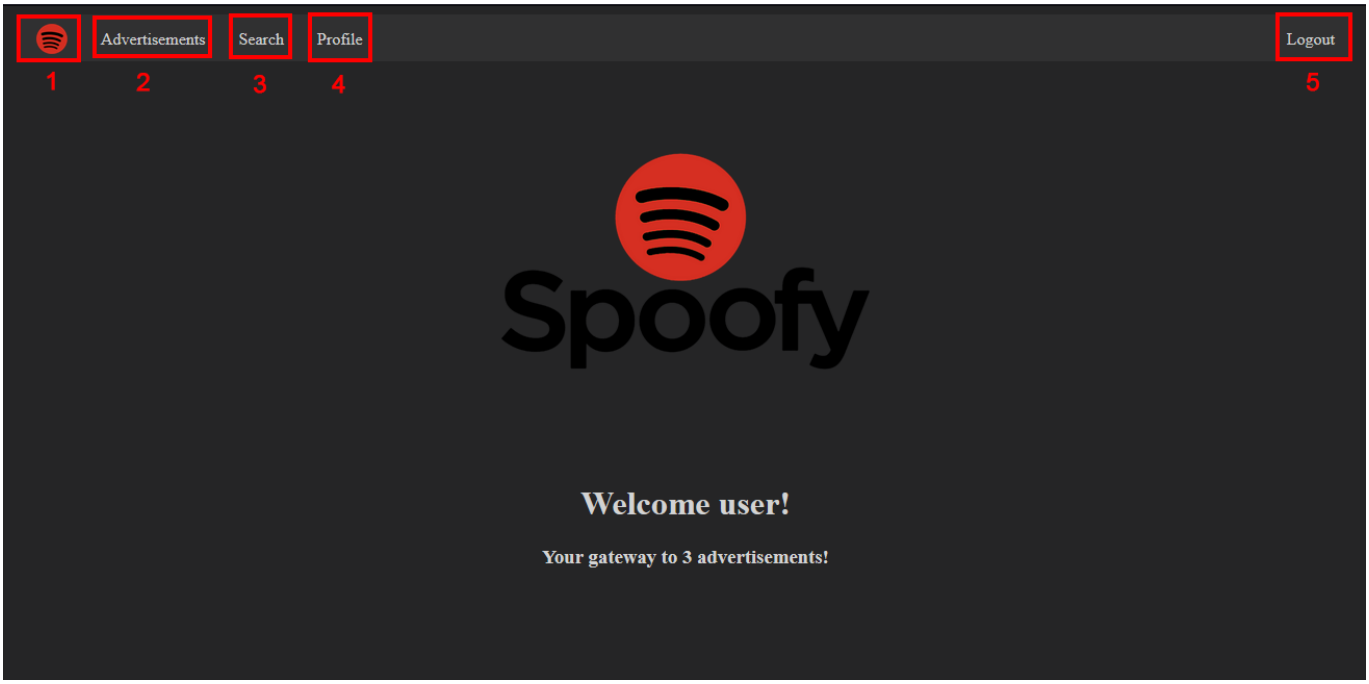
The "Login" button from the top menubar or the home screen will direct you here, prompting you to log in. Failing to log in will notify you of the error and allow you to try again.
1. Click to type in your username.
2. Click to type in your password.
3. Click to attempt to log in with the username and password you have typed.
4. Navigate to the register page if you do not have an account yet.

User Register

If you need to register you need to enter a unique username and a password. The password is securely stored in the database under the SHA256 hash algorithm. Once your account has been created you will be directed to the login page to securely log in with your new free account.
1. Click to type in a username you would like the new account to have.
2. Click to type in a password for the new account.
3. Click to type in the same password to verify that you have not made an error in the previous text box.
4. Click to submit a request to create the new account.
5. Click to reset the text fields if you wish to clear them and start over.
6. Navigate to the login page if you already have an account.

Free User

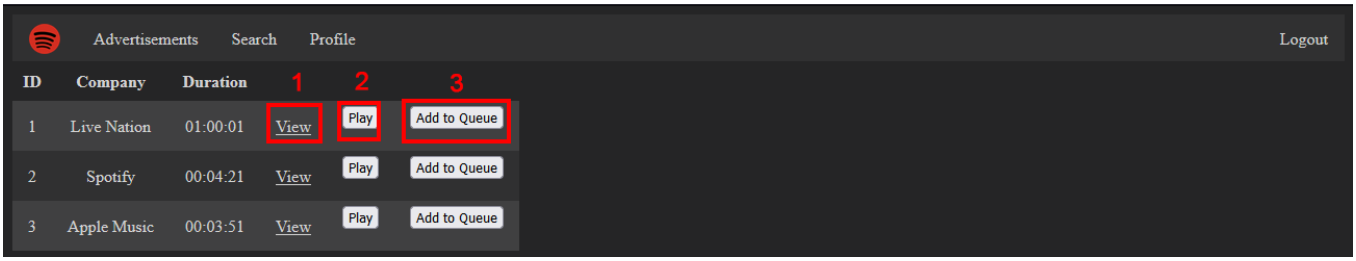
Home Page



After logging in as a free user you will see the home page. As free users can only access advertisements, the option to go to the Song List page is disabled and replaced by an Advertisements page. Note the addition of the Profile button in the menubar.

1. Website logo. Click at any point to return to this home page.
2. Navigate to Advertisements.
3. Navigate to Search.
4. Navigate to your account's Profile.
5. Click to log out of your account.

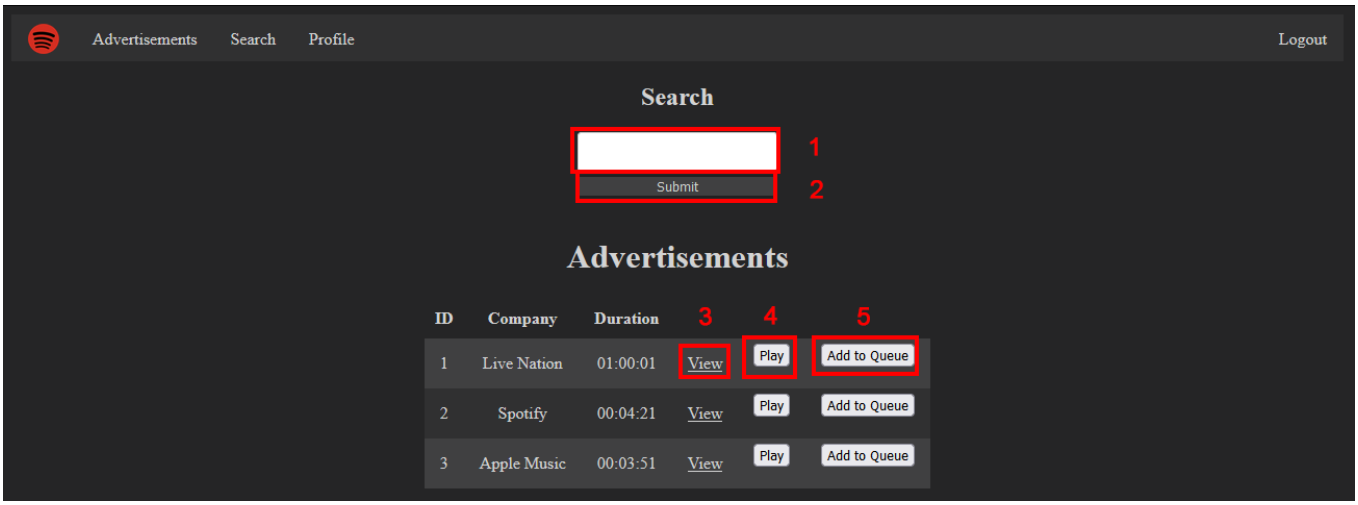
Advertisement List



Navigating to the advertisements page will show this list of ads. Ads can be viewed from here. As a registered user you can now play them, adding them to the queue.

1. Click to view the advertisement.
2. Click to play the advertisement.
3. Click to add the advertisement to your queue.

Search Page



Similar to the search page for music, this search page takes a string from the user as a query. This will return the list of all advertisements that contain the query as a substring. As with the advertisements page, ads can be viewed and played from here.

1. Click to type in a string to search for.
2. Click to submit your search. The list of songs, albums, and artists below will update to show only ones where some portion of their name matches your search query.
3. Click to view the advertisement.
4. Click to play the advertisement.
5. Click to add the advertisement to your queue.

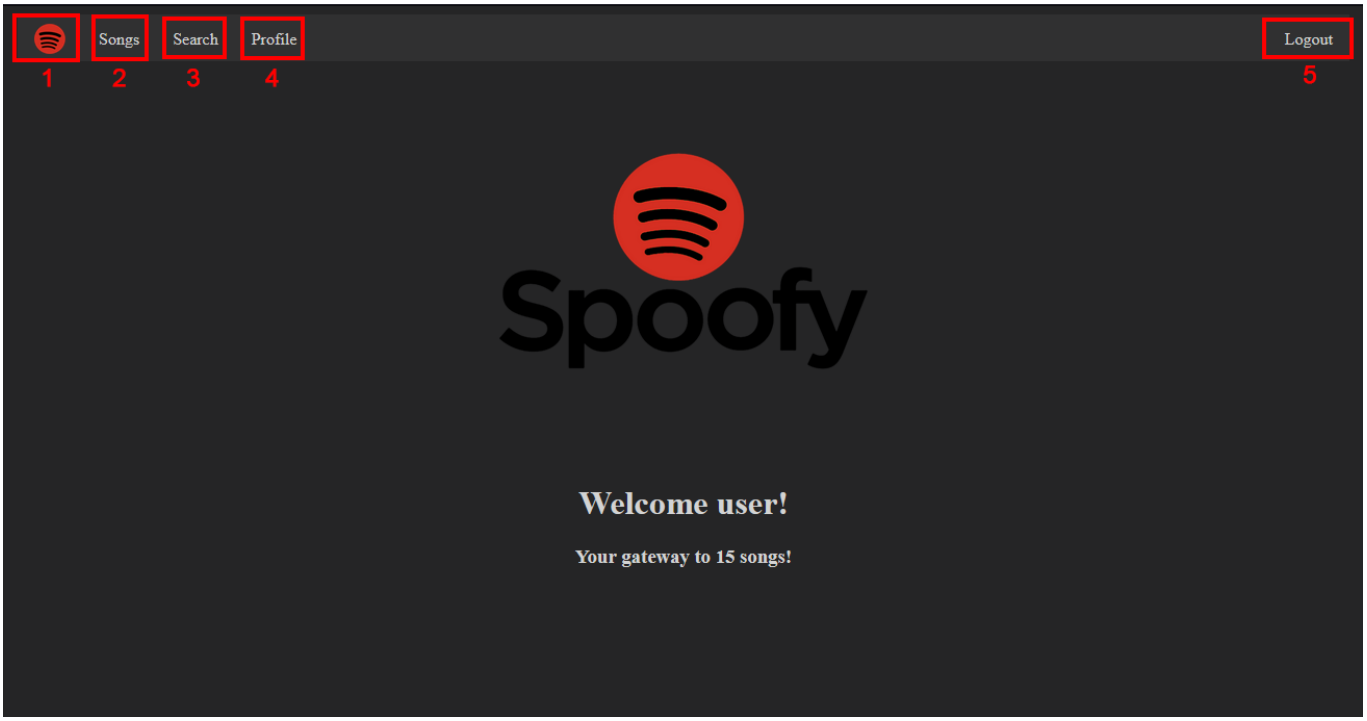
Profile Page

The profile page of a free user. Note that free accounts cannot create or use playlists, as that functionality is reserved for premium users.

1. Your username.
2. Your account type.
3. Click to log out of your account.
4. Click to upgrade your account's membership from Free, which can only view ads, to Premium, which can view songs, albums, and artists, and create playlists.

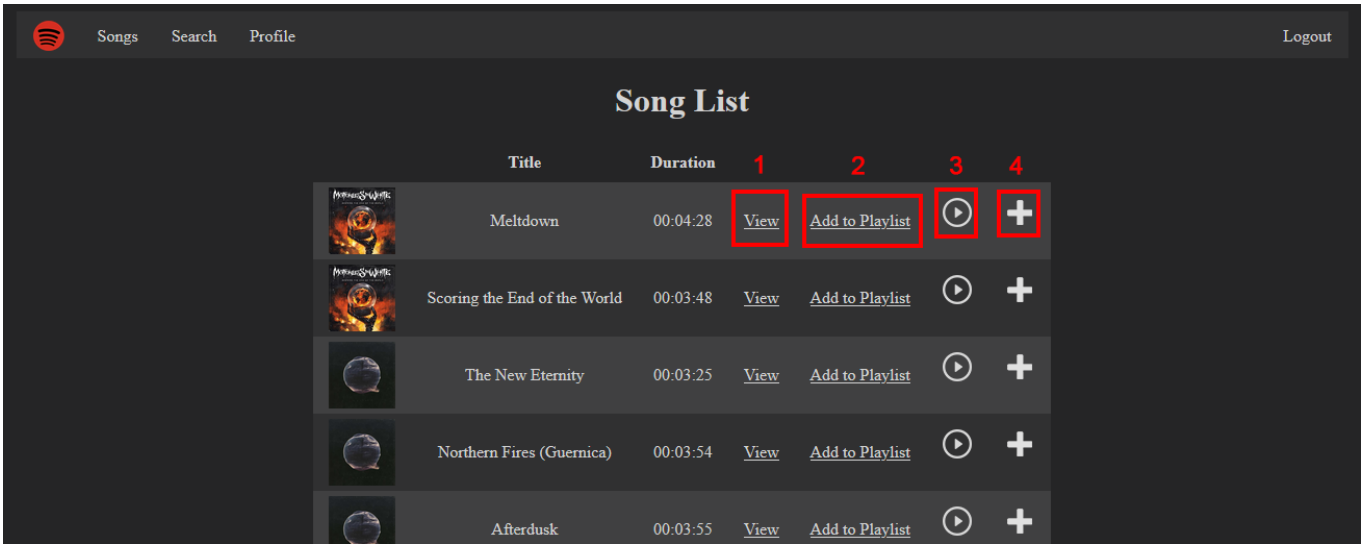
Premium User

Home Page



The home page for a premium user. Note that this user can view the Song List, instead of Advertisements.
1. Website logo. Click at any point to return to this home page.
2. Navigate to Advertisements.
3. Navigate to Search.
4. Navigate to your account's Profile.
5. Click to log out of your account.

Song List



The songs list page for a premium user. Similar to the song list page for an unregistered user, premium users have the additional functionality of being able to add songs to playlists, play songs, and add them to the queue.
1. Click to view the details of the song.
2. Click to add the song to one of your playlists.
3. Click to play the song.

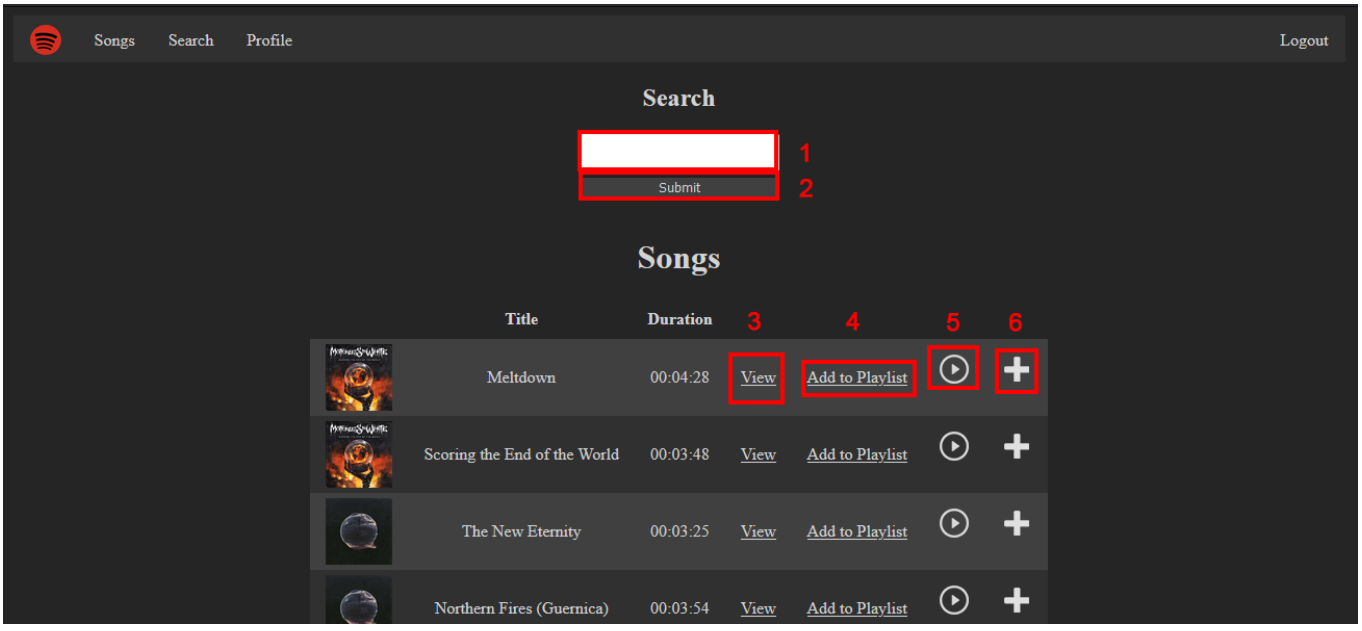
4. Click to add the song to your queue.

Song Page

A page dedicated to a single song. It shows metadata for that song, such as the title, number of plays and the duration. As a premium user, you are able to play the song, add it to the music queue, or add it to a playlist. You can also enable or disable different music stems that will remain in that state until you log out.

1. The song's title.
2. The song's cover art.
3. Click to play the song.
4. Click to add the song to your queue.
5. Click to add the song to one of your playlists.
6. Click to disable one of the stems of the song.

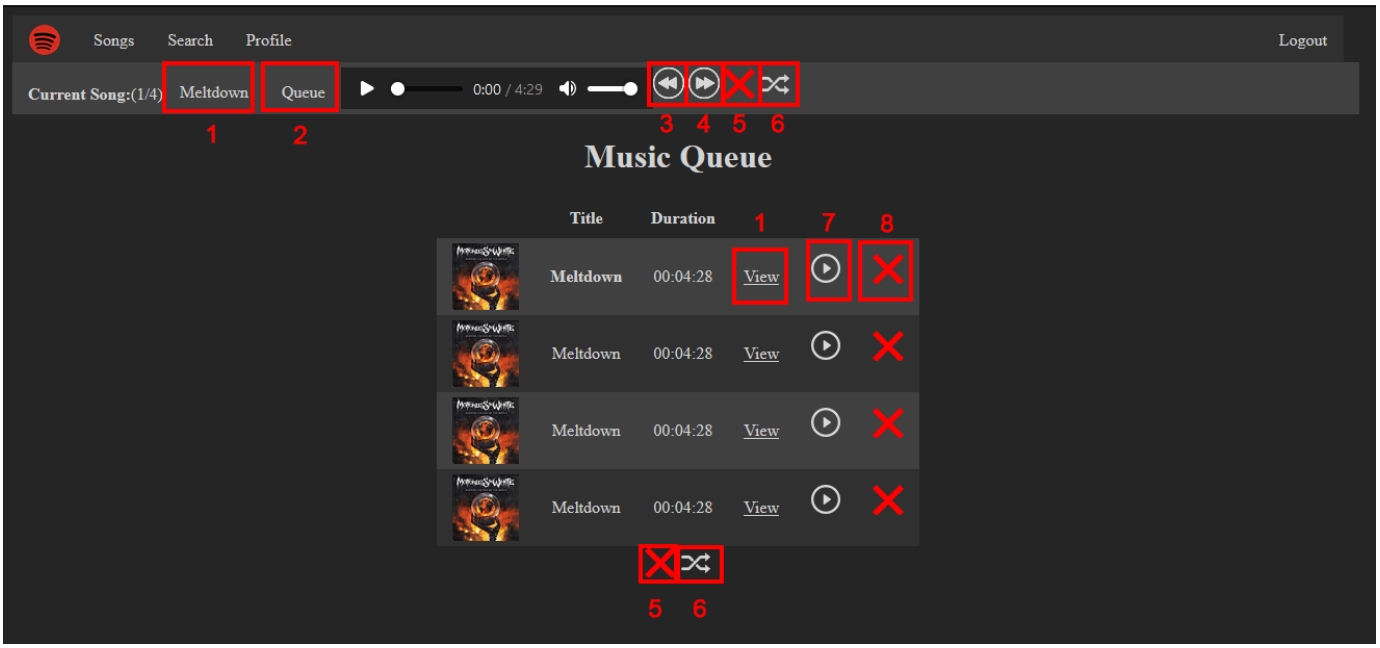
Search Page



The search page of a premium user. Similar to the search page of an unregistered user, it shows Songs, Albums, and Artists that fit the search query. Additional functionality is permitted as you can add songs to playlists and play songs or albums.

1. Click to type in a string to search for.
2. Click to submit your search. The list of songs, albums, and artists below will update to show only ones where some portion of their name matches your search query.
3. Click to view song details.
4. Click to add the song to one of your playlists.
5. Click to play the song.
6. Click to add the song to your queue.

Song Queue



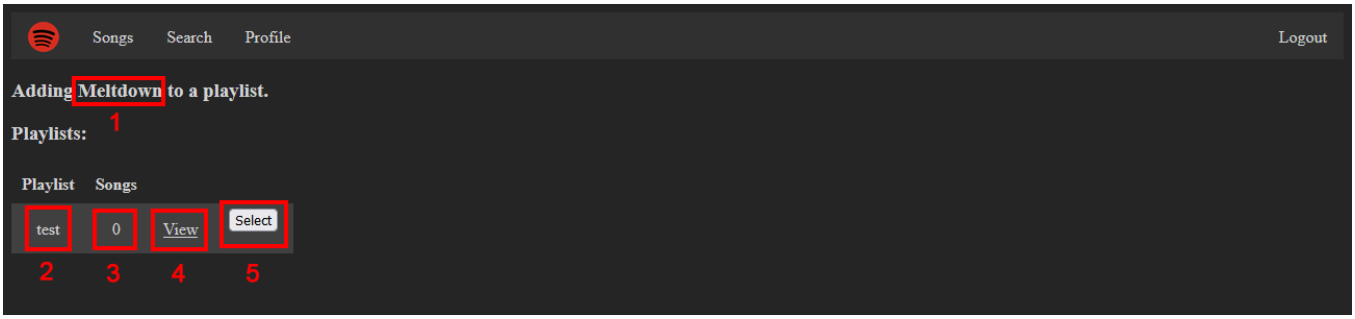
The current song queue. Whenever you play a song or album, or add a song to the queue, it will create a music queue that stores the list of songs you are listening to. You have several options to change the currently playing song, while also being able to shuffle and clear the queue. The queue bar at the top of the screen will persist on any other screen as long as you are logged in and there are songs in the queue.
1. The current song in the queue. Clicking it will navigate to that song's page.
2. Click to view the current queue (the page that is open in the screenshot).
3. Click this button to set the current song in the queue to the previous song, wrapping around to the last song if necessary.
4. Click this button to set the current song in the queue to the next song, wrapping around to the first song if necessary.
5. Click this button to clear the current queue and stop any music from playing.
6. Click this button to shuffle the queue, shuffling the list of songs and setting the current song to the first song of the list.
7. Click this button to play the specified song, setting the current song to the song that is being selected.
8. Click this button to remove the specified song from the queue.

Profile Page

Your profile page as a premium user. Similar to the free user, however you have the ability to create, edit, and play playlists.

1. Your username.
2. Your account type.
3. The playlist name of a given playlist, stored in the database under this user's ID.
4. The number of songs in the playlist. A playlist is simply a list of songs.
5. Click to go to the page dedicated to that playlist. To view the associated songs and make changes to the list.
6. Click to play this playlist, clearing the current queue and setting the queue to be the list of songs in the playlist.
7. Click to delete this playlist, removing it from your account.
8. The text field to enter the name of a new playlist.
9. Click this button to create the new playlist, under the name specified in 8.
10. Click to log out of your account.
11. Click to cancel your premium membership, reverting your account from premium to free that can only view ads.

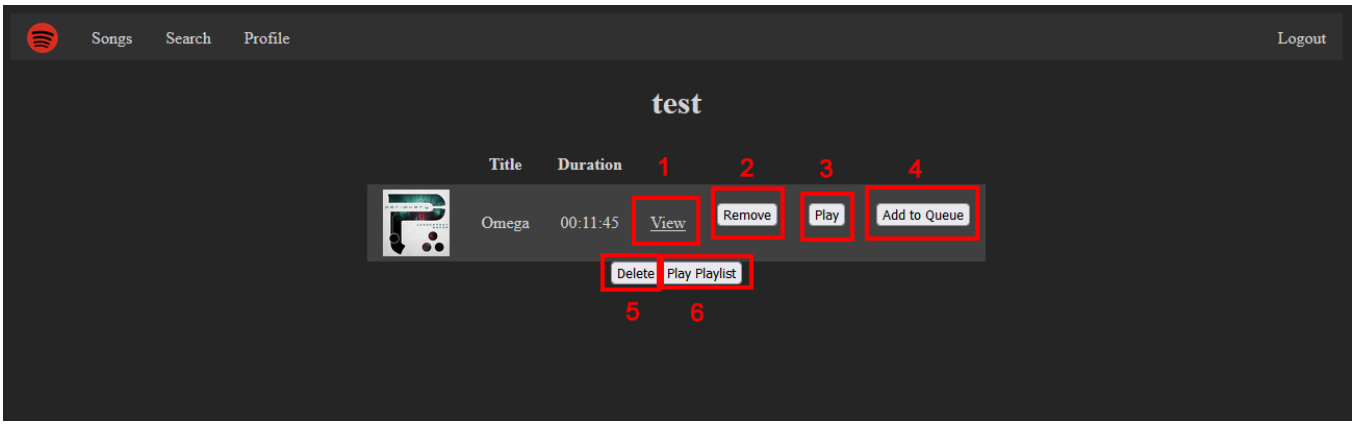
Add Song to Playlist



This menu comes up when you choose to add a song to a playlist. It will show all of the playlists attached to your account, and you can select the one to add it to. This will be stored in the database and will persist when you log out.

1. The title of the song that is currently being added to a playlist.
2. The name of a given playlist.
3. The number of songs in a playlist.
4. Click this button to view the playlist before adding the song to it.
5. Click this button to add the song to the chosen playlist.

View Playlist

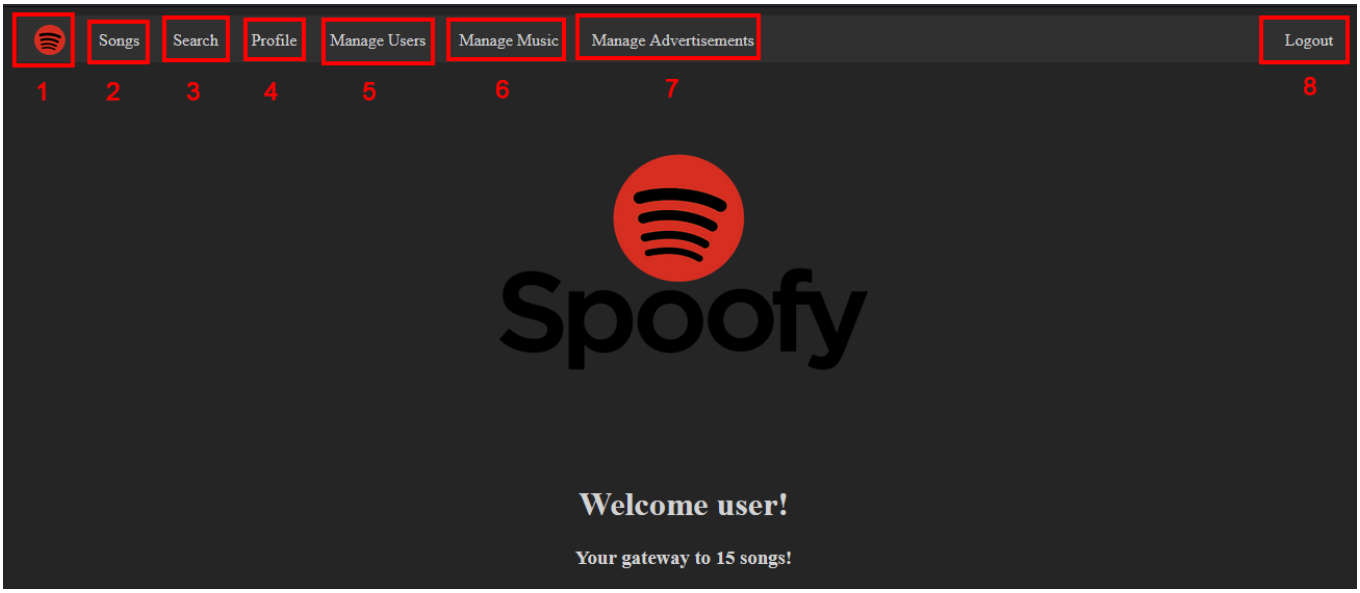


Viewing a playlist simply displays the list of songs present in it. You have options to play any song present in the playlist, or play the entire thing.

1. Click this button to view the song page of a given song.
2. Click this button to remove a song from the playlist.
3. Click this button to play this song, clearing the current queue if there is one.
4. Click this button to add the song to the end of the queue, creating a queue if necessary.
5. Click this button to delete the playlist from your account.
6. Click this button to play the whole playlist, clearing the current queue if there is one.

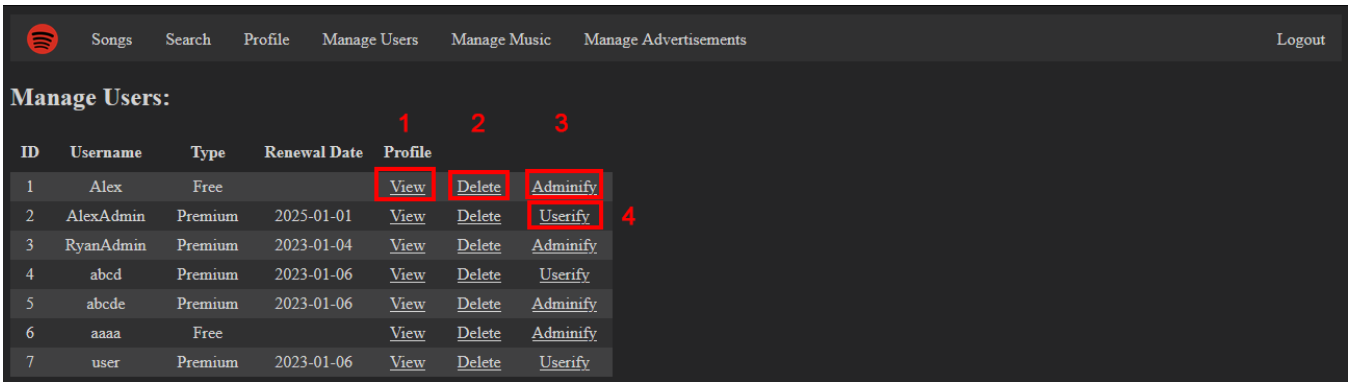
Admin

Home Page



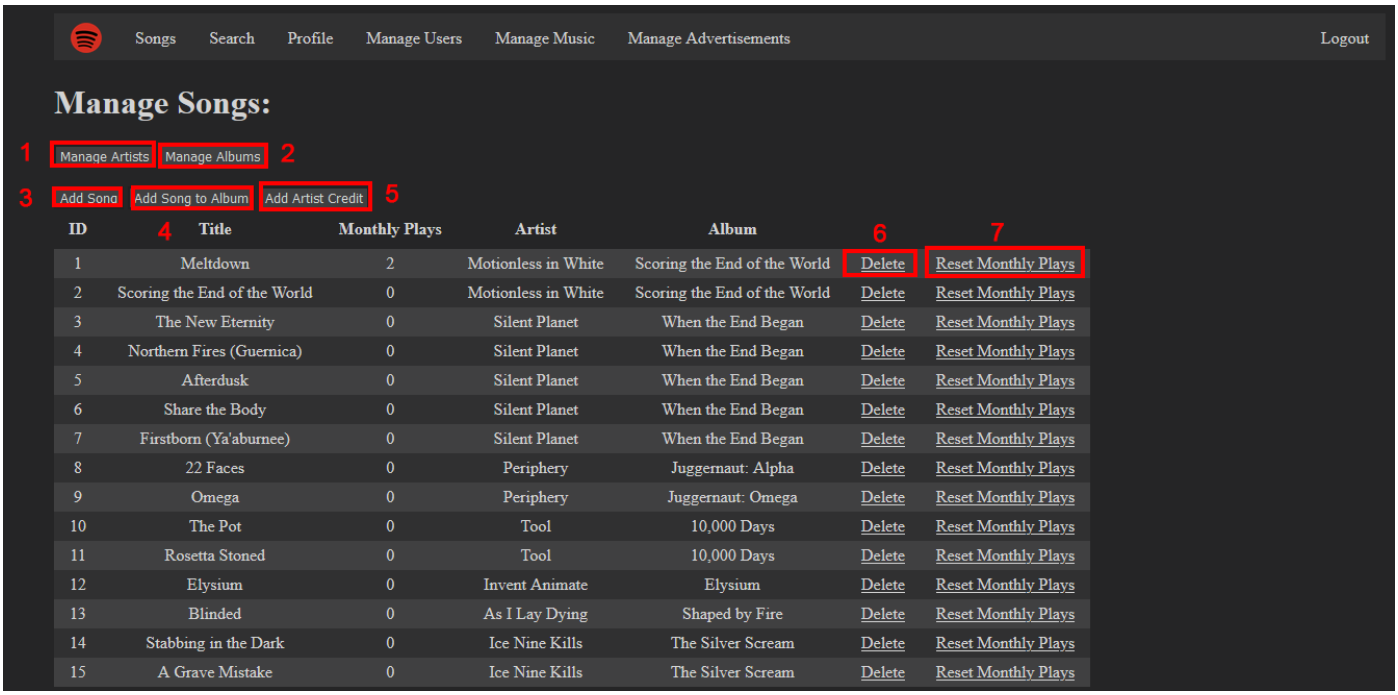
1. Website logo. Click at any point to return to this home page.
2. Navigate to Advertisements.
3. Navigate to Search.
4. Navigate to your account's Profile.
5. Navigate to the user management page.
6. Navigate to the song management page.
7. Navigate to the advertisement management page.
8. Click to log out of your account.

Manage Users



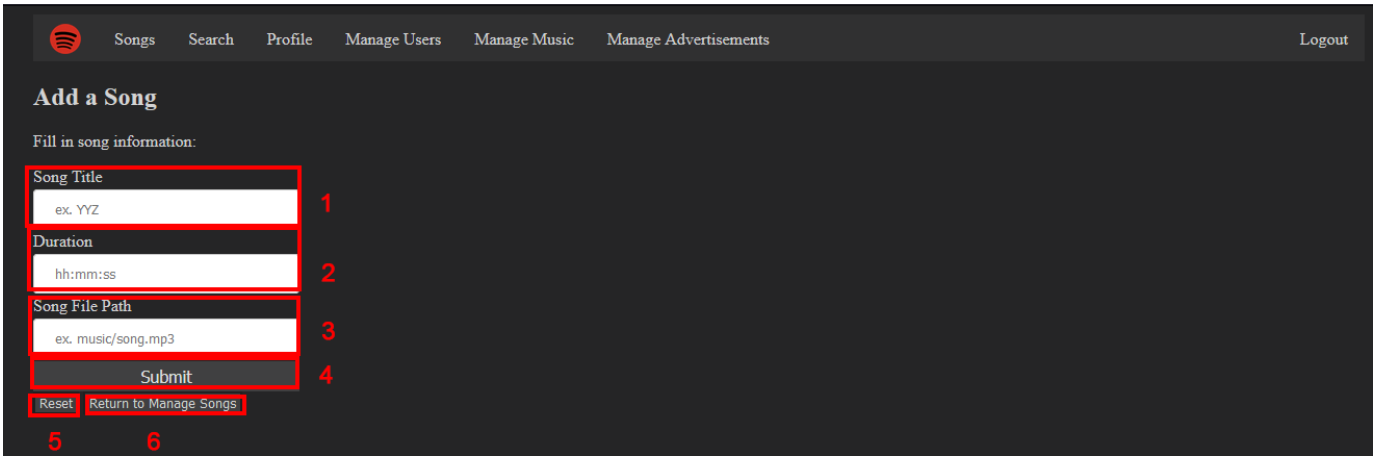
1. Click to view the profile of the user.
2. Click to delete the user's account from the service.
3. Click to turn a user account into an administrator account.
4. Click to turn an administrator account into a user account.

Manage Songs



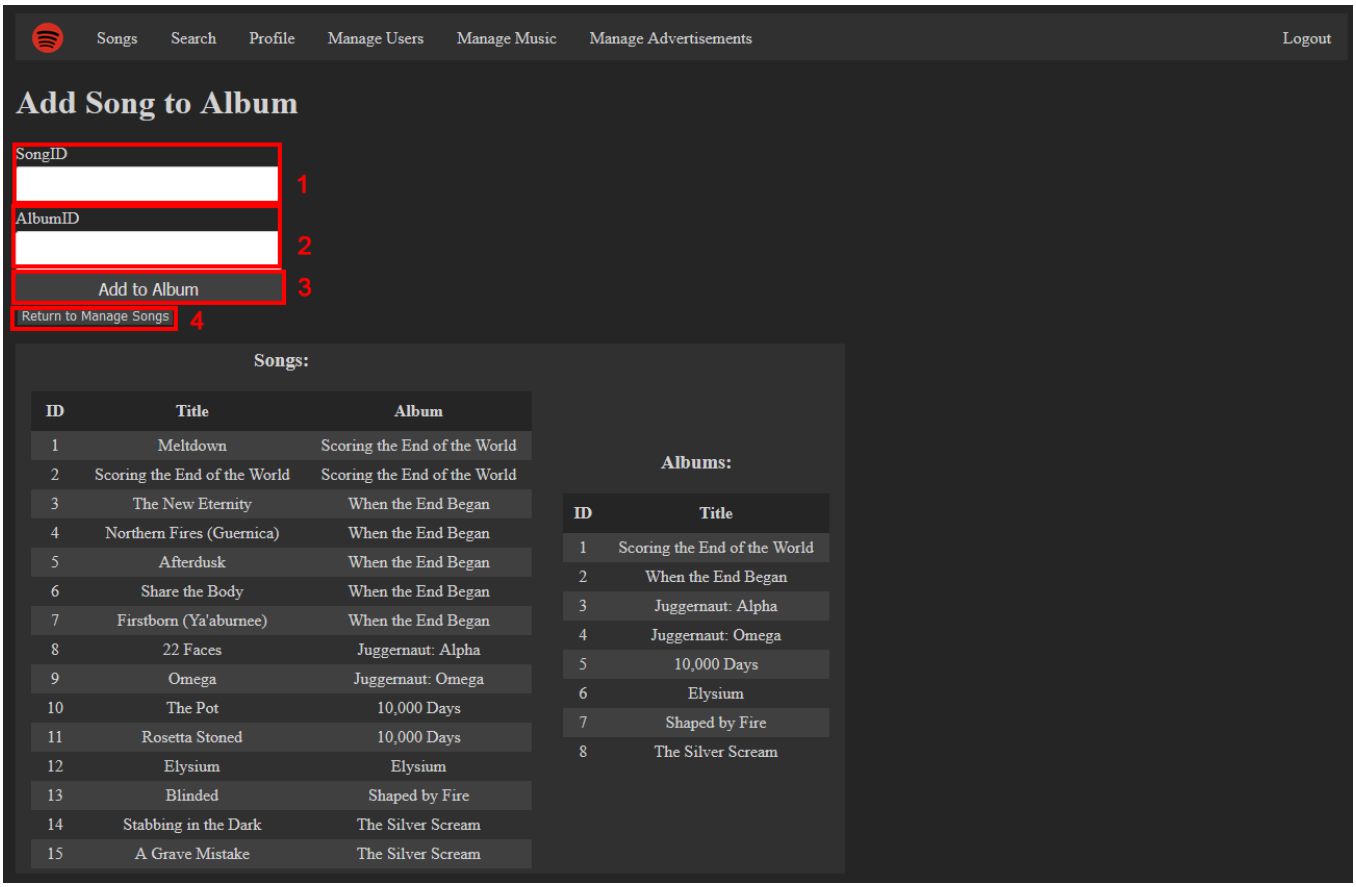
1. Navigate to the artist management page.
2. Navigate to the album management page.
3. Click to add a new song.
4. Click to add an existing song to an existing album.
5. Click to give an existing artist credit for an existing song.
6. Click to delete the song.
7. Click to reset the song's monthly plays.

Add Song



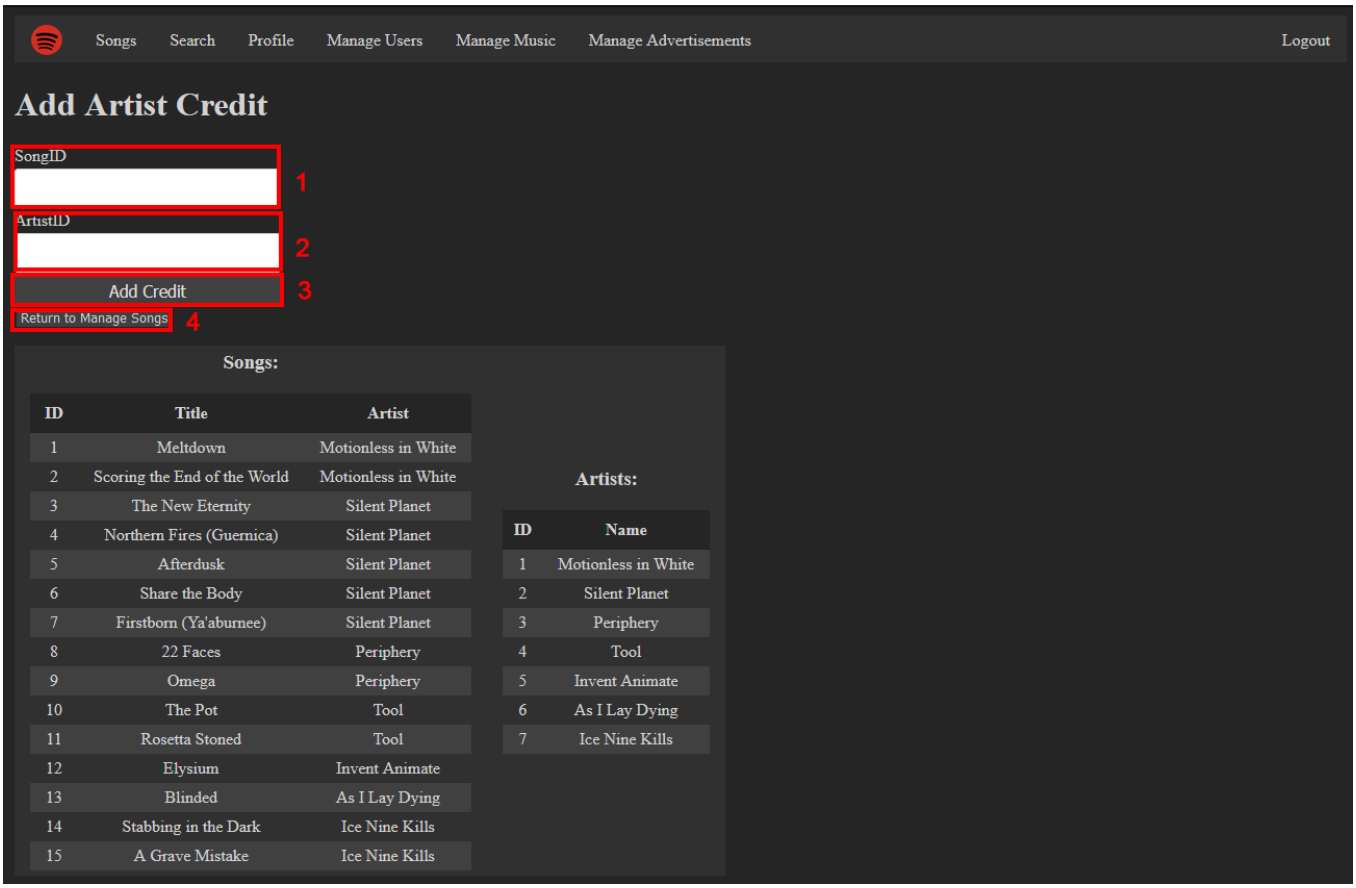
1. Click to type in the song's title.
2. Click to type in the song's duration in hh:mm:ss format.
3. Click to type in the file path to the song audio file.
4. Click to add the song to the database.
5. Click to reset the text boxes if you want to start fresh.
6. Navigate back to the song management page.

Add Song to Album



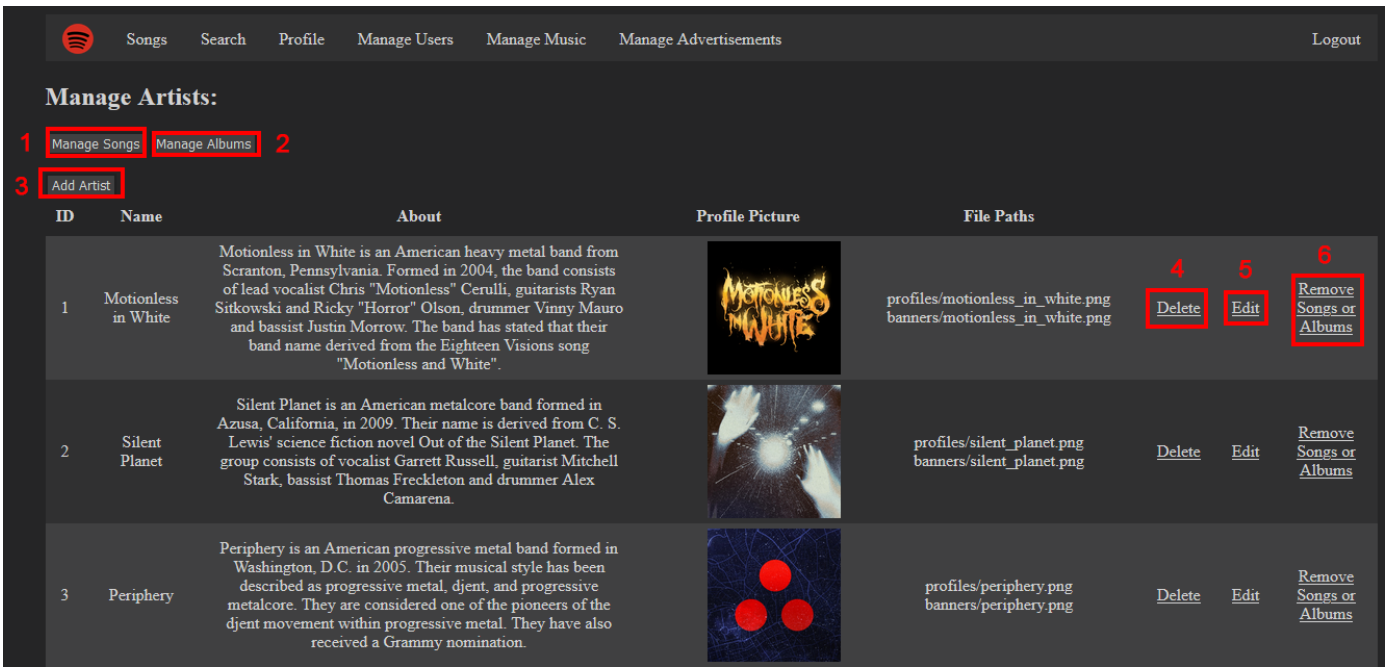
1. Click to input the ID number of the song you'd like to add to the album (listed below for reference).
2. Click to input the ID number of the album you'd like the song to be added to (listed below for reference).
3. Click to add the song to the album.
4. Navigate back to the song management page.

Add Song Credit



1. Click to input the ID number of the song you'd like to credit the artist for (listed below for reference).
2. Click to input the ID number of the artist you'd like to give song credit to (listed below for reference).
3. Click to add the credit for the song to the artist.
4. Navigate back to the song management page.

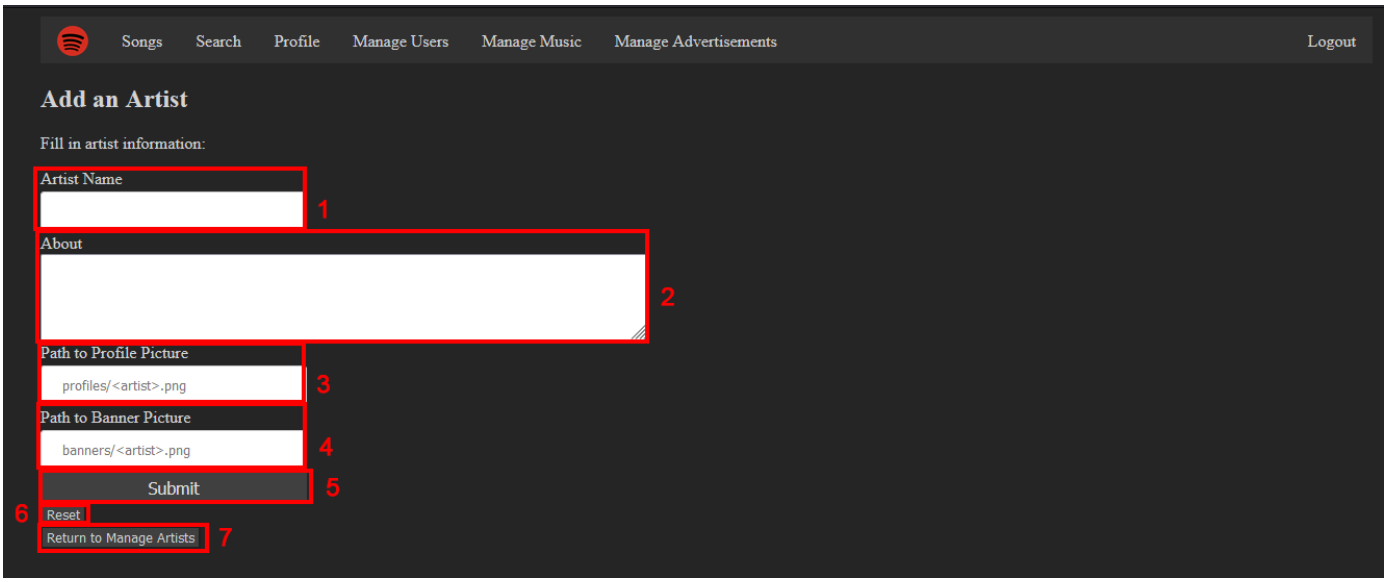
Manage Artist



1. Navigate to the song management page.

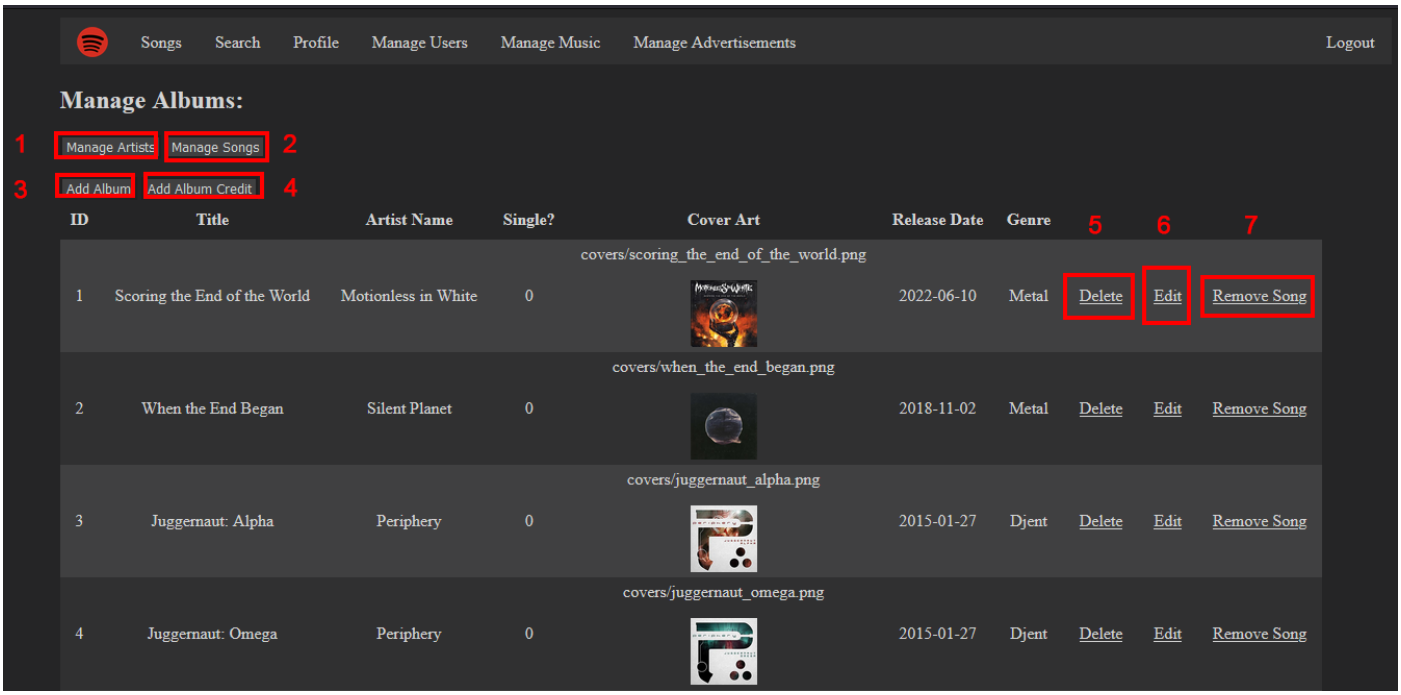
2. Navigate to the album management page.
3. Click to add a new artist.
4. Click to delete the artist from the database.
5. Click to edit the artist's details
6. Click to remove a song or an album from being associated with the artist.

Add Artist



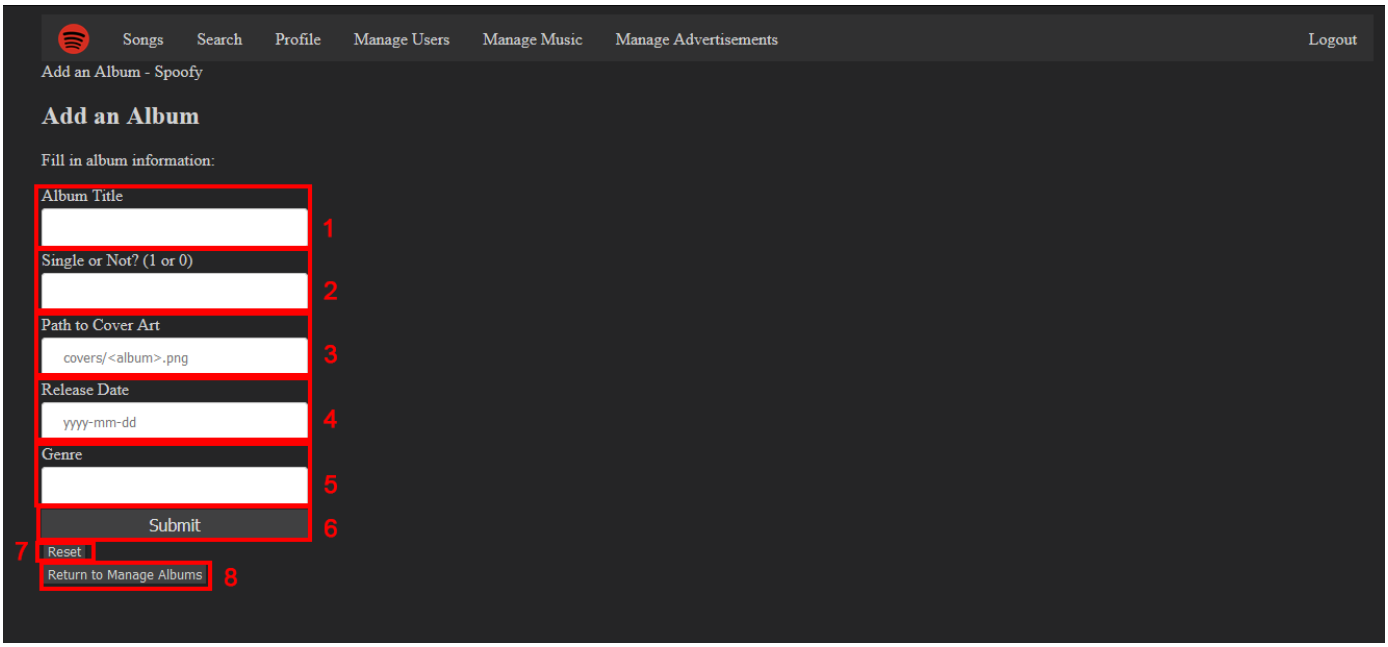
1. Click to type the name of the new artist.
2. Click to type a brief description of the artist.
3. Click to type the file path to the profile picture for the artist.
4. Click to type the file path to the banner picture for the artist.
5. Click to add the new artist to the database.
6. Click to reset the text boxes if you want to start fresh.
7. Navigate back to the artist management page.

Manage Album



1. Navigate to the artist management page.
2. Navigate to the song management page.
3. Click to add a new album.
4. Click to give an artist writing credit for an album.
5. Click to delete the album from the database.
6. Click to edit the album's details.
7. Click to remove a song from the album.

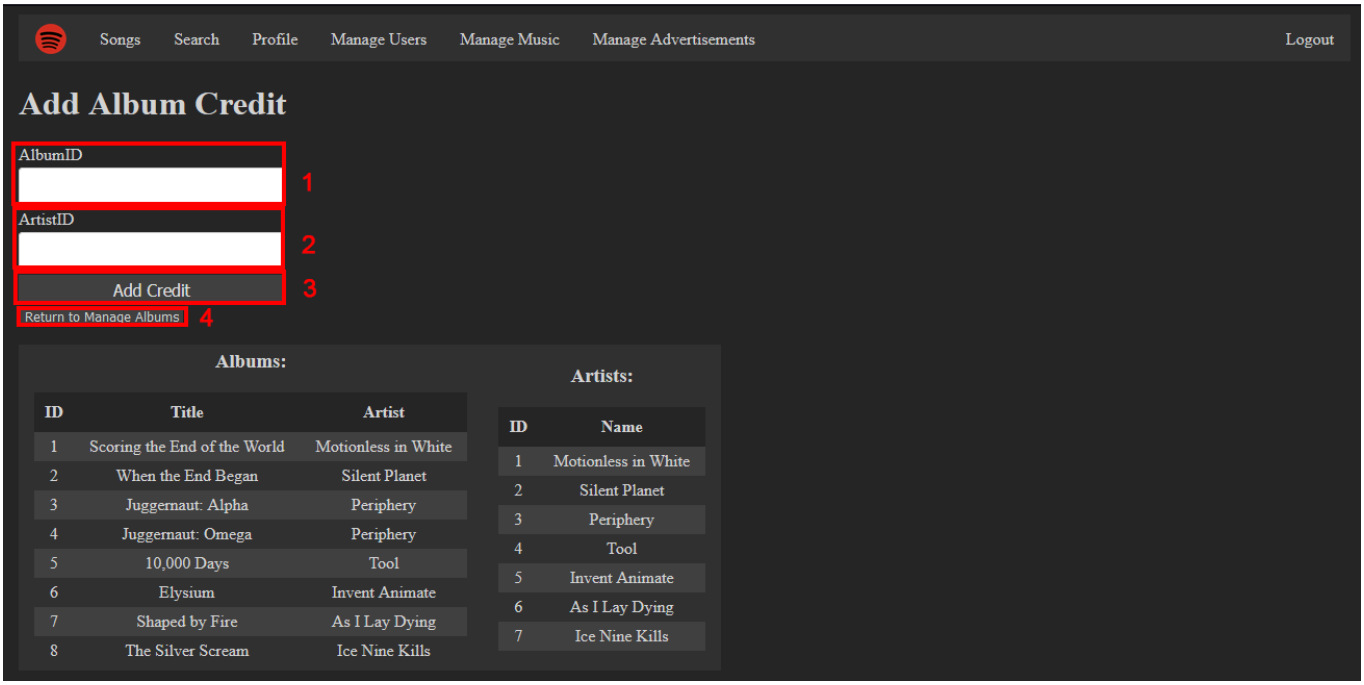
Add Album



1. Click to type the title of the new album.
2. Click to type a 1 if the new album is a single, or a 0 if it is not a single.
3. Click to type the file path to the album's cover art.
4. Click to type the release date of the album in yyyy-mm-dd format.
5. Click to type the genre of the album.

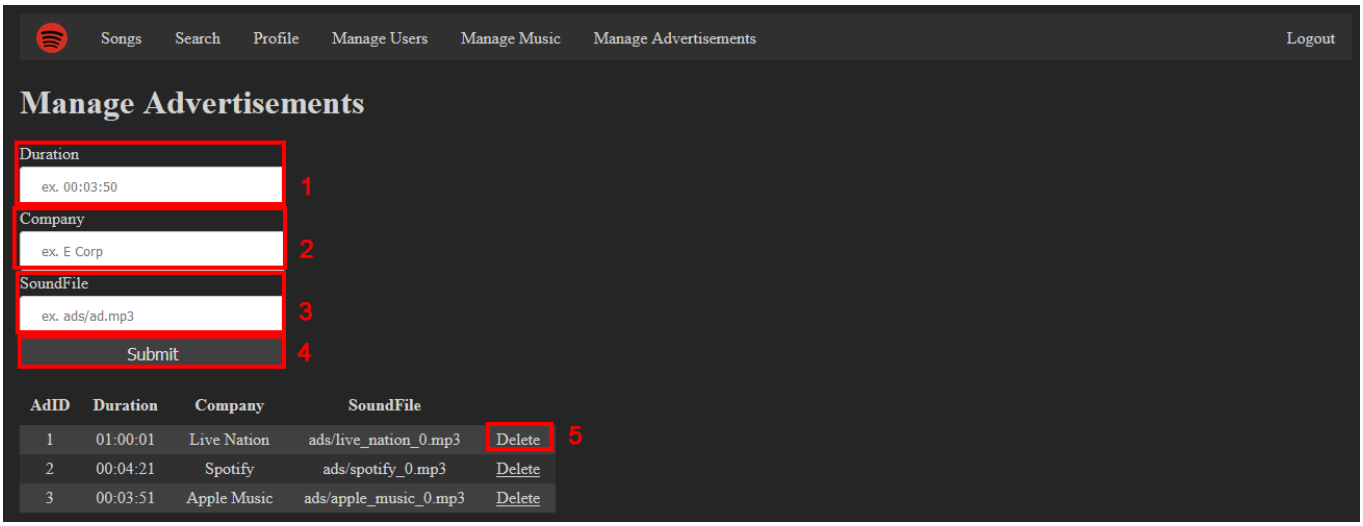
6. Click to add the new album to the database.
7. Click to reset the text boxes if you want to start fresh.
8. Navigate back to the album management page.

Add Album Credit



1. Click to input the ID number of the album you'd like to credit the artist for (listed below for reference).
2. Click to input the ID number of the artist you'd like credit for the album. (listed below for reference).
3. Click to credit the artist for the album.
4. Navigate back to the album management page.

Manage Ads



1. Click to type the duration of a new ad in hh:mm:ss format.
2. Click to type the name of the company a new ad is promoting.
3. Click to type the file path to the sound file of a new ad.
4. Click to add a new advertisement to the database.
5. Click to delete the advertisement from the database.

Appendix

Below is our mysql sample data. This sample data can be generated from a python script available in the github repository and usage is detailed in the readme for that repo.
https://github.com/alexs2112/CPSC471_Spoofy

Refreshing the default database:

```
C:\CPSC471_Spoofy>python InitDB/initialize.py refresh
Connection to SpoofyDB established.
Dropping tables in SpoofyDB.
Creating tables in SpoofyDB.
Populating tables with default data.
```

MySQL output from the MySQL 8.0 Command Line Client:

```
mysql> USE SpoofyDB;
Database changed
mysql> SHOW TABLES;
+--------------------+
| Tables_in_spoofydb |
+--------------------+
| admin              |
| advertisement      |
| album              |
| album_contains     |
| artist             |
| distributor        |
| has                |
| playlist           |
| playlist_contains  |
| represents         |
| song               |
| stem               |
| user               |
| writes             |
+--------------------+
14 rows in set (0.02 sec)

mysql> SELECT * FROM USER;
+--------+----------+--------------+-----------+--------------+
| UserID | Username | PasswordHash | IsPremium | SubRenewDate |
+--------+----------+--------------+-----------+--------------+
|      1 | Free     | <hash>       |         0 | NULL         |
|      2 | Admin    | <hash>       |         1 | 2025-01-01   |
|      3 | Free2    | <hash>       |         1 | 2023-01-04   |
+--------+----------+--------------+-----------+--------------+
3 rows in set (0.00 sec)z

mysql> SELECT * FROM ADMIN;
+---------+
| AdminID |
+---------+
|       2 |
+---------+
1 row in set (0.01 sec)
```

```
mysql> SELECT SongID, TotalPlays, MonthlyPlays, Title, Duration FROM SONG;
+--------+------------+--------------+------------------------------+----------+
| SongID | TotalPlays | MonthlyPlays | Title                        | Duration |
+--------+------------+--------------+------------------------------+----------+
|      1 |          0 |            0 | Meltdown                     | 00:04:28 |
|      2 |          0 |            0 | Scoring the End of the World | 00:03:48 |
|      3 |          0 |            0 | The New Eternity             | 00:03:25 |
|      4 |          0 |            0 | Northern Fires (Guernica)    | 00:03:54 |
|      5 |          0 |            0 | Afterdusk                    | 00:03:55 |
|      6 |          0 |            0 | Share the Body               | 00:03:33 |
|      7 |          0 |            0 | Firstborn (Ya'aburnee)       | 00:05:07 |
|      8 |          0 |            0 | 22 Faces                     | 00:03:51 |
|      9 |          0 |            0 | Omega                        | 00:11:45 |
|     10 |          0 |            0 | The Pot                      | 00:06:18 |
|     11 |          0 |            0 | Rosetta Stoned               | 00:11:13 |
|     12 |          0 |            0 | Elysium                      | 00:04:49 |
|     13 |          0 |            0 | Blinded                      | 00:03:22 |
|     14 |          0 |            0 | Stabbing in the Dark         | 00:04:40 |
|     15 |          0 |            0 | A Grave Mistake              | 00:03:04 |
+--------+------------+--------------+------------------------------+----------+
15 rows in set (0.00 sec)

mysql> SELECT SongID, MusicFILE FROM SONG;
+--------+-----------------------------------------------------------+
| SongID | MusicFILE                                                 |
+--------+-----------------------------------------------------------+
|      1 | songs/motionless_in_white/meltdown.mp3                    |
|      2 | songs/motionless_in_white/scoring_the_end_of_the_world.mp3 |
|      3 | songs/silent_planet/the_new_eternity.mp3                  |
|      4 | songs/silent_planet/northern_fires.mp3                    |
|      5 | songs/silent_planet/afterdusk.mp3                         |
|      6 | songs/silent_planet/share_the_body.mp3                    |
|      7 | songs/silent_planet/firstborn.mp3                         |
|      8 | songs/periphery/22_faces.mp3                              |
|      9 | songs/periphery/omega.mp3                                 |
|     10 | songs/tool/the_pot.mp3                                    |
|     11 | songs/tool/rosetta_stoned.mp3                             |
|     12 | songs/invent_animate/elysium.mp3                          |
|     13 | songs/as_i_lay_dying/blinded.mp3                          |
|     14 | songs/ice_nine_kills/stabbing_in_the_dark.mp3             |
|     15 | songs/ice_nine_kills/a_grave_mistake.mp3                  |
+--------+-----------------------------------------------------------+
15 rows in set (0.00 sec)
```

```
mysql> SELECT * FROM STEM;
+--------+--------+-----------------------------------------------------------+
| SongID | StemNo | MusicFile                                                 |
+--------+--------+-----------------------------------------------------------+
|      1 |      0 | stems/motionless_in_white/meltdown/0.mp3                  |
|      1 |      1 | stems/motionless_in_white/meltdown/1.mp3                  |
|      2 |      0 | stems/motionless_in_white/scoring_the_end_of_the_world/0.mp3 |
|      2 |      1 | stems/motionless_in_white/scoring_the_end_of_the_world/1.mp3 |
|      2 |      2 | stems/motionless_in_white/scoring_the_end_of_the_world/2.mp3 |
|      2 |      3 | stems/motionless_in_white/scoring_the_end_of_the_world/3.mp3 |
|      3 |      0 | stems/silent_planet/the_new_eternity/0.mp3                |
|      3 |      1 | stems/silent_planet/the_new_eternity/1.mp3                |
|      3 |      2 | stems/silent_planet/the_new_eternity/2.mp3                |
|      4 |      0 | stems/silent_planet/northern_fires/0.mp3                  |
|      4 |      1 | stems/silent_planet/northern_fires/1.mp3                  |
|      4 |      2 | stems/silent_planet/northern_fires/2.mp3                  |
|      5 |      0 | stems/silent_planet/afterdusk/0.mp3                       |
|      5 |      1 | stems/silent_planet/afterdusk/1.mp3                       |
|      5 |      2 | stems/silent_planet/afterdusk/2.mp3                       |
|      5 |      3 | stems/silent_planet/afterdusk/3.mp3                       |
|      6 |      0 | stems/silent_planet/share_the_body/0.mp3                  |
|      6 |      1 | stems/silent_planet/share_the_body/1.mp3                  |
|      7 |      0 | stems/silent_planet/firstborn/0.mp3                       |
|      7 |      1 | stems/silent_planet/firstborn/1.mp3                       |
|      7 |      2 | stems/silent_planet/firstborn/2.mp3                       |
|      7 |      3 | stems/silent_planet/firstborn/3.mp3                       |
|      8 |      0 | stems/periphery/22_faces/0.mp3                            |
|      8 |      1 | stems/periphery/22_faces/1.mp3                            |
|      8 |      2 | stems/periphery/22_faces/2.mp3                            |
|      9 |      0 | stems/periphery/omega/0.mp3                               |
|      9 |      1 | stems/periphery/omega/1.mp3                               |
|      9 |      2 | stems/periphery/omega/2.mp3                               |
|      9 |      3 | stems/periphery/omega/3.mp3                               |
|      9 |      4 | stems/periphery/omega/4.mp3                               |
|     10 |      0 | stems/tool/the_pot/0.mp3                                  |
|     10 |      1 | stems/tool/the_pot/1.mp3                                  |
|     10 |      2 | stems/tool/the_pot/2.mp3                                  |
|     11 |      0 | stems/tool/rosetta_stoned/0.mp3                           |
|     11 |      1 | stems/tool/rosetta_stoned/1.mp3                           |
|     11 |      2 | stems/tool/rosetta_stoned/2.mp3                           |
|     11 |      3 | stems/tool/rosetta_stoned/3.mp3                           |
|     11 |      4 | stems/tool/rosetta_stoned/4.mp3                           |
|     12 |      0 | stems/invent_animate/elysium/0.mp3                        |
|     12 |      1 | stems/invent_animate/elysium/1.mp3                        |
|     12 |      2 | stems/invent_animate/elysium/2.mp3                        |
|     13 |      0 | stems/as_i_lay_dying/blinded/0.mp3                        |
|     13 |      1 | stems/as_i_lay_dying/blinded/1.mp3                        |
|     13 |      2 | stems/as_i_lay_dying/blinded/2.mp3                        |
|     13 |      3 | stems/as_i_lay_dying/blinded/3.mp3                        |
|     14 |      0 | stems/ice_nine_kills/stabbing_in_the_dark/0.mp3           |
|     14 |      1 | stems/ice_nine_kills/stabbing_in_the_dark/1.mp3           |
|     15 |      0 | stems/ice_nine_kills/a_grave_mistake/0.mp3                |
|     15 |      1 | stems/ice_nine_kills/a_grave_mistake/1.mp3                |
|     15 |      2 | stems/ice_nine_kills/a_grave_mistake/2.mp3                |
+--------+--------+-----------------------------------------------------------+
50 rows in set (0.01 sec)
```

```
mysql> SELECT * FROM PLAYLIST;
+------------+--------------+-----------+
| PlaylistID | PlaylistName | CreatorID |
+------------+--------------+-----------+
|          1 | Playlist     |         2 |
+------------+--------------+-----------+
1 row in set (0.00 sec)

mysql> SELECT * FROM PLAYLIST_CONTAINS;
+------------+--------+
| PlaylistID | SongID |
+------------+--------+
|          1 |      3 |
|          1 |      5 |
|          1 |      8 |
+------------+--------+
3 rows in set (0.00 sec)

mysql> SELECT * FROM ADVERTISEMENT;
+------+----------+-------------+-----------------------+
| AdID | Duration | Company     | SoundFile             |
+------+----------+-------------+-----------------------+
|    1 | 01:00:01 | Live Nation | ads/live_nation_0.mp3 |
|    2 | 00:04:21 | Spotify     | ads/spotify_0.mp3     |
|    3 | 00:03:51 | Apple Music | ads/apple_music_0.mp3 |
+------+----------+-------------+-----------------------+
3 rows in set (0.00 sec)

mysql> SELECT * FROM DISTRIBUTOR;
+---------------+-----------------+
| DistributorID | DistributorName |
+---------------+-----------------+
|             1 | BigDistribute   |
|             2 | Yeet            |
|             3 | DistributorA    |
|             4 | Diztributor     |
+---------------+-----------------+
4 rows in set (0.00 sec)

mysql> SELECT ArtistID, Name, TotalPlays, MonthlyPlays FROM ARTIST;
+----------+--------------------+------------+--------------+
| ArtistID | Name               | TotalPlays | MonthlyPlays |
+----------+--------------------+------------+--------------+
|        1 | Motionless in White |          0 |            0 |
|        2 | Silent Planet      |          0 |            0 |
|        3 | Periphery          |          0 |            0 |
|        4 | Tool               |          0 |            0 |
|        5 | Invent Animate     |          0 |            0 |
|        6 | As I Lay Dying     |          0 |            0 |
|        7 | Ice Nine Kills     |          0 |            0 |
+----------+--------------------+------------+--------------+
7 rows in set (0.00 sec)
Note: About, ProfilePicture, BannerPicture omitted due to length
```

```
mysql> SELECT * FROM REPRESENTS;
+----------+--------------+
| ArtistID | DistributorID |
+----------+--------------+
|        1 |            1 |
|        2 |            1 |
|        3 |            1 |
|        4 |            2 |
|        5 |            3 |
|        6 |            3 |
|        7 |            4 |
+----------+--------------+
7 rows in set (0.00 sec)

mysql> SELECT * FROM WRITES;
+--------+----------+
| SongID | ArtistID |
+--------+----------+
|      1 |        1 |
|      2 |        1 |
|      3 |        2 |
|      4 |        2 |
|      5 |        2 |
|      6 |        2 |
|      7 |        2 |
|      8 |        3 |
|      9 |        3 |
|     10 |        4 |
|     11 |        4 |
|     12 |        5 |
|     13 |        6 |
|     14 |        7 |
|     15 |        7 |
+--------+----------+
15 rows in set (0.00 sec)

mysql> SELECT AlbumID, IsSingle, ReleaseDate, Genre, Title FROM ALBUM;
+---------+----------+-------------+-------+-----------------------------+
| AlbumID | IsSingle | ReleaseDate | Genre | Title                       |
+---------+----------+-------------+-------+-----------------------------+
|       1 |        0 | 2022-06-10  | Metal | Scoring the End of the World |
|       2 |        0 | 2018-11-02  | Metal | When the End Began          |
|       3 |        0 | 2015-01-27  | Djent | Juggernaut: Alpha           |
|       4 |        0 | 2015-01-27  | Djent | Juggernaut: Omega           |
|       5 |        0 | 2006-04-28  | Rock  | 10,000 Days                 |
|       6 |        1 | 2022-11-08  | Metal | Elysium                     |
|       7 |        0 | 2019-09-20  | Metal | Shaped by Fire              |
|       8 |        0 | 2018-10-05  | Metal | The Silver Scream           |
+---------+----------+-------------+-------+-----------------------------+
8 rows in set (0.00 sec)
```

```
mysql> SELECT ALBUMID, CoverArt FROM ALBUM;
+---------+----------------------------------------+
| ALBUMID | CoverArt                               |
+---------+----------------------------------------+
|       1 | covers/scoring_the_end_of_the_world.png |
|       2 | covers/when_the_end_began.png          |
|       3 | covers/juggernaut_alpha.png            |
|       4 | covers/juggernaut_omega.png            |
|       5 | covers/10000_days.png                  |
|       6 | covers/elysium.png                     |
|       7 | covers/shaped_by_fire.png              |
|       8 | covers/the_silver_scream.png           |
+---------+----------------------------------------+
8 rows in set (0.00 sec)

mysql> SELECT * FROM HAS;
+---------+----------+
| AlbumID | ArtistID |
+---------+----------+
|       1 |        1 |
|       2 |        2 |
|       3 |        3 |
|       4 |        3 |
|       5 |        4 |
|       6 |        5 |
|       7 |        6 |
|       8 |        7 |
+---------+----------+
8 rows in set (0.00 sec)

mysql> SELECT * FROM ALBUM_CONTAINS;
+---------+--------+
| AlbumID | SongID |
+---------+--------+
|       1 |      1 |
|       1 |      2 |
|       2 |      3 |
|       2 |      4 |
|       2 |      5 |
|       2 |      6 |
|       2 |      7 |
|       3 |      8 |
|       4 |      9 |
|       5 |     10 |
|       5 |     11 |
|       6 |     12 |
|       7 |     13 |
|       8 |     14 |
|       8 |     15 |
+---------+--------+
15 rows in set (0.00 sec)
```

References

Learning Material:
    Lecture slides from module 4, parts A and B
    https://www.w3schools.com/php/
    https://www.php.net/docs.php
    https://www.tutorialspoint.com/php/index.html

General Help:
    https://httpd.apache.org/docs/2.4/platform/windows.html
    https://www.w3schools.com/css/css_background.asp
    https://www.w3schools.com/css/css_navbar_horizontal.asp
    https://www.w3schools.com/css/css_table_style.asp
    https://www.w3schools.com/css/css_table.asp
    https://stackoverflow.com/questions/34011559/how-do-i-add-my-logo-to-my-website-using-html
    https://stackoverflow.com/questions/8090195/how-do-i-align-in-my-image-to-move-it-vertically-upcss-html
    https://www.geeksforgeeks.org/how-to-change-an-input-button-image-using-css/
    https://www.php.net/manual/en/function.empty.php
    https://stackoverflow.com/a/369761

Tools:
    https://www.appserv.org/en/
    https://discord.com/   (group discussion and virtual meetings)
    https://www.overleaf.com/   (latex formatting)
    https://app.diagrams.net/   (diagram creation)
    https://github.com/   (git repository host)

Resources:
    https://twitter.com/   (profile and banner pics for artists)
    https://en.wikipedia.org/   (artist about section, album covers)