Music Madness

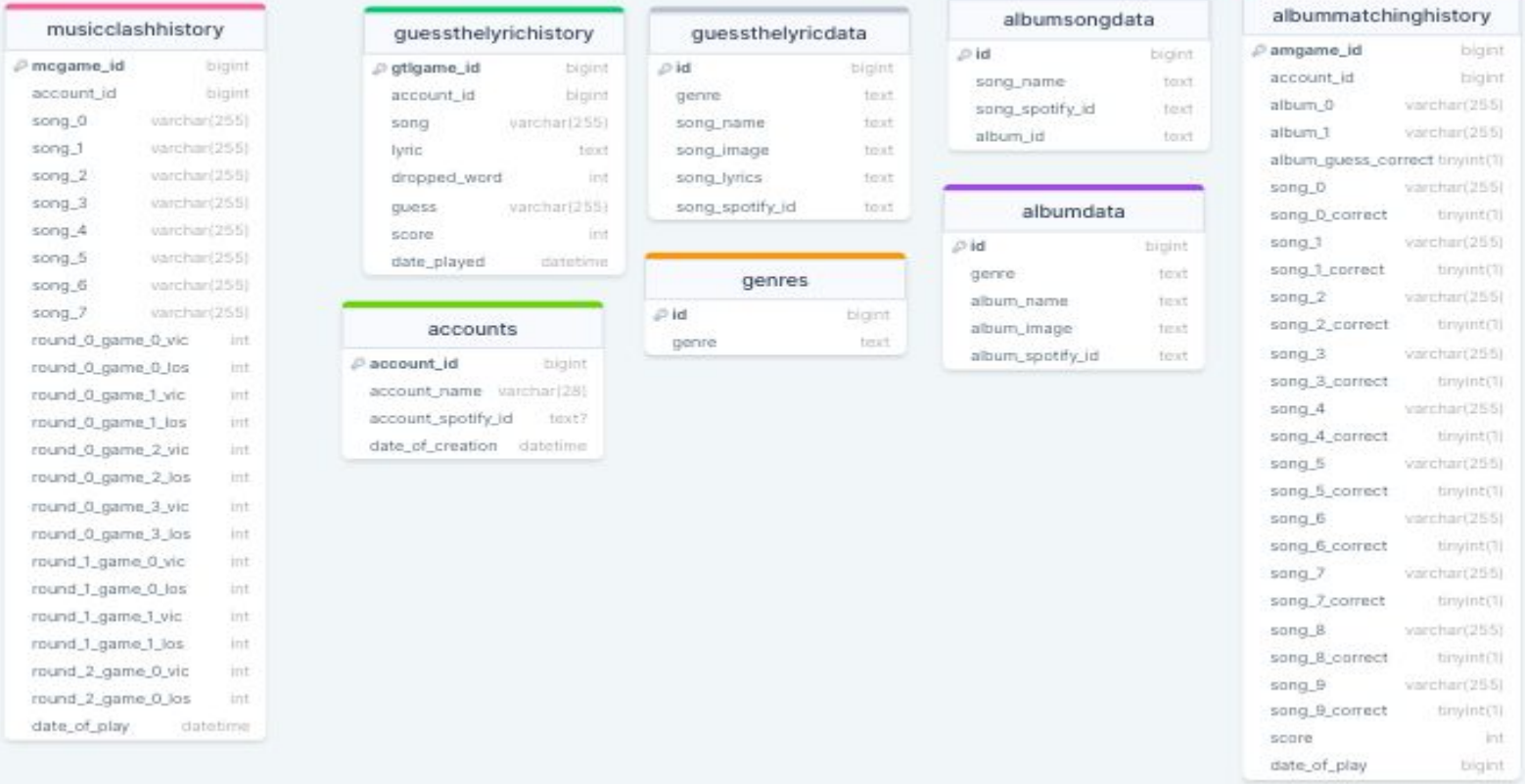
Jamaal Dublin Jr & Ryan Jakubielski

Introduction

We chose to create the website since we enjoy music and wanted to include it in the project. It's a music trivia website where you can test your knowledge of music and win points.

DEMO

DB


musicclashhistory
mcgame_id bigint
account_id bigint
song_0 varchar(255)
song_1 varchar(255)
song_2 varchar(255)
song_3 varchar(255)
song_4 varchar(255)
song_5 varchar(255)
song_6 varchar(255)
song_7 varchar(255)
round_0_game_0_vic int
round_0_game_0_los int
round_0_game_1_vic int
round_0_game_1_los int
round_0_game_2_vic int
round_0_game_2_los int
round_0_game_3_vic int
round_0_game_3_los int
round_1_game_0_vic int
round_1_game_0_los int
round_1_game_1_vic int
round_1_game_1_los int
round_2_game_0_vic int
round_2_game_0_los int
date_of_play datetime
guessthelyrichistory
gtlgame_id bigint
account_id bigint
song varchar(255)
lyric text
dropped_word int
guess varchar(255)
score int
date_played datetime
accounts
account_id bigint
account_name varchar(28)
account_spotify_id text?
date_of_creation datetime
guessthelyricdata
id bigint
genre text
song_name text
song_image text
song_lyrics text
song_spotify_id text
genres
id bigint
genre text
albumsongdata
id bigint
song_name text
song_spotify_id text
album_id text
albumdata
id bigint
genre text
album_name text
album_image text
album_spotify_id text
albummatchinghistory
amgame_id bigint
account_id bigint
album_0 varchar(255)
album_1 varchar(255)
album_guess_correct tinyint(1)
song_0 varchar(255)
song_0_correct tinyint(1)
song_1 varchar(255)
song_1_correct tinyint(1)
song_2 varchar(255)
song_2_correct tinyint(1)
song_3 varchar(255)
song_3_correct tinyint(1)
song_4 varchar(255)
song_4_correct tinyint(1)
song_5 varchar(255)
song_5_correct tinyint(1)
song_6 varchar(255)
song_6_correct tinyint(1)
song_7 varchar(255)
song_7_correct tinyint(1)
song_8 varchar(255)
song_8_correct tinyint(1)
song_9 varchar(255)
song_9_correct tinyint(1)
score int
date_of_play bigint

Tables & Functions

Features like "searchUserPlayed" and "getUserScore" for both games were utilized to get the user's game history and then take the score to show their all-time scores for every game they have ever played which they can also look at for better user interactivity.

API Connections & Session Variables

- Utilized two data fetching APIs
  - Spotify Web API
    - Retrieved song data and profile data.
  - MusixMatch API
    - Retrieved lyric data for each song.
- Sessions variables role in the project
  - User ID for saving and retrieving game results.
  - Authentication Token was needed for every spotify data retrieval call.
  - Album Matching game hidden data.

Successes

- A completed project.
- A completed DB for storing and retrieving game history.
- Successfully utilized two different API's for ever changing game data.
- Proper Error catching for the many issues that can occur when fetching from multiple APIs.

Failures

- Failed to implement a third game.
- Failed to utilize the Spotify Playback API.
- Could have been more animation and polish to reflect more of a game feel.
- Not much user engagement

Impediments

- Too many requests API error.
- Getting songs that didn't properly have lyrics associated with them.
- The length of time to create one game.
- Improper CSS units during initial frontend development leading to time spent fixing later to fix for responsiveness. IE: px not em, rem, or %.

What would we do differently?

- Setup
- Research
- User Engagement / Interactivity
- Communication
- Time Management
- Tackle the hard parts first

Going Forward

What we learned?

- Proper time management in a large project is key to success.
- Make sure to give extra time for bug management.
- Spend a little more time thinking through ideas before prototyping.

Questions?