Treble in Paradise

Bonnie Atelsek
CSC 4710 - Database Systems

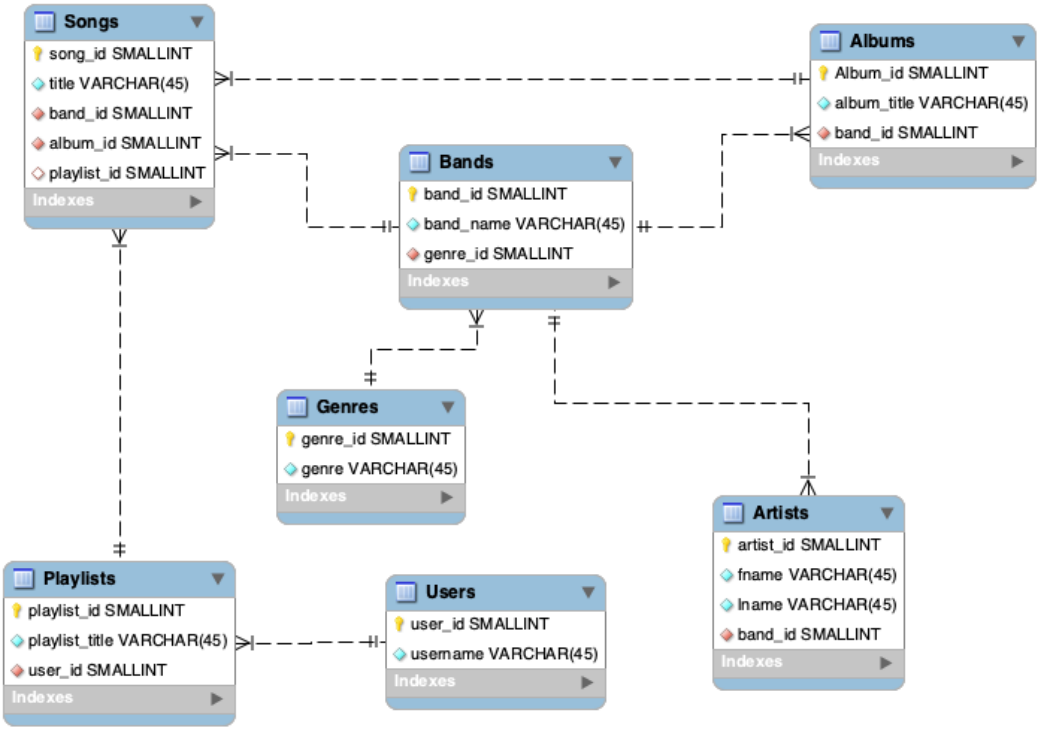
Introduction

The Treble in Paradise (TiP) database is meant to give users a medium with which to upload songs and create playlists based off of them. The front-end is a website which allows users to see all playlists, create a new playlist or edit a preexisting one, or add to the database a new song, band, or artist. The back-end hooks up to a MySQL database which stores artists, songs, albums, playlists, users, bands, and genres of music. The scope of this project is small but scalable- because of time constraints and limited resources the website and database are hosted locally and thus can only be accessed individually, but with relatively limited effort the project could be hosted by a third party in the future.

Implementation

The database was built in MySQL, and the interface was created using PHP, HTML, Javascript, and CSS. It is hosted and maintained using XAMPP and phpMyAdmin.

ER Diagram



Use Cases

1. **Add Song** - User clicks the “Add Song” button and is taken to a form which asks for a song title, an album name, and a band name (the band names are displayed as options in a drop-down menu and populated from the database).
    a. If the user attempts to submit the form without any of the fields filled out, the form will not submit and will notify the user that they must fill out the empty fields.
    b. If the album name submitted does not exist in the database, a new album will be created and the song will be added under it and connected to the submitted band.
    c. If the album name submitted exists in the database, the new song will be added under the existing album.
2. **Add Band** - User clicks the “Add Band” button and is taken to a form which asks for a band name and a musical genre (the genres are displayed as options in a drop-down menu and populated from the database).
    a. If the user attempts to submit the form without any of the fields filled out, the form will not submit and will notify the user that they must fill out empty fields.
    b. Upon submission, the form takes the first and last name values and creates a new artist entity connected to the selected band.
3. **Add Artist** - User clicks the “Add Artist” button and is taken to a form which asks for a first name, a last name, and a band name (the band names are displayed as options in a drop-down menu and populated from the database).
    a. If the user attempts to submit the form without any of the fields filled out, the form will not submit and will notify the user that they must fill out empty fields.
    b. Upon submission, the form takes the first and last name values and creates a new artist entity connected to the selected band.
4. **View Playlists** - User clicks the “View Playlists” button, the next page displays all playlists (playlist title, creator, and linked songs) currently in the database.
5. **Edit/Create Playlist** - User clicks the “Edit/Create Playlists” button and is taken to a form which asks for a username, a playlist name, and a song to add (the songs are displayed as options in a drop-down menu and populated from the database).
    a. If the user attempts to submit the form without any of the fields filled out, the form will not submit and will notify the user that they must fill out empty fields.
    b. If the playlist name already exists in the database, the song submitted will be added to that playlist and the username will be ignored.
    c. If the playlist name does not exist in the database, a new playlist will be created with the given playlist title and the song submitted will be added to that playlist.
        i. If the submitted username does not already exist in the database, a new user entity will be created with the given username.

Conclusion

The front-end structure is simplistic due to time-constraints; if given more time I would have cleaned up the styling, added more field checking to ensure that no faulty values entered the database, and added options to search the database for artists, songs, albums, genres, etc. However, the front-end functions as expected and interfaces correctly with the back-end.