Portfolio Project
Group Members: Jonathan Hirsch and Stephanie Conner
Project Title: Groovy DB

URL: http://flip1.engr.oregonstate.edu:1351/

Executive Summary

Our project changed a lot over the course of this quarter, there being 5 separate iterations (including this one) of our database. This summary will go through them beginning from the initial changes made, leading up to the most recent version of it. In step 1, the major changes that were made as a result of the critiques and change in design decisions were to clarify the design choices (specific to M:M relations), change genres to be affiliated with albums instead of artists.

In step 2, it was realized that 'Aritsts_albums' was not the greatest name for our intersection table as there is already a better name that exists for that exact use: discographies, which typically contains all the work of a peculiar artist. The relationship between albums and genres was changed from an M:M relationship to an M:1 relationship, as there was a lot of overlap that resulted when multiple genres and subgenres were associated with an album.

In step 3, instead of having the user manually enter IDs to the identifies they were looking for, a dynamic drop-down menu was developed, which would show all the current identities in the database. This makes it much more intuitive for a user as they do not have to remember nor manually enter all the IDs associated with entities in the database. Also, originally once a genre was made, it had to be tied to an album, however this wouldn't make sense as a genre should be able to exist without being tied to an album at the time of creation. Therefore, that relationship was made nullable, however, keep in mind that an album still must be associated with a genre when it's created as albums should be tied to a genre.

Finally, in the final draft (step 4), it was noticed that the user can enter a song into the database without a title. However, this should not be possible so a required statement was added so that a song title must be entered in order to enter a song into the database. In addition, the update function was not working as intended since the user had to manually refresh the page in order to get their updates to show, however, this was fixed and should no longer be an issue.

In summary, this project went through many development stages to get to its current state, and many changes were made along the way for functionality and accessibility. The goal of this project was to create an easy-to-use and styled database in order to please our users. Thanks to the peer feedback we've received as well as the iterations of that the project went through, it is believed that the goal has been accomplished.

Project Outline

Being a music fan can be challenging when there are so many new artists and songs to keep up with. If your music tastes encompass a wide variety of eras, genres, and energy levels, it becomes crucial to create a system that allows for quick and easy lookup of desired songs. Our database aims to assist in organizing and connecting users to music based on song title, album, artists, and genre in a user-friendly manner. With over 40 main genres of music, over 300 subgenres, and millions of songs and artists to choose from, our database provides an accessible solution for music enthusiasts to explore and enjoy their favorite tracks in an organized manner.

Database Outline

Songs

Records data pertaining to a specific song.

Attributes:

- songID: int, auto_increment, unique, not NULL, PK
- songName: varchar, not NULL
- albumID: int, not NULL, FK referencing the Album entity
- artistID: int, not NULL, FK referencing the Artist entity
- streamCount: int
- Relationships:
  - M:1 (Album) - multiple songs can be on 1 album
  - M:1 (Artist) - multiple songs can be from 1 Artist

Albums

Records and organizes data pertaining to an album. Many songs can appear on 1 album and many artists can work on multiple albums.

Attributes:

- albumID: int, auto_increment, unique, not NULL, PK
- albumName: varchar, not NULL
- artistID: int, NOT NULL, FK referencing the Artist entity
- genreID: int, NOT NULL, FK referencing Genre entity
- year: int, not NULL
- Relationship:
  - 1:M (Song) - 1 Album can have multiple songs
  - M:M (Artist) - Multiple Albums can have multiple artists
  - M:1(Genre) - Multiple albums can have 1 genre

Artists

Holds and organizes data pertaining to specific artists, holds relationships with albums and songs. To clarify, multiple artists can have many albums in cases such as for orchestral music, soundtracks, and other albums where there is collaboration.
Attributes:

- artistID: int, auto_increment, unique, not NULL, PK
- artistName: varchar, not NULL
- Relationship:
  - M:M (Album) - Multiple Artist can have many albums
  - 1:M (Songs) - 1 Artist can have many songs

Discographies

An intersection table for the M:N relationship between artists and albums.
Records an artist's relationships to the albums they've worked on/created.
Attributes:

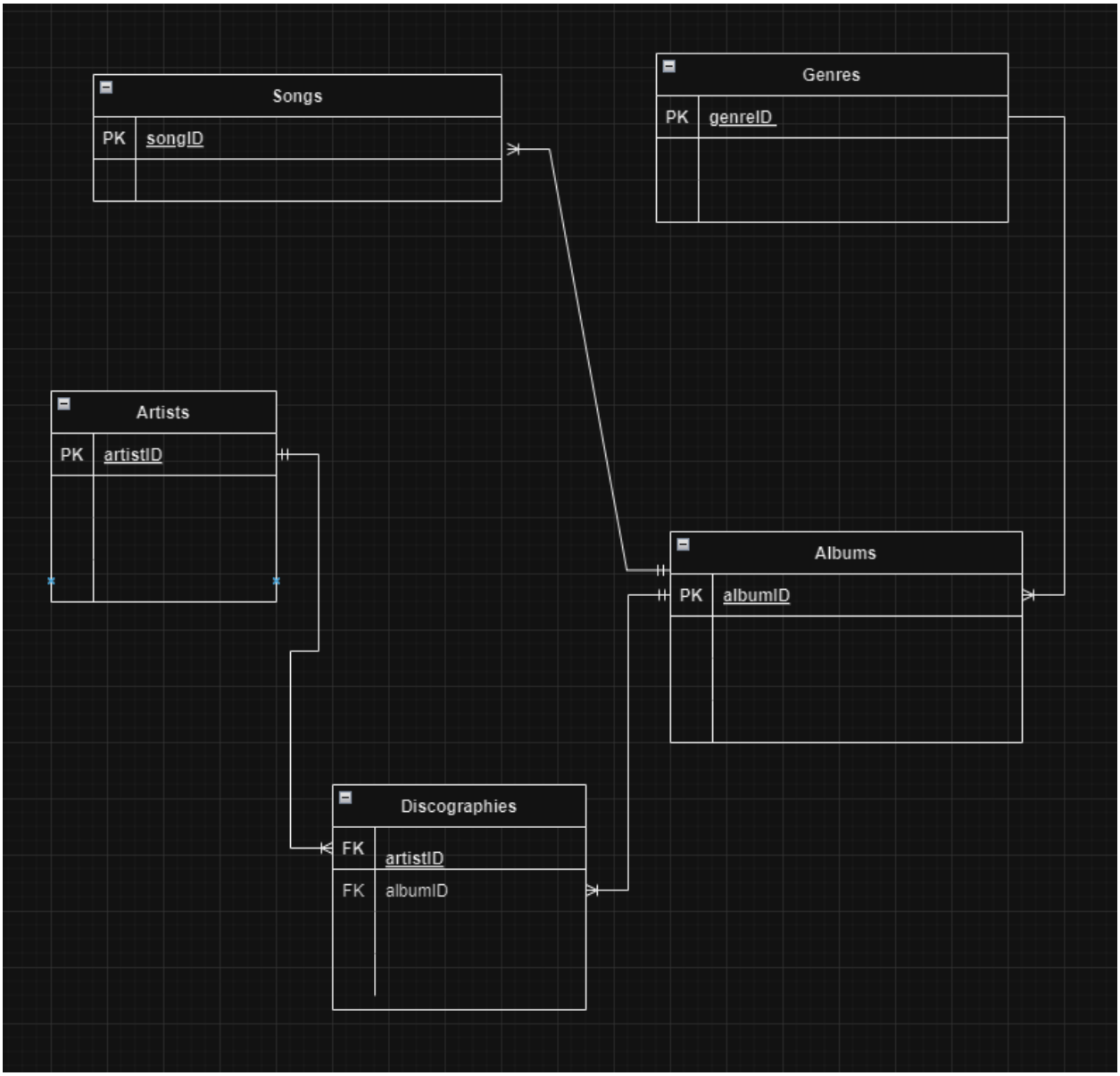
- artistID: int, FK, not NULL, unique
- albumID: int, FK, not NULL, unique

Genres

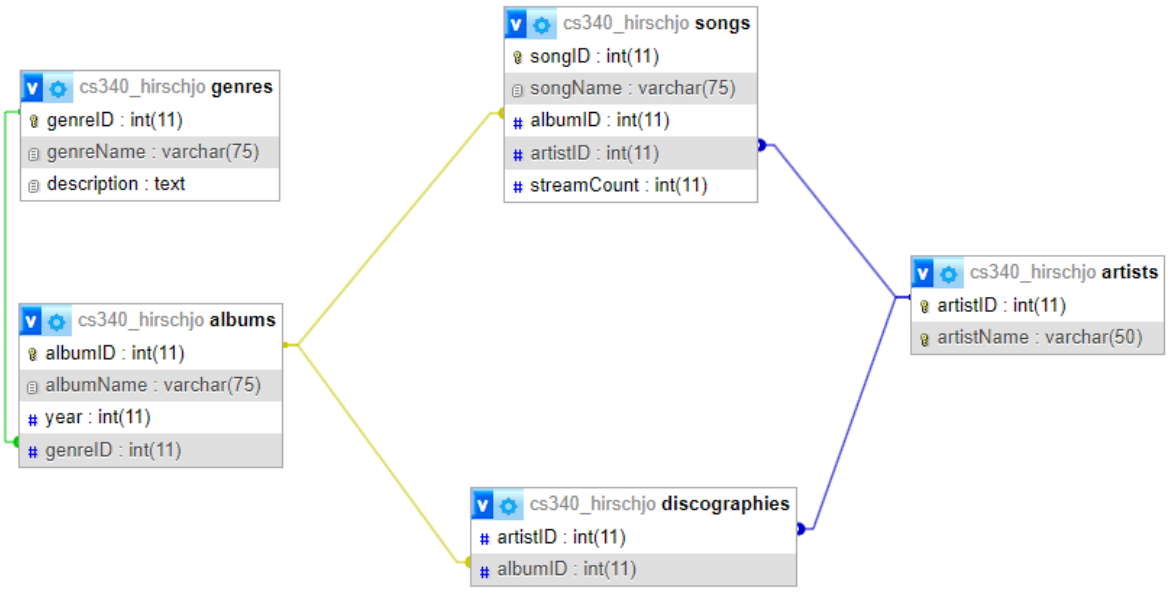
Holds data pertaining to genre for specific albums, albums can have multiple genres and subgenres, so it's important to hold data pertaining to that information.
Attributes:

- genreID: int, auto_increment, unique, not NULL, PK
- genreName: varchar, not NULL
- description: text
- albumID: int, FK referencing album table.
- Relationship:
  - 1:M (Album) - 1 genre can be used for multiple albums

ER Diagram


Songs
PK
songID
Genres
PK
genreID
Artists
PK
artistID
Albums
PK
albumID
Discographies
FK
artistID
FK
albumID

Schema



Sample Data

Songs				
songID	songName	albumID	artistID	streamCount
1	No Church In The Wild	1	1	8
2	Rich Flex	2	5	1
3	Me, Myself, and I	3	3	7
4	No Opp Left Behind	4	4	6
5	Champion	5	2	15

Albums				
albumID	albumName	artistID	genreID	year
1	Watch The Throne	1, 2	1	2011
2	Her Loss	4, 5	1	2016
3	Dangerously In Love	3	2	2003
4	SAVAGE MODE II	4	4	2020
5	Graduation	2	1	2007

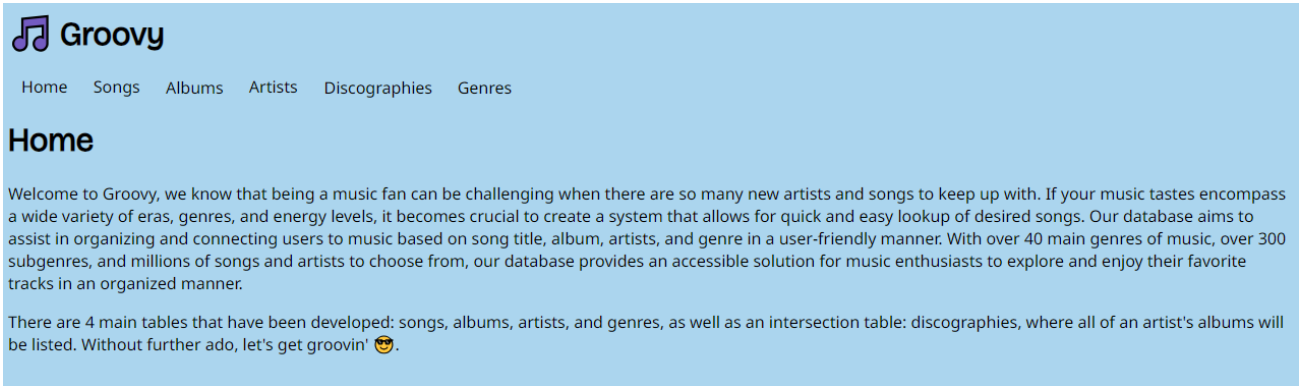
Artist	
artistID	artistName
1	Jay-Z
2	Kanye West
3	Beyonce
4	21 Savage
5	Drake

Discographies	
artistID	albumID
1	1
2	1

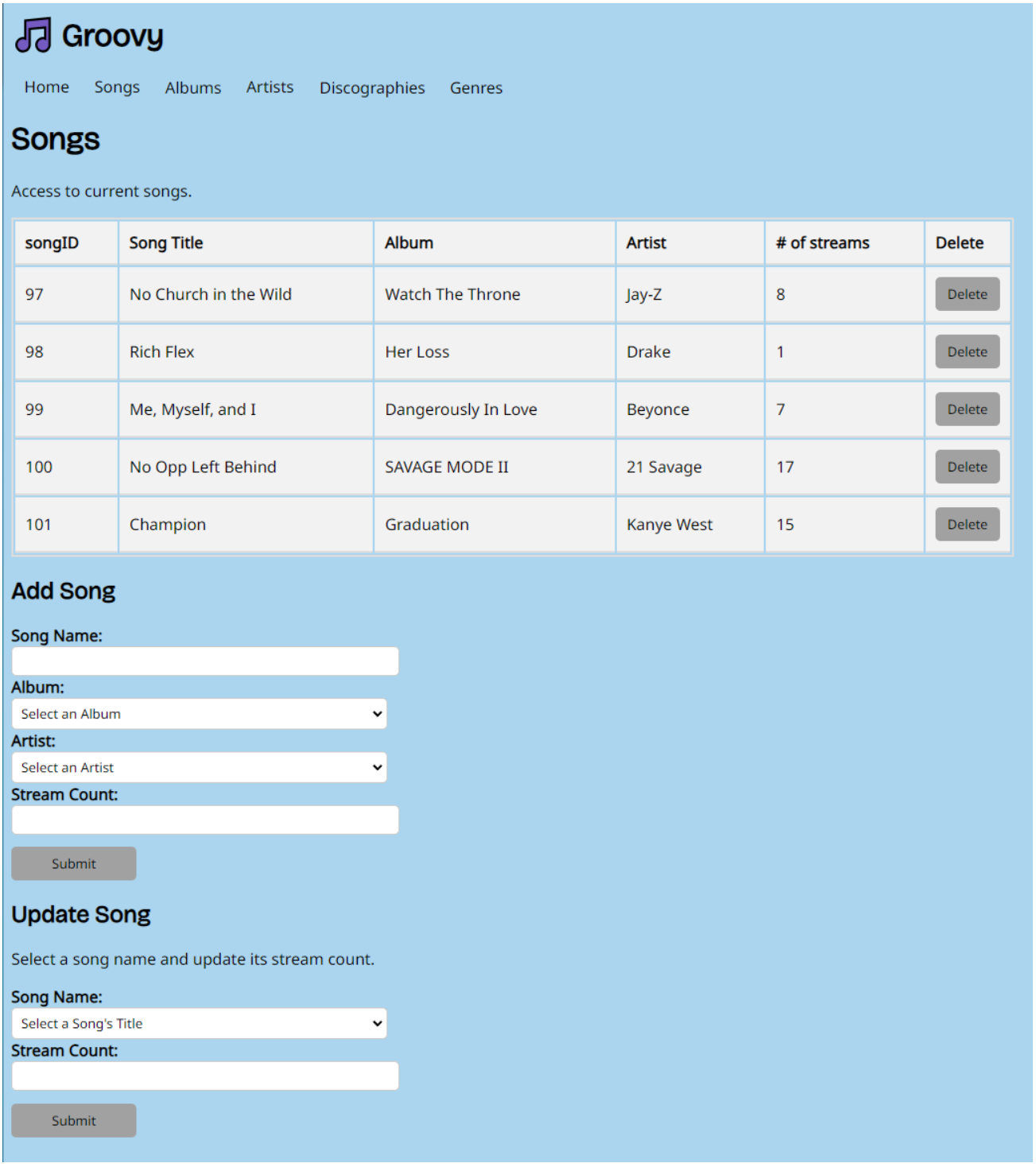
2	5
3	3
4	2
4	4
5	2

Genres		
genreID	genreName	description
1	Hip Hop	originated in the Bronx borough of New York City in the early 1970s
2	Rap	musical form of vocal delivery that incorporates "rhyme, rhythmic speech, and street vernacular"
3	Contemporary Rap	rap infused with content relevant to rap's “mature” listening audience
4	Trap	uses synthesized drums and is characterized by complex hi-hat patterns
5	Alternative Rap	genre of hip hop music that encompasses a wide range of styles that are not typically identified as mainstream

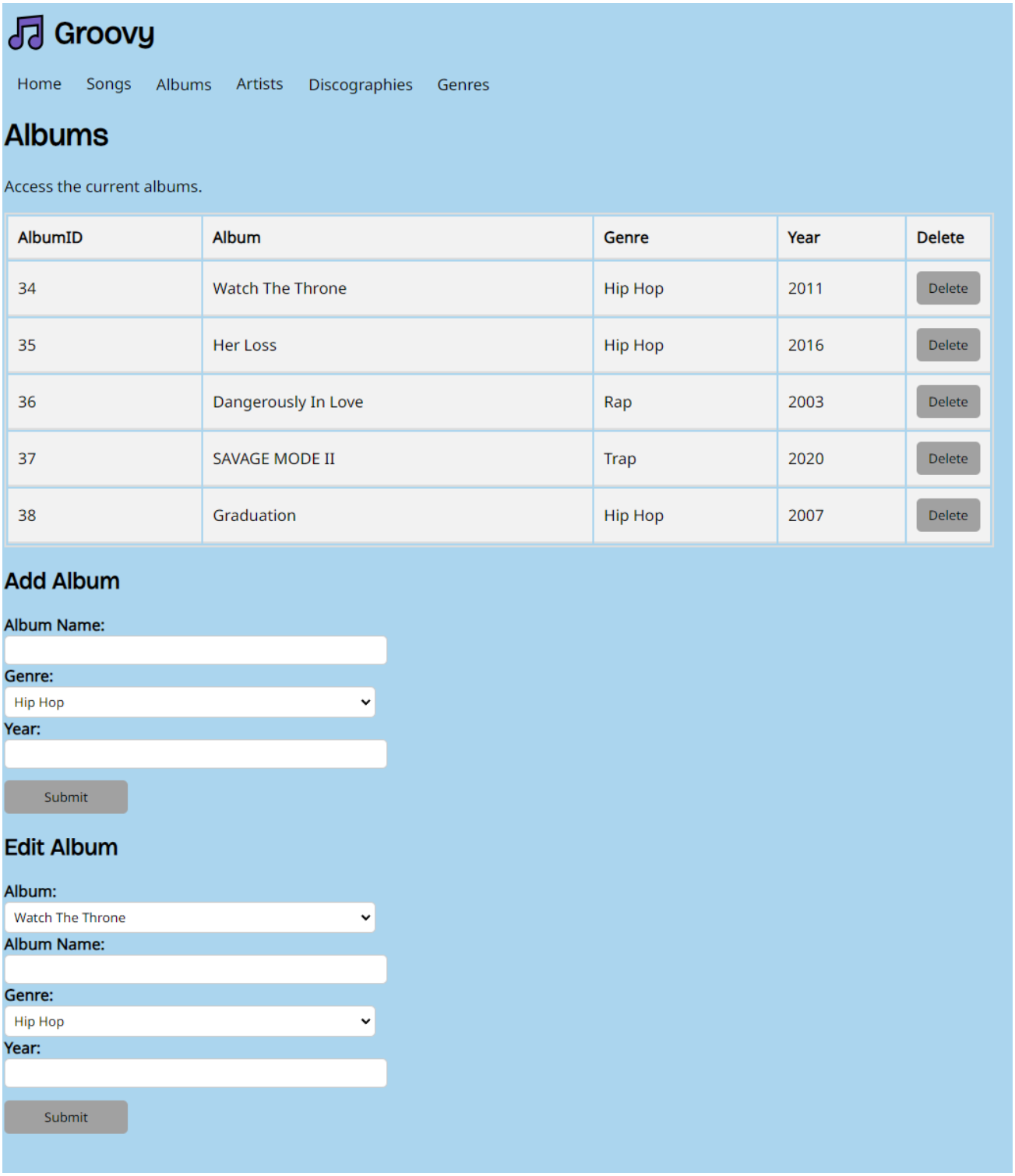
UI Screen Shots
Home Page:



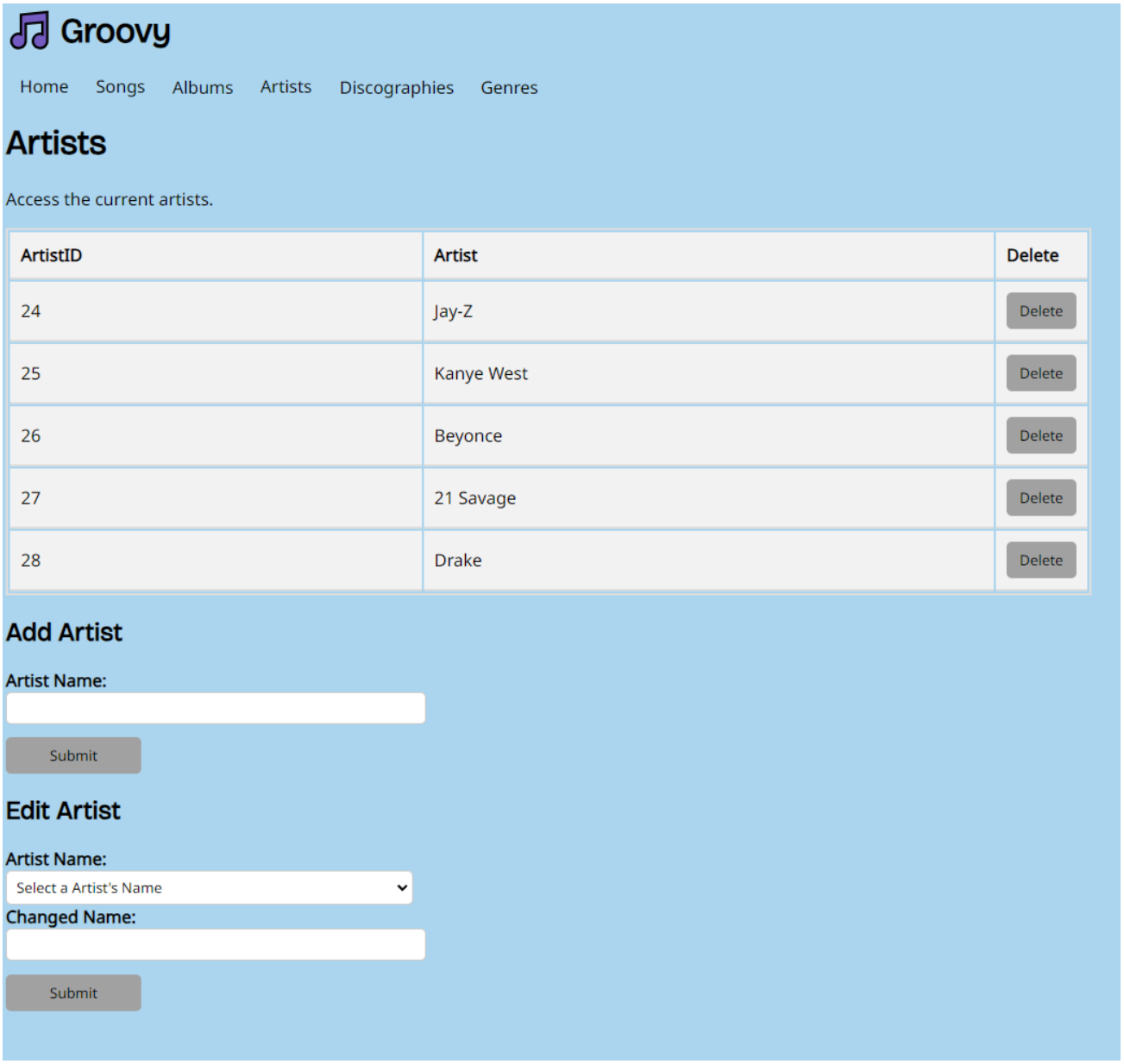
CREATE/READ/UPDATE/DELETE Songs Page:



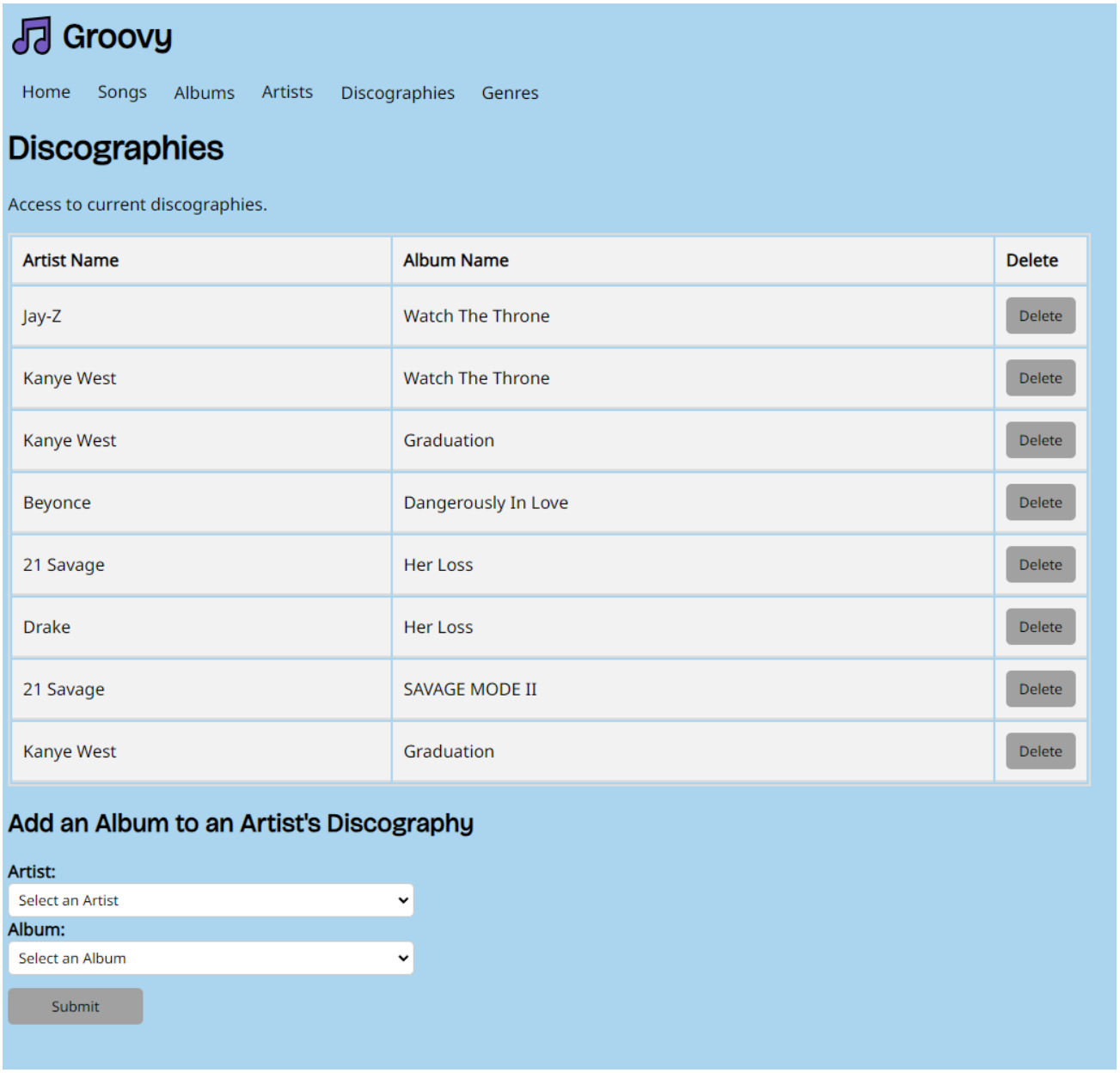
CREATE/READ/UPDATE/DELETE Albums Page:



CREATE/READ/UPDATE/DELETE Artists Page:



CREATE/READ/DELETE Discographies Page:



CREATE/READ/UPDATE/DELETE Genres Page:

Groovy

Home Songs Albums Artists Discographies Genres

Genres

Access to current genres.

Genre ID	Genre Name	Description	Delete
29	Hip Hop	Originated in the Bronx borough of New York City in the early 1970s	Delete
30	Rap	Musical form of vocal delivery that incorporates "rhyme, rhythmic speech, and street vernacular"	Delete
31	Trap	Uses synthesized drums and is characterized by complex hi-hat patterns	Delete
32	Alternative Rap	Genre of hip hop music that encompasses a wide range of styles that are not typically identified as mainstream	Delete

Add Genre

Genre Name:

Description:

Submit

Update Genre's Description

Select a genre and update its description.

Genre Name:

Select a Genre's Name

Description:

Submit