

# SPOTIFY SONG DATABASE

Deniz Çetin 28896  
İbrahim Ege Oral 29299  
Süda Şeker 28325

## Project Goal:

The original goal of our project was to create a system that would keep track of a user's music listening habits in order to recommend songs based on their playlists. However, we decided to pivot and focus on building a database that could store detailed information about various songs, albums, and artists. This database would allow users to search for and select songs for their playlists using various filters.

The project now consists of an interface that allows the user to manage and query this database of music information through a main page with buttons for adding and viewing records. This interface is intended to be used with the music streaming service Spotify.

### Main page:

Welcome to Spotify song database

What do you want to see?

Songs Longer Than 5 Minutes   Blues Songs   Artists from Turkey

What do you want to add?

Albums   Artists   Belongs   Creates Song   Creates Playlist   Podcast Episodes   Genres   Has   Is in Album   Is in Program   Playlists

Plays   Producers   Produces   Podcast Programs   Publishes   Releases   Shares   Songs   Users

What do you want to delete?

songID	song_name	duration
1	New Born	363
2	Bliss	251
3	blues song	331
4	asdf	231
5	Plug in Baby	218

4   5

2   DELETE with Song ID

In addition to the existing features, we added a button that allows the user to delete a song from the database using the song's ID. This gives the user the ability to remove any desired song from the database.

## SELECTION

When the user clicks on one of the buttons in the "What do you want to see?" section, they are taken to a new page that displays the results of the relevant query in a table format. This section contains three buttons: "Songs Longer Than 5 Minutes", "Blues Songs", and "Artists from Turkey". These buttons allow the user to access the results of specific queries on the database by clicking on them. Each button corresponds to a specific query that is run on the database when it is clicked.



- The "Songs Longer Than 5 Minutes" button runs the following SELECT query:

```
SELECT * FROM songs WHERE duration > 300
```

Song Name	Duration
New Born	363
blues song	331

This query selects all records from the "songs" table where the value of the "duration" attribute is greater than 300 seconds (5 minutes). The query returns the records that match this condition, and they are displayed in the table on the new page.

- The "Blues Songs" button runs the following SELECT query:

```
SELECT * FROM songs S, belongs B, genres G  
WHERE G.genre_name = 'blues' and G.genreID = B.genreID and B.songID = S.songID
```

Song Name
blues song

This query selects all records from the "songs" table that belong to the "blues" genre. It does this by first joining the "songs" table with the "belongs" and "genres" tables on the "songID" and "genreID" attributes, respectively. Then, it filters the resulting records by selecting only those where the "genre\_name" attribute in the "genres" table is equal to "blues". The query returns the records that match this condition, and they are displayed in the table on the new page.

- The "Artists from Turkey" button runs the following SELECT query:

```
SELECT * FROM Artists WHERE country = 'Turkey'
```

Artist Name
Mor ve Otesi

This query selects all records from the "Artists" table where the value of the "country" attribute is equal to "Turkey". The query returns the records that match this condition, and they are displayed in the table on the new page.

## INSERTION

The main page also has lots of buttons in the "What do you want to add?" section. When one of these buttons is clicked, the user is directed to a new page where they can enter data for the corresponding table in the database.

### *Artist insertion page:*

The screenshot shows a web application interface. At the top, there is a light blue rectangular form containing four input fields with placeholder text: "Type artist name", "Type artist ID", "Type country of the artist", and "Briefly describe who the arti". Below the form is a green "INSERT" button. To the right of the form is a table with four columns: "artist\_name", "artistID", "country", and "about". The table has two rows. The first row contains the values "Mor ve Ötesi", "1", "Turkey", and "they are awesome". The second row contains the values "Muse", "2", "England", and "harika".

artist_name	artistID	country	about
Mor ve Ötesi	1	Turkey	they are awesome
Muse	2	England	harika

The "Artists" button leads to a page with a form that allows the user to enter data for a new artist. The form includes fields for the artist's name, ID, country, and a brief description. When the user submits the form, the following INSERT query is run on the database:

```
INSERT INTO artists (artist_name, artistID, country, about)
VALUES ('$artist_name', '$artistID', '$country', '$about')
```

**Album insertion page (Albums button):**

The screenshot shows a user interface for inserting an album. At the top, there is a light blue rectangular form with rounded corners. It contains three input fields with placeholder text: "Type album name", "Type albumID", and "Type album duration". Below these fields is a green rectangular button labeled "INSERT". The background of the entire interface is a dark space-themed image with numerous small white stars.

albumID	album_name	duration
1	Origin of Symmetry	3306

The "Albums" button leads to a page with a form that allows the user to enter data for a new album. The form includes fields for the album's name, ID, and duration. When the user submits the form, the following INSERT query is run on the database:

```
INSERT INTO albums (album_name, albumID, duration)  
VALUES ('$album_name', '$albumID', '$duration')
```

**Song insertion page (Songs button):**

The image shows a user interface for inserting a song. At the top, there is a light blue rounded rectangle containing three input fields: "Type song name", "Type song ID", and "Type duration of the song(in)". Below these is a green rectangular button labeled "INSERT". To the right of this form is a table with five rows, each representing a song entry. The columns are labeled "songID", "song\_name", and "duration". The data in the table is as follows:

songID	song_name	duration
1	New Born	363
2	Bliss	251
3	blues song	331
4	asdf	231
5	Plug in Baby	218

**Album-Song relation insertion page (Is in Album button):**

songID	song_name	duration	albumID	album_name	duration	songID	albumID
1	New Born	363				1	1
2	Bliss	251				2	1
3	blues song	331	1	Origin of Symmetry	3306		
4	asdf	231				5	1
5	Plug in Baby	218					

We can see which song is in which album based on the 3rd table(songId | albumID), same goes for other insertion pages, hence, we can determine which is which.

**Song-Artist relation insertion page (Creates Song button):**

songID	song_name	duration	artist_name	artistID	country	about	songID	artistID
1	New Born	363	Mor ve Ötesi	1	Turkey	they are awesome	1	2
2	Bliss	251	Muse	2	England	harika	2	2
3	blues song	331					5	2
4	asdf	231						
5	Plug in Baby	218						

**Album-Artist relation insertion page (Releases button):**

artist_name	artistID	country	about	albumID	album_name	duration	artistID	albumID	release_date
Mor ve Ötesi	1	Turkey	they are awesome	1	Origin of Symmetry	3306	2	1	2001-06-18
Muse	2	England	harika						

**Genre insertion page (Genres button):**

The screenshot shows a mobile application interface. At the top, there is a light blue rounded rectangular box containing two input fields and a green button. The first input field is labeled "Type genre" and the second is labeled "Type genre ID". Below these is a green button with the text "INSERT". In the bottom right corner of the screen, there is a small table with two rows and two columns.

Genre_ID	Genre_Name
1	Rock
2	blues

**Song-Genre relation insertion page (Belongs button):**



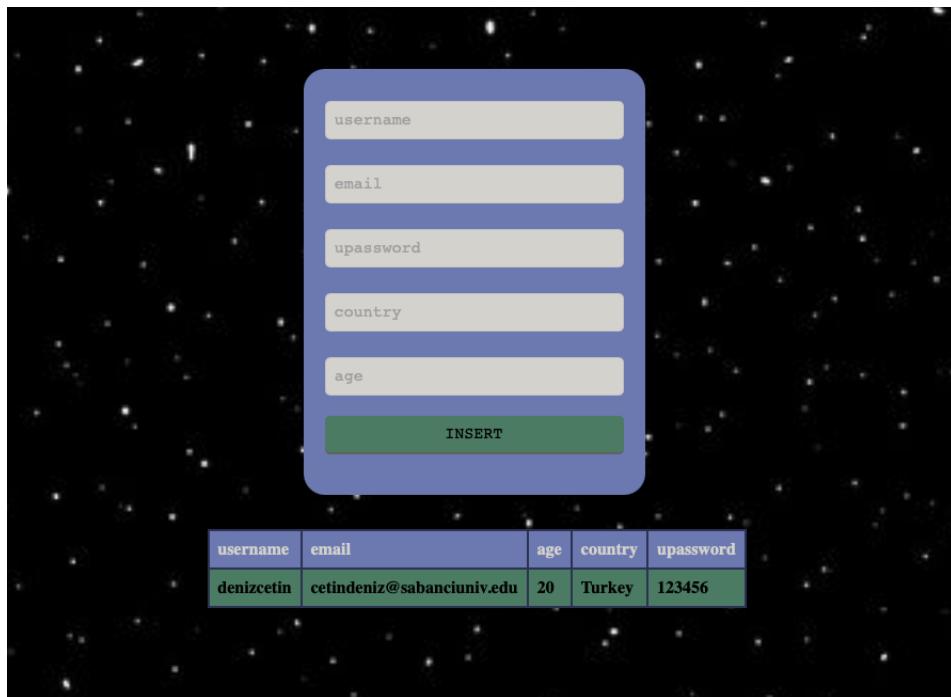
A screenshot of a mobile application interface. At the top, there is a light blue rounded rectangular input field with a white background and a thin gray border. Inside the field, the placeholder text "Type song ID" is centered. Below this is another similar input field with the placeholder "Type genre ID". At the bottom of the screen is a large green rectangular button with the word "INSERT" written in white capital letters.

songID	song_name	duration
1	New Born	363
2	Bliss	251
3	blues song	331
4	asdf	231
5	Plug in Baby	218

Genre_ID	Genre_Name
1	Rock
2	blues

songID	genreID
1	1
2	1
3	2
5	1

**User insertion page (Users button):**



A screenshot of a mobile application interface. It features a vertical stack of five input fields, each with a white background and a thin gray border. The first field is labeled "username", the second "email", the third "upassword", the fourth "country", and the fifth "age". Below these input fields is a large green rectangular button with the word "INSERT" in white capital letters.

username	email	age	country	upassword
denizcetin	cetindeniz@sabanciuniv.edu	20	Turkey	123456

**Playlist insertion page (Playlists button):**

The screenshot shows a mobile application interface. At the top, there is a navigation bar with icons for back, forward, and search. Below the navigation bar is a header labeled "Playlists". The main content area contains a form for inserting a playlist:

- A text input field with placeholder text "Type playlist name".
- A text input field with placeholder text "Type playlist ID".
- A text input field with placeholder text "Type song count".
- A green "INSERT" button at the bottom of the form.

Below the form, a table displays the inserted data:

pl_ID	pl_name	song_count
1	myPlaylist	3

**User Creates a Playlist relation insertion page (Creates Playlist button):**

The screenshot shows a mobile application interface. At the top, there is a navigation bar with icons for back, forward, and search. Below the navigation bar is a header labeled "Creates Playlist". The main content area contains a form for creating a user-playlist relation:

- A text input field with placeholder text "Type username".
- A text input field with placeholder text "Type playlist ID".
- A green "INSERT" button at the bottom of the form.

Below the form, a table displays the created data:

username	email	age	country	upassword	pl_ID	pl_name	song_count	username	pl_ID
denizcetin	cetindeniz@sabanciuniv.edu	20	Turkey	123456	1	myPlaylist	3	denizcetin	1

**Song-Playlist relation insertion page (Has button):**

The screenshot shows a mobile application interface. At the top, there is a table with columns: songID, song\_name, duration, pl\_ID, pl\_name, and song\_count. The data in the table is as follows:

songID	song_name	duration	pl_ID	pl_name	song_count
1	New Born	363			
2	Bliss	251			
3	blues song	331	1	myPlaylist	3
4	asdf	231			
5	Plug in Baby	218			

Below the table is a blue rounded rectangle containing three input fields and a green "INSERT" button. The first input field is labeled "Type playlist ID". The second input field is labeled "Type song ID". The third input field is empty. The "INSERT" button is green with white text.

**User Plays Playlist relation insertion page (Plays button):**

The screenshot shows a mobile application interface. At the top, there is a table with columns: username, email, age, country, upassword, pl\_ID, pl\_name, song\_count, username, and pl\_ID. The data in the table is as follows:

username	email	age	country	upassword	pl_ID	pl_name	song_count	username	pl_ID
denizcetin	cetindeniz@sabanciuniv.edu	20	Turkey	123456	1	myPlaylist	3	denizcetin	1

Below the table is a blue rounded rectangle containing three input fields and a green "INSERT" button. The first input field is labeled "Type username". The second input field is labeled "Type playlist ID". The third input field is empty. The "INSERT" button is green with white text.

***Podcast Program insertion page (Podcast Programs button):***

The screenshot shows a mobile-style interface for inserting a podcast program. It features a light blue rounded rectangle containing three input fields and a green "INSERT" button. The first field is labeled "Type program name", the second "Type program ID", and the third "Type episode count". Below this form is a table with three columns: "programID", "program\_name", and "episode\_count". The table contains one row with values 1, myProgram, and 3.

programID	program_name	episode_count
1	myProgram	3

***Podcast Episode insertion page (Podcast Episodes button):***

The screenshot shows a mobile-style interface for inserting a podcast episode. It features a light blue rounded rectangle containing three input fields and a green "INSERT" button. The first field is labeled "Type episode name", the second "Type episode ID", and the third "Type duration of the episode". Below this form is a table with three columns: "episodeID", "episode\_name", and "duration\_e". The table contains three rows with values 1, myEpisode1, 123; 2, myEpisode2, 124; and 3, myEpisode3, 125.

episodeID	episode_name	duration_e
1	myEpisode1	123
2	myEpisode2	124
3	myEpisode3	125

***Episode-Program relation insertion page (Is in Program button):***

Type program ID

Type episode ID

INSERT

			episodeID	episode_name	duration_e
programID	program_name	episode_count			
1	myProgram	3	1	myEpisode1	123
			2	myEpisode2	124
			3	myEpisode3	125

***User-Program relation insertion page (Publishes button):***

Type username

Type program ID

INSERT

username	email	age	country	upassword	programID	program_name	episode_count	programID	username
denizcetin	cetindeniz@sabanciuniv.edu	20	Turkey	123456	1	myProgram	3	1	denizcetin

**User-Episode relation insertion page (Shares button):**

Type username

Type episode ID

Type share date

INSERT

			episodeID	episode_name	duration_e				
username	email	age	country	upassword			episodeID	username	share_date
denizcetin	cetindeniz@sabanciuniv.edu	20	Turkey	123456	1	myEpisode1	123		
					2	myEpisode2	124	2	denizcetin
					3	myEpisode3	125		2022-12-21

**Producer insertion page (Producers button):**

Type producer name

Type producer ID

INSERT

producerID	producer_name
1	myProducer

### **Album-Producer relation insertion page (Produces button):**

Type producer ID  
Type album ID  
INSERT

producerID	producer_name	albumID	album_name	duration	producerID	albumID
1	myProducer	1	Origin of Symmetry	3306	1	1

## **DELETION**

If the 'ids' parameter is present, the script will execute a DELETE statement on the 'songs' table, using the provided ID as a WHERE clause to specify which record should be deleted.

`DELETE FROM songs WHERE songID = $selection_id`

*Main page's delete section before deletion:*

What do you want to delete?

songID	song_name	duration
1	New Born	363
2	Bliss	251
3	blues song	331
4	asdf	231
5	✓ 1 2 3 4 5	218

1

DELETE with Song ID

*After deletion:*

The screenshot shows a table of songs with columns: songID, song\_name, and duration. The table has five rows. A modal dialog box titled "What do you want to delete?" is overlaid on the page, containing a list of song IDs (2, 3, 4, 5) and a "DELETE with Song ID" button.

songID	song_name	duration
2	Bliss	251
3	blues song	331
4	asdf	231
5	Plug in Baby	218

2      DELETE with Song ID

Overall, this website allows users to view and modify the database of songs and artists by executing various SQL queries based on the user's input.

In the end, we were able to achieve the initial goals for this step of the project. We implemented a web-based administration panel for managing our website, which allows us to perform various operations on the database using web forms, including insertion, deletion, and selection (using filters applied through web forms that generate SQL statements to list relevant entries from the database on the browser). In addition to the administration panel, we also implemented other specified parts of the project according to the initial project specifications. This included making the website functional and enhancing its appearance with HTML and CSS (though we plan to continue improving the design before the final submission). Finally, we added features that covered at least the number of tables equal to the number of group members.