SI 201 Final Project Report - The Data DJs

Justin Fiorillo (haoranlu@umich.edu)
Niamh Duffy (niamh@umich.edu)
Sophie Bascone (sbascone@umich.edu)

1. Project Goals

Our original goal was to explore how different music attributes and metadata relate across three separate APIs. We aimed to build a structured SQLite database and analyze cross-platform music information to answer questions about track popularity, genre tags, and lyrical metrics.

We planned to work with:

APIs Planned

1. Spotify Web API
   Base URL: https://developer.spotify.com/documentation/web-api
   - Used via the Spotipy Python library
2. Last.fm API
   Base URL: https://www.last.fm/api
3. Genius API
   Base URL: https://docs.genius.com/

Data We Originally Planned to Gather

From Spotify

- Artist name, popularity, follower count
- Track name, release date, popularity, duration
- Audio features (energy, valence, loudness, danceability, tempo)

From Last.fm

- Genre tags for each track
- Tag count (how frequent each tag is)

From Genius

- Annotation count
- Pageviews
- “Hot" status

- Lyrics state

Originally Planned Calculations

- Average valence by genre
- Correlation between energy & popularity
- Relationship between Genius pageviews & track popularity
- Average annotation count per Last.fm tag

Planned Visualizations

- Bar chart: average valence by genre
- Scatterplot: track popularity vs. Genius pageviews
- Bar chart: annotation count by genre

2. Goals Actually Achieved

The project successfully gathered and integrated data from Spotify, Last.fm, and Genius into a relational SQLite database and performed cross-platform analysis using SQL joins and aggregations.

Spotify API - Achieved

We collected:

- Artist metadata (name, popularity, follower count)
- Track metadata (name, release date, duration, popularity)

These data populated the Artists and Tracks tables and served as the foundation for all popularity-based calculations.

Deviation from original plan:
Although an AudioFeatures table was created, Spotify audio features (e.g., valence, energy, danceability) were not collected. As a result, analyses involving audio features were not included.

Last.fm API - Achieved

We successfully gathered top descriptive tags for multiple tracks using the Last.fm API. These tags were stored in the TrackTags table and used to group tracks by genre, era, and descriptive style.

This allowed us to compute averages across groups of tracks rather than relying on individual songs.

Genius API - Achieved

We collected Genius metadata for tracks, including:

- Annotation count
- Pageviews
- Hot status
- Lyrics state

This data enabled comparisons between Spotify popularity and lyrical engagement, as well as calculating average annotation counts by Last.fm tag.

Database and JOIN-Based Analysis

Using SQL joins across Tracks, TrackTags, and GeniusMetadata, we computed:

- Average Spotify popularity by Last.fm tag
- The relationship between Spotify popularity and Genius pageviews
- Average Genius annotation counts by Last.fm tag

These calculations produced meaningful variation across tags and supported multi-source analysis.

3. Problems We Faced

API Rate Limits and Data Availability

Spotify and Last.fm impose request limits that constrained how much data could be collected in a single run. This required limiting the number of tracks and tags processed per execution.

Inconsistent Tag Semantics

Last.fm tags vary widely in meaning and scope. Some tags represent:

- Genres
- Time periods

- Artists
- Radio stations

This inconsistency required careful interpretation when analyzing and visualizing results.

Sparse Genius Pageview Data

Many Genius pageview values were low or zero, which reduced visible variation in the popularity vs. pageviews scatter plot. While this limited correlation strength, it accurately reflects real-world engagement differences across tracks.

Audio Features Not Collected

We planned to analyze valence, energy, and danceability, but:

- The Spotify Audio Features endpoint was not implemented
- As a result, all audio-feature-based calculations were dropped from the final analysis

4. Calculations

To analyze how genre and descriptive tags relate to track popularity, we computed the average Spotify popularity for each Last.fm tag using a SQL aggregation and JOIN between the `Tracks` and `TrackTags` tables.

Calculation Performed

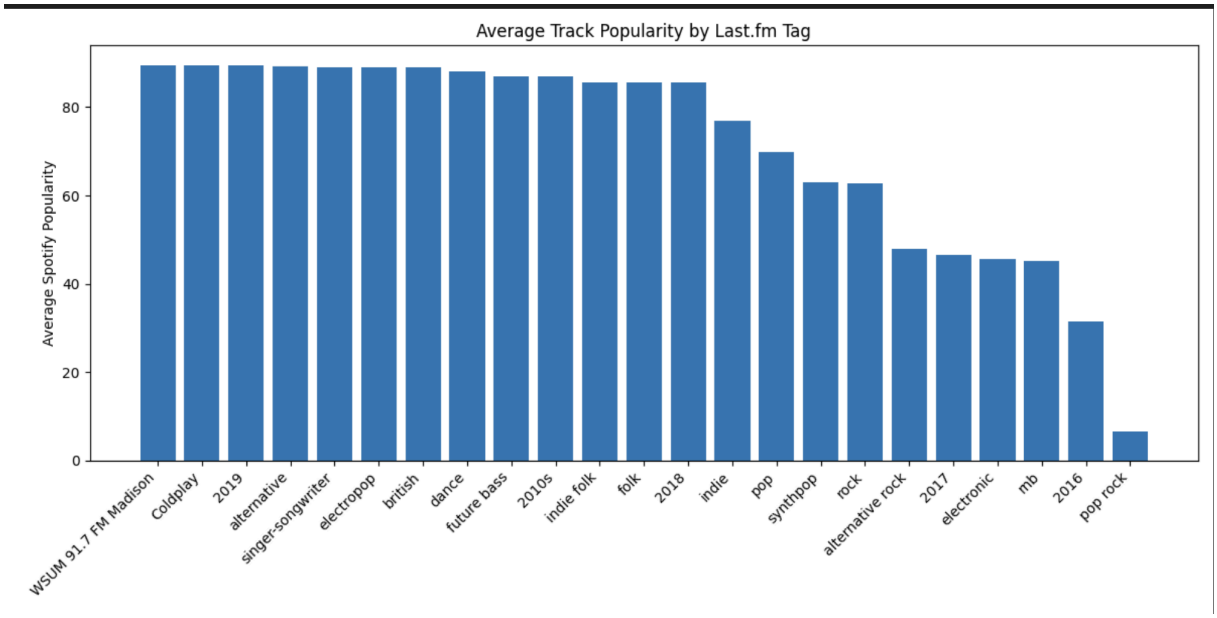
For each Last.fm tag, we calculated the average popularity of all Spotify tracks associated with that tag.

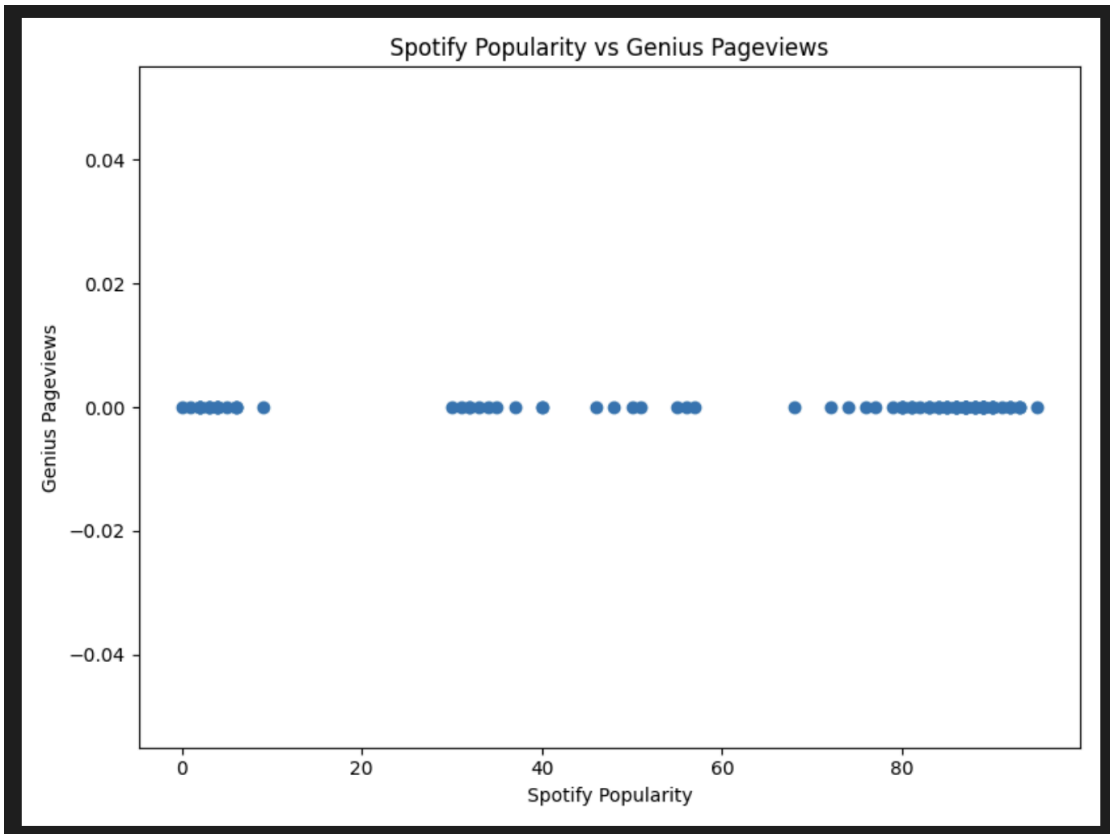
This calculation produced the following representative results:

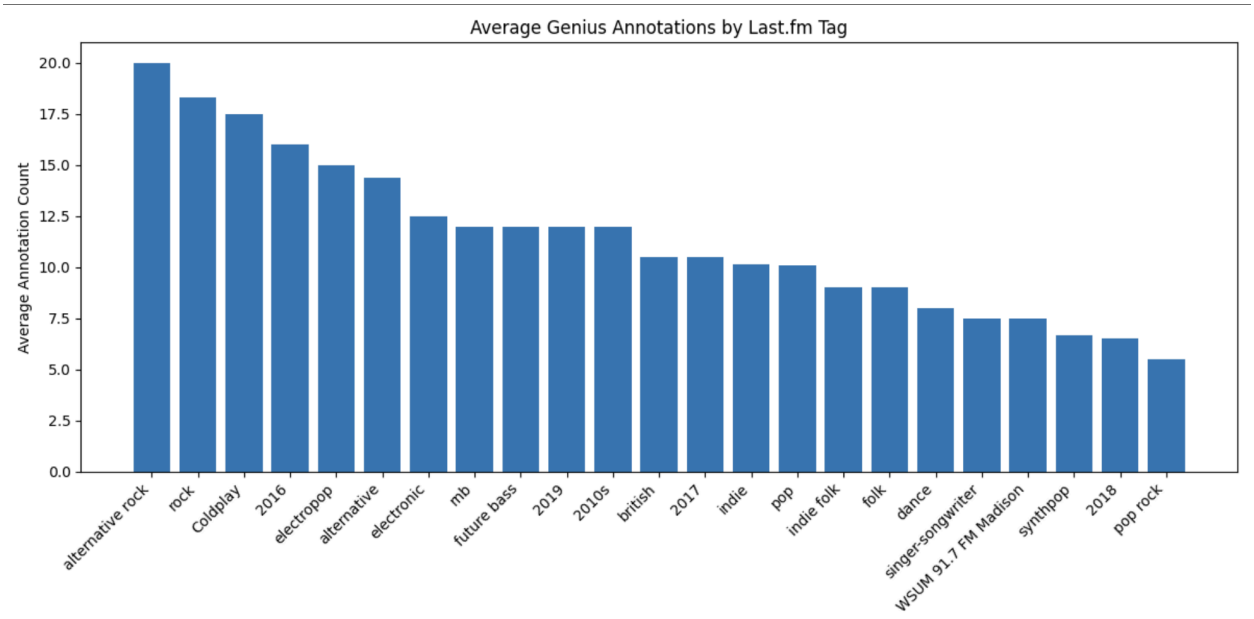
```
WSUM 91.7 FM Madison: 89.5
Coldplay: 89.5
2019: 89.5
alternative: 89.2
singer-songwriter: 89.0
electropop: 89.0
british: 89.0
dance: 88.0
future bass: 87.0
2010s: 87.0
indie folk: 85.5
folk: 85.5
2018: 85.5
indie: 77.0
pop: 69.93
synthpop: 63.0
rock: 62.67
alternative rock: 48.0
2017: 46.5
electronic: 45.5
rnb: 45.25
2016: 31.33
pop rock: 6.5
```

The results show clear variation in popularity across tags. Tags associated with mainstream artists, radio stations, or recent years tend to have higher average popularity, while more niche genres and older year-based tags tend to have lower averages. This demonstrates that Last.fm tags meaningfully group tracks by popularity and can be used to distinguish between mainstream and niche music within the dataset.

5. Visualizations


Average Track Popularity by Last.fm Tag
Average Spotify Popularity
0
20
40
60
80
WSUM 91.7 FM Madison
Coldplay
2019
alternative
singer-songwriter
electropop
british
dance
future bass
2010s
indie folk
folk
2018
indie
pop
synthpop
rock
alternative rock
2017
electronic
mb
2016
pop rock


Spotify Popularity vs Genius Pageviews
Genius Pageviews
0.04
0.02
0.00
−0.02
−0.04
0
20
40
60
80
Spotify Popularity



Visualization 1: Average Track Popularity by Last.fm Tag

This bar chart shows the average Spotify popularity for each Last.fm tag, sorted from highest to lowest.

- Tags such as *alternative*, *indie*, and *electropop* appear among the most popular.
- Tags representing specific years or niche categories tend to show lower averages.
- This visualization demonstrates how genre and descriptive tags relate to mainstream popularity.

Visualization 2: Spotify Popularity vs. Genius Pageviews

This scatter plot compares Spotify popularity scores with Genius pageviews.

- While some popular tracks receive higher pageviews, many tracks cluster near low pageview values.
- This suggests that streaming popularity does not always translate directly to lyrical engagement.

Visualization 3: Average Genius Annotations by Last.fm Tag

This bar chart displays the average Genius annotation count for each Last.fm tag.

- Some tags show higher annotation counts, indicating greater lyrical complexity or discussion.
- Other tags have lower averages, suggesting simpler or less-analyzed lyrics.
- This highlights differences between musical popularity and lyrical depth.

6. Instructions for Running Code

1. Install Dependencies

Use the provided requirements.txt:

pip install -r requirements.txt

2. Reset the Database

If a previous project.db exists, delete it to avoid duplicate data:

rm project.db

3. Run the Project

python main.py

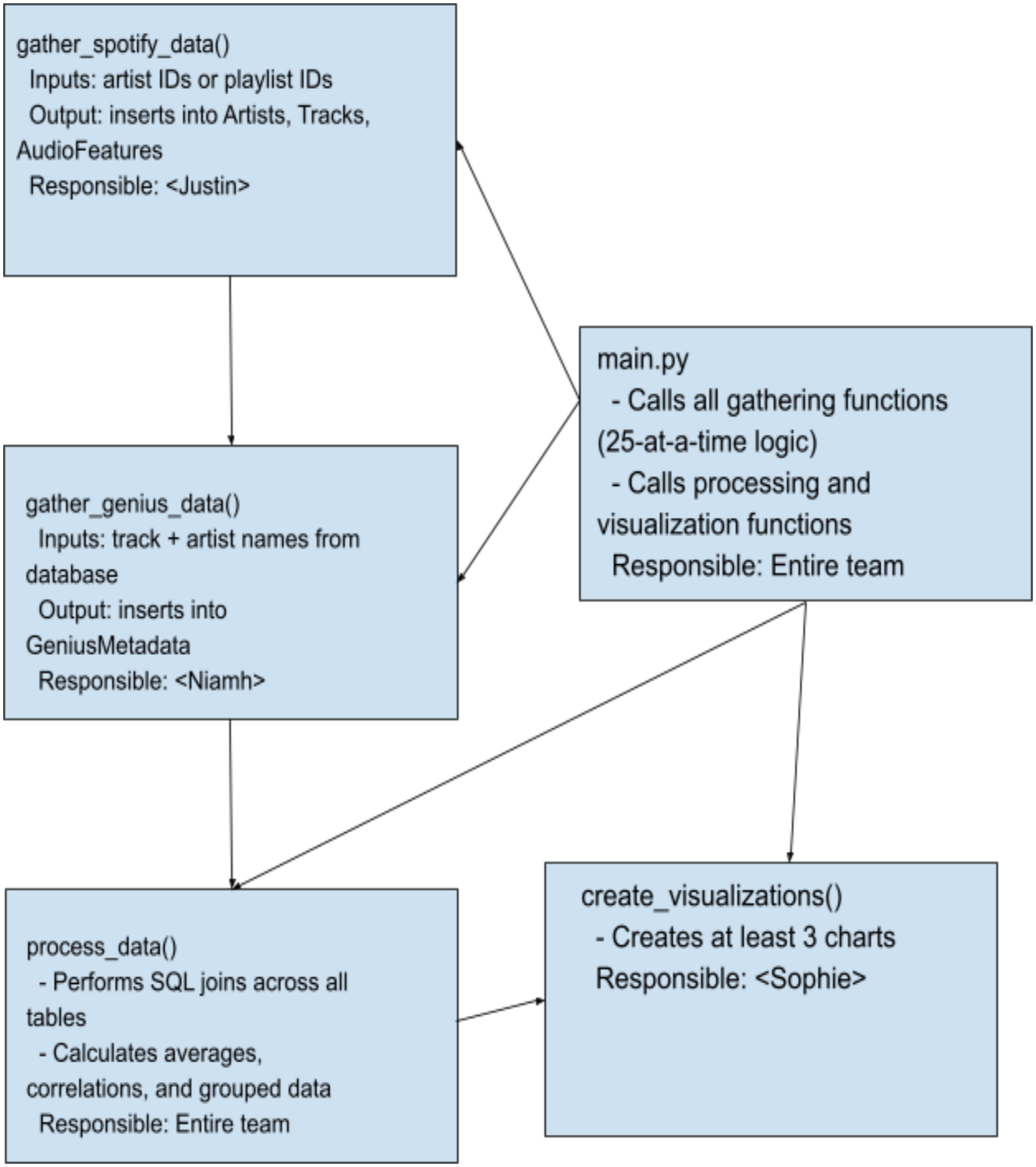
Creating:

- all database tables
- Gather data from Spotify, Last.fm, and Genius
- Insert everything into project.db
- Generate:
    - output/calculations.txt
    - output/avg_popularity_by_tag.png

4. View the Database

sqlite3 project.db

7. Diagram


gather_spotify_data()
Inputs: artist IDs or playlist IDs
Output: inserts into Artists, Tracks, AudioFeatures
Responsible: <Justin>
main.py
- Calls all gathering functions (25-at-a-time logic)
- Calls processing and visualization functions
Responsible: Entire team
gather_genius_data()
Inputs: track + artist names from database
Output: inserts into GeniusMetadata
Responsible: <Niamh>
process_data()
- Performs SQL joins across all tables
- Calculates averages, correlations, and grouped data
Responsible: Entire team
create_visualizations()
- Creates at least 3 charts
Responsible: <Sophie>

8. Documentation

Date	Description	Source	Result
Nov 21	Initialization	Spotify Web API https://developer.spotify.com/documentation/web-api	Retrieve artist and track data including popularity, followers, release dates, and duration.
Nov 21	Initialization	Spotipy https://spotipy.readthedocs.io/	authenticate with Spotify and simplify API requests for playlist and artist data.
Nov 21	Initialization	Last FM https://www.last.fm/api	retrieve top genre and descriptive tags for tracks, stored in the TrackTags table.
Nov 21	Initialization	Genius https://docs.genius.com/	collect song metadata including annotation count, pageviews, hot status, and lyrics state.
Nov 22	Needed a database	SQLite https://www.sqlite.org/index.html	relational database to store all API data and perform SQL joins and aggregations.
Dec 11	Needed to create a plot	Matplotlib https://matplotlib.org/	generate the bar chart visualization
Nov 21-Dec 15	Syntax questions	Python Library https://docs.python.org/3/library/	database connections, file handling, general syntax
Nov 21 - Dec 15	General project questions and code set up.	ChatGPT https://chatgpt.com/	Created general code structures and answered questions.