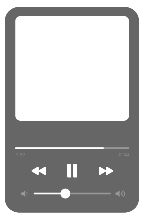
COM-480

Sound Sculptors

What's Inside



Introduction

Our project aims to provide a comprehensive understanding of music preferences around the world and explore the intricate relationships between different countries' musical tastes. Music serves as a universal language that transcends borders, cultures, and languages, connecting people on a deeply emotional level. We seek to uncover the threads that bind diverse peoples together through shared musical experiences.

This process book outlines the journey we took to achieve our final result, including the the evolution of our initial sketches and plans, the challenges we faced and the design decisions we made.

Path to the Final Result

Conceptualization

We began by brainstorming ideas for our project. We found a common interest which is music. We started looking for examples and we were inspired by existing visualizations such as the Musical Map of the World. We identified the key features we wanted to include, such as music taste similarities, top songs characteristics, music genres. The idea of including a visualization of related artists came to us later on.

Backend Development

Our backend served as the bridge between the Spotify API and our frontend. We implemented features to fetch and process data efficiently, ensuring smooth communication between the backend and frontend components.

Frontend Development

Using Svelte, CSS and D3.js, we designed a responsive and visually appealing frontend interface. We implemented interactive visualizations, allowing users to explore global music preferences and receive personalized recommendations based on their tastes, through the related artists visualization.

Testing and Optimization

We conducted thorough testing to identify and resolve any bugs or issues. We optimized performance to ensure seamless user experience, especially when handling real-time data updates and complex visualizations.

Documentation and Deployment

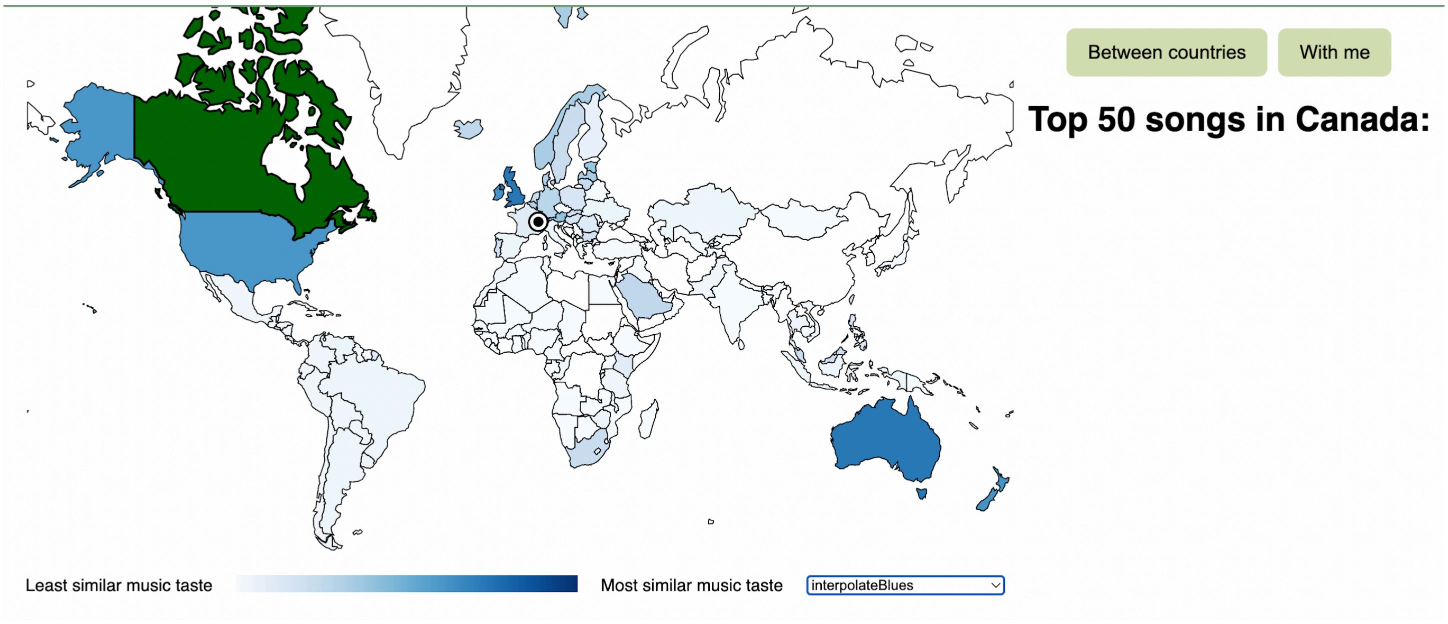
We created clear documentation in README for users on how to interact with our application. We deployed the application to “Netlify”, making it accessible to a wider audience. Our backend is hosted on “pythonanywhere”.

Evolution of Sketches and Plans

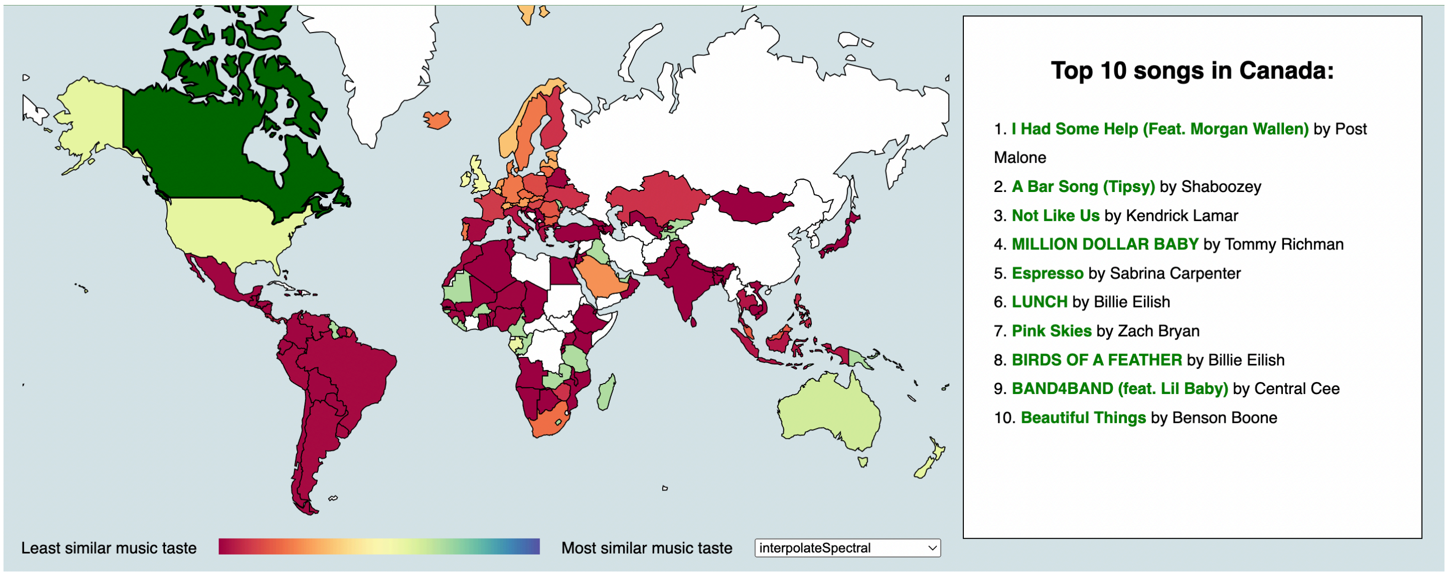
Music Taste Similarities

We expanded our initial sketch to include dynamic color changes based on similarity calculations. We enhanced the user interface to provide intuitive navigation and display real-time top songs for selected countries. Before clicking on a specific country, the user can see the current top 10 songs in the world. We also changed the background color and added a frame for the top 10 songs. The visualization is now more visually appealing and complete.

Before:



Now: (example)

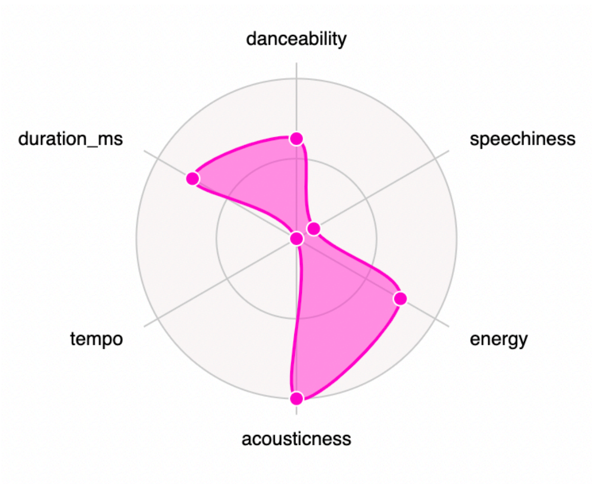
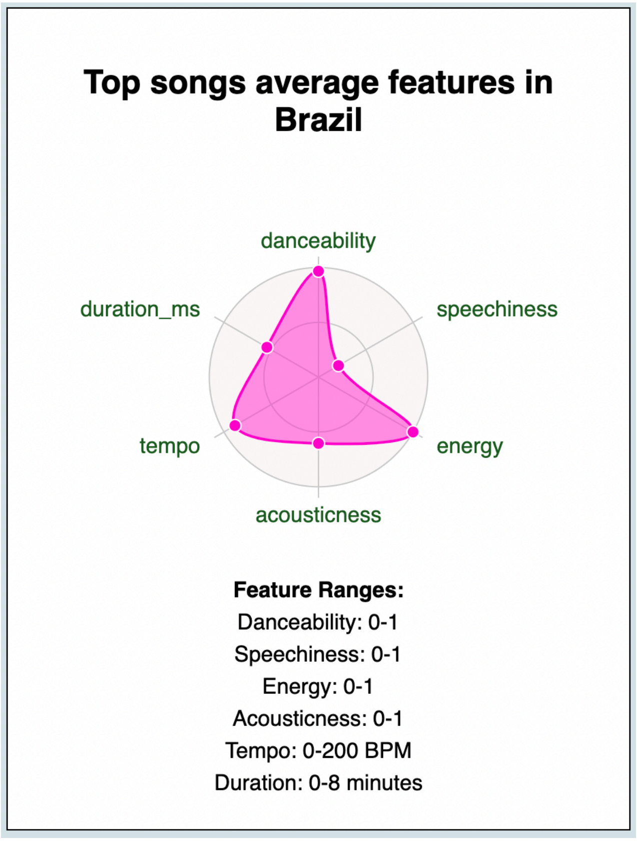


Top 10 songs in Canada:

1. **I Had Some Help (Feat. Morgan Wallen)** by Post Malone
2. **A Bar Song (Tipsy)** by Shaboozey
3. **Not Like Us** by Kendrick Lamar
4. **MILLION DOLLAR BABY** by Tommy Richman
5. **Espresso** by Sabrina Carpenter
6. **LUNCH** by Billie Eilish
7. **Pink Skies** by Zach Bryan
8. **BIRDS OF A FEATHER** by Billie Eilish
9. **BAND4BAND (feat. Lil Baby)** by Central Cee
10. **Beautiful Things** by Benson Boone

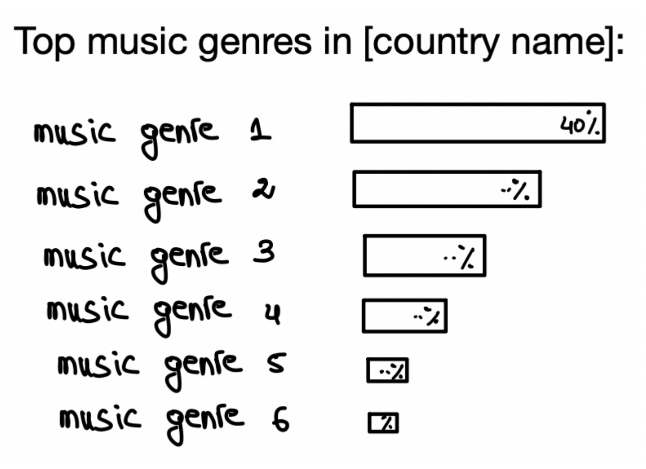
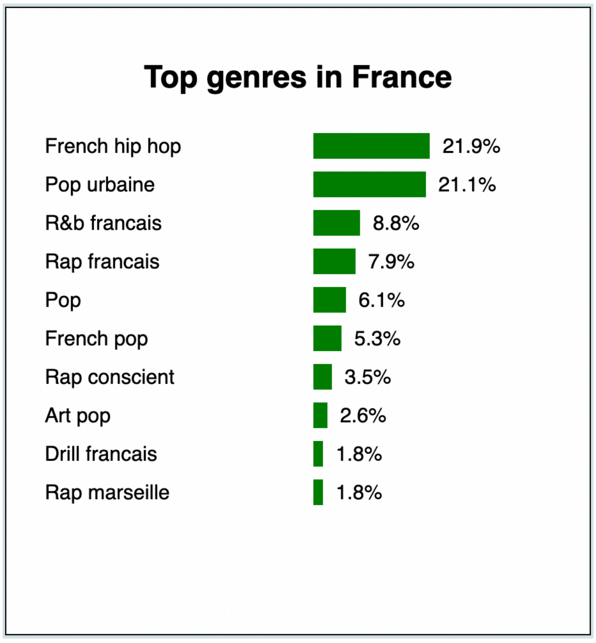
Top Songs Characteristics

Our radar graph visualization evolved to include interactive features, allowing users to explore characteristics of top songs in different countries. We refined the design to make it more visually appealing and informative by including the ranges of the different song features.

Before: (using dummy data)	**Now: (example)**
	 **Top songs average features in Brazil** **Feature Ranges:** Danceability: 0-1 Speechiness: 0-1 Energy: 0-1 Acousticness: 0-1 Tempo: 0-200 BPM Duration: 0-8 minutes

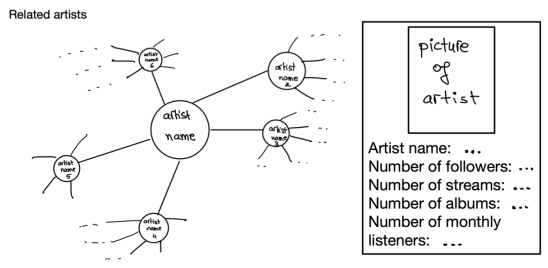
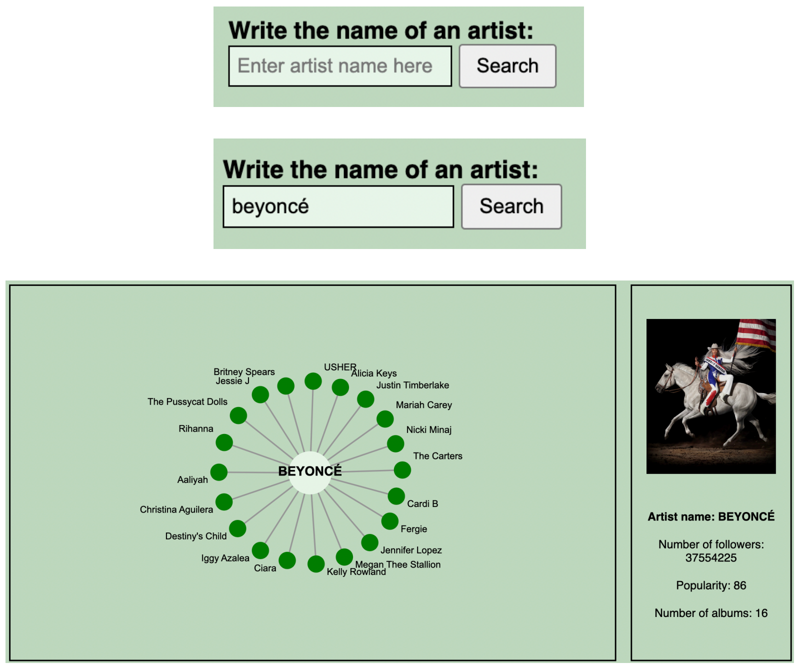
Music Genres

The idea of the Music Genres visualization is that when you click on a country, a bar chart will appear on the right of the map with the top genres in the selected country. In milestone 2 we put a sketch of this idea and now we implemented it so it became concrete!

Sketch:	**Example Visualization:**
 Top music genres in [country name]: music genre 1 40% music genre 2 ..% music genre 3 ..% music genre 4 ..% music genre 5 ..% music genre 6 %	

Related Artists

We expanded our initial plan for the related artists visualization to include interactive search functionality and detailed artist information. The design was optimized for ease of use and exploration. Once you type the name of your favorite artist, you get a graph of related artists. You can click on any of these nodes to have details of this related artist and a new graph. You can also enter the name of any other artist in the box. The goal of this visualization is to discover new artists that are similar to your favorite ones.

Sketch:	**Example Visualization:**
 Related artists artist name artist name 1 artist name 2 artist name 3 artist name 4 artist name 5 artist name 6 picture of artist Artist name: ... Number of followers: ... Number of streams: ... Number of albums: ... Number of monthly listeners: ...	 Write the name of an artist: Enter artist name here Search Write the name of an artist: beyoncé Search Artist name: BEYONCÉ Number of followers: 37554225 Popularity: 86 Number of albums: 16 BEYONCÉ Britney Spears Jessie J USHER Alicia Keys Justin Timberlake Mariah Carey Nicki Minaj The Carters The Pussycat Dolls Rihanna Aaliyah Cardi B Christina Aguilera Fergie Destiny's Child Jennifer Lopez Iggy Azalea Ciara Kelly Rowland Megan Thee Stallion

Challenges

Count and Duration of API Calls

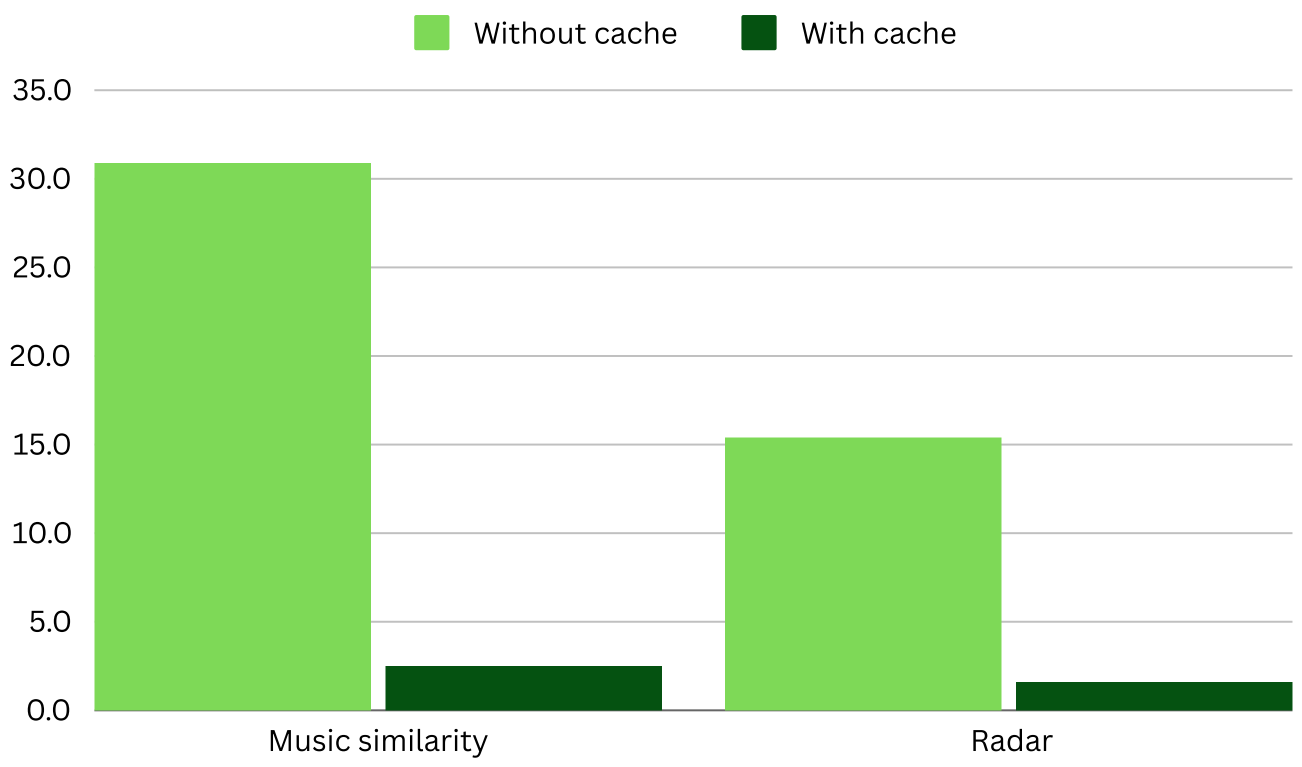
One of the main challenges we faced was managing the count and duration of API calls. The frequent and lengthy API calls to Spotify's service affected our ability to retrieve data quickly and efficiently. This issue was compounded by the rate limits imposed by the Spotify API, which restricts the number of calls we can make within a specific timeframe.

To mitigate this issue, we decided to implement a **caching mechanism**. This cache stores the top tracks for various countries locally, significantly reducing the number of API calls needed. By retrieving data from the cache instead of making frequent API calls, we were able to improve response times and ensure that our application remained efficient and responsive.

Another challenge is the limited data available to us. Spotify discontinued their charts API, and we had to bodge together a replacement, which manually searches for the charts data that Spotify publishes as playlists.

Response Time Before and After Caching

The bar chart compares the response times of two API endpoints (music_similarity and radar) before and after implementing caching. Where X-Axis represent Endpoints and Y-Axis represent Response Time in Seconds.



Technical Implementation

Backend

The backend of our project is a Python-based application that serves as the core engine for data retrieval, processing, and aggregation. It communicates directly with the Spotify API to gather music-related data and compute various statistics, which are then used to generate insights for the frontend visualizations.

Frontend

The frontend was created in Svelte, a modern JavaScript framework. Visualizations were created using D3.js.

Contributions

Alexander Müller

- Spotify Charts API replacement
- Map + Top 10 Songs
- Deployment
- Process book

Yasmine Chaker

- Radar Visualization
- Graph Visualization
- Top Genres Visualization
- CSS / Landing page
- Screen cast
- Process book

Tymur Tytarenko

- Backend
- Data processing
- Results Caching
- Spotify API integration
- Process book

🔗**Website link:** https://soundsculptors.netlify.app

💻**Github repository:** https://github.com/com-480-data-visualization/SoundSculptors

📽️**Screen cast link:** https://www.youtube.com/watch?v=uzzclIc422g