Spotify Stats - Discover your sound

University of CINCINNATI CEAS

CEAS EXPO

Abstract

Spotify Stats is a web application that will transform users' Spotify listening data into meaningful insights and personalized music recommendations. Through an engaging and innovative interface, users will explore two main features: a comprehensive genre distribution and an intelligent music recommendation system via a user comparison social aspect. The application will leverage Spotify's API and intelligent data analysis to analyze listening patterns and suggest new music that aligns with users' tastes. By combining music genre analysis with future recommendations in a visually distinctive package, Spotify Stats will offer music enthusiasts a unique platform for understanding their listening habits and discovering new artists.

Project Overview

By connecting to a user's Spotify account, our application provides both insight and discovery through two core features:

Data Visualization

Data visualizations are presented in a way to best articulate the insights discovered through our genre comparisons and intelligent data analysis used. We used a combination of polar area, spider, and cartesian planes to show insights along with direct playlist recommendations.

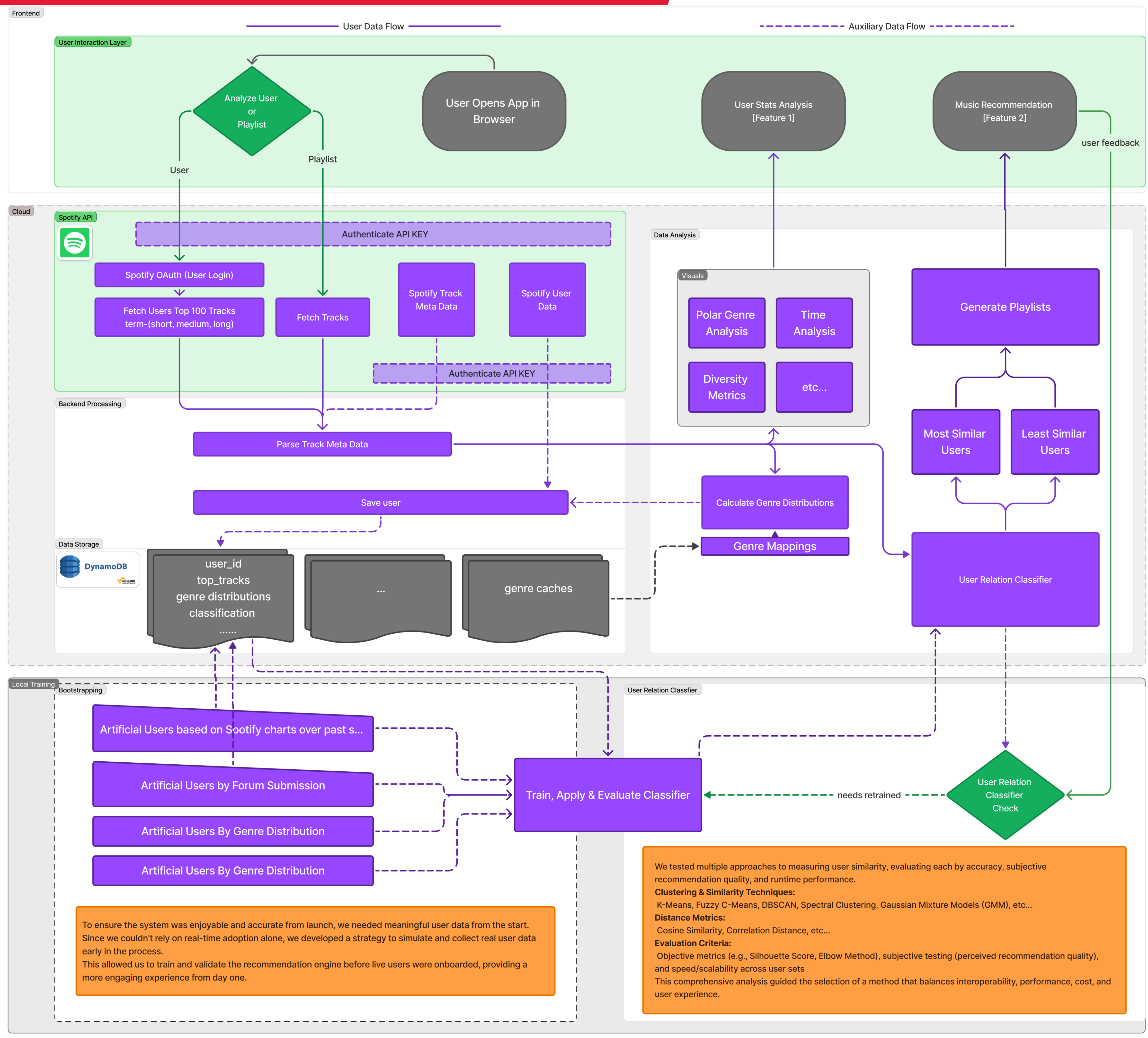
Playlist Generation

We analyze listening history across all users to identify patterns and similarities. By training a model that calculates user similarities, we can pinpoint the most and least similar listeners.

Using this similarity data, we **generate playlists** from:

- The **most similar users**, to discover new music of familiar taste
- The **least similar users**, to encourage exploration and discovery

The result is a **private**, **personalized**, yet **social** discovery experience, helping users find new music to love, and music they may never have heard before.



The Danger of 3rd Party APIs – "The Pivot"

Our project initially **planned to use Spotify's 13 track-level audio attributes** (e.g., danceability, energy, tempo) to train a machine learning model that could generate new playlists by requesting songs with similar characteristics. **However, midway through development, Spotify deprecated key parts of their API,** limiting access to the very data we relied on. This change, intended to protect their data from being used in external models, broke core parts of our approach.

This forced a major architectural shift:

- Adopted **user-based similarity** and **genre-driven** clustering
- Reduced API dependency with **local processing and caching**

A Blessing in Disguise

This led us to create a more social, user-centered experience.
System built around shared tastes rather than raw audio features, resulting in a more unique, gratifying, and personal platform

Key Design Considerations

1. **User-Centered Experience**
   - We prioritized recommendations that feel right over those that are mathematically perfect
   - Preprocessing and caching ensure a smooth, responsive experience without heavy real-time computation
2. **Performance & Cost Efficiency**
   - To reduce computation, we rely on lightweight classifiers, retraining only when deemed necessary
   - AWS services were chosen for affordability, modularity, and scalability
3. **Compliance and Security**
   - User data privacy and API usage guidelines were strictly followed, especially when interacting with Spotify's platform
   - System design considered minimal data retention and secure handling of user tokens and metadata

Technologies Used

- **AWS**
  - DynamoDB: Amazon's NoSQL storage service that allows us to efficiently store and receive data without constantly accessing user's Spotify data.
- **Python**
  - Flask: A lightweight web framework that handles HTTP requests, routing, and session management.
  - Spotipy: A Python library that provides easy access to the Spotify Web API for authentication and music data retrieval.
  - Boto3: The AWS SDK for Python that enables programmatic interaction with AWS services, particularly DynamoDB for this project.
- **Render**
  - Render for hosting the Python flask app

User Similarity & Recommendation Pipeline

Real Users or Playlists

The pipeline begins by gathering listening data from the Spotify API, and transforming it into structured insights about the users preferences.[Feature One]
This information is placed into user profiles, which are securely stored in AWS DynamoDB to support scalable access and comparison.
The data is then abstracted into *high-dimensional genre vectors* and passed through a similarity modeling system, where *user affinities are inferred, clustered, and used to drive tailored music discovery and recommendation experiences.*[Feature Two]

Artificial Training Data and Bootstrapping

To support early development and improve model robustness, we built a bootstrapping system that generates artificial user profiles based on historical Spotify chart data and custom genre patterns.
These synthetic users allow us to validate the system's behavior before deployment and ensure that recommendations remain high-quality—even when real user data is limited.




Seamus Collins Computer Science

Chris Laney Computer Science

Justin Tisch Computer Science

Noah Falanga Computer Science

Advisor: Prof William Hawkins III
Department: Computer Science