Project Name: Track Tales
Team name: Audible Algorithms
Team Members: Mia Patrikios, Lena Young, Shipra Patel

Section 1: Introduction

The developed project uses the Spotify API to organize a user's playlist into a data graphic. The visualization breaks down their listening data to show top songs by genre, artists, albums, and songs. Track Tales allows Spotify users to visualize and discover insights into their music preferences.

The motivation behind this project is to create a more user-friendly interface to view listening stats. There is a similar feature available to Spotify members called "Spotify Wrapped," but it is only available at the end of the year and it requires a premium subscription. Additionally, the insights it shows are very generic by only showing top 5 artists, most listened-to songs, etc. Track Tales would expand on the listener's stats, showing in-depth customized data visualizations. The user would be able to navigate through all of their data, rather than getting to see limited, generic insights.

In the context of the market, users would be able to share their stats with their friends and on social media. Additionally, it would not require them to wait until the end of the year, as they would be able to use the program anytime they wanted. People are always looking to expand their music interests and learn more about themselves. Track Tales would allow them to visualize the data in an intuitive way, and learn more about what kind of music they prefer.

Overall, we were able to complete all of our initial goals for this project, and further add more features, such as filtering by genre and adding the option to switch between a treemap and sunburst view.

Section 2: Customer Value *- Note: No changes from previous report*

The primary customers of Track Tales are younger Spotify users, roughly ages 10-25. This Software provides Spotify users with an insightful experience where they can visualize their music listening. Because music is such a self-expressive form of entertainment, by being able to see what kind of music they like the most, they will be able to learn more about themselves. By gaining these insights, it will help them find more music and explore similar artists.

The main issue this software intends to solve is the availability of music data visualization. Many music platforms offer yearly music insights, but they are not customized or available at any other time. Additionally, many of the existing solutions are more of a slideshow

view of the most generic stats, not allowing the user to delve into more niche data. To gauge user approval, we initially wanted each team member to test the software with their data, as well as other selected participants. However, due to issues with authentication, we were not able to test it with multiple logins. We chose to instead gauge our project's success based on the fact that we were able to meet all of our intended goals for this project. The software provided viewable data insights and improved user experience, as well as ran without noticeable bugs.

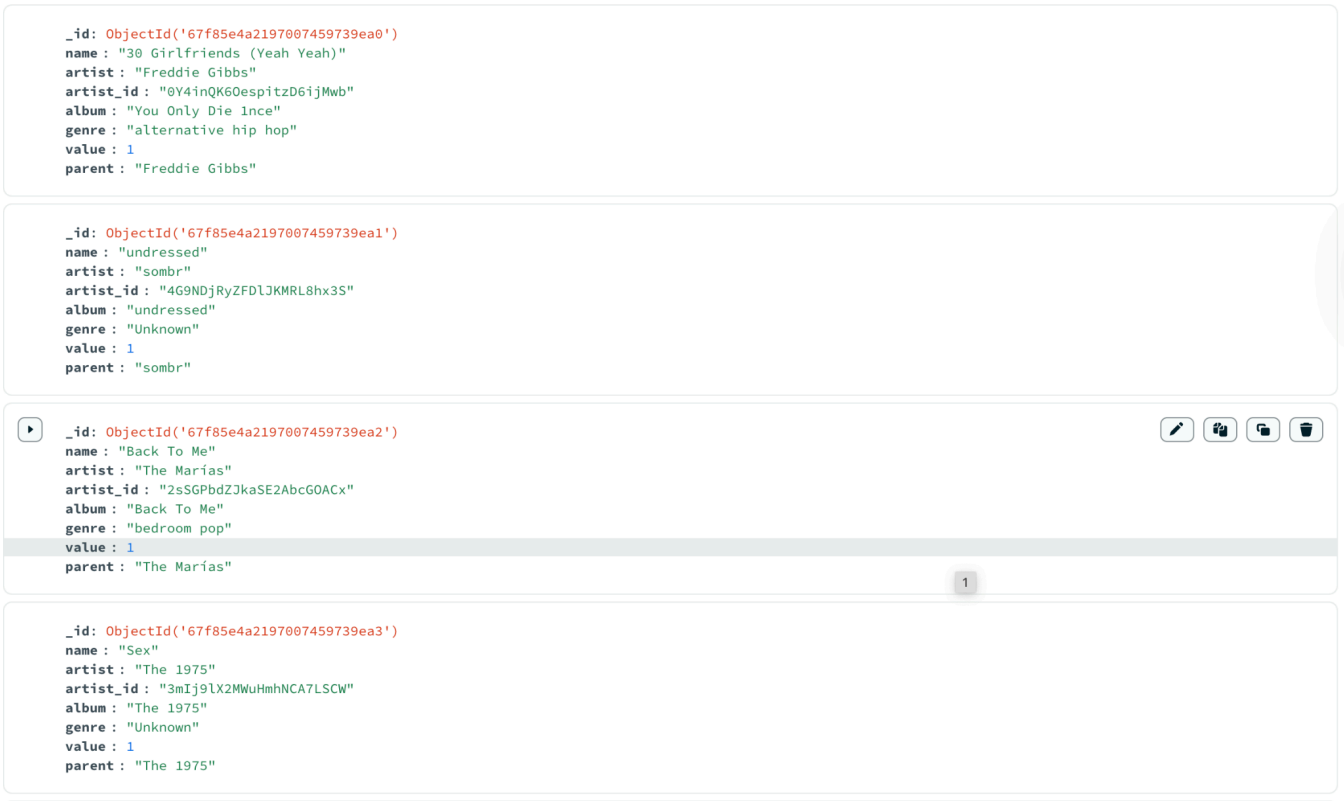
Section 3: Technology

The software uses Spotify's API to get a user's liked songs playlist. It first parses the JSON data and submits it to MongoDB to organize it. The data is used to visualize the user's liked songs broken down by genre, album, artist, and song. The main components of the system are API calls, MongoDB, and Plotly/Pandas for data visualization. The team used Python and Flask for the backend development, Spotify for the API calls, Pandas/Plotly for data visualization, and HTML/CSS for the front end.

Our implementation went beyond the the minimum value and even target versions. At the most basic level, it is capable of parsing the JSON data and providing the Mongo database their playlist breakdown. On target level, it features a visually appealing front end based on the category of the data graphic you are in.

During testing, we ran into issues with Spotify's API call rate limit. Unless we switched to a paid version of their API, we could not make the number of calls needed to get the genre of each artist. Thus, we limited the number of songs fetched to 100, and called the artist genres in batches of 20. This allows the app to run faster and not crash due to API limits. After this change, we also noticed the front end visualization looked better because there was not as much information to explain.

Database Document Example:



Section 4: Team

The contributions of each team member was virtually equivalent throughout the project. We chose to use static roles that played to each of our strengths and what each team member wanted to get out of working on the project. Mia acted primarily as the project manager, assigning sprints, helping troubleshoot any issues, and guiding the others through the software. Lena and Shipra acted as Software Developers, as they wrote the functions and figured out how the different technology components interacted with each other.

Section 5: Project Management

We were able to complete all of our goals for this project. We even had time to add more features, such as filtering by genre and adding the option to switch between a treemap and sunburst view. The spotify python API library and plotly's integration with pandas streamlined the progress of this project, and made development faster than we anticipated. The only problem we encountered during development was Spotify's API call limit, which was easily solved by reducing the number of calls we made. The only feature we considered adding that

we did not implement was giving the user music recommendations based on their liked songs. We later chose not to incorporate it as it did not feel like it addressed a relevant need, as Spotify already provides users with intuitive and curated music recommendations.

Section 6: Reflection

Overall, the project went very smoothly. The team dynamic worked well and played to the strength of each team member. Because Mia was more familiar with the softwares used in development, it made sense for her to act more as the project manager. Lena and Shipra then were able to learn from Mia and help contribute to writing the code. The team worked well together, with all members contributing equally throughout the project. The project was definitely a success and we gained valuable insights into the process of software development.