Final Project Report: What Spotify Knows About You

Jiayao Jin
Brown University

Zhinuo Wang
Brown University

Abstract

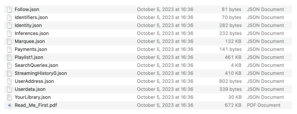
Nowadays, even though many users can retrieve their personal data from the applications they use, it's still really difficult for non-technical users to understand their data because many services like Spotify just return personal data in JSON format. To contribute to such a problem, this project implements a full-stack application to provide the visualization report for the users to understand their own Spotify data. While the front end gives a detailed explanation of each visualization and provides a neat layout, the back end aggregates the original songs' data with its features obtained from Spotify API so that more insights can be provided to the users. Such implementation allows users to register and log into their account in this application. Then simply by uploading the downloaded data zip file, the user can see their own visualized Spotify data report.

1 Introduction

According to the GDPR, the data subjects (typically users) should have the right to access their personal data saved or processed by the companies [1]. Nowadays, more and more companies are complying with GDPR and provide API or graphical interfaces for users to retrieve their personal data.

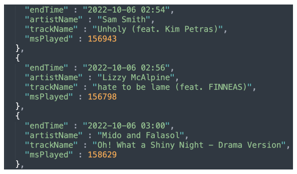
However, the data retrieved from different companies arrives in different formats. The personal data that users can obtain from Spotify is all in the form of JSON 1, like their streaming histories, search queries, and playlist information. Such data may be common and understandable for technical users who deal with JSON a lot, but normal users can hardly get anything useful from their personal data. Moreover, it is just a plain listing of all the raw data and provides nearly no insights for users to understand the impact of their data. In this sense, their GDPR rights to read seem not to be perfectly complied with.

Therefore, this project would like to help users understand their personal data in Spotify by utilizing data visualization techniques. In other words, their data will be displayed in an understandable way, and certain analyses will be provided so that they can learn more than the plain listing of their personal data. Moreover, inspired by the paper by Wei et al. [2], this project could also help the users learn about the impact of the data because the company could profile them and analyze them with the same data. As a result, they know what to expect of their data being processed by the companies, and thus can take actions accordingly.



Follow.json	October 5, 2023 at 16:36	81 bytes	JSON Document
Identifiers.json	October 5, 2023 at 16:36	70 bytes	JSON Document
Identity.json	October 5, 2023 at 16:36	282 bytes	JSON Document
Inferences.json	October 5, 2023 at 16:36	232 bytes	JSON Document
Marquee.json	October 5, 2023 at 16:36	132 KB	JSON Document
Payments.json	October 5, 2023 at 16:36	141 bytes	JSON Document
Playlist1.json	October 5, 2023 at 16:36	461 KB	JSON Document
SearchQueries.json	October 5, 2023 at 16:36	4 KB	JSON Document
StreamingHistory0.json	October 5, 2023 at 16:36	410 KB	JSON Document
UserAddress.json	October 5, 2023 at 16:36	802 bytes	JSON Document
Userdata.json	October 5, 2023 at 16:36	339 bytes	JSON Document
YourLibrary.json	October 5, 2023 at 16:36	30 KB	JSON Document
Read_Me_First.pdf	October 5, 2023 at 16:36	672 KB	PDF Document

Figure 1: All JSON files



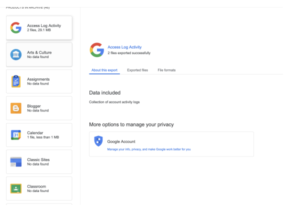
```
    "endTime" : "2022-10-06 02:54",
    "artistName" : "Sam Smith",
    "trackName" : "Unholy (feat. Kim Petras)",
    "msPlayed" : 156943
  },
  {
    "endTime" : "2022-10-06 02:56",
    "artistName" : "Lizzy McAlpine",
    "trackName" : "hate to be lame (feat. FINNEAS)",
    "msPlayed" : 156798
  },
  {
    "endTime" : "2022-10-06 03:00",
    "artistName" : "Mido and Falasol",
    "trackName" : "Oh! What a Shiny Night - Drama Version",
    "msPlayed" : 158629
  },
```

Figure 2: Sample content

2 Related Works

2.1 Personal Data Visualization

Though more and more companies are complying with the GDPR rights to enable people to retrieve their personal data, not all of them also provide a user-friendly way to visualize or demonstrate the data. Companies like Spotify only return the personal data in its raw form, containing only JSON files 1. Though it also includes a PDF that explains the meaning of each field in the JSON data files, it is difficult for non-technical users to understand and see their data. Companies like Instagram and Google not only return the raw data but also include HTML files that provide a clearer graphical interface for the user to see their data 2.1. However, this is just a plain list of a user's personal data and it provides no insights for the user. This project aims to solve this problem and provides meaningful visualizations to let people better understand the data they obtained from Spotify.



PRODUCTS IN ARCHIVE (46)

Access Log Activity
2 files, 29.1 MB

Arts & Culture
No data found

Assignments
No data found

Blogger
No data found

Calendar
1 file, less than 1 MB

Classic Sites
No data found

Classroom
No data found

Access Log Activity
2 files exported successfully

About this export | Exported files | File formats

Data included

Collection of account activity logs

More options to manage your privacy

Google Account
Manage your info, privacy, and make Google work better for you

Figure 3: Google display

Instagram

About your download

Your information can be found in the HTML files in this download. We recommend viewing your download on a desktop device in order to use these links to navigate to your information. The links don't work on mobile devices.

On a mobile device, use the folders in the download to go to the HTML files. Each type of information is stored in its own HTML file and can be viewed by opening the file.

Comments
Comments you've posted on your own content or on other people's content
Comments

Contacts
Contact information you've added for other people
Synced contacts

Your topics
A collection of topics determined by your activity on Instagram that is used to create recommendations for you in different areas of Instagram, such as Reels, feed recommendations, and Shopping
You have no data in this section

Messages
Messages you've exchanged with other people
Secret conversations
Chats

Guides
Guides you've created
You have no data in this section

Media Settings
Changes you've made to your media controls in settings
Comments allowed from
Use cross-app messaging

Information about you
Information that may be associated with your account

Figure 4: Instagram display

2.2 Visualization Techniques

As noted by Tamara Munzner, visualization plays a pivotal role in aiding users to efficiently understand data. Munzner emphasizes that computer-based visualization systems are designed to create visual representations of datasets, aiming to streamline task completion [3]. This understanding underscores the importance of research in visualization techniques, particularly in crafting visuals that are not only easily comprehensible but also insightful for non-technical users, enhancing their ability to interpret and interact with data effectively.

Shneiderman's paper outlines a key visualization design principle: "Overview first, zoom and filter, then details-on-demand [4]." Building on these fundamentals, Cario summarized the key features for creating easily interpretable visualizations, which include artistic embellishments for aesthetic appeal, a minimal approach to information presentation for clarity, a focus on presenting data from a single perspective, and the use of familiar chart formats to enhance user comprehension [5].

2.3 Spotify Web API and Emotion-Based Music Classification

Since one goal of this project is to provide insights into user's personal data, the data directly retrieved from Spotify is not enough. For example, there are many songs in the Playlist.json and StreamingHistory.json files, but they only include the basic information of the song like its ID and name. To enrich the data, this project applies Spotify Web API [6] to obtain richer information like a song's features so that further analysis could be made. According to Partick's paper on Emotion-Music Selection and Recommendation [7], the energy and valence features of a song can be used to classify the mood of the song in two dimensions. With such additional features obtained from the Spotify Web API, this project can provide users with more insights into their data.

3 Design

Since this project aims to provide an easy way for people to understand their personal data in Spotify and get more insights than just a plain listing of the raw data, it intends to create a web service so that a user can easily access it and

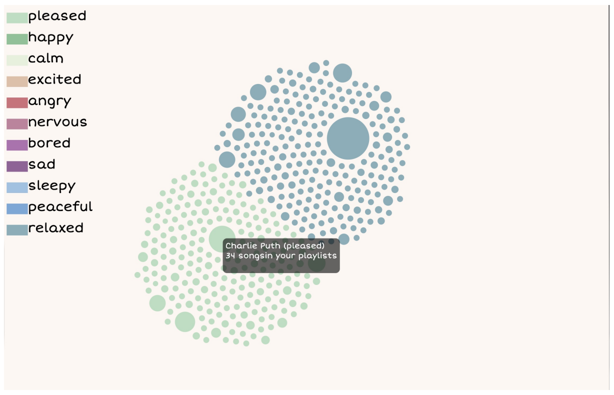


Figure 5: Force layout for artists

interact with the graphical interfaces instead of the raw data. The web service is divided into two parts: the front end and the back end.

3.1 The Frontend

3.1.1 General Layout and Interaction

As this project aims to produce a visualization report, the interaction design of the website is fairly easy, after users register and successfully log in, they just need to upload their downloaded general data zip file and download extensive streaming data zip to view their reports. These two zip files should be uploaded separately as Spotify only provides separate downloads for these two data. The report is divided into three parts: general information display, visualization of playlists as well as artists, and visualization of streaming history. After the user successfully uploads the files, the website will automatically generate the general information textual report for the users to see the data Spotify collected about their emails, addresses, and payment methods. Then the user can go to see their different visualization report by simply clicking the displayed buttons.

3.1.2 Visualization

As for the visualization, there are four charts for the two different visualizations in total. For the visualization of playlists and artists, the first is a force layout of artists' data 5. This visualization groups artists into distinct groups based on the emotional tone of their songs, with each artist represented by a circle. The size of each circle correlates with the frequency of the artist's songs in a user's playlists. Such a layout not only provides an overview of the types of music a user prefers but also offers deeper insights into their favorite artists, effectively blending quantitative data with qualitative emotional analysis. The second visualization is a radar chart to display how different playlists vary from each other in songs' features like danceability, energy level, and so on 6. Such comparison

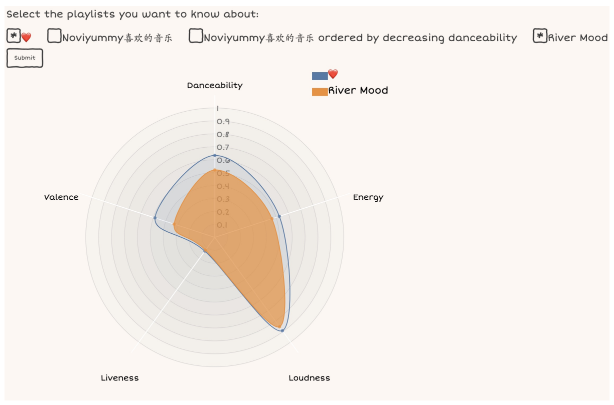

Select the playlists you want to know about:
❤️
Noviyummy喜欢的音乐
Noviyummy喜欢的音乐 ordered by decreasing danceability
River Mood
Submit
❤️
River Mood
Danceability
Energy
Loudness
Liveness
Valence

Figure 6: Radar chart with a filter

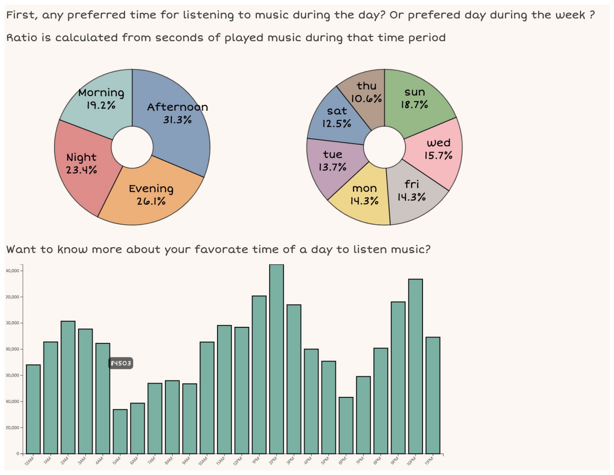

First, any preferred time for listening to music during the day? Or prefered day during the week ?
Ratio is calculated from seconds of played music during that time period
Morning 19.2%
Afternoon 31.3%
Night 23.4%
Evening 26.1%
thu 10.6%
sun 18.7%
sat 12.5%
wed 15.7%
tue 13.7%
mon 14.3%
fri 14.3%
Want to know more about your favorate time of a day to listen music?
24503

Figure 7: Pie charts and bar chart for listening habits

contributes to users' understanding of their playlist style better and potentially understanding their preferred music styles better(for example, from the similar feature values they found in different playlists).

The visualization of streaming history in this project employs pie charts and a bar chart to offer users a clearer understanding of their listening patterns 7. One pie chart illustrates the distribution of listening times across different parts of the day—morning, afternoon, evening, and night—while another pie chart displays how listening time is spread across the days of the week. Additionally, a bar chart is utilized to provide a more detailed view of preferred listening times, showing how the total listening duration fluctuates hour by hour throughout the day. This multifaceted approach allows users to gain a comprehensive insight into their music listening habits.

3.2 The Backend

The back-end part is needed for this project mainly for two reasons. First, since there is a lot of data to visualize and

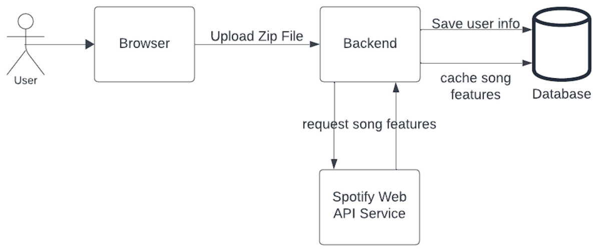


Figure 8: Back-end service structure

different data provides different insights, it is wise to separate the pages of the visualizations. Therefore, certain states and data need to be memorized in the back end. Second, to use the Spotify Web API, a client ID and secret need to be created using the Spotify developer account, which is used to request a bearer for the following API requests. Since all of those keys can not be exposed to the client side, a back-end service is needed to separate that information from the clients.

3.2.1 Structure

The main structure of the back-end service is shown in the figure 8. The user accesses the service from the browser and can upload their data zip file directly. Then, the back-end server will parse the zip file and take out the needed JSON files to process. The server will take out the data needed for visualization and save it to the database accordingly. Besides, it will also request the Spotify API for songs' features and cache that information in the database for future use. Finally, the user can get the visualizations by accessing different pages.

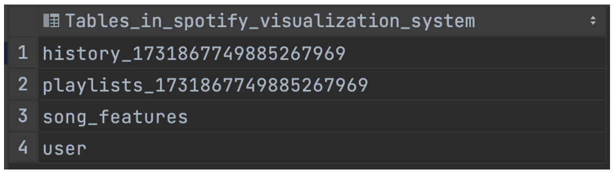
3.2.2 Database Design

There are four types of tables. First, the User table. It is a single table that holds the information of users using this service. It contains users' basic information obtained from the zip file they upload. A unique ID will also be generated for each user.

Second, the Playlist table. It saves all the songs of a user's playlists and will be created and overridden each time the user uploads its zip data file. Each user will have a separate playlist table, so the table is named "playlists_userId". This design is to ensure that when different users request their playlist data from the database, the requests can be processed concurrently because their data lies in different tables.

Third, the History table. It is similar to the Playlist table in that it is also per-user based, which is named "history_userId". It holds all the listening history of a certain user obtained from the extended listening history zip file.

Fourth, the Song Features table. This table is used to cache the song features obtained from the Spotify API. The idea is that if different users or even the same user have the same



	Tables_in_spotify_visualization_system
1	history_1731867749885267969
2	playlists_1731867749885267969
3	song_features
4	user

Figure 9: Sample database tables after a user has used

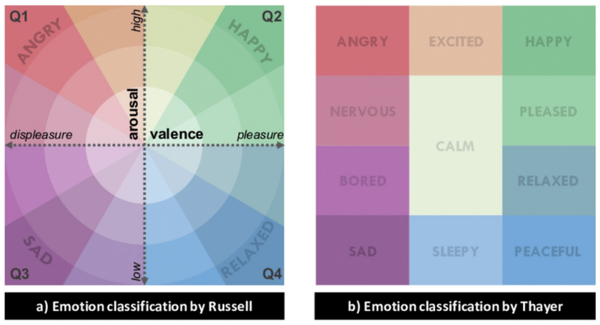


Figure 10: Human emotion classification model by Russell and by Thayer

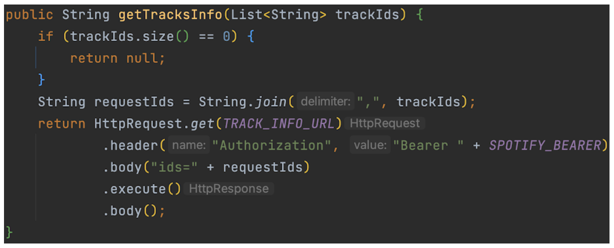
songs in their playlists, the song features should only be requested once from the Spotify API. This table increases the speed of the processing and reduces the requests needed to make to Spotify. A sample database after a user has uploaded his data zip file and extended streaming history zip file is shown in figure 9.

4 Implementation

4.1 Front-end

For a cohesive and visually appealing design, the website utilizes a CSS library named “Doodle CSS” to ensure consistency in style. The visualizations are created using SVG (Scalable Vector Graphics) to guarantee that they remain crisp and clear at any scale, avoiding any loss in quality. Furthermore, D3.js is employed for these visualizations due to its versatility and extensive customization options, enabling the creation of highly detailed and tailor-made visual designs. In total, it contains about 1590 lines of code.

As for the emotion classification of the artists, this project implemented the human emotion classification model by Russell and Thayer as Helmhloz suggests a direct implementation of these models in classifying music [7]. In this project, the *Math.atan2()* function is utilized to calculate the specific area for each artist on the visualization, based on their songs’ average energy and valence values according to the human emotion classification models proposed by Russell and Thayer10. Additionally, the color scheme chosen to represent various emotions is consistent with these models, ensuring that the visual representation accurately reflects the emotional spectrum



```
public String getTracksInfo(List<String> trackIds) {
    if (trackIds.size() == 0) {
        return null;
    }
    String requestIds = String.join( delimiter: ",", trackIds);
    return HttpRequest.get(TRACK_INFO_URL) HttpRequest
            .header( name: "Authorization",  value: "Bearer " + SPOTIFY_BEARER)
            .body("ids=" + requestIds)
            .execute() HttpResponse
            .body();
}
```

Figure 11: Batch requests with a list of songs’ id

as defined in the established psychological frameworks.

4.2 Back-end

The back-end part applies Java Spring Boot as the web framework to handle requests and responses, and uses MySQL as the database storage. It also applies libraries like MyBatis Plus, Hutool, and Jackson for faster development. In total, it contains about 1600 lines of Java code and about 200 lines of SQL code.

Several optimizations have been made when implementing the back-end service. First, creating separate tables and caching tables. As mentioned in section 3.2.2, the playlists table and the history table are created and maintained separately for each user. As a result, when different users request their playlist or streaming data from the database, their SQL statements can be executed concurrently since their data lies in different tables. It can also reduce the size of each table and thus improve the performance of the SQL statements. The caching table can reduce the response time of the back-end server if the song’s features have already been cached.

Second, using batch requests when invoking Spotify Web API. There is a rate limit in requesting third-party API. For Spotify, it monitors the number of requests in a certain period (30 seconds) [6]. Therefore, to reduce unnecessary requests, each time song features need to be requested, batch requests are used to hold a list of song IDs (maximum 100 songs for each batch request) 11.

Third, using asynchronous processing when dealing with Spotify data zip files. When a user uploads a Spotify data zip file, the server will take out his account’s basic information and save it to the database. At the same time, it will also process his playlist information and save it to the database. If there are songs in his playlist that are not in the cached table, the server will also request their song features from Spotify. It may take the server some time to execute all those steps, especially requesting the Spotify API, before it can respond to the client, which may cause the client side to stuck. To solve this problem, when the back-end server has processed and saved the basic information of a user, it will generate a response to the client side and all logic involving the playlist will be executed asynchronously. This is achieved by Java’s CompletableFuture, ThreadPoolExecutor, and ArrayBlockingQueue.

Fourth, processing JSON using stream. The JSON file server needs processing might be very large, especially for the extended streaming history that contains listening histories for years. Processing such large files creates huge overhead on both space and time. To solve this problem, the server applied the Jackson library to use a stream to process the JSON file. Tested it on an extending streaming history JSON file with 10k entries and 6.7MB in size, it is 3 times faster than loading it completely and processing it as a string.

5 Evaluation

Overall, this project delivers a comprehensive visualization report tailored for non-technical users, adhering to the principle of "Overview first, zoom and filter, then details-on-demand." The visualizations provide clear and direct overviews, offering users a general understanding of their data from various perspectives. SVG is utilized for its high-quality zoom capabilities, enhancing detail visibility. The radar chart's filter function facilitates clear comparisons between playlists or focuses on individual playlist features. Additionally, the use of tooltips for displaying detailed information when users hover over data points allows for a deeper exploration of their data, seamlessly blending overview, detailed analysis, and interactive elements [4].

In addition, this project's design aligns with the principles for creating easily interpretable visualizations as outlined by Cario. [5]. The use of a CSS library ensures unified and artistically enhanced visual elements. Each visualization limits itself to no more than three attributes simultaneously, maintaining simplicity and clarity. Additionally, every visualization focuses on a single aspect of the data, adhering to the principle of unidimensionality. Familiar formats like radar charts, pie charts, and bar charts are employed, except for the more unique force layout. These familiar formats ensure ease of recognition and understanding for the users, further enhancing the accessibility of the visualizations.

6 Future Work

One limitation of this work is how to deal with large zip files and JSON files. Though we have currently optimized some parts of the data processing logic by using the stream as mentioned in section 4.2, further optimizations can be applied to reduce the processing overhead so that the server can handle more requests.

Another limitation lies in the parsing logic of the JSON files. Though zip files from different users normally contain a similar structure, some of the files like Payments and Playlists may differ. For example, if a user has too many playlists or payments, Spotify may either return several JSON files or a JSON file with an array of objects to hold all information. To make our system more robust, we could write more flexible parsing logic so that it can cover all cases of the zip file structure.

For future development, this project intends to add more visualization to provide more insights for the user to know how much more Spotify knows about them. One of the potential visualizations is to visualize the users' Spotify login history by using their IP address to show how their privacy can be revealed with detailed data collected.

Finally, this project could also be deployed to a public website, enabling everyone to access this service and create their own Spotify reports.

7 Individual Contributions

Jiayao mainly worked on the implementation of the back end server. Zhinuo mainly worked on the implementation of the front end client. All other works like the presentation and report are completed together.

You can visit the GitHub repository of this project. The back end part can be accessed by this link. The front end part can be accessed by this link.

References

[1] European Parliament and Council of the European Union. Regulation (EU) 2016/679 of the European Parliament and of the Council.

[2] Miranda Wei, Madison Stamos, Sophie Veys, Nathan Reitinger, Justin Goodman, Margot Herman, Dorota Filipczuk, Ben Weinshel, Michelle L. Mazurek, and Blase Ur. What twitter knows: Characterizing ad targeting practices, user perceptions, and ad explanations through users' own twitter data. In *29th USENIX Security Symposium (USENIX Security 20)*, pages 145–162. USENIX Association, August 2020.

[3] T. Munzner. *Visualization Analysis and Design*. AK Peters Visualization Series. CRC Press, 2015.

[4] B. Shneiderman. The eyes have it: A task by data type taxonomy for information visualizations. In *Visual Languages, IEEE Symposium on*, page 336, Los Alamitos, CA, USA, sep 1996. IEEE Computer Society.

[5] A. Cairo. *The Functional Art: An Introduction to Information Graphics and Visualization*. Voices That Matter Series. New Riders, 2013.

[6] Spotify. Spotify web api documentation, 2023.

[7] Patrick Helmholz, Michael Meyer, and Susanne Robra-Bissantz. Feel the moosic: Emotion-based music selection and recommendation. 06 2019.