Department of Computer Science & Engineering
The University of Texas at Arlington

Project Charter
CSE 4316: Senior Design I
Fall 2021

Team Silk Sonic
Track Records

Patrick Arzoumanian
Gustavo Chavez
Abraham Mookhoek
Ahmed Ullah
Spencer Whitehead

Revision History

Revision	Date	Author(s)	Description
0.1	09.18.2023	AM	Document creation
0.2	09.25.2023	PA, GC, AM, AU, SW	Complete first draft

CONTENTS

1 Problem Statement **7**

2 Methodology **7**

3 Value Proposition **7**

4 Development Milestones **7**

5 Background **8**

6 Related Work **8**

7 System Overview **9**

8 Roles & Responsibilities **10**

9 Cost Proposal **11**
9.1 Preliminary Budget 11
9.2 Current & Pending Support 11

10 Facilities & Equipment **11**

11 Assumptions **11**

12 Constraints **12**

13 Risks **12**

14 Documentation & Reporting **12**
14.1 Major Documentation Deliverables 12
14.1.1 Project Charter 12
14.2 Recurring Sprint Items 12
14.2.1 Product Backlog 12
14.2.2 Sprint Planning 13
14.2.3 Sprint Goal 13
14.2.4 Sprint Backlog 13
14.2.5 Task Breakdown 13
14.2.6 Sprint Burn Down Charts 13
14.2.7 Sprint Retrospective 13
14.2.8 Individual Status Reports 14
14.3 Closeout Materials 14
14.3.1 System Prototype 14
14.3.2 Project Poster 14
14.3.3 Web Page 14
14.3.4 Demo Video 14
14.3.5 Source Code 14
14.3.6 Source Code Documentation 14
14.3.7 Installation Scripts 14

14.3.8 User Manual 14

List of Figures

1 Example sprint burn down chart 13

1 Problem Statement

With the advent of music streaming services like Apple Music or Spotify, people have been given access to an overwhelmingly large collection of musical content. It has become far easier and intuitive to refine or expand musical interests. With so much content at people's fingertips, an enormous amount of insight could be gleaned from their music streaming history. Although music streaming services often offer limited analysis on recent listening history, they lack a robust set of analytical data for a user's entire history.

2 Methodology

Our music analytics web app, Track Records, seeks to empower listeners with a wide variety of data sourced from their music streaming services. The app will offer people a better understanding of their relationship with music and how it influences and complements their daily lives. This will be accomplished by creating an interactive web application that mines data from a user's registered music streaming apps. This data will be presented to users in a calendar view by default. This will provide users with an intuitive timeline of their listening history. Other data visualization formats will be provided for a more in-depth analysis of a user's data. Modular graphs such as histograms, pie-charts, and line graphs will offer users a way to compare different values. Examples of potential graph values are how much a user has listened to a specific song, artist, or genre.

3 Value Proposition

The global music streaming market size was estimated at USD 34.53 billion in 2022 and is expected to grow at a compound annual growth rate (CAGR) of 14.4% from 2023 to 2030 [1]. The leaders in this space are Spotify and Apple Music which provide great streaming quality alongside small features. Despite their massive platforms, they both miss the mark on providing a consistent way to view their listening data and other user-focused retrospective features. Our product aims to enhance the user experience by providing a visually stimulating analytic and auditory diary experience to fill in the gaps left by Apple Music and Spotify. Furthermore, we also act as a data collection resource for record labels curious about when users listen to a piece of music, how often a particular user replays a track, and other alike metrics.

4 Development Milestones

- Project Charter first draft - Sep. 25, 2023
- System Requirements Specification - Oct. 16, 2023
- Architectural Design Specification - Nov. 06, 2023
- Demonstration of Spotify Calendar View - Dec. 01, 2023
- Detailed Design Specification - Feb. 13, 2024
- Demonstration of User Journal and Tone Analysis - Mar. 06, 2024
- Demonstration of Modular Graph Views - Apr. 03, 2024
- CoE Innovation Day poster presentation - Apr. 2024
- Demonstration of Apple Music Integration - Apr. 24, 2024
- Final Project Demonstration - May 2024

5 BACKGROUND

The musical availability offered by streaming services have given listeners a great deal of freedom in how, when, and where they listen to music. Despite this, these services are lacking in retrospective features. Music streaming services store a great deal of data for each of their users. Unfortunately, this data has been severely underutilized for the purposes of offering clarity to users. Currently, music streaming services do not offer a simple or intuitive way to see how one's musical interests have evolved over time. Certain features like Spotify Wrapped offer some information on users' listening habits, but a great deal of this information is left untouched. This is a problem for anyone hoping to better understand how they listen to music.

Consider an instance where someone wants to know which of two specific artists or songs they listen to more. Perhaps they want to know how varied their listening has been in terms of genre, artist, or some other metric. As of right now, these individuals are left to determine all of this without the help of their streaming services. It would be unreasonably cumbersome for a person to constantly track when and what they listened to. Yet this is what someone must do without the assistance of a dedicated set of music analytics tools.

Some have already tried to remedy this predicament with their own web apps. These products offer analysis on users' listening metrics, but they do not capitalize on the personalized nature of this data. The focus of these services is simply to provide metrics, not to relate this information to the user's individual experiences. Furthermore, using preexisting apps to see these statistics will only offer the data, not focusing on how to use this information to encourage personal retrospectives.

This problem presents a unique opportunity for those capable of accessing music streaming service APIs. By collecting user data and storing it for users, one could provide easy access to a wealth of information through data visualization. Although this is a service offered by a handful of other web applications, there is still an untapped market. The people who want to utilize these statistics for the purpose of reminiscing on their past with music and to document this musical journey have yet to be catered to. This will be the target market for Track Records.

6 RELATED WORK

As briefly mentioned in the previous section, music streaming services offer limited data analytics to their users. One such example is Spotify Wrapped [2]. Spotify wrapped is an annual feature provided by Spotify that shows users their top songs, artists, and albums. These statistics are provided on both an individual user scale, as well as platform-wide. Unfortunately, this is the only data analytics functionality Spotify Wrapped offers users. For anyone that wants a more in-depth view of their past listening, they must find it elsewhere.

Beyond music streaming services themselves, there already exist web applications that seek to provide their users with statistics from music streaming services. One such product is LastFM [3]. This web application collects listening metrics from its users and offers this information in the form of a user profile. However, instead of leaning into personal retrospection, LastFM focuses on the future by providing music recommendations. Not only this, but LastFM emphasizes social connections by making users' statistics public and placing its users into various leader-boards. Due to these differences in intended features, this solution does not meet our target market's needs.

Another application that offers music streaming statistics is StatsFM [4]. This mobile application is closer to the vision of our product. StatsFM emphasizes the usage of their statistics as being a way to paint a clear picture of users' personal journeys with music. At the same time, StatsFM implements a social aspect to their product by letting users compare stats with others.

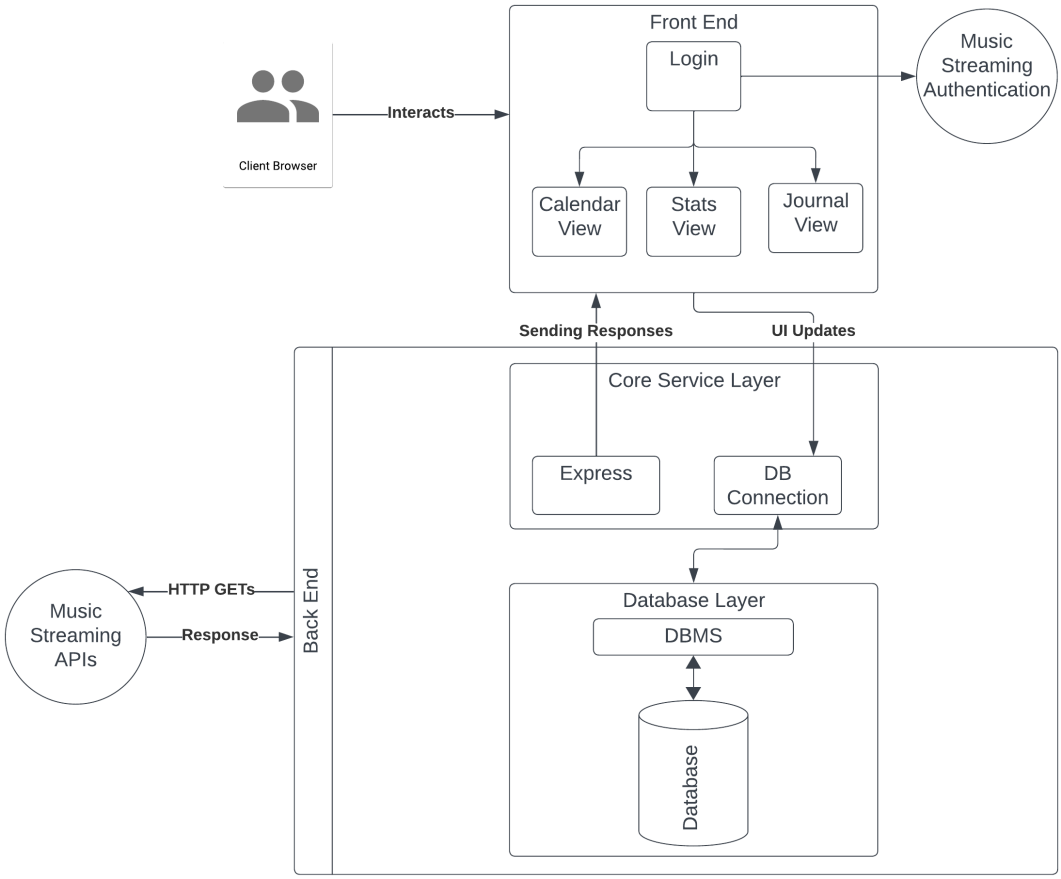
Obscurify is another music statistics website, but with a focus on different aspects of users' taste, such as obscurity, danceability, and acousticness [5]. More importantly it provides users with a comparison

of their current taste with their overall taste, which allows the user to see how their taste has evolved over time. This is similar to how Track Records will focus on the history of users' listening patterns, and provide more insight past just what the user is listening to right now. However, it differs by only providing a general overview of different aspects of users' taste, rather than giving users the ability to access specific times they have listened to a song or added it to a playlist in the past, which is how our app will likely implement this aspect.

An additional aspect to consider is the implementation of a music player. Since the ability to play music from the app would be convenient, but is not the focus of our app, a simple player without too many options would be preferable. A good example of an app that implements this kind of player is Discover Quickly [6]. This app allows you to connect your Spotify account and gives you large lists of songs related to any artist or song that you request. Because this app is focused on allowing the user to browse through the music as quickly as possible, the entirety of the player interface is playing a snippet of a song as the user hovers over/click on a song, and stopping when the user clicks away. This level of simplicity works well for the focus of the app, and while our app has a different focus, it could still benefit from a similar level of simplicity for the player we plan to implement.

7 System Overview

Our product offers a solution through its retrospective, personalized focus on music streaming service data analytics. After registering an account and allowing access to their personal music streaming account(s), we will ensure to handle the user's private and sensitive information. Once the system has proper authorization to access the user's information, it will proceed to extrapolate and organize the data meaningfully and aesthetically. Authentication will be handled on a back-end server that will handle processing, managing user requests, and ensuring data integrity. As the user interfaces with the front-end side of the application, different calls will be made to the back-end to fulfill the user's request whether that be viewing a specific data point, account management, or modifying their personal journal entries. Due to the personalized nature of our analytics service, many of our front-end components must be dynamic with conditional rendering to tailor to each user's experience. We will also utilize a database in which personalized user information will be stored such as weekly recaps of listening habits and small personal notes that the user decides to write down for individual songs. This information will be requested from the database and be presented once the user has successfully logged in to the application. As mentioned earlier, for the application to retrieve the user's music data, we need to utilize the developer API that each respective service has created and made available to the public. Unfortunately, these APIs limit the amount of calls a system may request within a given amount of time. Thus, the task of limiting the amount of requests a user makes will also be handled in the back-end. Together, in unison, all these components will be critical to providing the user a clean and smooth experience in viewing their data and habits as they continue using their music streaming service(s).



8 Roles & Responsibilities

For this project, the role of stakeholder and sponsor will be held by Professor Gieser. Anytime the team would have met with a representative for a sponsored project, the meeting will be with Gieser instead. Although the expectations for the project are open-ended, the team will maintain consistent and clear communication with Gieser to ensure a satisfactory product. The team consists of five members: Patrick Arzoumanian, Gustavo Chavez, Abraham Mookhoek, Ahmed Ullah, and Spencer Whitehead.

Although each team member will pivot and assist anywhere they are needed, the areas of concentration are as follows. Patrick Arzoumanian will helm the front-end development of the app with the assistance of Spencer Whitehead. This will consist largely of creating the user interface layout and visual design using web development tools. Gustavo Chavez will lead in back-end development with the assistance of Ahmed Ullah. This will revolve around connecting to music streaming services' APIs, setting up the structure of user accounts, and working with database queries. Naturally, there will be a fair amount of collaboration between front-end and back-end in order to fully test and finish certain features. This is why we anticipate some flexibility in what team members do throughout different sprints. Abraham Mookhoek will handle the majority of documentation and reporting, assisting in development wherever he is needed most. In terms of documentation, this means handling each sprint plan presentation, sprint review presentation, sprint backlogs, and documentation deliverables. For development, this means working on tasks that are either running behind or consuming more development time than other tasks.

Since Gustavo Chavez was the individual who thought of and pitched the project, the team has

decided that he will act as product owner. Finally, the position of scrum master will be cycled between team members. This position will involve scheduling and ensuring attendance of weekly meetings and keeping tabs on teammate progress. Specifically, each sprint will have a new scrum master. At the end of each sprint, the team will vote on who serves as scrum master for the next sprint. Abraham will serve as the first scrum master.

9 Cost Proposal

This project, being a web application that pulls information from music streaming services' APIs, will require a variety of expenses. First, the team will need an apple developer's license. This license will be necessary if we are to implement any meaningful amount of Apple Music functionality. Furthermore, it is possible that we will need to pay for a dedicated server to store user data. These expenses alone will cost hundreds of dollars.

9.1 Preliminary Budget

Item	Price	Amount
Apple Developer License	$100	1
Database Server	$250	1

9.2 Current & Pending Support

Current financial support comes exclusively from the Computer Science and Engineering (CSE) department from the University of Texas Arlington (UTA). Through the CSE department, the team has been allotted $800 that can be spent with approval from Dr. Shawn Gieser. Although it is not anticipated to exceed this $800 budget, if expenses did so, then remaining expenses would have to come out of pocket.

10 Facilities & Equipment

Since the product itself is a web application, it has no physical components. Consequently, there is no dedicated physical space necessary for development. Instead, team members will choose a physical location on campus to hold weekly meetings. Besides weekly progress update meetings, communication will be held online. This allows for a very flexible and fluid development structure. Members will be able to make contributions to development wherever and whenever they have access to a computer.

In terms of equipment, the team will need to access an Apple Developer's license when implementing apple music functionality. Furthermore, it is possible the team will need a Mac computer. If the Mac is truly necessary, then it can be lent from the university for the duration of the project's development.

11 Assumptions

- We will have access to the Apple Developer License by the time we start developing.
- The user of our product will own a premium subscription with either Apple Music or Spotify.
- The music statistics provided by Apple and Spotify API's don't get depreciated.
- Apple's and Spotify API's continue being commercially available.
- React.JS will continue to be commercially available.

12 Constraints

- Total development costs must not exceed $800.
- All user data obtained from our product must be reviewed and approved for release by the Information Security Office prior to being copied to any internet connected storage medium
- The Web-Application must run on the major Web Browsers such as Chrome, Edge, and FireFox.
- Our API Calls-Per-Second limit for both Apple Music and Spotify must not exceed a certian amount.

  Apple: "user-hour-lim:3500;user-hour-rem:500"
  Spotfiy: Based on the number of calls that your app makes to Spotify in a rolling 30 second window.
- Final prototype demonstration must be completed by May 1st, 2024

13 Risks

The following high-level risk census contains identified project risks with the highest exposure. Mitigation strategies will be discussed in future planning sessions.

Risk description	Probability	Loss (days)	Exposure (days)
Delayed access to apple developer license	0.50	60	30
Delayed access to lifetime Spotify history	0.70	30	21
Access limitations with Spotify API	0.20	20	4
Access limitations with Apple API	0.10	10	1
Cloud database temporarily unavailable	0.10	4	0.4

Table 1: Overview of highest exposure project risks

14 Documentation & Reporting

14.1 Major Documentation Deliverables

14.1.1 Project Charter

This document will be maintained as a LaTeX document within Overleaf that all team members have access to. It is expected to be updated every sprint as new information becomes available.

Initial version expected delivery: September 25th, 2023
Final version expected delivery: November 27th, 2023

14.2 Recurring Sprint Items

14.2.1 Product Backlog

All potential features will be taken in from User feedback, Classmates, Professor Gieser, Teammates and more. We will be keeping track of these items through Overleaf and a Discord Text Channel. During our weekly team meetings, we will discuss which item is most important and order/delegate the tasks accordingly.

14.2.2 Sprint Planning

Each sprint will be planned during our weekly meeting where all team members will give input on their current progress. After this we will look at the items on the backlog and assign the next items for our team. Since there are 4 sprints per semester, we will have a total of 8 sprints during our academic school year.

14.2.3 Sprint Goal

The sprint goal will be constructed based off our last sprints progress, preset project deadlines, professor recommendations, and overall team input. It will also be constructed to ensure that we continuously make progress each sprint.

14.2.4 Sprint Backlog

Due to the likelihood that the web application will be broken down into numerous components, each handled by a separate team member, a variety of items from the backlog may be promoted to the current sprint depending on our individual progress. The backlog will be maintained via our Team Discord Channel.

14.2.5 Task Breakdown

The team will divide tasks from the sprint backlog among members during our sprint planning session. Individual team members will voluntarily claim tasks from the backlog, and the finalization of task assignments is conducted by the product owner, Gustavo Chavez. Each member will document their individual time spent on completing tasks, and the team will compile the total time at the end of each sprint.

14.2.6 Sprint Burn Down Charts

The responsibility of generating the burn down chart will be Abraham Mookhoek's. He will be responsible for assessing the total amount of effort by referring to the documentation of time spent per backlog task for each member. The format used by the burn down charts will be excel, and an example burn down chart can be seen below.

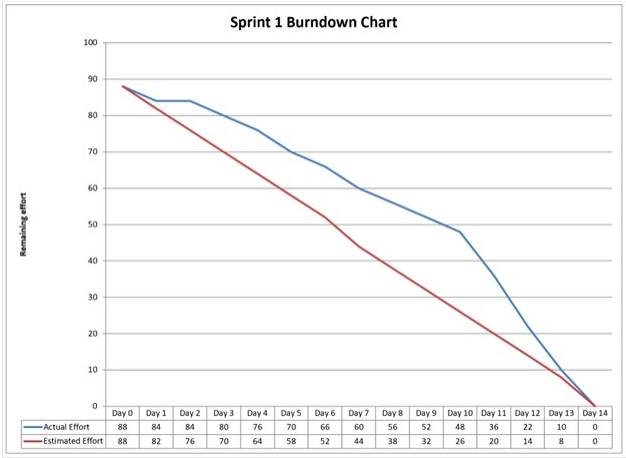


Figure 1: Example sprint burn down chart

14.2.7 Sprint Retrospective

After each sprint, the team will conduct a sprint retrospective in a meeting either in person or through our Discord text channel. During the meeting, each member will contribute their documentation of

any difficulties encountered during the sprint, and the team will compile documentation to assess what went well and what could be improved for the next sprint.

14.2.8 Individual Status Reports

Each individual team member will report an overview of progress achieved in completing tasks assigned from the sprint backlog. This will occur every week in order to adequately assess the status of the project's development. In each report, a member will include personal accomplishments during the week, the current items they've worked on and that are incomplete, any difficulties encountered, and an estimate of time spent per task.

14.3 Closeout Materials

14.3.1 System Prototype

In the final system prototype, the team will feature a web application which will undergo a Prototype Acceptance Test (PAT) with test customers. The prototype will be demonstrated on the College of Engineering (CoE) Innovation Day, April 16, 2024, presented on sight at the University of Texas at Arlington.

14.3.2 Project Poster

The project poster will provide a cohesive overview of the application, including an abstract, introduction methodology review, system functionality architecture, and sample images of the application's demonstration. The final dimensions of the poster will be 36" x 48". The poster will be presented on the CoE Innovation Day, April 16, 2024.

14.3.3 Web Page

The project web page will feature a cohesive overview of the application as well as provide a means to demo the application. The web page will be available to the public, and the team hopes to deliver the web page by May 2024, provided at the end of Senior Design II.

14.3.4 Demo Video

The demonstration video will show a full display of the application's functionality, providing a walkthrough of the app's various features. The video will likely range between 5 - 8 minutes.

14.3.5 Source Code

The project source code will be maintained via a GitHub repository, in which all the team members have access to. As for the version control system, the team will adopt Git. The customer will have access to the source code of the project. The customers as well as the general public will have access to a GitHub repository containing all project source code.

14.3.6 Source Code Documentation

The team will employ the documentation generation tool Doxygen, and will abide by the IEEE standards. The final documentation will be provided as a Microsoft Word document.

14.3.7 Installation Scripts

Since the product is a web application, no installation scripts will be needed.

14.3.8 User Manual

The user will be provided with apt instructions in the README.txt file, and the user will only require the provided link to access the web application.

REFERENCES

[1] G. V. R. Staff, "Music streaming market size & share analysis report, 2030," Jul 2023.

[2] C. Eggertsen, "How does spotify wrapped work?" Dec 2021.

[3] LastFM, "Explore top music powered by your scrobbles," 2023.

[4] S. Staff, "Stats.fm," 2020.

[5] A. Olivero and N. Patel, "Obscurify music."

[6] A. Aufrichtig and E. Lee, "Discover quickly," 2019.