

Knight Foundation School of Computer & Information Sciences

Spring 2026 Senior Design Project

AudioAura

Students: Lauren Stone, Daniel Gonzalez Lemus, Juan de Dios Nunez, Nicholas Brotherhood, Daniel Bencomo
Instructor/Faculty: Seyedmasoud Sadjadi, Florida International University

PROBLEM

Modern trends are highly focused on aesthetics, “vibes”, and the trendy concept of “aura”, however, standard music players don’t reflect on this and fail to hop on the latest trends. Platforms like Spotify and Apple music offer your own music “wrap” with outdated visuals, that do not capture the emotional essence of the music. Users lack a highly shareable, music engaging way to see their “aura” based on their favorite songs and artists.

CURRENT SYSTEM

1. **Recycled recaps:** Features like “Spotify wrapped” attempt to capture music aesthetics, but the format and visuals have remained unchanged for the past five years resulting in boring features.
2. **Yearly trend recaps:** Only available once a year rather than providing the user an option to choose between the last 4 weeks, 6 months or last several years.
3. **Flawed representation:** Current wrap algorithms rely on a narrow set of metrics. This limited scope generates inaccurate profiles and fail to capture a user’s true aesthetic.

REQUIREMENTS

- Provide secure user authentication using Spotify OAuth 2.0 with PKCE
- Retrieve and display the user’s top 5 tracks, top artists, and profile data via API
- Extract and analyze audio features such as tempo, energy, danceability, valence, acousticness , and popularity.
- Generate an “AudioAura” based on collected listening behavior and genre distribution
- Present all results in a clear, visually engaging, and user-friendly interface.
- Maintain secure session management, allowing users to log out and terminate sessions safely.

SYSTEM DESIGN

The system design follows a client-server architecture integrated with Spotify services:

- **Authentication Flow:**
 - Users securely log in with their Spotify account through Spotify OAuth
 - Authorization allows for access to personal listening data
- **Client-Server Interaction:**
 - The React frontend sends requests to the backend
 - The backend manages communication with Spotify services
- **Data Retrieval & Processing:**
 - The backend fetches user data from the Spotify Web API and Track Analysis API
 - Data such as top artists, tracks, genres, and audio features is processed into meaningful insights
- **Report Generation & Delivery:**
 - Processed analytics are returned to the frontend
 - The frontend renders a personzalied AudioAura report

OBJECT DESIGN

Key objects are used around the system such as User, Track, Artist, and AuraProfile

- **User:** Uses Spotify to store authentication data and session information
- **Track:** Represents the users top songs and audio features such as energy, valence, and acousticness
- **Artist:** Represents the users top artist and related data that is used in the report
- **AuraProfile:** Contains calculated attributes such as energy level, aura classification and summary description.

IMPLEMENTATION

The application was implemented with the following:

Frontend:

- Built using React + Vite (TypeScript) for a responsive and component based interface.
- Integrated Spotify data visualization using animation libraries such as motion/react and lucide-react.

Backend:

- Built with Node.js + Express to handle API requests and data processing.
- Implemented secure authentication using Spotify’s OAuth.
- Communicates with the Spotify Web API and Track Analysis API to retrieve user listening data.

VERIFICATION

System functionality was verified through using real Spotify user data for testing.

- Ensured accurate retrieval and parsing of listening data from users
- Tested logic of aura generation across different listening profiles
- Ensure consistency and responsiveness across components using UI testing
- Used feedback from team members to refine functionality and user experience

SUMMARY

AudioAura transforms a user’s Spotify activity into a personalized, interactive visual report. By integrating Spotify’s API with external track analysis and visual charts, the system provides a responsive and intuitive application that allows users to explore and better understand their Spotify listening habits.

REFERENCES

- Spotify Web API documentation: <https://developer.spotify.com/documentation/web-api>
- Track Analysis API documentation: <https://rapidapi.com/soundnet-soundnet-default/api/track-analysis>
- React.js documentation: <https://react.dev/>
- Obscurify - music taste analysis platform (used as inspiration for design and feature): <https://www.obscurifymusic.com/login>
- Research on audio features (energy, valence, acousticness): <https://developer.spotify.com/documentation/web-api/reference/get-audio-features>