Ridhim

(Music Streaming Application)

Submitted for the requirement of

Project course

BACHELOR OF ENGINEERING

COMPUTER SCIENCE & ENGINEERING

Submitted to:

Parvez Rahi (E14563)

(Supervisior)

Submitted By:

Pushpinder Singh(20BCS7300)

Sujata Singh(20BCS7295)

Gurwinder Singh(20BCS7634)

Co-Supervisior
Arvind Gautam
(E13182)

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CHANDIGARH UNIVERSITY, GHARUAN

Abstract

The project ‘Ridhim’ (Music Streaming Application) is a application that can be used for services related to music. It provides an easy and convenient way to access music anytime, anywhere. Users can stream or download millions of songs to their device with just a few clicks, making it easy for them to enjoy their favorite music anytime, anywhere. These apps use advanced algorithms to customize the user experience and offer recommendations based on the user's listening and preferences. This helps users discover new music they might not find otherwise and can create a more engaging and enjoyable listening experience. This app provides access to a large library of songs spanning many genres, artists, and eras.

TABLE OF CONTENT

1.Feature/characteristics identification

- **User Authentication:** The application must require users to register and log in to access the music. User authentication ensures that only authorized users can access the content, and their preferences and playlists can be saved and accessed across devices.

- **Music Library:** The application must have a vast collection of music that users can choose from. The music library should be regularly updated to provide users with the latest releases and popular tracks.

- **Audio Quality:** The application must provide high-quality audio playback to enhance the user experience. The audio quality must be consistent across different devices and internet speeds.

- **Search Functionality:** The application should allow users to search for music based on artists, albums, genres, and other parameters. The search function must be quick, accurate, and user-friendly.

- **Playlists:** Users should be able to create and manage their playlists, and the application should provide features like automatic playlist generation based on user preferences, smart shuffle, and repeat options.

- **Offline Playback:** The application should allow users to download music and listen to it offline without an internet connection. This feature is particularly important for users with limited internet connectivity.

- **Social Integration:** Users should be able to share music and playlists on social media platforms like Facebook, Instagram, and Twitter.

- **Personalization:** The application should provide personalized recommendations based on user listening history, preferences, and search history.

- **Recommendation Engine:** The application must provide personalized recommendations to users based on their listening history, preferences, and search history. This feature involves implementing machine learning algorithms that analyze user behavior to make accurate recommendations.

- **Security:** The application must ensure that user data such as login credentials and payment information is secure and protected from unauthorized access.

2. Constraints Identification

- **Scalability:** The application must be able to handle a large number of users, requests, and traffic without sacrificing performance. Scalability involves considering issues such as load balancing, caching, and database optimization.

- **Compatibility:** The application must be compatible with different platforms, devices, and web browsers. This constraint involves considering different operating systems, screen resolutions, and hardware specifications.

- **Performance:** The application must provide a fast and responsive user experience, including quick loading times, smooth playback, and accurate search results. Performance involves optimizing code, minimizing latency, and reducing server load.

- **Availability:** The application must be available and accessible to users 24/7 without downtime or interruptions. This constraint involves implementing backup and recovery mechanisms, load balancing, and failover strategies.

- **Resource Constraints:** The application must be optimized to run on limited resources, including processing power, memory, and storage. This constraint involves implementing efficient algorithms, reducing redundant data, and minimizing the use of external APIs.

- **Budget Constraints:** The application must be developed within the allocated budget, which includes development costs, hosting fees, and marketing expenses. This constraint involves prioritizing features and functionalities based on their impact on user experience and revenue generation

3.Analysis of features and finalization, subject to constraints

Legal Constraints: The application must comply with legal constraints, including music licensing agreements, copyright laws, and data privacy regulations.

Scalability: The application must be able to handle a large number of users, requests, and traffic without sacrificing performance. Scalability involves considering issues such as load balancing, caching, and database optimization.

Compatibility: The application must be compatible with different platforms, devices, and web browsers. This constraint involves considering different operating systems, screen resolutions, and hardware specifications.

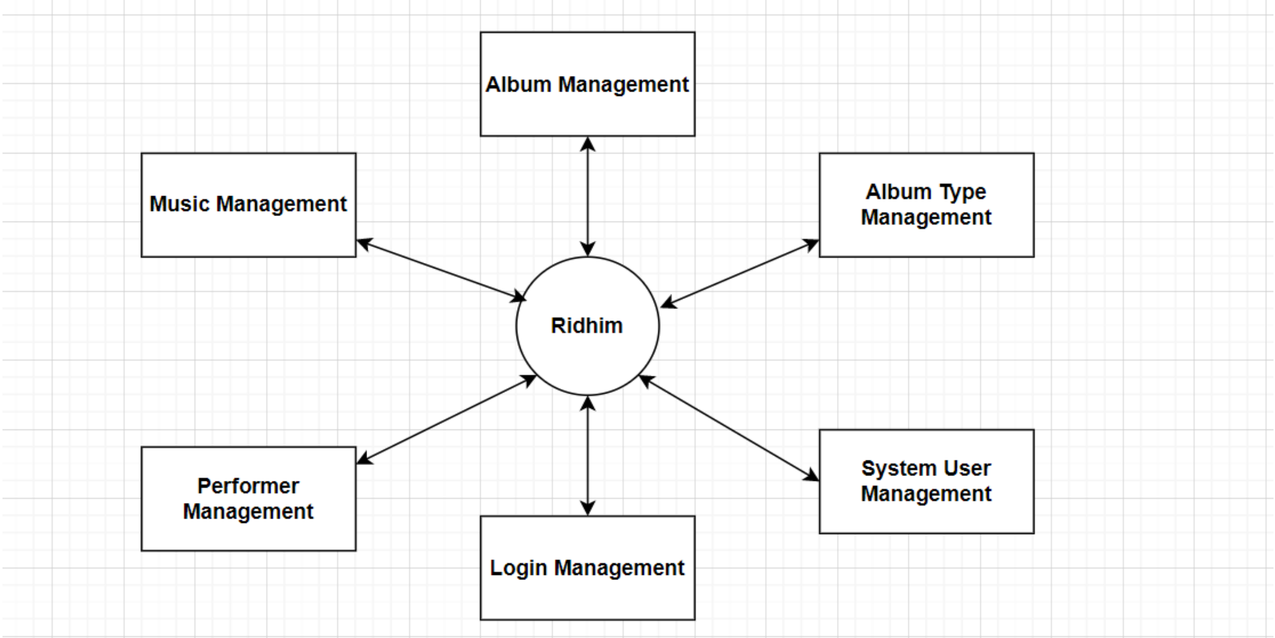
Performance: The application must provide a fast and responsive user experience, including quick loading times, smooth playback, and accurate search results. Performance involves optimizing code, minimizing latency, and reducing server load.

Security: The application must ensure that user data such as login credentials and payment information is secure and protected from unauthorized access. This constraint involves implementing encryption, firewalls, an access controls.

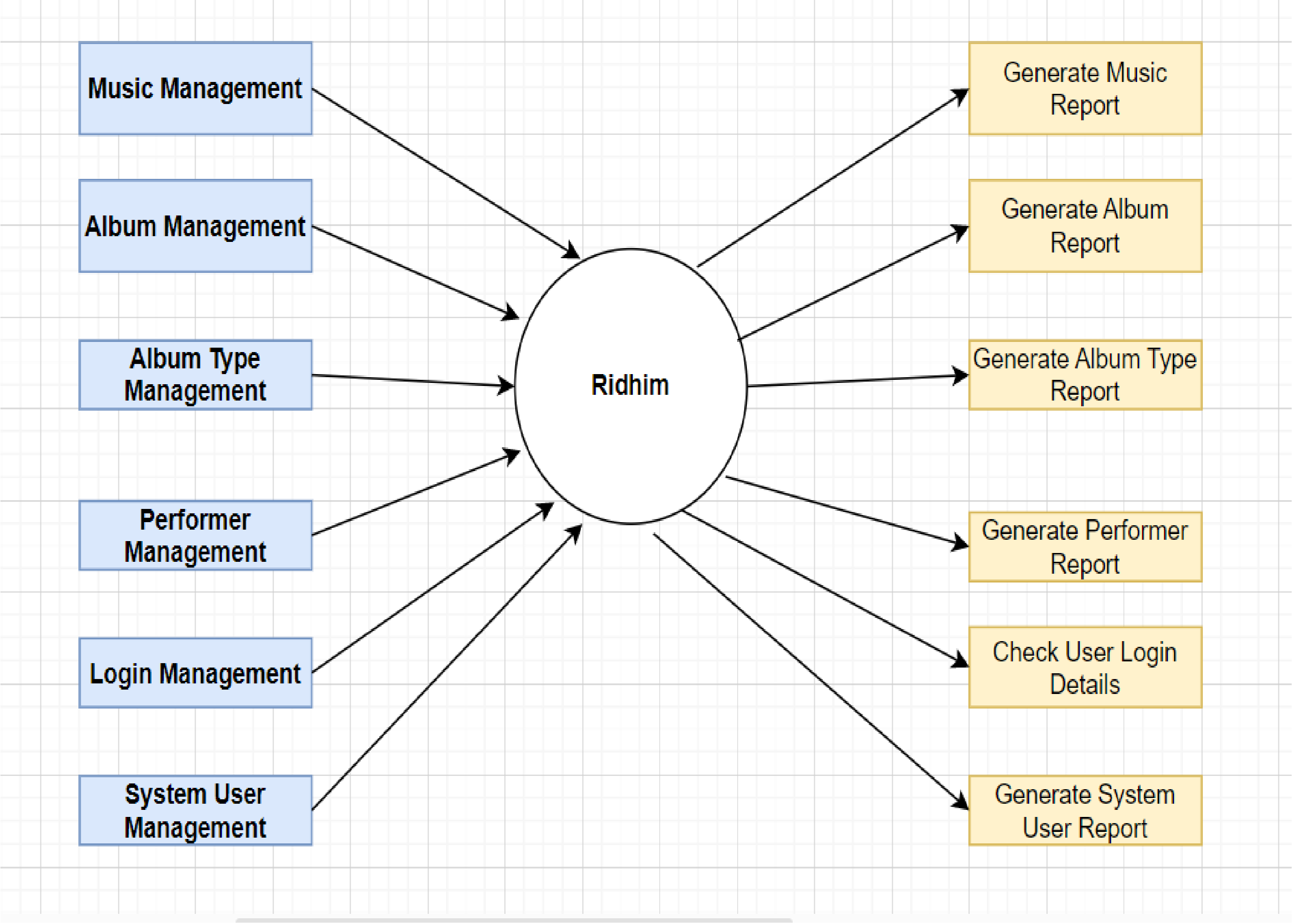
Budget Constraints: To ensure the project stays within the allocated budget, the application should prioritize features based on their impact on user experience and revenue generation. The development team should also consider open-source solutions, use cost-efficient cloud infrastructure, and optimize development processes to minimize costs.

4.Design Selection

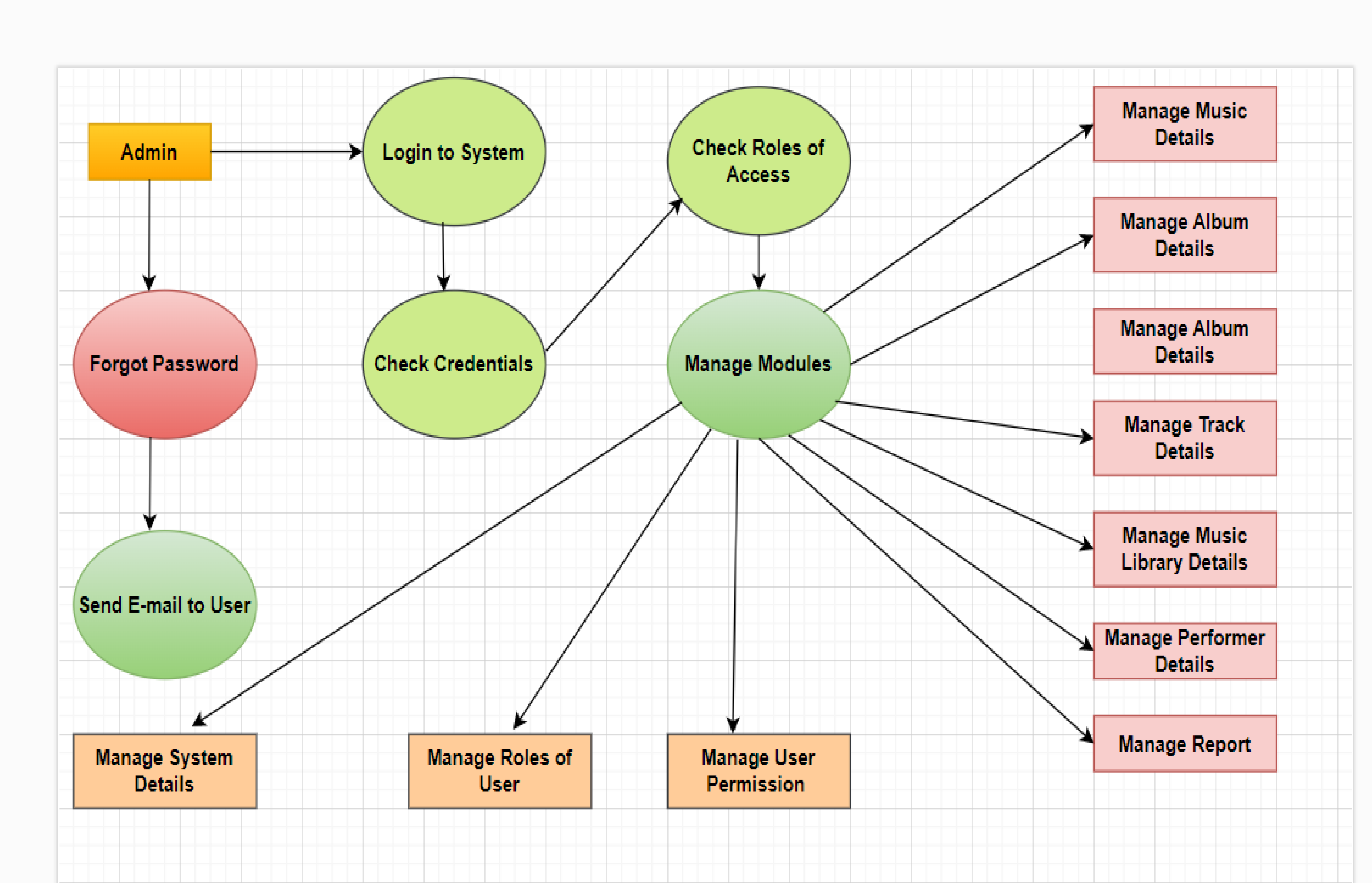
i) DFD (level 0)



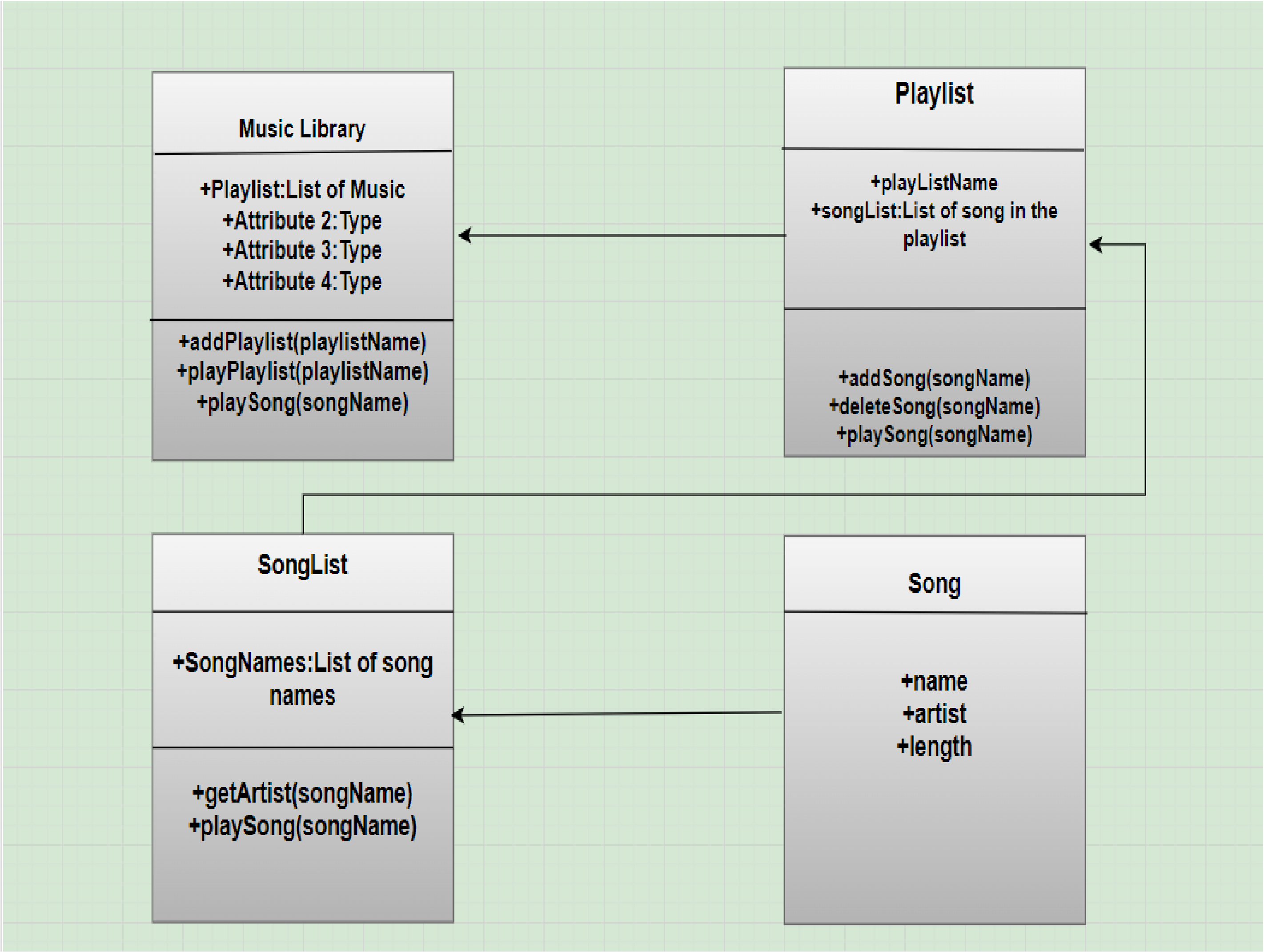
ii) DFD (level 1)



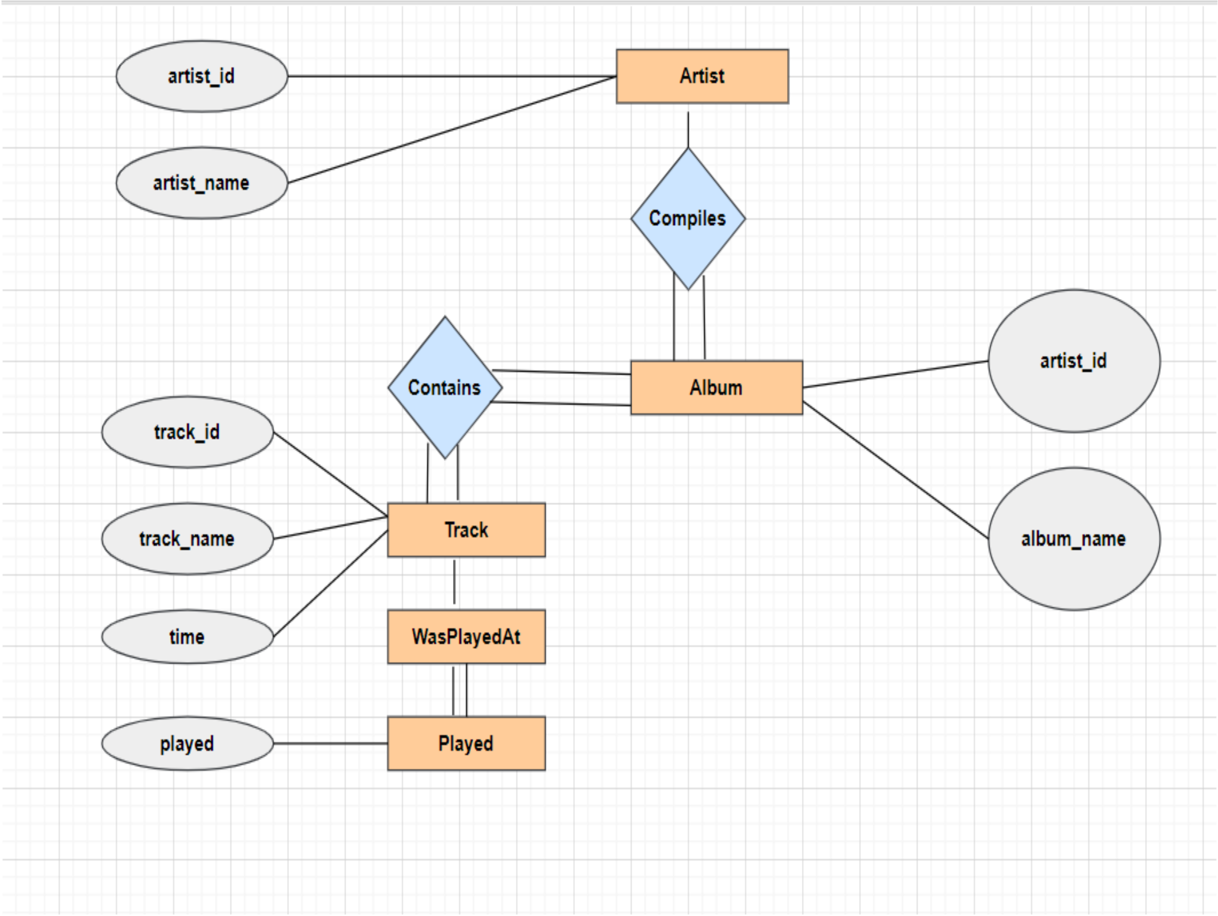
iii) DFD (level 2)



iv) UML Class Diagram



v) ER Diagram


artist_id
Artist
artist_name
Compiles
artist_id
Contains
Album
track_id
track_name
Track
album_name
time
WasPlayedAt
played
Played

References

1) The State of the Music Streaming Industry in 2021" by MIDiA Research: This report provides an overview of the music streaming market, including key players, trends, and challenges.

2) "Music Streaming Services: A Review" by J.M. Lammers et al.: This academic article examines the technical aspects of music streaming services, including audio quality, codec formats, and metadata.

3) "The Impact of Music Streaming on the Music Industry: A Global Perspective" by A. Montagnani and E. Losavio: This academic article explores the economic and legal implications of music streaming for the music industry, including revenue models and copyright issues.

4) "Designing the User Experience of Music Streaming Apps: A Literature Review" by M. Ma and L. Yang: This academic article reviews existing research on user experience design for music streaming apps and provides recommendations for creating engaging and user-friendly interfaces.

5) "How Spotify Engineered the Perfect Release Day" by J. Cowen: This article from The Verge examines the marketing strategies and data analytics behind successful music releases on Spotify.