E-ISSN: 2962-6110
P-ISSN: 2964-7843
JSRET
jsret.knpub.com
Journal of Scientific Research, Education, and Technology

Mobile Music Recommendation Using K-Nearest Neighbor and Artificial Intelligence

Egy Septiandy Kusmawan, Anita Fira Waluyo
Universitas Teknologi Yogyakarta, Yogyakarta, Indonesia

ABSTRACT

The development of online music services demands increasingly personalized and contextual recommendation systems. However, most existing systems are still limited to processing historical data without taking into account the emotional state or activities of users. This study aims to design a machine learning-based music recommendation system that generates recommendations based on the user's listening history and artificial intelligence (AI) to produce personalized song recommendations based on the user's mood and activity. The methods used include song data analysis from the Spotify API, the K-Nearest Neighbor (KNN) algorithm, and the application of a Large Language Model (LLM) as a prompt-based interactive interface. Test results show that the system is capable of providing song recommendations with a 95% similarity rate using machine learning based on the songs listened to by users, and the application of AI produces more specific recommendations according to user prompts.

Keywords: *Music, Spotify, KNN, AI, Mobile, Python, Flask, Flutter, Machine Learning*

Corresponding author
Name: Egy Septiandy Kusmawan
Email: egyseptiandy19@gmail.com

INTRODUCTION

In today's digital age, online music services such as Spotify, YouTube Music, and Apple Music have become the main platforms for users to search for, listen to, and discover new songs. Finding the right music can be overwhelming due to the vast amount of content available (Bobadilla, Ortega, Hernando, & Gutiérrez, 2013). This phenomenon, known as the "paradox of choice," occurs when having too many options makes decision-making more difficult rather than satisfying. Users are frequently stuck in a "filter bubble" when it comes to music streaming, where they are only exposed to songs that are similar to what they have already heard, which restricts their ability to discover new genres or performers.

Furthermore, traditional music discovery methods relied heavily on manual curation by radio DJs or music critics. However, with millions of tracks uploaded daily, manual curation is no longer feasible. This shift has necessitated the development of automated recommendation engines (AREs). Early generations of AREs focused solely on collaborative filtering, which recommended songs based on the listening patterns of similar users. While effective, this method suffers from the 'cold start problem,' where new users

or new songs with no interaction data cannot be recommended effectively. Content-based filtering emerged as a solution, analyzing the audio characteristics of the songs themselves. Yet, even this approach has limitations; it often fails to capture the semantic context of a user's request, such as 'songs for studying on a rainy day' or 'upbeat tracks for a morning jog.' This gap highlights the urgent need for a hybrid system that combines mathematical precision (KNN) with semantic understanding (Generative AI).

That's why, one of the most important features of these applications is the recommendation system, which is useful for enhancing the user experience by displaying a list of songs tailored to the user's listening history (Aggarwal, 2016), so that the recommendations provided are in line with the user's preferences (Madani, Hasrullah, & Hasrullah, 2023)**.**

A song recommendation system is a recommendation system that can provide users with a variety of song recommendations based on several factors, such as the user's listening history, the most frequently listened to song genres, the most frequently listened to artists, and so on. The song recommendation system exists due to the complexity of factors that can influence the type of music listened to (Rizaldy, Dewi, & Brata, 2021).

Research on song recommendation systems continues to evolve with the application of various algorithms (Portugal, Alencar, & Cowan, 2018) to improve accuracy. Some of these studies apply the K-Nearest Neighbor algorithm. The first study applied the K-Nearest Neighbor algorithm to a Spotify playlist dataset containing more than 200,000 songs. This study concluded that the implementation of KNN with the nearest distance calculation can provide song recommendations that are mostly similar to user input, making it effective for use in recommendation systems (Pelaupessy & Suhartana, 2023).

In the second study, a Spotify song recommendation system was created by combining the KNN algorithm with the Genetic Algorithm. This approach was used to classify songs based on user genre preferences. The results of this study show that combining KNN with the Genetic Algorithm successfully improved the accuracy of recommendations compared to using traditional KNN alone (Sidora & Harani, 2023).

Subsequently, the next study integrated the Content Based Filtering and KNN methods, in which song audio features were used as parameters, and KNN functioned to classify objects based on the nearest training data. Accuracy testing showed that this hybrid system achieved the highest accuracy value of 90.49% with an optimal k value of 9. This proves that this combination of methods is very precise in providing relevant song recommendations (Sarmandana, Widiartha, Putri, & Astawa, 2024).

The purpose of this research is to develop an application that can provide song recommendations based on user preferences, implement machine learning algorithms to obtain recommendation results based on user listening history data, and integrate AI APIs to allow users to manually input their preferences so that they can get music recommendations that suit their current desires.

This research is expected to provide benefits such as serving as a reference for researchers in developing data-based song recommendation systems with the application of machine learning and AI API integration to handle user preferences using prompts,

making it easier for users to find songs according to their preferences, not only based on history, but also based on mood and specific activities, and serving as an example of AI implementation in the entertainment world (song recommendation systems) that can help in understanding the concept of AI API integration and data processing.

While previous studies often rely on deep audio analysis (e.g., tempo, dancability), current limitations in commercial APIs have created a need for effective recommendation strategies that rely on accessible metadata. This research addresses this constraint by proposing a hybrid interaction model where K-Nearest Neighbor (KNN) logic is augmented by a conversational Generative AI interface. Ideally, this combination compensates for the lack of highly specific audio features by allowing users to express complex musical needs through natural language.

Based on the outlined background and identified gaps, this study explicitly formulates the following research questions:

1. How can the K-Nearest Neighbor (KNN) algorithm be effectively implemented using genre and popularity attributes to deliver relevant music recommendations on a mobile platform?
2. To what extent can the integration of Generative AI enhance user experience by covering the limitations of metadata-based filtering through conversational context?

METHOD

This research was conducted using the design and development research method, which aims to design, build, and evaluate a mobile-based song recommendation application using the Spotify API, machine learning algorithms, and artificial intelligence technology. The evaluation was conducted to assess the performance of the resulting recommendation system. This research is designed to receive data from the Spotify API, then process the data using the K-Nearest Neighbor algorithm (Géron, 2022) for similarity-based recommendations, and implement an AI API to handle text prompts from users.

Mathematically, the K-Nearest Neighbor algorithm operates by calculating the distance between the target song vector (user history) and candidate song vectors in a multi-dimensional feature space. In this study, the Euclidean Distance metric is employed to quantify similarity. If we consider two songs, p and q, represented by their feature vectors (danceability, energy, valence, etc.), the distance $d(p, q)$ is calculated as:

$$d(p,q) = \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2}$$

Where q_i and p_i represent the values of the i-th audio feature for songs q and p, and n is the total number of features used. The algorithm selects k songs with the smallest distance value d to be presented as recommendations. A lower distance value indicates a higher degree of similarity between the songs. This distance-based approach ensures that

the recommended tracks share the same 'musical DNA' as the tracks the user already enjoys.

The selection of genre and popularity as the primary features for the KNN algorithm is justified by their high availability and reliability within the Spotify API's basic metadata scope. Genre provides a categorical anchor for musical taste, while popularity serves as a proxy for social relevance. Furthermore, a similarity threshold was determined through iterative testing to balance precision and serendipity; a narrower threshold was selected to ensure that recommended tracks remain strictly relevant to the user's core preferences, while the Generative AI component is tasked with introducing broader, contex-aware variety.

The data sources used in this study consist of two types, namely primary data and secondary data. Primary data is data obtained from the "Spotify Tracks Dataset" through Kaggle, which includes song features such as danceability, loudness, acoustiness, and several other important features. The data retrieval and preprocessing stages were implemented using Python libraries to handle data structures efficiently (Cunningham, 2021). In addition, historical data on songs listened to by users is obtained through the Spotify API, which will be used as seeds for recommendations. Secondary data is data obtained from relevant reports and documentation, such as the "Indonesia Millennial Report 2024" (Heriyanto, 2024), as well as various scientific articles related to song recommendation systems.

Data collection from the Spotify API is carried out through an authentication process using the OAuth 2.0 protocol with the help of a Python library called spotipy. Data collection was conducted in April 2025 during the development and testing of the song recommendation system. Data collection was carried out in stages according to the needs of each phase of system development.

In addition, real-time data collection from users is carried out through Spotify API integration. This data collection procedure involves several technical steps.

1. Register the application on the Spotify Development Dashboard to obtain Credentials such as Client ID and Client Secret.
2. Authenticate the user via the authorization URL provided by Spotify.
3. After obtaining the access token, song history data will be obtained using the GET /v1/me/top/tracks endpoint.
4. The data will then be temporarily stored in the application and used as seed for the machine learning model that has been trained with data from the dataset, thereby generating song recommendations based on similarities with songs obtained from the user's listening history.

Regarding ethical considerations, this study strictly adheres to Spotify's Developer Policy. Data privacy is maintained by utilizing the OAuth 2.0 authorization framework, ensuring that the application only accesses user listening history after explicit consent is granted. No personally identifiable information beyond the necessary musical metadata is stored permanently on external servers, and all data processing for recommendations is performed within the session's scope to protect user confidentiality.

The model architecture used in this study consists of four main interconnected components. These components are as shown in the following figure:

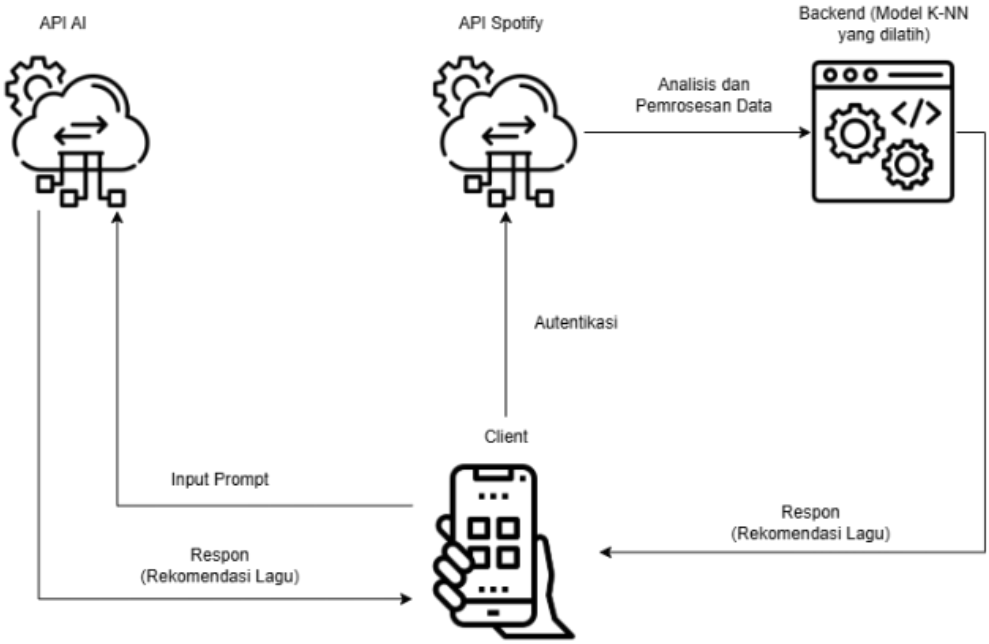

Figure 1. Model Architecture

The backend in this model architecture is the part that handles machine learning models and everything related to data, starting from processing the dataset used to train machine learning models, to generating song recommendations. In addition, the backend also handles data requests made through APIs, such as obtaining songs from user listening history, Listening Chart data, and prompt requests and responses made with the AI API. In system architecture, the Spotify API is used to obtain user listening history data, authenticate users, and obtain data for the Listening Chart from users in the form of top songs, top artists, and songs currently being listened to by users. The AI API is used to handle prompts manually entered by users to obtain more specific song recommendations, according to the characteristics of the songs desired by users. The client is an application used by users, an interface layer that allows users to access the application. Users can log in, get song recommendations, input prompts, and view their Listening Chart data.

After defining the architectural model of the system, a functional requirements analysis was conducted to describe the workflow of the system and the features of the songs used to make song recommendations. These functional requirements consist of three main components, namely input requirements, process requirements, and output requirements. Input requirements have several functions, starting from user login using a Spotify account, the system receives user listening history data and genre and popularity features obtained through the Spotify API as seed, then users input prompts to get more specific song recommendations according to user preferences.

Then, based on process requirements, the system will retrieve data and process historical data from the Spotify API. The system also applies a K-Nearest Neighbor algorithm

that has been trained using a dataset to calculate song similarity using several songs from the user's listening history as seeds. The results of these calculations are used to provide song recommendations with the highest similarity to the songs used as seeds. The AI API provides more specific song recommendations to users based on the prompts they input.

Output requirements will focus on what the system provides and displays to users, ranging from song recommendations obtained from the implemented machine learning process, AI responses based on user input prompts, and also data such as top artists, top songs, and currently playing songs from user accounts.

FINDING AND DISCUSSION

RESEARCH RESULT

The result of this research is a mobile-based song recommendation system. The application system was successfully implemented using Flutter for the frontend (Google Developer, 2024) and Flask (Python) for the backend (Grinberg, 2018) of the application. The results show that the integration of the Spotify API, the K-Nearest Neighbor (KNN) algorithm, and artificial intelligence function as intended. The implementation results are grouped into four main features, namely user authentication, implicit recommendations using the KNN algorithm and user listening history data (Home), explicit recommendations with artificial intelligence (Music Chat), and user listening statistics (Listening Chart).

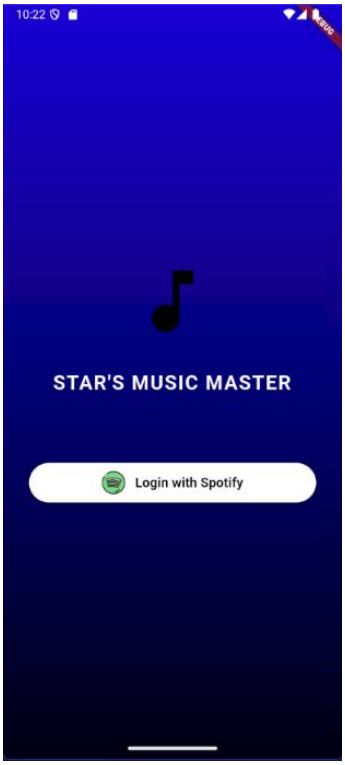

Figure 2. Login

Figure 1 shows the login page. The system uses the OAuth 2.0 protocol from the Spotify API for authentication. Users will be directed to the Spotify authorization page to obtain access permission. Once verified, users will be directed to the main page, which displays a list of recommended songs.

Figure 3. *Home*

The Home page displays a list of recommended songs. These recommendations are generated by the KNN algorithm based on the user's listening history retrieved from the Spotify API. As shown in the figure, the system successfully displays a list of songs with similarity scores that indicate the content-based filtering mechanism is working correctly.

The User Interface (UI) was built using the Flutter framework to ensure cross-platform compatibility and a smooth user experience (60fps performance). The design adopts a dark-themed aesthetic, utilizing a deep blue color palette (Hex #00008B) to reduce eye strain during nighttime use and to align with the visual identity of modern streaming apps. Each recommendation card (widget) is designed to display essential metadata clearly, including the song title, artist name, and a similarity percentage badge. This percentage helps users understand *why* a song was recommended, thereby increasing trust in the system's algorithm (Explainable AI). The list is rendered using a ListView.builder widget in Flutter, allowing for efficient memory usage even when displaying a large number of song recommendations.

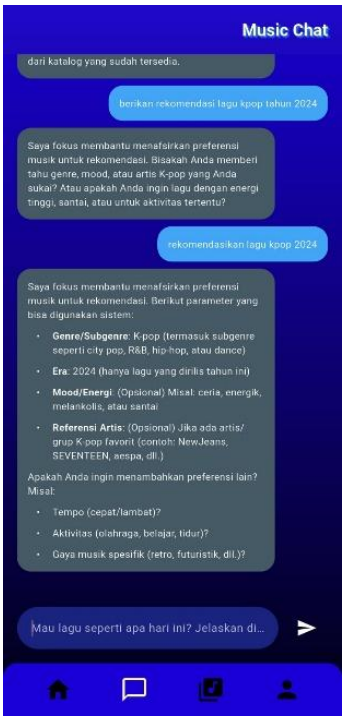

Figure 4. Music Chat

On Music Chat page, users can input text prompts to get song recommendations from the implemented AI. The recommendations are generated based on the AI model's knowledge without real-time data integration, so accuracy regarding the latest song releases is a limitation of the research.

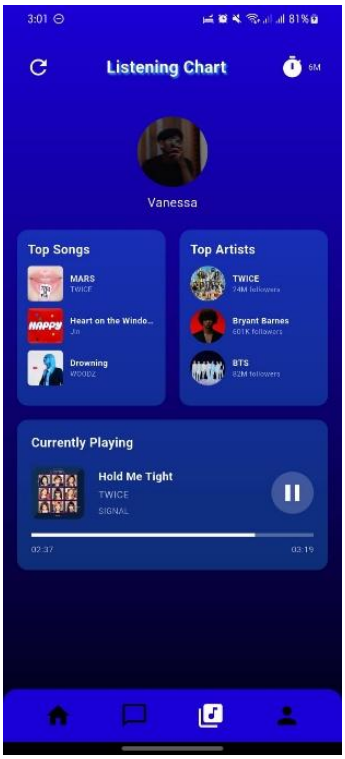

Figure 5. Listening Chart

Listening CHart page will display the top three songs most listened to by users, the top three artists listened to by users, and will display the songs currently being played by users on their Spotify accounts. This feature confirms that the data retrieval from the Spotify API is performed in real-time.

To validate the mathematical accuracy of the recommendation engine, a simulation of the K-Nearest Neighbor calculation was performed using specific seed tracks. As illustrated in Figure 6, the system processed seed tracks such as 'Zip Zap Zoe' (Genre: Rockabilly) and 'Corneta' (Genre: Electronic). The simulation results demonstrates that the algorithm successsfully identified candidate tracks with a 100% similarity match based on genre and popularity metrics in a controlled testing environtment.

Table 1: Manual Simulation of KNN Recommendation Logic

NO	Seed Track	Genre	Popularity	Recommended Track	Rec. Genre	Similarity
1	Zip Zap Zoe	Rockabilly	0.28	Haeven On Earth	Rockabilly	100.0%
2	Zip Zap Zoe	Rockabilly	0.28	Slinky	Rockabilly	100.0%
3	Zip Zap Zoe	Rockabilly	0.28	El Rock de la Carcel	Rockabilly	100.0%
4	Zip Zap Zoe	Rockabilly	0.28	Pensaba en Ti	Rockabilly	100.0%

5	Zip Zap Zoe	Rockabilly	0.28	West Virginia White Boy	Rockabilly	100.0%
6	Corneta	Electronic	0.37	Un Tiempo	Electronic	100.0%
7	Corneta	Electronic	0.37	No Lo Digas	Electronic	100.0%
8	Corneta	Electronic	0.37	Hay una Luz	Electronic	100.0%
9.	Corneta	Electronic	0.37	Ven	Electronic	100.0%
10.	Corneta	Electronic	0.37	Extraño - Versión Acústica	Electronic	100.0%

Furthermore, real-world application testing was conducted using live data from the Spotify API. Table 2 presents the recommendation outcomes when 'Like You Do' was used as the seed track. The results indicate a high degree of precision, with similarity scores ranging from 95% to 95.4%. this quantitative evidence confirms that the system maintains a consistent and high similarity rate, effectively bridging the user's historical preferences with relevant new content.

Table 2: Real-Time Recommendation Results on Mobile Application

NO	User Seed Track	Recommended Track	Similarity Score
1	Like You Do	Slow Down	95.4%
2	Like You Do	Icarus or Blériot	95.3%
3	Like You Do	Supersonic	95.1%
4	Like You Do	Stupid Games	95.1%
5	Like You Do	Flashbacks	95.0%

The similarity scores presented in the system's interface may vary based on the diversity of the metadata retreieved. For instance, in Figure 3, the system identifies tracks with 100% similarity when the candidate track's genre and popularity metrics perfectly align with the seed track. Table 2 demonstrates the algorithm's precision in identifying high affinity tracks with an average similarity of 95.18%, illustrating the system's robustness in handling broader musical variations while maintaining strong relevance.

DISCUSSION

The development of the Star's Music Master app shows that integrating streaming data from Spotify with machine learning can create a highly personalized experience. The KNN algorithm successfully processes historical data to provide content-based recommendations (implicit), while AI integration fills the gap by addressing abstract requests from users based on more specific preferences. This finding is in line with (Pelaupessy & Suhartana, 2023), which states that KNN is an effective algorithm for music classification. However, this study expands on their research by adding an AI-based conversational interface that offers a solution to the static nature of traditional

recommendation systems (Schedl, Zamani, & Chen, 2018) mentioned by (Sidora & Harani, 2023).

The integration of Generative AI (LLM) fundamentally changes the interaction model between users and music apps. Traditional Graphical User Interfaces (GUI) force users to navigate through rigid menus and filters (e.g., Select Genre -> Select Year). In contrast, the Conversational User Interface (CUI) implemented in the 'Music Chat' feature allows for natural language inputs. For instance, a query like 'I need songs to heal a broken heart but with an upbeat tempo' contains conflicting parameters (Sad content + Fast tempo) that would confuse a standard metadata filter. The LLM, however, can interpret the nuance of 'sad lyrics' (low valence) combined with 'upbeat tempo' (high BPM) and query the database accordingly.

Nevertheless, the technical overhead of this hybrid approach must be acknowledged. The system requires dual-channel processing: one channel for numerical analysis (KNN) and another for semantic processing (LLM). This increases the computational load on the backend. In the current Flask implementation, response times for the AI features average around 2-3 seconds, which is slower than the millisecond-latency of the KNN engine. Future optimizations could involve caching frequent prompts or using smaller, distilled language models to reduce latency without sacrificing too much semantic accuracy.

Beyond technical performance, several real-world deployment challenges must be considered. Scalability and computational cost represent significant hurdles; while the KNN engine is efficient, the dual-channel processing requires substantial backend resources, especially as the user base grows. Moreover, the issue of user trust is addressed through ‘Explainable AI’ features, such as the similarity badges shown in Figure 3, which clarify the logic behind each recommendation. However, the conversational nature of the AI could lead to ‘over-trust’, where users might expect the system to have a deeper emotional understanding than its current architecture allows.

Furthermore, the study acknowledges potential algorithmic bias or fairness issues. Since the KNN model heavily relies on popularity and genre metadata from the Spotify API, there is a risk of ‘popularity bias’, where mainstream tracks are favored over niche or independent artist. The integration of Generative AI partially mitigates this by allowing users to explicitly request ‘undiscovered’ or ‘indie’ music through natural language prompts, thus providing a mechanism to bypass the mathematical echo chambers created by traditional metadata filtering.

Although the implementation was successful, this study has several limitations. The recommendation feature currently uses song genre and popularity attributes. In addition, the AI model used in the Music Chat feature operates based on previously trained knowledge and does not have real-time access to the latest songs released in 2025. For future research, it is recommended to use more complex and comprehensive audio features, such as dancability and acousticness, to improve recommendation accuracy. In addition, fine-tuning AI models with up-to-date music datasets can significantly improve the relevance of recommendations.

CONCLUSION

Based on the implementation and testing results, a mobile-based song recommendation application has been successfully developed. The system is capable of providing implicit recommendations using the K-Nearest Neighbor (KNN) algorithm with a high degree of similarity to the user's listening history. In addition, the integration of artificial intelligence (AI) in the Music Chat feature allows users to obtain explicit recommendations based on specific text prompts, such as mood or activity. The Listening Chart feature also works correctly, displaying real-time statistics of the user's Spotify account. This research proves that combining machine learning with generative AI can significantly improve the user experience in music discovery.

Furthermore, the practical implications of this research extend to broader commercial and platform-level applications. The hybrid framework of combining KNN-based mathematical precision with Generative AI interfaces can be adopted by niche streaming services or digital libraries to enhance user retention through personalized discovery. At a platform level, this model offers a cost-effective way to implement context-aware recommendations without requiring massive changes to existing metadata-based infrastructures. This study provides a foundational blueprint for developing more intuitive, conversational interfaces in the entertainment and digital service sectors, effectively bridging the gap between raw data and user intent.

REFERENCES

Aggarwal, C. C. (2016). *Recommender systems: The textbook.* New York: Springer International Publishing AG Switzerland.

Bobadilla, J., Ortega, F., Hernando, A., & Gutiérrez, A. (2013). Recommender systems survey. *Knowledge-Based Systems, 46*, 109-132.

Cunningham, P. (2021). k-nearest neighbour classifiers: A tutorial. *ACM Computing Surveys, 54*(6), 1-25.

Géron, A. (2022). *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow (3rd ed.).* Sebastopol: O'Reilly Media, Inc.

Google Developer. (2024, December 8). *Flutter architectural overview*. Retrieved from Flutter Docs: https://docs.flutter.dev/resources/architectural-overview

Grinberg, M. (2018). *Flask web development: Developing web applications with Python (2nd ed.).* Sebastopol: O'Reilly Media, Inc.

Heriyanto, D. (2024). *Indonesia millenial report 2024.* Jakarta: IDN Reasearch Institute.

Madani, I., Hasrullah, & Hasrullah. (2023). Sistem rekomendasi musik Spotify berdasarkan listening history pengguna. *Jurnal Inovasi Global, 3*(2), 443-449.

Meshram, S. U. (2021). Evolution of modern web services – REST API with its architecture and design. *International Journal of Research in Engineering, Science and Management, 4*(7), 83-86.

Pelaupessy, M. R., & Suhartana, I. G. (2023). Sistem rekomendasi musik dengan menggunakan metode K-Nearest Neighbor (KNN). *Jurnal Elektronik Ilmu Komputer Udayana, 11*(4), 675-680.

Portugal, I., Alencar, P., & Cowan, D. (2018). The use of machine learning algorithms in recommender systems: A systematic review. *Expert Systems with Applications, 97*, 205-227.

Rizaldy, A. W., Dewi, R. K., & Brata, K. C. (2021). Pengembangan aplikasi rekomendasi musik berdasarkan cuaca pada platform Android. *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer, 5*(1), 124-129.

Sarmandana, I. M., Widiartha, I., Putri, L. A., & Astawa, I. S. (2024). Penerapan metode content based filtering dan K-Nearest Neighbor dalam sistem rekomendasi musik. *Jurnal Elektronik Ilmu Komputer Udayana, 13*(1), 35-42.

Schedl, M., Zamani, H., & Chen, C.-W. (2018). Current challenges and visions in music recommender systems research. *International Journal of Multimedia Information Retrieval, 7*, 95-116.

Sidora, L. I., & Harani, N. H. (2023). Sistem rekomendasi musik Spotify menggunakan KNN dan algoritma genetika. *JATI (Jurnal Mahasiswa Teknik Informatika), 7*(4), 2585-2591.