Jurnal Inovasi Teknik dan Edukasi Teknologi, 1(9), 2021, 707-714
ISSN: 2797-7196 (online)
DOI: 10.17977/um068v1i92021p707-714

Aplikasi Music Streaming Menggunakan Flutter dilengkapi Music Recognizer

Eka Rahayu Setyaningsih*, Iwan Chandra, William
Universitas Negeri Malang, Jl. Semarang No. 5 Malang, Jawa Timur, Indonesia
*Penulis korespondensi, Surel: eka.rahayu.2205349@students.um.ac.id

Paper received: Paper received: 01-9-2021; revised: 11-9-2021; accepted: 17-9-2021

Abstract

Considering the current development of music streaming services, not many are equipped with a search facility using the music recognizer. Even though there are not a few service users who search for songs only based on the song snippet or the humming tone of the song in question, without knowing the title or artist of the song. Based on this, in this study, a mobile music streaming application was created equipped with a music recognizer using audio fingerprinting that utilizes spectrogram imagery and hash data from audio. The testing process was carried out in 3 scenarios: the first scenario was a recognized test in a quiet environment, and out of 90 trials the results obtained an accuracy of 96.6%. The second scenario is to recognize the song in a noisy environment, from 90 trials, the accuracy is 93.3%. The third scenario is to recognize the song by humming or humming, from 90 trials, the accuracy is 55.5%. All these trials were carried out by 10 participants with a composition of 5 males and 5 females. Each participant will test the application made with 3 scenarios. Each scenario will be tested for 3 songs with different genres.

Keywords: music recognizer; music streaming; mobile apps; flutter; node.js

Abstrak

Menilik perkembangan layanan *music streaming* saat ini, belum banyak yang dilengkapi dengan fasilitas pencarian dengan memanfaatkan *music recognizer*. Padahal tidak sedikit pengguna layanan yang melakukan pencarian lagu hanya berdasarkan potongan lagu atau *humming* nada dari lagu yang bersangkutan, tanpa mengetahui judul atau penyanyi lagu tersebut. Berdasarkan hal tersebut, maka pada penelitian ini dibuatlah sebuah aplikasi mobile music streaming yang dilengkapi dengan music recognizer dengan *audio fingerprinting* yang memanfaatkan citra spectrogram dan data hash dari sebuah audio. Untuk proses ujicobanya dilakukan dalam 3 skenario: skenario pertama dilakukan uji coba recognize pada lingkungan yang hening, dari 90 kali uji coba mendapatkan hasil akurasi 96,6%. Skenario kedua yaitu dengan melakukan recognize pada lagu di lingkungan yang bising, dari 90 kali uji coba mendapatkan hasil akurasi 93,3%. Skenario ketiga yaitu dengan melakukan recognize pada lagu dengan humming atau bersenandung, dari 90 kali uji coba mendapatkan hasil akurasi 55,5%. Semua uji coba ini dilakukan oleh 10 peserta dengan komposisi 5 laki-laki dan 5 perempuan. Setiap peserta akan menguji aplikasi yang dibuat dengan 3 skenario. Pada setiap skenarionya akan diujikan 3 lagu dengan genre yang berbeda-beda.

Kata kunci: music recognizer; music streaming; mobile apps; flutter; node.js

1. Pendahuluan

Music recognition atau pengenalan musik pertama kali didirikan oleh Chris Barton pada tahun 1999 yang kemudian meluncurkan aplikasi pengenalan musik bernama Shazam. Terdapat beberapa aplikasi yang menyematkan music recognizer yaitu Shazam, SoundHound, MusixMatch, Genius, dll. Pada awal pengembangan Shazam, mereka menemukan beberapa masalah. Masalah pertama adalah manusia mengenali suara bukan dengan membandingkan setiap bit yang didengar dengan lagu yang tersimpan kedalam memori, tetapi sebaliknya manusia mengenali nada tertentu secara berurutan yang memicu ingatan kita. Sedangkan komputer hanya dapat membandingkan data secara harfiah dan tidak memiliki cara untuk mengenali pola secara implisit seperti manusia (Pełzyński, 2012). Masalah inilah merupakan

masalah utama dari Shazam sebagai pengembang pertama dari music recognizer. Masalah kedua adalah struktur tempat penyimpanan jutaan bahkan miliaran musik serta perincian data musik seperti judul, nama penyanyi, dll yang cepat untuk dicari. Untuk mengatasi kedua masalah tersebut, Shazam menggunakan audio hash dan tabel hash. Hal inilah yang kemudian diadaptasi dalam penelitian ini.

Pada penelitian ini dibuat aplikasi music streaming yang lagunya tidak hanya dapat dicari berdasarkan judul lagu, nama penyanyi, melainkan dapat juga dicari melalui music recognition. Dengan menggunakan music recognition, pengguna dapat melakukan pencarian lagu dengan hanya menggunakan microphone dari ponsel yang pengguna miliki. Lagu yang dihasilkan pun dapat langsung dinikmati. Selain menghasilkan lagu, aplikasi yang dibangun dalam penelitian ini juga dilengkapi web crawling yang secara terjadwal menyusuri dan melakukan fetching terhadap informasi lagu-lagu terbaru, sehingga pengguna dapat menikmati streaming lagu sekaligus melihat liriknya. Dengan menggunakan aplikasi yang dibuat, memungkinkan untuk lagu-lagu yang diputar di lingkungan yang bising sekalipun dapat dicari dengan mudah tanpa menyertakan judul lagu ataupun lirik lagu. Dalam penelitian ini terdapat tujuan yang akan dicapai, yang terdiri dari:

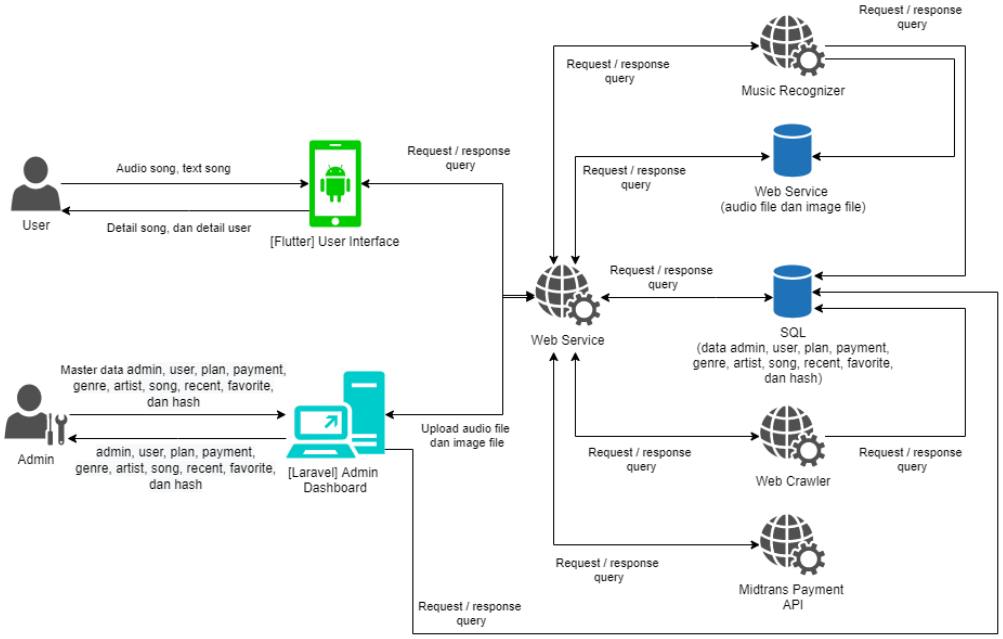
- Membantu admin untuk dapat menambahkan lagu yang dapat dikenali oleh music recognizer.
- Membantu pengguna untuk dapat melakukan pencarian lagu dengan mudah menggunakan music recognizer yang disematkan pada aplikasi.
- Membantu pengguna untuk dapat melihat lirik yang sesuai dengan lagu yang diputar.

2. Metode

Pada bagian ini akan dijelaskan mengenai rancangan sistem pada penelitian ini. Penjelasan mengenai rancangan sistem akan dibagi menjadi 2 bagian yaitu arsitektur sistem dan music recognition, dimana setiap komponen yang berperan didalamnya juga akan dibahas secara terperinci pada subbab berikut ini:

2.1. Arsitektur Sistem

Seperti yang dapat dilihat pada gambar 1 dibawah ini, sistem yang dibangun memiliki 2 aktor, yaitu pengguna dan admin. Aktor admin dapat melakukan CRUD semua data yang tersedia, seperti data admin, user, plan, payment, genre, artist, lagu, recent, favorite, dan hash. Admin juga dapat mengakses laporan data seperti laporan user yang belum dan telah melakukan pembayaran pada bulan tertentu. User interface admin dibuat dengan menggunakan framework PHP yaitu Laravel. Salah satu fitur admin yang penting adalah master data lagu. Fitur master data lagu ini memuat beberapa fungsi seperti CRUD link streaming lagu, judul dan lirik yang diperoleh dari web crawler, serta fungsi HTML editor untuk mengubah struktur HTML yang di akses saat proses fetching data yang dilakukan oleh web crawler. Pada music recognizer, audio file akan dikonversi menjadi audio hash yang nantinya dapat digunakan untuk pencarian lagu. Penelitian ini menggunakan SQL sebagai tempat penyimpanan data.



Gambar 1. Arsitektur Diagram

Aktor user/ pengguna dapat melakukan pencarian dengan mengetikan judul lagu, nama penyanyi, genre lagu, ataupun dengan memberikan audio yang dapat diperoleh langsung dari microphone ponsel pengguna. Nantinya input dari user akan diproses oleh web service dan web service akan melakukan request kepada music recognizer. Hasil response dari music recognizer akan berupa kumpulan id lagu yang dirasa cocok oleh music recognizer, dan diurutkan berdasarkan derajat kecocokannya. Selanjutnya web service akan mencari informasi lebih lengkap pada database berdasarkan id lagu yang diperoleh dari hasil response music recognizer dan akan mengembalikannya kepada pengguna untuk dinikmati.

Karena aplikasi yang dikembangkan bersifat subscription, maka pengguna juga dapat melakukan pembayaran. Pembayaran dapat dilakukan dengan menggunakan Midtrans API. Apabila pembayaran berhasil dilakukan, maka bukti dari pembayaran akan ditambahkan kedalam database dan juga akan dikirimkan kepada pengguna melalui email.

2.2. Music Recognition

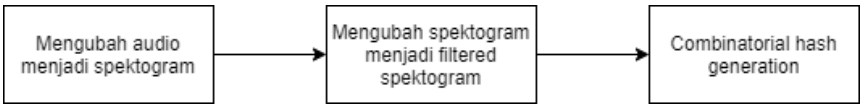
Music recognition merupakan suatu algoritma yang digunakan untuk mengenal suatu musik. Terdapat 3 cara untuk mengenali sebuah musik yaitu, Query by Text (QBT), Query by Example (QBE), dan Query by Humming (QBH).

2.2.1. Query by Text

Query by Text (QBT) menggunakan metadata konseptual seperti text query untuk mencari kesamaan antara lagu tertentu. Aplikasi seperti Spotify menggunakan fitur ini untuk sistem pencarian musik. Cara ini merupakan cara pertama yang diperkenalkan di bidang music recognition karena kemudahan penggunaannya yang bergantung pada teks yang diketahui sebelumnya dan dapat dicari melalui database. Teks yang dapat dicari dapat berupa beberapa hal meliputi judul lagu, penyanyi lagu, dan genre lagu.

2.2.2. Query by Example

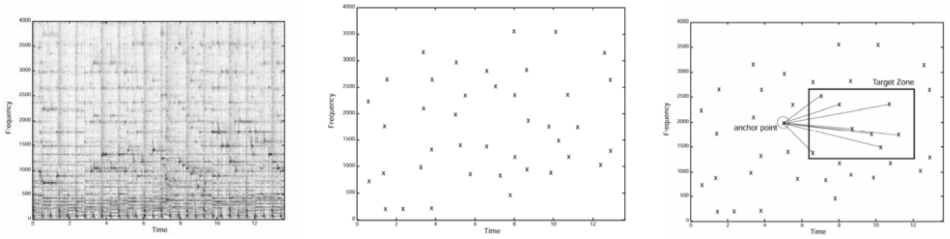
Query by Example (QBE) menggunakan potongan dari musik asli yang telah direkam sebelumnya dan mencari pada database untuk mengambil lagu yang mirip. Aplikasi seperti Shazam menggunakan fitur ini untuk sistem pencarian musik. Pada saat pengembangan Shazam, mereka menemukan beberapa masalah. Masalah pertama adalah manusia mengenali suara bukan dengan membandingkan setiap bit yang didengar dengan lagu yang tersimpan kedalam memori, tetapi sebaliknya manusia mengenali nada tertentu secara berurutan yang memicu ingatan kita. Sedangkan komputer hanya dapat membandingkan data secara harfiah dan tidak memiliki cara untuk mengenali pola secara implisit seperti manusia. Masalah inilah merupakan masalah utama dari Shazam sebagai pengembang pertama dari Query by Example ini. Untuk mengatasi masalah diatas dan masalah struktur penyimpanan jutaan bahkan miliaran musik yang dapat dicari, Shazam menggunakan Audio Hash.



Gambar 2. Alur Pembuatan Audio Hash

Audio hash merupakan suatu solusi yang dikembangkan oleh Shazam untuk mengatasi permasalahan pengenalan suatu musik dari jutaan bahkan miliaran musik yang ada di dunia. Untuk dapat membuat audio hash, seperti yang ditunjukkan pada gambar 2, audio file akan dikonversi menjadi bentuk spektogram dengan menggunakan FFT melalui window kecil pada setiap detik sample lagu. Spektogram adalah grafik visual frekuensi suara sepanjang waktu di sepanjang sumbu x, sumbu y, dan sumbu z . Sumbu z sendiri adalah gradien warna untuk mewakili amplitudo frekuensi (Wang, 2003).

Hanya dengan menggunakan spektogram (gambar 3a), sebenarnya pencarian lagu sudah dapat dilakukan; yaitu dengan membandingkan antara spektogram dari user dengan spektogram yang ada di database. Akan tetapi, untuk membandingkan keduanya membutuhkan terlalu banyak perhitungan dan waktu. Sehingga spektogram yang semula 3 dimensi diubah menjadi 2 dimensi yang disebut filtered spektogram seperti yang dapat dilihat pada gambar 3b.

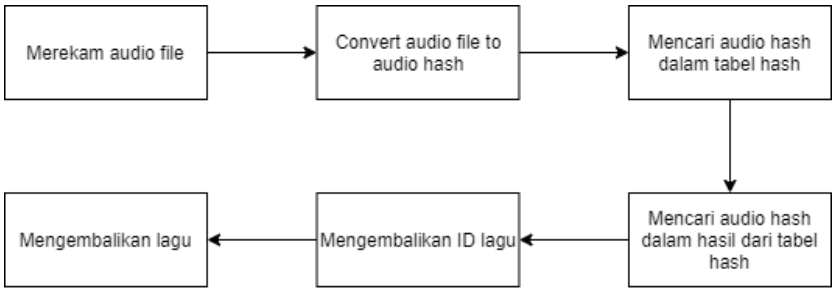


(a) **(b)** **(c)**

Gambar 3. Spectrogram

Dalam filtered spektogram, sumbu z yang mewakili amplitudo dari frekuensi akan dihapus dan hanya frekuensi dengan amplitudo tertinggi atau nada yang kuat yang disimpan di spektogram ini. Hal ini akan mengurangi jumlah data yang diperlukan untuk mewakili satu

lagu. Tidak hanya itu, frekuensi dengan amplitudo rendah yang bisa dianggap noise atau bukan termasuk dalam lagu aslinya akan otomatis hilang. Langkah terakhir akan dilakukan combinatorial hash generation. Melalui proses ini, salah satu titik pada filtered spektogram dipilih untuk menjadi jangkar dan dihubungkan dengan titik lain pada spektogram pada waktu tertentu. Hal ini dinamakan dengan target zone, seperti yang dapat dilihat pada gambar 3c. Setiap pasangan titik jangkar akan disimpan kedalam tabel yang berisi frekuensi titik jangkar, frekuensi titik lain, dan waktu antara kedua titik (T. Jie, 2009). Data inilah yang kemudian akan digabungkan dengan tanda pengenal musik yaitu id musik dan disimpan dalam bentuk hashnya ke dalam tabel yang terpisah dari detail informasi lagu.



Gambar 4. Alur Proses Pencarian Lagu

Seperti yang dilihat pada gambar 4, untuk pencarian lagu memiliki beberapa tahap. Pertama, pengguna akan merekam audio file menggunakan microphone ponsel pengguna. Setelah itu akan dilakukan konversi dari audio file menjadi audio hash. Pada proses ini, data audio file pengguna akan diubah menjadi spektogram dan diubah menjadi filtered spektogram. Setelah itu filtered spektogram akan dilakukan combinatorial hash generation. Semua hasil dari combinatorial hash generation tadi akan diubah menjadi audio hash.

Audio hash tadi akan dilakukan pencarian kecocokan pada tabel hash yang tersedia berdasarkan hash dan waktu antara kedua titik. Setelah itu hasil dari kecocokan tabel hash tersebut, akan dilakukan pencocokan lagi terhadap audio file pengguna tetapi yang dicocokkan adalah jarak antar hash-nya. Dengan mencocokkan jarak antar hash-nya, memungkinkan untuk mencari musik yang sesuai tanpa melihat adanya waktu. Dengan kata lain, input musik dapat dilakukan di awal, tengah atau bahkan di akhir lagu. Apabila terdapat kecocokan pada jarak antar hash-nya, maka akan menghasilkan id lagu. Data id lagu tersebut akan dicari pada tabel lagu. Hasil dari tabel lagu merupakan hasil akhir yang berupa data perincian lagu yang lengkap seperti id lagu, judul lagu, nama penyanyi, dll. Data inilah yang menjadi output dari Query by Example ini.

2.2.3. Query by Humming

Query by Humming (QBH) menggunakan suara senandung alami yang dipancarkan dari manusia untuk melakukan query pada database (Patel, 2019). Selain itu, pendekatan ini sangat cocok karena suara senandung terjadi secara alami dan dapat melekat dipikiran manusia. Aplikasi seperti SoundHound menggunakan fitur Query by Example dan Query by Humming sebagai sistem pencarian musik (M. Antonelli, 2010) (Wei, 2011). Query by Humming memiliki beberapa pendekatan yang dapat dilakukan. Salah satunya adalah dengan menggunakan Dynamic Time Warping (DTW).

Dynamic Time Warping (DTW) adalah sebuah algoritma yang secara dinamis mengukur dua deret waktu yang berbeda dan menghasilkan sebuah kesamaan (Amin & Mahmood, 2008). Hal ini dapat memudahkan Query by Humming karena manusia memiliki kemampuan bersenandung yang berbeda-beda dalam segi kecepatan, dan ketepatan nada. Algoritma ini dirancang untuk ketahanan dan dapat mencapai pengambilan tinggi ketepatan. Algoritma ini memiliki fungsi untuk menghitung sebuah cost atau biaya yang dibutuhkan agar satu variabel menyerupai variabel yang lain. Algoritma Dynamic Time Warping ini akan digunakan untuk Query by Humming (Putri & Lestari, 2015). Tujuannya adalah mencari cost terkecil dari input humming manusia dengan lagu asli atau lagu yang terdapat pada database. Hasil dari algoritma Dynamic Time Warping inilah yang akan menjadi hasil dari Query by Humming.

3. Hasil dan Pembahasan

Pada tahap uji coba, digunakan 150 data lagu dengan komposisi 25 lagu tiap genrenya. Terdapat 5 genre lagu, dimana pada setiap genrenya terdapat 5 penyanyi yang berbeda, dan pada setiap penyanyi terdaftar sekurang-kurangnya 5 lagu. Uji coba ini akan dilakukan oleh 10 peserta dengan komposisi 5 pria dan 5 wanita. Setiap peserta melakukan 9 kali uji coba, 3 lagu pada 3 genre yang berbeda dengan 3 skenario uji coba, yaitu peserta uji coba bernyanyi pada kondisi lingkungan tenang, pada kondisi lingkungan bising, dan proses pencarian berdasarkan suara humming.

Pada tabel 1 telah menunjukkan bahwa sebanyak 90 kali uji coba dengan menggunakan metode 1 yaitu melakukan recognize lagu dengan situasi sekitar tenang menghasilkan 87 hasil yang tepat. Hasil dari metode pertama menghasilkan akurasi 96,6%. Sebanyak 90 kali dengan menggunakan metode 2 yaitu melakukan recognize lagu dengan situasi sekitar yang ramai menghasilkan 84 hasil yang tepat. Hasil dari metode kedua ini menghasilkan akurasi 93,3%. Sebanyak 90 kali uji coba dengan menggunakan metode 3 yaitu melakukan recognize lagu dengan humming atau bersenandung menghasilkan 50 hasil yang tepat sehingga akurasi untuk metode 3 ini 55,5%.

Tabel 1. Hasil Uji Coba dengan 3 Skenario

Metode	Jumlah Uji Coba	Jumlah Ketepatan
Recognize lagu dengan situasi tenang	90	87
Recognize lagu dengan situasi ramai (noise 10db)	90	84
Recognize lagu dengan humming	90	50

Rata-rata durasi yang dibutuhkan untuk melakukan proses recognize pada setiap uji coba adalah 14,5 detik. Panjang potongan audio untuk di-recognize merupakan 7 detik. Panjang potongan 7 detik ini didapatkan karena merupakan nilai yang paling optimal. Sebelumnya sudah dilakukan uji coba oleh developer dengan menggunakan 4 detik sampai 6 detik, tetapi hasil akurasi yang diberikan tidak optimal. Lalu juga dicoba dengan menggunakan 8 detik sampai 10 detik, tetapi mengorbankan durasi proses recognize, sehingga nilai 7 detik dirasa paling optimal yaitu menghasilkan hasil yang cukup memuaskan tanpa mengorbankan durasi proses recognize.

4. Simpulan

Dari hasil uji coba dapat disimpulkan bahwa dari sebanyak 90% peserta uji coba melaporkan bahwa hasil yang didapatkan dari music recognizer cukup akurat dengan sumber potongan lagu asli. Sedangkan jika menggunakan sumber suara dengan humming menampilkan hasil yang tidak optimal karena sumber suara menghasilkan frekuensi yang tidak sesuai dengan database

Daftar Rujukan

Wang, A. (2003). An Industrial Strength Audio Search Algorithm. *ISMIR 2003, 4th International Conference on Music Information Retrieval* (p. 27). Baltimore, Maryland, USA: ISMIR.

Patel, P. (2019). *Music Retrieval System Using Query-by-Humming.* San Jose: San Jose State University.

Putri, R. A., & Lestari, D. P. (2015). Music information retrieval using Query-by-humming based on the dynamic time warping. *International Conference on Electrical Engineering and Informatics, ICEEI* (pp. 65-70). Denpasar, Indonesia: IEEE.

M. Antonelli, A. R. (2010). A Query by Humming System for Music Information Retrieval. *2010 10th ICISDA* (pp. 586-591). Kairo: IEEE.

Pełzyński, K. A. (2012). Melody recognition system. *2012 Joint Conf. NTAV/SPA, Lodz* (pp. 115-118). Poland: IEEE.

T. Jie, L. G. (2009). Improved Algorithms of Music Information Retrieval Based on Audio Fingerprint. *2009 Third ICIITAW* (pp. 367-371). Nanchang: IEEE.

Wei. (2011). An improved feature extraction algorithm of humming music. *roc. 2011 Int. Conf. on Trans., Mech., and Elect. Eng. (TMEE)* (pp. 2500-2503). Changchun: IEEE.

Amin, T. B., & Mahmood, I. (2008). Speech Recognition using Dynamic Time Warping. *2008 2nd International Conference on Advances in Space Technologies* (pp. 74-79). Islamabad, Pakistan: IEEE.