Jutisi: Jurnal Ilmiah Teknik Informatika dan Sistem Informasi
Jl. Ahmad Yani, K.M. 33,5 - Kampus STMIK Banjarbaru
Loktabat – Banjarbaru (Tlp. 0511 4782881), e-mail: puslit.stmikbjb@gmail.com
e-ISSN: 2685-0893
p-ISSN: 2089-3787

Pengembangan *Bot Discord* Sebagai Pemutar dan Rekomendasi Musik Menggunakan Metode *K-Means*

Deva Dwi Satrio[1*], Fawwaz Ali Akbar[2], Muhammad Muharrom Al Haromainy[3]
Informatika, UPN Veteran Jawa Timur, Surabaya, Indonesia
*e-mail *Corresponding Author. 19081010185@student.upnjatim.ac.id*

Abstract

In the digital age, video gaming and music streaming have experienced rapid growth. Many gamers face challenges in time efficiency while playing, particularly when listening to music and communicating with teammates during video games. The Discord platform, providing voice channel spaces for gamers to communicate and command Discord bots, offers a potential solution to time efficiency issues. This paper aims to create an efficient Discord bot capable of searching, streaming, and providing recommendations to users to streamline their communication and gaming experiences. By integrating Spotify and YouTube APIs for attribute retrieval and music streaming, the K-Means method can be applied within the Spotify API for music recommendations. Implementing a Discord bot that can stream and recommend music aims to enhance the focused and enjoyable gaming experience for players.

Keyword: *K-Means Clustering; Bot Discord; Spotify Music Recommendation System; Application*

Abstrak

Dalam era serba digital, permainan video dan *streaming* musik berkembang sangat pesat. Banyak gamer menghadapi tantangan efisiensi waktu saat bermain, terutama dalam hal mendengarkan musik dan berkomunikasi dengan rekan tim dalam permainan *video*. Dengan adanya platform *Discord* yang menyediakan ruang kanal suara untuk gamer berkomunikasi serta memerintah *bot Discord* maka efisiensi waktu akan sangat memungkinkan. Penulisan paper ini bertujuan menciptakan *bot Discord* yang secara efisien dapat melakukan pencarian, *streaming*, dan memberikan rekomendasi kepada pengguna agar dapat meringkas waktu ketika berkomunikasi dan bermain permainan *video*. Dengan mengintegrasikan API *Spotify* dan API Youtube untuk melakukan pengambilan atribut dan menyiarkan musik, *metode K-Means* dapat diterapkan dalam API *Spotify* untuk melakukan rekomendasi musik. Dengan mengimplementasikan *bot Discord* yang bisa menyiarkan serta merekomendasikan musik, maka akan tercipta pengalaman bermain yang lebih terfokus serta menyenangkan bagi para gamer.

Kata kunci: *Klasterisasi K-Means; Bot Discord; Sistem Rekomendasi Musik Spotify; Aplikasi*

1. Pendahuluan

Dalam menghadapi era digital yang terus berkembang, permainan video dan layanan streaming musik menjadi aspek penting dalam gaya hidup modern, terutama bagi komunitas gamer. Penggunaan *Discord* sebagai platform komunikasi untuk koordinasi dalam permainan membuka peluang untuk meningkatkan efisiensi dan pengalaman bermain. Tidak jarang, banyak gamer yang mendengarkan musik ketika bermain game. Musik adalah sesuatu yang dianggap menyenangkan, ceria, mempunyai ritme/irama, melodi, serta *timbre* tertentu yang dapat mengolaborasikan antara tubuh dan pikiran [1]. Penelitian ini bertujuan untuk mengefisiensikan penggunaan komputer bagi *gamer* ketika bermain sembari mendengarkan musik. Dengan mengintegrasikan layanan steaming musik khususnya *Spotify* dan *YouTube*, dengan *Discord*, untuk menciptakan solusi yang dapat mengefisiensikan pengalaman mendengarkan musik dan bermain permainan video.

Faktanya, dalam kehidupan sehari-hari masih banyak gamer yang menggunakan tiga aplikasi sekaligus secara berbarengan sebagai media komunikasi sesama tim, mendengarkan musik, dan aplikasi permainan video. Hal ini menyebabkan turunnya performa kecepatan komputer atau laptop dikarenakan penggunaan aplikasi secara tidak efisien. Banyaknya

aplikasi yang berjalan dapat membuat *RAM* bekerja terlalu keras dalam mengalokasikan seluruh ruang yang dimilikinya pada aplikasi yang berjalan saat start-up. Hal ini tentu membuat *RAM* sangat terbebani [2]. Namun, aplikasi komunikasi dan *streaming* musik tersebut dapat diringkas dengan menggunakan platform *Discord* sebagai media komunikasi, dan memanfaatkan *bot Discord* sebagai media pemutar musik.

Discord adalah *platform* komunikasi yang memungkinkan pengguna untuk melakukan percakapan teks, gambar, suara, video, dan bahkan berbagi layar. *Discord* merupakan aplikasi yang seringkali digunakan oleh para gamers untuk berkomunikasi [3]. *Platform* ini memiliki banyak fitur seperti *channel, live streaming,* serta panggilan video. Untuk memenuhi kebutuhan penggunanya maka *Discord* membuat sebuah wadah yaitu *Bot Discord Developer* yang bisa mengembangkan *Bot* secara kreatif, bebas, dan tanpa batas. *Bot Discord* mulanya dibuat dengan tujuan membuat *server Discord* menjadi lebih hidup yang lebih interaktif dan menarik [4]. Namun dalam perkembangannya, *Bot Discord* saat ini bisa dijadikan berbagai macam aplikasi tersendiri dan salah satunya pemutar musik.

Paper ini akan membahas pembuatan *bot Discord* sebagai pemutar musik dengan tujuan untuk mengefisienkan para pengguna *Discord* dalam mendengarkan lagu ketika bermain *game* dan rekomendasi musik menggunakan *metode k-means*. *K-means* merupakan salah satu algoritma dalam *data mining* yang bisa digunakan untuk melakukan pengelompokan atau *clustering* pada suatu data [5]. Ada banyak pendekatan untuk membuat *cluster*, diantaranya adalah membuat aturan yang mendikte keanggotaan dalam *grup* yang sama berdasarkan tingkat persamaan diantara anggota-anggotanya. K-means merupakan metode yang termasuk dalam algoritma *clustering* berbasis jarak yang membagi data ke dalam sejumlah cluster dan algoritma ini hanya berkerja pada atribut numerik [6]. Metode ini dipilih karena dapat digunakan untuk memperoleh kelompok-kelompok musik yang memiliki kesamaan dalam karakteristik musiknya melalui sembilan atribut numerik, yaitu *danceability, energy, accousticness, instrumentalness, loudness, liveness, speechiness, tempo,* dan *valence* [7]. Dengan adanya alasan tersebut, *Spotify* mendapatkan respon dan apresiasi positif melalui kumpulan musik yang dimilikinya secara lebih lengkap [8]. Sehingga dapat memberikan rekomendasi musik yang sesuai dengan selera musik *gamer*.

2. Tinjauan Pustaka

Penelitian terdahulu oleh Buchori Anantya Firdaus, Dian Eka Ratnawati, dan Buce Trias Anggara dalam jurnal "Klasterisasi Popularitas Artis pada Playlist Today's Top Hits Menggunakan Metode *K-Means* dengan Integrasi *Spotify Web API* dan Teknologi *Amazon SageMaker*" (2021) menyebutkan bahwa penerapan penelitian klasterisasi *playlist today top hits* Spotify ini dapat menggunakan metode *K-Means* secara berguna, dikarenakan metode partisi ruang dapat digunakan untuk mempercepat proses rekomendasi [9]. Dalam metode partisi ruang, *K-means* dapat menangani data musik yang besar secara efektif. Pada proses pengujian kluster menggunakan *Silhouette Coeficient*, menunjukkan nilai direntang 0.540, yang berarti hasil klaster yang terbentuk baik karena nilai menunjukkan +1. Sedangkan menggunakan *Davies-Bouldin Index* menunjukkan hasil direntang 0.553, yang berarti hasil kluster yang terbentuk baik karena nilai DBI rendah.

Penelitian terdahulu oleh Ully Laili Musyarofah, Suci Nur Alima, dan Dhian Satria Yudha K dalam jurnal "Klasifikasi Top 50 Spotify Tahun 2010-2019 Menggunakan Metode K-Means Clustering" (2022) menyebutkan bahwa dalam penerapan penelitiannya, klasifikasi ini dipengaruhi oleh *valence* (valensi musik), genre, tempo, ketukan atau *beat perminute* (BPM), dan sebagainya [10]. Dalam menemukan klaster yang sesuai, penelitian diperoleh dari hasil analisis *cluster pie chart*, klaster diagram batang, dan titik koordinat hubungan valensi musik.

Penelitian terdahulu oleh Rizki Septiansyah, Sabriansyah Rizqika Akbar, dan Rizal Maulana dalam jurnal "Perancangan Bot Discord Untuk Pembacaan Sensor Di Raspberry Pi Dengan Sistem Learning Yand Dinamis" (2018) meyebutkan bahwa pada implementasinya bot yang dirancang mampu untuk melakukan beberapa perintah, diantaranya adalah perintah learn digunakan agar bot dapat mempelajari suatu pembacaan sensor berdasarkan perintah yang diberikan oleh user pada discord [11]. Perintah *command* digunakan agar bot dapat menampilkan nilai pembacaan sensor pada *Raspberry* Pi, nilai di tampilkan dalam teks chat pada *discord* perintah *forgot* digunakan agar bot dapat menghapus sebuah perintah pembacaan sensor yang telah dipelajari sehingga tidak dapat digunakan kembali sampai dipelajari kembali.

Penelitian Terdahulu oleh Muhammad Ikhsan Firmansyah, Ramdhan Saepul Rohman, dan Eva Marsusanti dalam jurnal "Penerapan Algoritma Klastering *K-Means* Untuk Fitur Atribut Pada Layanan *Streaming* Musik *Spotify*" (2023) menyebutkan bahwa Setiap lagu memiliki karakteristik nada yang unik, seperti *danceability, energy, loudness, speechiness, acousticness, instrumentalness, liveness, valence*, dan tempo [12] dapat dilakukan penerapan klasterisasi *K-Means* untuk membuat rekomendasi musik sesuai anggota terdekat dalam klaster.

Penelitian terdahulu oleh Ana Rohmah Zaidah, Chandra Indira Septiarani, Mar'atus Sholikhatun Nisa, Ahmad Yusuf, dan Noor Wahyudi dalam jurnal "Komparasi Algoritma *K-Means, K-Medoid, Agglomeartive Clustering* Terhadap *Genre Spotify*" (2021) menyebutkan bahwa *Spotify* sebagai *platform* yang memiliki pengguna yang banyak tentu perlu sebuah penelitian lebih lanjut tentang streaming musik yang ditawarkan untuk menawarkan *user experience* yang lebih baik dan upaya meningkatkan persaingan dengan platform streaming lain melalui analisis data mining. Pada penelitian tersebut, penulis menggunakan data publik Global Top 50 yang akan dikelompokkan berdasarkan genrenya menggunakan *algoritma K Means* [13].

Pada prinsipnya, *bot Discord* yang dikembangkan dalam penelitian ini adalah sama dengan [9-13], perbedaannya adalah pada pengaplikasian bentuk program klasifikasi *K-Means* sebagai rekomendasi musik yang diterapkan dalam *bot Discord* dengan perintah secara tekstual serta memiliki *output* suara yang dapat dirasakan secara langsung dengan fitur pemutar musik.

3. Metodologi

SLDC (*Systems Development Life Cycle*) merupakan point yang sangat vital, krusial, dan keputusan didalam *Software development* pada sebuah proyek. sukses atau tidaknya sebuah proyek sudah bisa diprediksi pada saat manajer proyek menentukan model *SLDC* mana yang akan diambil. *Model Waterfall* adalah model pertama digunakan dan umum digunakan dan umum digunakan oleh *project-project* pemerintahan dan perusahaan besar. Model ini juga menekankan pentingnya dokumentasi sehingga model ini cocok untuk proyek yang mengedepankan kualitas.[14]

3.1 Analisa Kebutuhan

Analisa kebutuhan sistem informasi memiliki peran yang cukup besar dalam pengembangan suatu sistem karena merupakan titik awal yang menjadi acuan dari langkah-langkah selanjutnya [15]. Analisa ini dilakukan untuk mendefinisikan berbagai macam hal yang diperlukan dalam pembuatan pemutaran dan rekomendasi musik menggunakan *bot Discord* seperti permasalahan dan kebutuhan pengguna, serta fitur-fitur fungsional yang diharapkan bisa memenuhi keinginan pengguna oleh *bot Discord* yang dikembangkan penulis. Pengguna *bot Discord*, dalam hal ini adalah para pengguna platform Discord dilibatkan bersama oleh penulis mengenai pengembangan bot Discord serta analisis kebutuhan fungsionalnya.

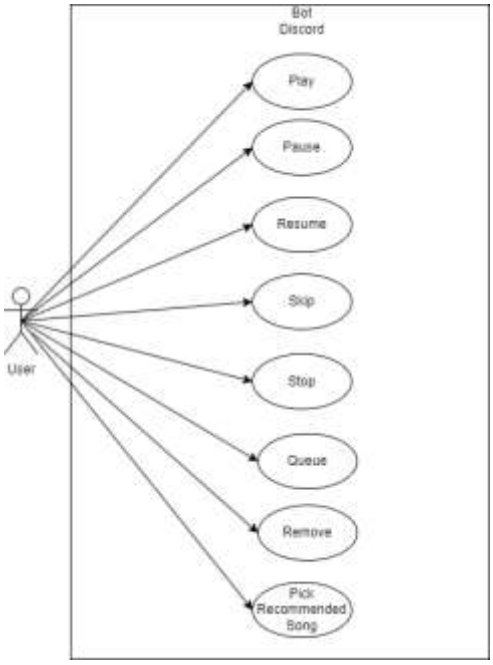
Pokok utama kebutuhan fungsionalnya adalah: dapat memutar musik secara lancar, memiliki perintah untuk melihat antrean musik, jeda atau lanjut, melewati musik, dan menghapus musik dari antrean, serta menghentikan bot Discord dalam melakukan pemutaran.

3.2. Tahap Desan Sistem

Desain sistem merupakan salah satu tahapan setelah melakukan analisa kebutuhan sistem untuk menetapkan proses yang akan diperlukan untuk pembuatan *bot Discord*. Desain-desain tersebut berupa *flowchart*, yaitu:

1) *Use Case Diagram*

Dalam diagram *use case* pada gambar 1 dijelaskan mengenai fitur-fitur ataupun kegiatan yang bisa dilakukan pengguna ketika menggunakan atau mengakses *bot Discord* yang dibuat oleh penulis.

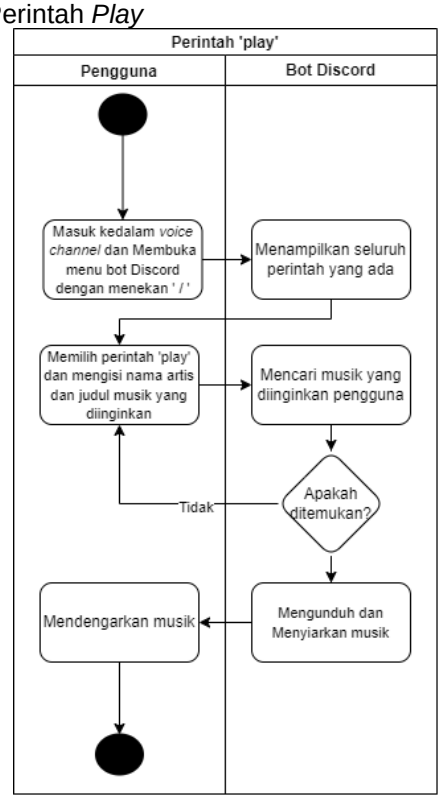


Gambar 1. *Use Case Diagram*

2) *Activity Diagram*

Activity Diagram menggambarkan *workflow* atau alur kerja serta aktivitas fungsional dalam sebuah sistem atau proses bisnis serta fungsi-fungsi yang ada pada *bot Discord*.

a. *Activity Diagram* Perintah *Play*

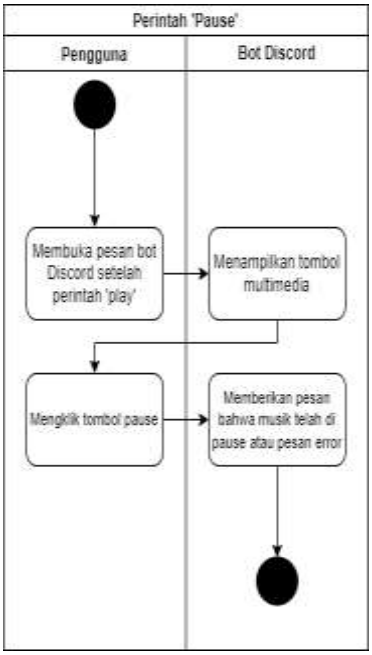


Gambar 2. *Activity Diagram* Perintah *Play*

Gambar 2 merupakan *Activity Diagram* pengguna melakukan perintah *play* pada bot Discord. Diagram pada gambar 2 menunjukkan bahwa penggunaan *bot Discord* dimulai dengan

mengetikkan ‘/’ sebagai tanda pembuka menu lalu memilih perintah *play* untuk melakukan pemutaran musik.

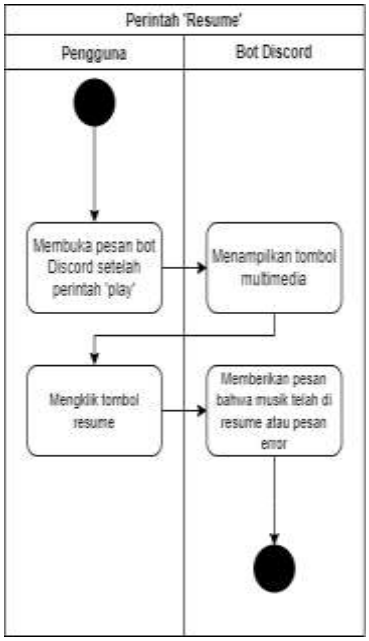
b. *Activity Diagram* Perintah *Pause*



Gambar 3. Activity Diagram Perintah *Pause*

Gambar 3 merupakan *Activity Diagram* pengguna melakukan perintah *pause* pada bot Discord. Diagram pada gambar 3 menunjukkan bahwa penggunaan *bot Discord* untuk melakukan perintah *pause* adalah dimulai dengan perintah pertama yaitu *play* lalu menekan tombol *pause*.

c. *Activity Diagram* Perintah *Resume*



Gambar 4. *Activity Diagram* Perintah *Resume*

Gambar 4 merupakan *Activity Diagram* pengguna melakukan perintah *resume* pada bot Discord. Diagram pada gambar 4 menunjukkan bahwa penggunaan *bot Discord* untuk melakukan perintah *resume* adalah dimulai dengan perintah pertama yaitu *play* lalu menekan tombol *resume*.

d. *Activity Diagram* Perintah *Skip*

Gambar 5. *Activity Diagram* Perintah *Skip*

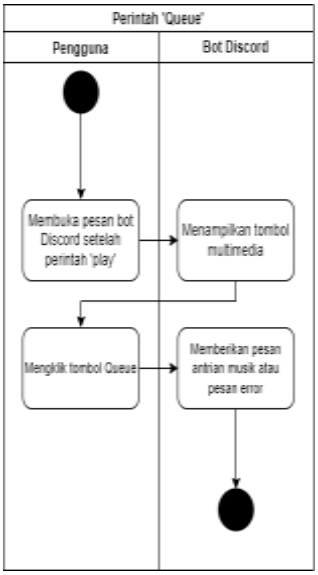
Gambar 5 merupakan *Activity Diagram* pengguna melakukan perintah *skip* pada bot Discord. Diagram pada gambar 5 menunjukkan bahwa penggunaan *bot Discord* untuk melakukan perintah *skip* adalah dimulai dengan perintah pertama yaitu *play* lalu menekan tombol *skip*. Kemudian muncul menu untuk memasukkan berapa banyak musik yang ingin dilewati, kemudian menekan tombol konfirmasi.

e. *Activity Diagram* Perintah Stop

Gambar 6. *Activity Diagram* Perintah Stop

Gambar 6 merupakan *Activity Diagram* pengguna melakukan perintah stop pada bot Discord. Diagram pada gambar 6 menunjukkan bahwa penggunaan *bot Discord* untuk melakukan perintah *stop* adalah dimulai dengan perintah pertama yaitu *play* lalu menekan tombol stop. Kemudian, pengguna melakukan konfirmasi.

f. *Activity Diagram* Perintah *Queue*



Gambar 7. *Activity Diagram* Perintah *Queue*

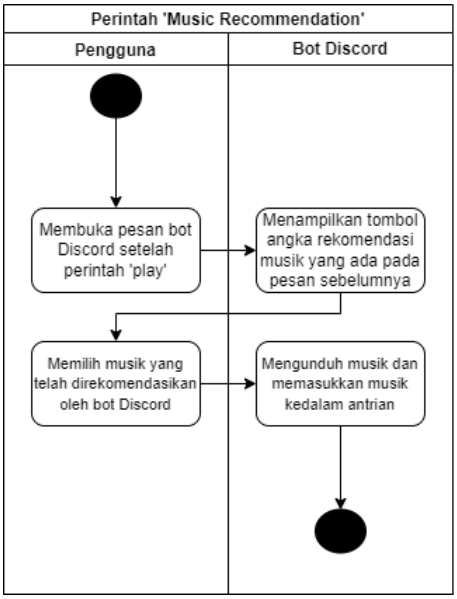
Gambar 7 merupakan *Activity Diagram* pengguna melakukan perintah queue atau melihat antrean musik pada *bot Discord*. Diagram pada gambar 7 menunjukkan bahwa penggunaan *bot Discord* untuk melakukan perintah *queue* adalah dimulai dengan perintah pertama yaitu *play* lalu menekan tombol *queue*. Bot Discord akan mengirimkan pesan berupa urutan antrean musik.

g. *Activity Diagram* Perintah *Remove*

Gambar 8. *Activity Diagram* Perintah *Remove*

Gambar 8 merupakan *Activity Diagram* pengguna melakukan perintah *remove* musik pada bot Discord. Diagram pada gambar 8 menunjukkan bahwa penggunaan *bot Discord* untuk melakukan perintah *remove* adalah dimulai dengan perintah pertama yaitu *play* lalu menekan tombol *remove*. Kemudian muncul menu untuk memasukkan urutan keberapa musik yang ingin dihapus, kemudian menekan tombol konfirmasi.

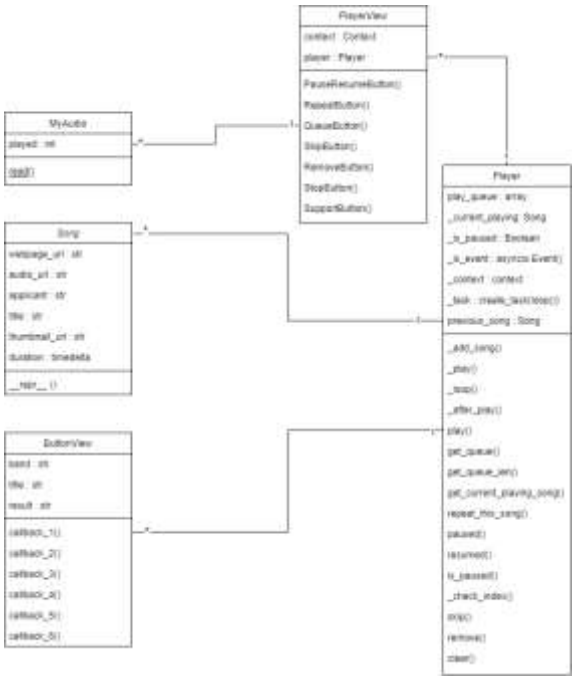
h. *Activity Diagram* Rekomendasi Musik



Gambar 9. *Activity Diagram* Rekomendasi Musik

Gambar 9 merupakan *Activity Diagram* pengguna melakukan pemilihan rekomendasi musik pada *bot Discord*. Diagram pada gambar 9 menunjukkan bahwa penggunaan *bot Discord* untuk melakukan pemilihan rekomendasi musik adalah dimulai dengan perintah pertama yaitu *play* lalu menekan tombol rekomendasi musik sesuai dengan pesan yang muncul ketika perintah *play* dijalankan.

3) *Class Diagram*



Gambar 10. *Class Diagram* Program Bot Discord

Gambar 10 merupakan sebuah *class diagram* yang menggambarkan kelas-kelas dalam sebuah sistem dan hubungannya antara satu dengan yang lain.

4. Hasil Dan Pembahasan

4.1 Hasil

Beberapa tampilan menu dan tampilan fungsi yang ada pada *bot Discord* disajikan dalam bentuk gambar sebagai berikut:

1) Urutan Menu

Gambar 11. Menu Bot Discord

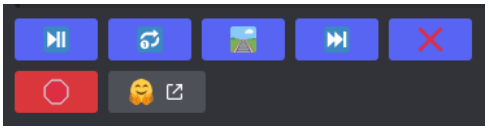
Menu diatas adalah perintah yang bisa dilakukan oleh bot Discord. Perintah utamanya adalah *play* yang digunakan untuk memulai pemutaran musik. Untuk perintah ping adalah mengecek keadaan koneksi internet bot Discord. Perintah force_quit digunakan untuk mereset paksa bot Discord sebagai pembaruan untuk pengembang.

2) *Interface* Perintah *Play*

Gambar 12.Interface *Play*

Tampilan *interface* perintah *play* menampilkan nama pengguna, judul, situs lagu, informasi rekomendasi musik, tombol multimedia dan tombol rekomendasi musik. Untuk menggunakan fitur ini, pengguna diharuskan masuk kedalam saluran suara terlebih dahulu. Kemudia, *bot Discord* akan mengikuti pengguna yang mengirimkan perintah serta memulai pemutaran musik. Dalam prosesnya, ketika pengguna memasukkan judul musik, maka judul musik tersebut akan diproses oleh bot sehingga menghasilkan rekomendasi musik yang sesuai.

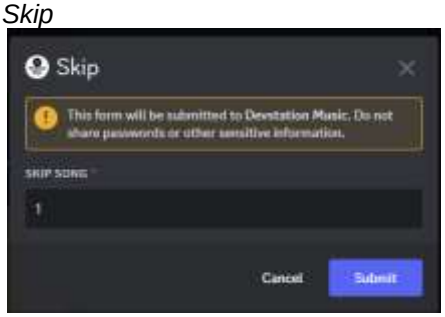
3) *Interface* Tombol Multimedia



Gambar 13. Tombol Multimedia

Tampilan *interface* tombol multimedia, dengan fitur urut tombol dari kiri adalah *resume* atau *pause*, *repeat* musik, menampilkan antrian musik, melewati musik, menghapus musik, stop pemutaran musik, dan tombol donasi untuk pengembang atau penyedia *bot Discord*.

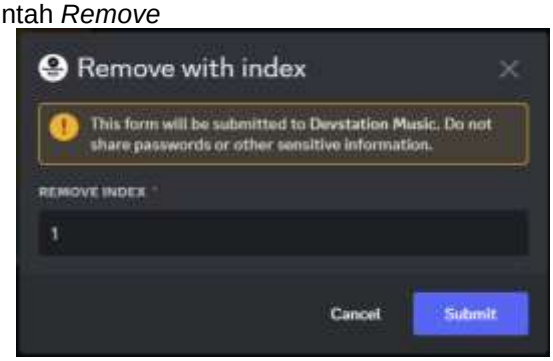
4) *Interface* Perintah *Skip*



Gambar 14. *Interface Skip*

Pengguna dapat melewati musik sesuai urutannya. Urutan bisa dilihat terlebih dahulu dengan menggunakan tombol antrean musik. Lalu melakukan konfirmasi dengan menekan tombol Submit.

5) *Interface* Perintah *Remove*

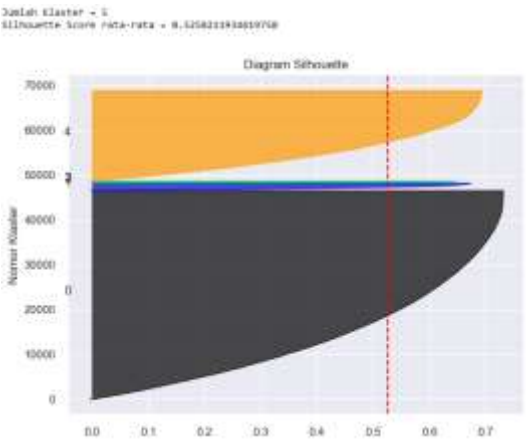


Gambar 15. *Interface Remove*

Interface diatas adalah *modal* dari perintah *remove*. Urutan bisa dilihat terlebih dahulu dengan menggunakan tombol antrean musik. Pengguna dapat melakukan penghapusan lagu sesuai dengan urutan yang ada dalam antrian *bot Discord*.

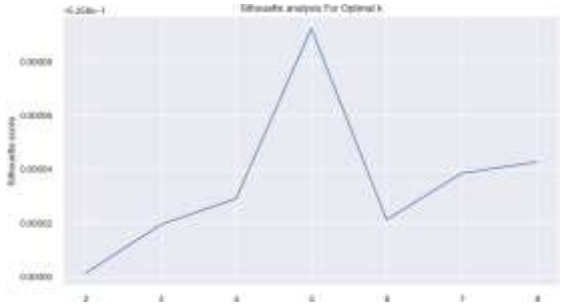
4.2 Pembahasan

Dengan menggunakan dataset yang tersedia pada situs *kaggle*, dengan jumlah sebanyak 133.638 musik yang diklasterisasi menggunakan metode *K-Means*. Didapatkan hasil *silhouette scores* rata-rata yaitu 0.525 dan nilai K optimal adalah 5.



Gambar 16. Rata-rata *Silhouette Scores*

Pencarian nilai K optimal dilakukan dengan menggunakan metode *Silhouette Analys* dengan mencari nilai atau angka tertinggi dalam grafik.



Gambar 17. Grafik *Silhouette Analysis*

4.3 Pengujian Sistem

Pengujian *Blackbox Testing* berfokus pada pengujian fungsional dalam sistem yang dikembangkan. Hasil pengujian disajikan pada Tabel 1.

Tabel 1. Aktivitas Pengujian *Blackbox* pada *Bot Discord*

Aktivitas Pengujian	Keterangan	Hasil Uji
Perintah *Play*	Memutar musik dari URL atau kata kunci pencarian yang diberikan pengguna	Sukses
Pause	Menghentikan musik secara sementara yang sedang diputar oleh pengguna	Sukses
Resume	Melanjutkan pemutaran musik yang sedang dihentikan secara sementara oleh pengguna	Sukses
Tombol *Repeat*	Melakukan aktivasi perulangan untuk lagu yang sedang diputar dan menonaktifkan perulangan	Sukses
Tombol *Queue*	Menampilkan daftar lagu atau antrean lagu yang akan diputar serta lagu yang sedang diputar	Sukses
Tombol *Skip*	Melewati musik yang sedang diputar	Sukses
Tombol *Remove*	Menghapus musik yang ada pada antrean	Sukses
Tombol *Stop*	Menghentikan pemutaran musik bot Discord dan keluar dari *voice channel*	Sukses
Tombol Rekomendasi Musik	Menambahkan musik kedalam antrean bot Discord	Sukses

Hasil dari uji fungsionalitas menggunakan metode pengujian *blackbox* pada Tabel 1 menunjukkan bahwa keseluruhan fitur dalam *bot Discord* dapat berjalan dengan baik sesuai apa yang diharapkan penulis. Pengujian dilakukan menyerupai keadaan yang sesungguhnya. Dengan demikian, *bot Discord* yang telah dikembangkan dapat diimplementasikan kedalam platform *Discord*.

5. Simpulan

Hasil Pengujian fungsional menunjukkan bahwa fitur-fitur pada *bot Discord* telah berjalan dengan baik. Dengan dikembangkannya sistem *bot* pemutaran musik ini, pengguna Discord dapat mengefisiensikan waktu mereka ketika sedang bermain permainan video sambil mendengarkan musik. Serta dengan fitur tombol multimedia dan rekomendasi musik akan memudahkan para pengguna Discord dalam mengendalikan pemutaran musik.

Daftar Referensi

[1] C.D. Andita, D. Desyandri, “Pengaruh Penggunaan Musik Terhadap Konsentrasi Belajar Anak Sekolah Dasar”, *EDUKATIF Jurnal Ilmu Pendidikan*, Vol. 1, No. 3, pp. 205-209, 2019. 10.31004/edukatif.v1i3.50

[2] H. Rasminto, “8 Cara Mengatasi Komputer Lemot atau Lambat, Mudah dan Cepat”, Universitas Stekom, 31 Maret 2022, [Online]. Tersedia: https://komputerisasi-

akuntansi-d3.stekom.ac.id/informasi/baca/8-Cara-Mengatasi-Komputer-Lemot-atau-Lambat-Mudah-dan-Cepat/f111f376352dc911db43324cb1010603a820b93c [Diakses: 20 November 2023].

[3] R. Aditya, E.J. Dase, N. Lukman, T.B. Lulu, R.B. Rohimah, D.I. Suryani, Mudmainah V., Vica Dian A. R., "Analisis Pemanfaatan Aplikasi Discord Dalam Pembelajaran Daring Di Era Pandemi Covid-19", *Prosiding Seminar Nasional Pendidikan FKIP,* Vol. 3, No. 1, pp. 55-59, 2020.

[4] A. Verma, S. Tyagi, & G.A. Mathur, "A Comprehensive Review on Bot - Discord Bot." *International Journal of Scientific Research in Computer Science, Engineering and Information Technology*, Vol. 7, No. 2, pp.532-536, Apr 2021. 10.32628/cseit2172100.

[5] R.K. Dinata, S. Safwandi, N. Hasdyna, & N. Azizah, "Analisis K-Means Clustering pada Data Sepeda Motor", Vol. 5, No. 1, pp.10-17, April 2020. 10.19184/isj.v5i1.17071.

[6] I. Witten, E. Frank, *Data Mining, Practical Machine Learning, Tools and Techniques, 2nd Edition.* San Fransisco: Morgan Kaufmann Publisher, 2005.

[7] S.A. Fitriani, "Analisis Klaster Atribut Musik pada Global Top 50 dengan Data Spotify dengan Menggunakan Algoritma K-Means", Researchgate, Jan 2023. [Online]. https://www.researchgate.net/publication/348335831 [Diakses: 31 Juli 2023]

[8] Z.R. Karyono, Y.T. Mursityo, & H.M. Az-Zahra, "Analisis Perbandingan Pengalaman Pengguna Pada Aplikasi Music Streaming Menggunakan Metode UX Curve (Studi Pada Spotify dan JOOX)", *J-PTIIK*, Vol. 3, No. 7, pp. 6422–6429, Agu 2019.

[9] B.A. Firdaus, D.E. Ratnawati, & B.T. Hanggara, "Klusterisasi Popularitas Artist pada Playlist Today's Top Hits Menggunakan Metode K-Means dengan Integrasi Spotify Web API dan Teknologi Amazon SageMaker", *J-PTIIK*, Vol. 5, No. 1, pp.373-380, Januari 2021.

[10] U.L. Musyarofah, S.N. Alima, & D.S.Y. Kartika, "Klasifikasi Top 50 Spotify Tahun 2010-2019 Menggunakan Metode K-Means Clustering", *Prosiding Seminar Nasional Teknologi Dan Sistem Informasi*,Vol. 2, No. 1, pp. 215-220, Sept 2022. 10.33005/sitasi.v2i1.300

[11] R. Septiansyah, S.R. Akbar, & R. Maulana, "Perancangan Bot Pada Discord Untuk Pembacaan Sensor Di Raspberry Pi Dengan Sistem Learning Yang Dinamis", *J-PTIIK*, vol 2, No. 10, pp. 4202-4212, Feb 2018.

[12] M.I. Firmansyah, R.S. Rohman, E. Marsusanti, "Penerapan Algoritma Klastering K-Means Untuk Fitur Atribut Pada Layanan Streaming Musik Spotify", *Indonesian Journal Computer Science*, Vol. 2, No. 2, pp.675-680, Sept 2023.

[13] A.R. Zaidah, C.I. Septiarani, N.M. Sholikhatun, A. Yusuf, & N. Wahyudi, "Komparasi Algoritma K-Means, K-Medoid, Agglomeartive Clustering Terhadap Genre Spotify", *Jurnal Ilmiah Ilmu Komputer,* Vol. 7, No. 1, pp. 49-54, Apr 2021. 10.35329/jiik.v7i1.186

[14] H. Pangestu, H. Alianto, & S.F. Wijaya, "Hasil Rancang Bangun Sistem ERP dengan SDLC Model Waterfall: Studi Kasus Sistem Inventori PT Pan Brothers", ComTech, Vol. 3, No. 2, pp. 1036, Des 2012. 10.21512/comtech.v3i2.2360

[15] M. H. Prayitno, "Analisa Kebutuhan Sistem Informasi Dengan Menggunakan Analisis Value Change Dan Critical Success Factor Pada PT. LHE", *BINA INSANI ICT JOURNAL*, Vol. 3, No.1, pp. 269 – 278, Juni 2016.