Jurnal Teknologi Sistem Informasi dan Aplikasi **ISSN: 2654-3788**
Penerbit: Fakultas Ilmu Komputer Universitas Pamulang **e-ISSN: 2654-4229**
Vol. 9, No. 1, January 2026 (13-22) **DOI: https://doi.org/10.32493/jtsi.v9i1.57819**

Penerapan Arsitektur Kappa dengan Kafka dan Spark untuk Pemrosesan Data Hipertensi di Media Sosial X

Legawan Perkasa[1*], Tikaridha Hardiani[2]

[1]Teknologi Informasi, Universitas 'Aisyiyah Yogyakarta, Jl. Siliwangi (Ring Road Barat) No. 63 Nogotirto, Gamping, Sleman, Yogyakarta, Indonesia, 55292
e-mail: [1]2211501046@student.unisayogya.ac.id, [2]Tikaridha@unisayogya.ac.id
*Corresponding author

Submitted Date: January 2nd, 2026 Reviewed Date: January 12nd, 2026
Revised Date: January 17th, 2026 Accepted Date: January 24th, 2026

Abstract

Hypertension is one of the major public health problems with a continuously increasing prevalence and is widely discussed on social media platform X. The dynamic and continuously flowing nature of social media data requires a Big Data-based processing approach capable of operating in real-time and in a scalable manner. This study aims to implement a streaming-based Big Data architecture (Kappa Architecture) using Apache Kafka and Apache Spark to process and analyze conversations about hypertension on the social media platform X in real-time. The proposed system integrates the X API as the data source, Apache Kafka as the immutable event log and streaming backbone, Apache Spark Structured Streaming as the real-time data processing engine, and MongoDB as the serving layer. The research methodology includes a literature review, system design, streaming-based data collection, real-time text cleaning and feature extraction, and performance evaluation using throughput, latency, and success rate parameters. A total of 10,000 tweets were collected over a two-month period and processed through a unified streaming pipeline. The implementation results show that the system successfully established a consistent end-to-end processing workflow, enabling real-time data ingestion and processing without separating batch and speed layers. The system achieved an average throughput of 19.23 tweets per second, a latency of approximately 520 seconds, and a success rate of 100%. This study concludes that the Kappa Architecture is effective, stable, and scalable for real-time processing and analysis of social media data in monitoring public health issues such as hypertension.

Keywords: Kappa Architecture; Apache Kafka; Apache Spark; Hipertensi; X

Abstrak

Hipertensi merupakan salah satu masalah kesehatan masyarakat dengan prevalensi yang terus meningkat dan banyak diperbincangkan di media sosial X. Karakteristik data media sosial yang bersifat dinamis dan mengalir membutuhkan pendekatan pemrosesan berbasis *Big* Data yang mampu bekerja secara *real-time* dan skalabel. Penelitian ini bertujuan menerapkan arsitektur *Big* Data berbasis *streaming* (*Kappa Architecture*) menggunakan Apache Kafka dan Apache Spark untuk mengolah dan menganalisis data percakapan tentang hipertensi di media sosial X secara *real-time*. Sistem yang diusulkan mengintegrasikan API X sebagai sumber data, Apache Kafka sebagai *immutable event log* dan tulang punggung *streaming*, Apache Spark *Structured Streaming* sebagai pemroses data *real-time*, serta MongoDB sebagai *serving layer*. Metode penelitian meliputi studi literatur, perancangan sistem, pengumpulan data berbasis *streaming*, pembersihan dan ekstraksi fitur teks secara *real-time*, serta evaluasi kinerja menggunakan parameter *throughput*, *latency*, dan *success rate*. Sebanyak 10.000 *tweet* dikumpulkan selama dua bulan dan diproses melalui satu jalur *streaming* terpadu. Hasil penerapan menunjukkan bahwa sistem berhasil membangun alur pemrosesan *end-to-end* yang konsisten, memungkinkan *ingest* dan pemrosesan data secara *real-time* tanpa pemisahan *batch* dan *speed layer*, dengan *throughput* rata-rata 19,23 *tweet* per detik, *latency*

This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International (CC BY-NC 4.0) License
Copyright © 2026 Legawan Perkasa, Tikaridha Hardiani

sekitar 520 detik, serta *success rate* sebesar 100%. Penelitian ini menyimpulkan bahwa Arsitektur Kappa efektif, stabil, dan skalabel untuk pemrosesan serta analisis data media sosial secara *real-time* dalam pemantauan isu kesehatan seperti hipertensi.

Kata Kunci: Arsitektur Kappa; Apache Kafka; Apache Spark; Hipertensi; X

1. Pendahuluan

Hipertensi menjadi salah satu masalah kesehatan utama yang berkontribusi besar terhadap meningkatnya angka kematian di seluruh dunia. Menurut laporan WHO pada tahun 2023, sebanyak 1,28 miliar penduduk berusia 30–79 tahun tercatat mengalami kondisi ini (Mita Permatasari, 2025).

Hipertensi adalah salah satu tipe penyakit tidak menular (PTM) yang angka kejadiannya terus meningkat di seluruh dunia, termasuk di Indonesia. Penyakit ini sering disebut sebagai *silent killer* karena jarang menunjukkan gejala jelas hingga mengarah pada komplikasi berbahaya yang dapat mengancam nyawa. Menurut *World Health Organization* (WHO), angka prevalensi hipertensi secara global diperkirakan sebesar 33,1%, sedangkan di kawasan Asia Tenggara mencapai 32,4% pada populasi dewasa usia 30–79 tahun. Di Indonesia sendiri, berdasarkan data tahun 2018, prevalensi hipertensi tercatat sebesar 34,1% pada penduduk berusia 18 tahun ke atas. (Sabrina et al., 2026).

Media sosial merupakan *platform online* yang memungkinkan penggunanya berinteraksi melalui pertukaran informasi, pendapat, dan berbagai minat. Media sosial mengacu pada serangkaian aplikasi berbasis internet yang dibangun berdasarkan ideologi dan teknologi Web 2.0 dan memungkinkan terciptanya pertukaran konten buatan pengguna. Media sosial tidak hanya digunakan sebagai alat untuk berkomunikasi dan berinteraksi, tetapi juga sebagai alat untuk mengekspresikan diri (*self-expression*) dan membangun citra diri (Tri Buana, 2022). Twitter alias X adalah aplikasi media sosial yang memungkinkan pengguna melakukan interaksi melalui pesan yang disebut *tweet*. Warganet memiliki kemampuan untuk saling berbagi informasi dan pendapat tentang berbagai topik. Aplikasi X memungkinkan penyebaran informasi secara cepat dan luas, sehingga dapat menjadi media edukasi yang efektif untuk meningkatkan pengetahuan dan pemahaman masyarakat mengenai pentingnya isu hipertensi (Nursinggah et al., 2024).

Arsitektur Kappa, yang dibuat oleh Jay Kreps, menekankan model pemrosesan aliran terpadu untuk pemrosesan *batch* dan *real-time*. Arsitektur ini memperlakukan setiap data sebagai aliran, yang menyederhanakan alur pemrosesan data dan memungkinkan pendekatan yang lancar dan skalabel untuk menangani jumlah besar data waktu nyata (*real-time*) (Abirami T and Dr. Chandrasekar B S, 2024). Apache Kafka merupakan platform yang широко digunakan untuk pengelolaan aliran data *real-time* dalam lingkungan terdistribusi. Kafka dirancang untuk menangani volume data yang sangat besar dengan tingkat latensi yang rendah, sehingga mampu mendukung pengiriman pesan secara cepat dan efisien antar sistem yang saling terhubung. Teknologi ini banyak dimanfaatkan dalam berbagai kebutuhan *streaming data*, termasuk analisis sentimen secara *real-time*. Selain itu, Kafka menyediakan sistem yang bersifat *fault-tolerant* dan skalabel, sehingga memungkinkan pemrosesan data dalam skala besar dengan performa tinggi dan stabil (Zulkifli, 2025). Apache Spark merupakan salah satu framework komputasi *Big* Data yang dikembangkan untuk menangani pemrosesan data dalam skala besar dengan performa tinggi melalui teknologi *in-memory processing* (Maulana et al., 2026).

Berbagai penelitian telah menunjukkan bahwa Lambda *Architecture* unggul dalam ketahanan karena menggabungkan pemrosesan *batch* dan *streaming*, sementara Kappa *Architecture* lebih sederhana karena berfokus sepenuhnya pada pemrosesan real-time (Studies & Guntupalli, 2023). Hal ini diperkuat oleh penelitian dalam konteks *Big* Data telekomunikasi yang menyimpulkan bahwa Kappa lebih efisien untuk pemantauan jaringan dan pemrosesan data 5G secara real-time, sedangkan Lambda lebih sesuai untuk analisis historis yang membutuhkan tingkat akurasi tinggi (Abirami T and Dr. Chandrasekar B S, 2024). Penelitian berikutnya juga mengusulkan versi optimal arsitektur Kappa dengan memanfaatkan kombinasi Apache Samza dan Apache Druid, yang terbukti meningkatkan kecepatan pemrosesan, efisiensi memori, serta menurunkan penggunaan CPU hingga 20% (Bertin et al., 2022).

Selain itu, hasil penelitian ini menegaskan bahwa Apache Kafka memegang peranan yang signifikan dalam mendukung proses pemrosesan data secara *real-time*. Salah satu studi membuktikan bahwa Kafka mampu meningkatkan *throughput* hingga 213% serta menurunkan latensi sebesar 56% pada pemrosesan sentimen media sosial (Zulkifli, 2025). Penelitian lain menunjukkan bahwa Kafka dapat mengirimkan data

This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International (CC BY-NC 4.0) License
Copyright © 2026 Legawan Perkasa, Tikaridha Hardiani

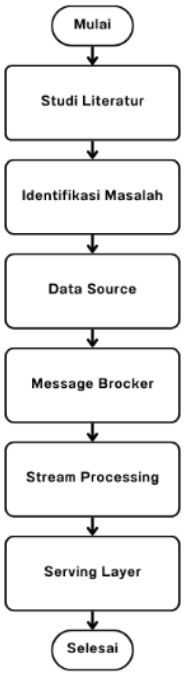
ISSN: 2654-3788
e-ISSN: 2654-4229
DOI: https://doi.org/10.32493/jtsi.v9i1.57819

antar aplikasi secara cepat dan stabil dalam lingkungan perusahaan (Nabawi, 2022). Selain itu, sistem *e-ticketing* berbasis pendekatan *event-driven* menggunakan Kafka terbukti lebih responsif dan mudah diskalakan (Pradinata et al., 2023). Penelitian lainnya menunjukkan bahwa Kafka mampu menangani data besar hingga 200 GB dengan penggunaan CPU tetap rendah melalui replikasi dan enkripsi AES (Park & Huh, 2023). Penelitian lain juga memperlihatkan bahwa kombinasi Kafka dengan Flink dan Hadoop melalui mekanisme *hot path* dan *cold path* mampu meningkatkan efisiensi serta skalabilitas pemrosesan data (Parmar, 2025). Di samping Kafka, teknologi pendukung lainnya seperti Apache Spark mengolah data curah hujan periode 2005–2024 secara terdistribusi dan menerapkan beberapa algoritma *machine learning*, yaitu *Linear Regression, Decision Tree,* dan *Random Forest* (Maulana et al., 2026). Selain itu, penelitian terkait pemrosesan hybrid menunjukkan bahwa penggabungan batch processing dengan real-time processing menggunakan Spark dapat memberikan ketelitian sekaligus kecepatan analisis (Vaghani Divyeshkumar, 2024).

Tujuan penelitian ini adalah untuk menerapkan Arsitektur Kappa sebagai pendekatan pemrosesan data real-time dalam pengolahan konten terkait isu hipertensi di media sosial X. Penerapan ini mencakup perancangan alur *streaming*, integrasi komponen *ingest–process–store*, serta optimalisasi proses pemrosesan data agar mampu menangani data yang terus mengalir. Peningkatan kecepatan dan efisiensi dievaluasi melalui pengukuran *throughput* (jumlah data yang dapat diproses setiap detik), *latency* (waktu dari data diterima hingga selesai diproses), serta penggunaan sumber daya dan stabilitas aliran data. Kemampuan pemantauan dinilai dari konsistensi sistem dalam menangkap setiap *tweet* baru tanpa jeda berarti. Selain itu, skalabilitas diamati dari kemampuan sistem untuk tetap stabil ketika volume data meningkat. Melalui penerapan Arsitektur Kappa, penelitian ini bertujuan menunjukkan bagaimana pemrosesan data dengan satu jalur kerja dapat meningkatkan kecepatan pengolahan, efisiensi proses, dan kemampuan sistem dalam memantau percakapan publik di media sosial.

2. Metode

Berdasarkan Gambar 1, alur penelitian arsitektur kappa merupakan pendekatan yang lebih tepat untuk menangani data berskala besar yang memerlukan pemrosesan cepat dan *real-time*, seperti data yang dihasilkan oleh sensor, robot, maupun jaringan. Arsitektur ini menekankan penggunaan satu jalur pemrosesan terpadu untuk menangani aliran data secara *real-time* maupun melalui mekanisme *micro-batching* (Gede et al., 2024).



Gambar 1. Alur Penelitian Penerapan Arsitektur Kappa.

This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International (CC BY-NC 4.0) License
Copyright © 2026 Legawan Perkasa, Tikaridha Hardiani

a. Studi Literatur

Tahap pertama penelitian ini adalah melakukan studi literatur untuk memperoleh landasan teoretis yang kuat terkait hipertensi, perilaku masyarakat di media sosial, teknik analisis teks, serta teknologi pemrosesan data *real-time*. Literatur yang dikaji meliputi penelitian mengenai Apache Kafka, Apache Spark, Arsitektur Kappa, *Big* Data, serta beberapa studi terdahulu yang memanfaatkan media sosial sebagai sumber data kesehatan. Studi literatur ini bertujuan memperkuat dasar teori, mengidentifikasi celah penelitian, serta memastikan bahwa metode yang digunakan relevan dan sesuai dengan kebutuhan penelitian dalam konteks arsitektur yang ditunjukkan pada Gambar 1.

b. Identifikasi Masalah

Identifikasi masalah dilakukan untuk memahami urgensi penelitian, yaitu meningkatnya prevalensi hipertensi dan aktifnya pembahasan kesehatan di media sosial X, namun pemantauan percakapan publik masih terbatas. Sebagaimana terlihat pada alur Gambar 1, data media sosial bersifat mengalir dan dinamis sehingga membutuhkan pendekatan pemrosesan berbasis streaming. Penerapan arsitektur Kappa dengan Kafka dan Spark dalam konteks kesehatan masih jarang dilakukan. Data dari API X *Free Tier* yang hanya menyediakan 500 postingan per bulan digunakan sebagai data sampel untuk menerapkan alur kerja Arsitektur Kappa. Untuk meningkatkan jumlah data yang diperoleh, pengambilan data dilakukan menggunakan 10 akun secara bertahap selama periode dua bulan, sehingga total data yang berhasil dikumpulkan mencapai 10.000 postingan. Meskipun terdapat keterbatasan pada masing-masing akun, pendekatan ini tetap relevan karena arsitektur Kappa menekankan pemrosesan data secara *real-time* dalam satu jalur kerja yang sederhana, efisien, dan siap menangani skala data yang lebih besar.r.

c. Data Source

Data source adalah sumber data yang berperan penting dalam integrasi informasi pada era *Big* data (Zhou & Zhou, 2024). Mengacu komponen data *Source* menggambarkan proses pengumpulan data yang dilakukan dari media sosial X melalui API dengan skema *Free Tier* yang menyediakan 500 postingan per bulan di tingkat aplikasi serta batas tambahan di tingkat pengguna. Untuk meningkatkan jumlah data, proses pengambilan dilakukan menggunakan 10 akun secara bertahap selama dua bulan. Setiap akun memanfaatkan kuota 500 postingan per bulan sehingga total data yang diperoleh mencapai 10.000 postingan. Data dikumpulkan dalam format JSON yang berisi teks *tweet*, waktu pembuatan, serta metadata interaksi. Dataset tersebut menjadi input utama dalam *pipeline streaming* untuk proses analisis selanjutnya.

d. Message Brocker

Message broker adalah komponen yang memisahkan (*decouple*) produsen data dan konsumen data serta menyediakan penyimpanan dan mekanisme pengiriman data yang andal (Puthenpariyarath, 2025). *Message Broker* berfungsi sebagai perantara aliran data *real-time* yang dikirim dalam format JSON ke Apache Kafka melalui *topic hipertensi_x_stream* dengan konfigurasi tiga *partition* dan *replication factor* satu. Kafka bertindak sebagai *immutable event log* yang menyimpan data secara berurutan sehingga dapat diproses ulang jika diperlukan. Konfigurasi producer menggunakan *acks=all* untuk memastikan pesan diterima seluruh replica sebelum dinyatakan berhasil, *compression.type=gzip* untuk mengurangi ukuran data saat transmisi, serta *batch.size=16384* untuk optimalisasi pengiriman berbasis *batch*. *Consumer* dikonfigurasi dengan *auto.offset.reset=earliest* agar pembacaan dimulai dari data paling awal ketika *offset* belum tersedia serta *enable.auto.commit=false* guna menjaga konsistensi pengolahan data melalui mekanisme *commit* manual. Volume data yang digunakan tetap representatif dalam memvalidasi *pipeline streaming* sesuai arsitektur sistem yang diterapkan.

e. Stream Processing

Stream processing adalah hal yang semakin penting untuk menganalisis serta memperoleh wawasan secara real-time dari data yang masuk, baik yang berasal dari eksperimen, simulasi (Musababa et al., 2025). Tahap *Stream Processing* memproses data yang tersimpan dalam *log* Kafka diproses menggunakan Apache Spark *Streaming* dengan mode *Structured Streaming*. Pemrosesan dilakukan menggunakan dua *core*, RAM

This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International (CC BY-NC 4.0) License
Copyright © 2026 Legawan Perkasa, Tikaridha Hardiani

4 GB, interval *batch* lima detik, serta mekanisme *checkpointing* pada direktori /*checkpoint/kappa_hipertensi* untuk menjaga keberlanjutan proses.

Tahapan pemrosesan meliputi pembersihan teks, penghapusan duplikasi, penyaringan konten tidak relevan, tokenisasi, normalisasi, serta identifikasi kata kunci terkait hipertensi secara *real-time*. Proses ini mengikuti prinsip Arsitektur Kappa yang memproses data dalam satu jalur *streaming* tanpa memisahkan *batch layer* dan *speed layer*.

f. Serving Layer

Tahap terakhir dalam arsitektur sistem adalah *Serving Layer*, *Serving layer* berfungsi untuk memfasilitasi proses pengambilan data yang telah diproses agar dapat digunakan dalam analisis dan pelaporan (Abirami T and Dr. Chandrasekar B S, 2024). Penyimpanan dilakukan menggunakan MongoDB pada *database hipertensi_db* dan *collection tweet_stream_clean* dengan *write concern w:1* dalam format dokumen JSON.

MongoDB digunakan karena mendukung penyimpanan data semi-terstruktur secara fleksibel. Data akhir kemudian diekspor ke dalam format CSV agar lebih mudah dikelola dan digunakan untuk analisis lanjutan.

Untuk mengukur efektivitas *Serving Layer* dalam menangani aliran data dari Apache Spark Structured Streaming ke MongoDB, dilakukan evaluasi berdasarkan tiga indikator utama, yaitu *Throughput*, *Latency*, dan *Success Rate*.

- *Throughput*

  Menurut Anom *et al*., (2024) *Throughput* didefinisikan sebagai ukuran laju efektif dalam proses pengiriman atau pemrosesan data (Anom et al., 2024). *Throughput* dapat dihitung dengan membandingkan jumlah total dokumen yang berhasil dikirim dengan lamanya waktu pengiriman, sebagaimana ditunjukkan pada Persamaan (1).

  $$T = \frac{\sum N_{berhasil}}{t} \quad (1)$$

  - T= Throughput (dokumen/detik)
  - $N_{berhasil}$= Jumlah dokumen yang berhasil disimpan
  - t= Total waktu pemrosesan (detik)

  Semakin besar nilai T, semakin tinggi kapasitas sistem dalam menangani aliran data.

- *Latency*

  Menurut Mikola *et al*. (2022), *latency* merupakan waktu keterlambatan yang terjadi selama proses pengiriman data dari satu titik sumber menuju titik tujuan. Nilai latency menunjukkan selisih waktu antara saat proses dimulai dan saat proses tersebut selesai. (Mikola et al., 2022). Secara matematis, *latency* dapat dihitung menggunakan rumus sebagaimana ditunjukkan pada Persamaan (2).

  $$L = t_{tersimpan} - t_{kejadian} \quad (2)$$

  - L= Latency (detik)
  - $t_{kejadian}$= Waktu data diterima dari API
  - $t_{tersimpan}$= Waktu data berhasil disimpan di MongoDB

  Semakin kecil nilai L, semakin cepat sistem merespons aliran data.

- *Succes Rate*

  Menurut Nielsen dan Budiu (2021), *success rate* merupakan salah satu metode paling sederhana untuk mengukur tingkat kegunaan (*usability*) suatu sistem. Metrik ini menunjukkan persentase tugas yang berhasil diselesaikan pengguna dengan benar dibandingkan dengan total tugas yang diberikan (Farid et al., 2022). Adapun perhitungan *success rate* dapat dinyatakan menggunakan rumus sebagaimana ditunjukkan pada Persamaan (3).

This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International (CC BY-NC 4.0) License
Copyright © 2026 Legawan Perkasa, Tikaridha Hardiani

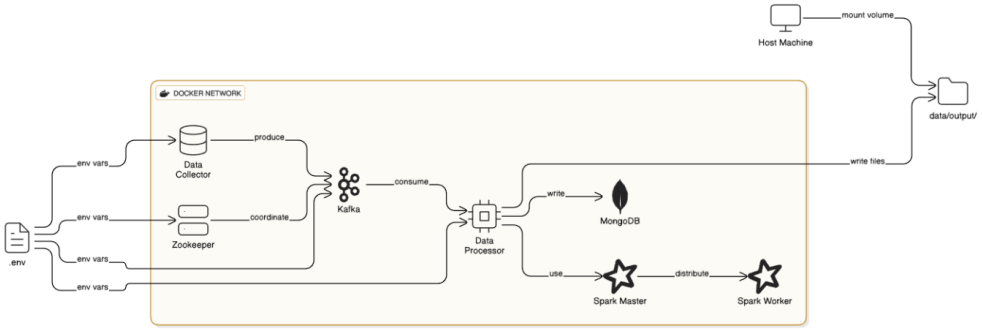
$$SR = \frac{N_{db}}{N_{spark}} x\ 100\% \quad (3)$$

- SR= Success Rate (%)
- N_{db}= Jumlah dokumen yang tersimpan di MongoDB
- N_{spark}= Jumlah dokumen yang dikirim oleh Spark

Hasil pengujian, sistem menunjukkan nilai *Success Rate* sebesar 100%, yang berarti seluruh data yang dikirim berhasil tersimpan tanpa kehilangan maupun redundansi.

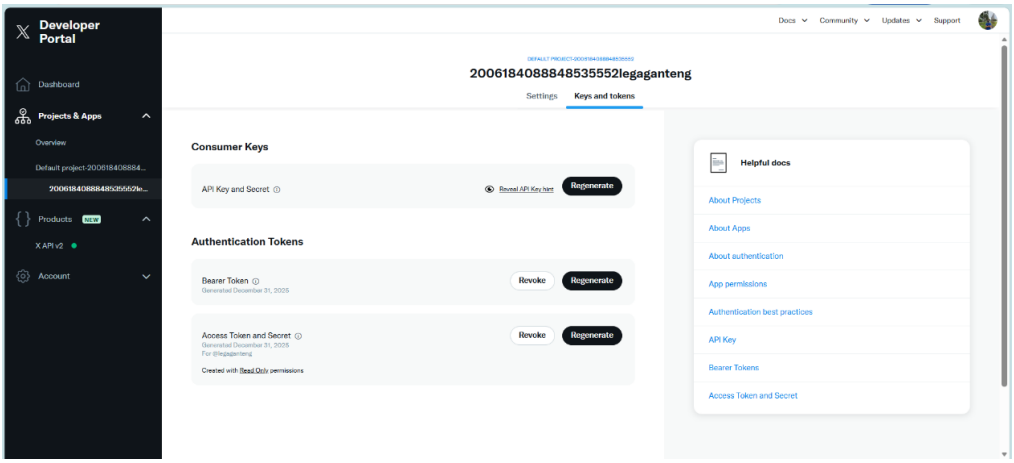
3. Hasil dan Pembahasan

Penelitian menunjukkan bahwa penerapan Arsitektur Kappa berhasil mewujudkan konsep *immutable event log* dengan memanfaatkan *topic* Apache Kafka sebagai *single source of truth* dalam pemrosesan data media sosial X. Seluruh data *tweet* yang diperoleh dari API X terlebih dahulu dicatat ke dalam *event log* Kafka, sehingga membentuk jejak audit (*audit trail*) yang lengkap dan konsisten. Pendekatan ini memungkinkan sistem untuk mempertahankan integritas data sekaligus menyediakan fleksibilitas dalam melakukan *reprocessing* apabila terjadi perubahan logika pemrosesan. Dengan menggunakan satu *processing pipeline* terpadu, seluruh kebutuhan pemrosesan data mulai dari pembersihan hingga ekstraksi informasi terkait hipertensi dilakukan dalam satu alur kerja yang berkelanjutan.



Gambar 2. Alur Sistem Pemrosesan Data

Gambar 2 menjelaskan alur sistem pemrosesan data dimulai dari Data *Collector* yang mengambil data dari sumber eksternal dengan konfigurasi yang dikelola melalui file .env, kemudian mempublikasikan data tersebut ke Kafka yang dikoordinasikan oleh *Zookeeper*. Data dalam Kafka selanjutnya dikonsumsi oleh Data *Processor* untuk dilakukan pemrosesan seperti pembersihan dan transformasi, lalu hasilnya disimpan ke MongoDB sebagai basis data dokumen. Untuk pemrosesan skala besar, Data *Processor* memanfaatkan Apache Spark, di mana *Spark Master* mendistribusikan tugas ke *Spark Worker* secara paralel, dan hasil akhir dapat ditulis ke file pada volume yang terhubung dengan *Host Machine*.

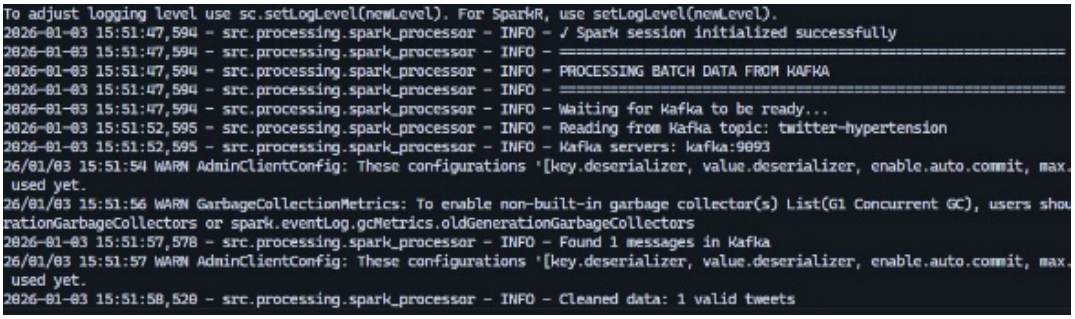


Gambar 3. API X

This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International (CC BY-NC 4.0) License
Copyright © 2026 Legawan Perkasa, Tikaridha Hardiani

Pada tahap *autentikasi* di Gambar 3, producer terlebih dahulu melakukan koneksi ke API X dengan menggunakan token atau kredensial autentikasi yang valid. Proses ini bertujuan untuk memastikan bahwa hanya aplikasi yang memiliki izin resmi yang dapat mengakses data *tweet*. Setelah autentikasi berhasil, tahap *data fetching* dilakukan dengan mengambil data *tweet* berdasarkan kata kunci tertentu, "hipertensi". Data yang diperoleh pada tahap ini masih berupa data mentah (*raw data*) yang mencakup teks *tweet*, metadata pengguna, waktu unggah, serta informasi pendukung lainnya.

Data mentah tersebut melalui tahap *serialization*, yaitu proses konversi data ke dalam format JSON agar memiliki struktur yang konsisten dan mudah diproses oleh sistem *downstream*. Format JSON dipilih karena bersifat ringan, fleksibel, dan kompatibel dengan berbagai platform pemrosesan data. Setelah proses serialisasi selesai, tahap *publishing* dilakukan dengan mengirimkan data JSON tersebut ke *Kafka topic* menggunakan *KafkaProducer*. Dengan mekanisme ini, data *tweet* dapat diproses secara *real-time* oleh konsumer lain dalam arsitektur sistem, seperti untuk analisis sentimen atau pemantauan isu hipertensi secara berkelanjutan.



Gambar 4. Tampilan *Log* (*Console Output*)

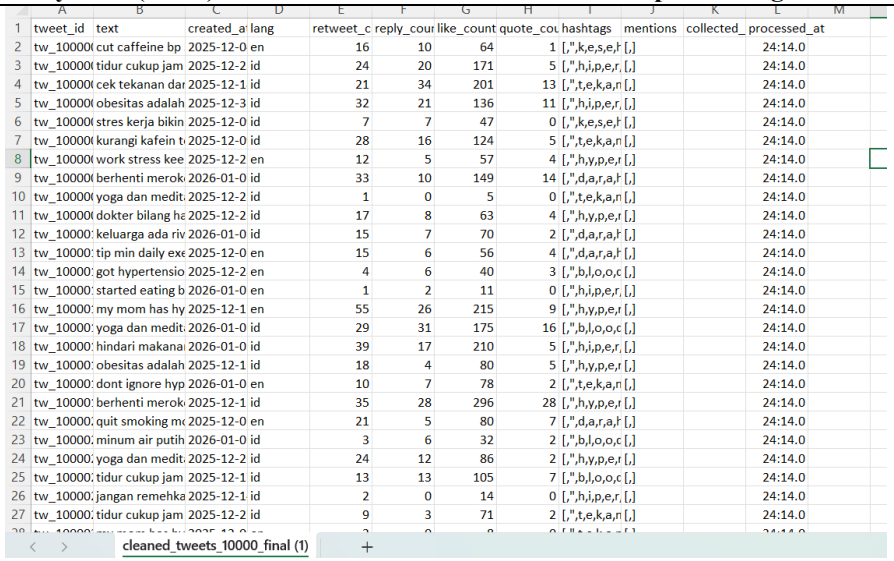
Berdasarkan *log* pada Gambar 4. proses dimulai dari tahap *connection*, di mana Apache Spark berhasil melakukan koneksi ke *Kafka broker* yang ditandai dengan inisialisasi *Spark session* dan kesiapan Kafka untuk mengirimkan data. Setelah koneksi terbentuk, Spark melakukan *subscription* ke topik tertentu, misalnya *twitter-topic*, sehingga Spark dapat menerima aliran data *tweet* secara berkelanjutan dari Kafka. Data yang diterima dari Kafka masih dalam bentuk *bytes*, sehingga dilakukan tahap *deserialization* untuk mengonversi data tersebut ke dalam struktur *DataFrame Spark* agar dapat diproses lebih lanjut secara terstruktur.

DataFrame tersebut memasuki tahap *transformation*, yang meliputi penghapusan data duplikat (*remove duplicates*), penyaringan nilai kosong (*filter empty values*), serta proses *data cleaning* dan validasi untuk memastikan kualitas data. Pada tahap ini juga dilakukan *feature extraction* guna mengekstraksi atribut penting yang relevan dengan kebutuhan analisis. Setelah seluruh proses transformasi selesai, tahap *preparation* dilakukan dengan menyiapkan data akhir agar sesuai dengan skema penyimpanan MongoDB, sehingga data siap ditulis dan digunakan untuk analisis lanjutan atau pemrosesan berikutnya.

Tahap *configuration*, sistem terlebih dahulu menyiapkan MongoDB *connection string* yang berisi informasi penting seperti alamat *host*, *port*, nama *database*, serta kredensial autentikasi. Konfigurasi ini digunakan oleh Apache Spark sebagai dasar untuk membangun koneksi yang aman dan stabil. Setelah konfigurasi selesai, tahap *connection* dilakukan dengan menghubungkan Spark ke MongoDB *cluster* atau *instance* yang dituju. Selanjutnya, pada tahap *specification*, ditentukan secara eksplisit *database* dan *collection* target yang akan digunakan sebagai lokasi penyimpanan data hasil pemrosesan.

Tahap *writing*, data yang telah melalui proses pembersihan dan transformasi akan ditulis ke MongoDB menggunakan mekanisme *insert* atau *upsert*, sehingga data baru dapat ditambahkan atau data lama dapat diperbarui tanpa menimbulkan duplikasi. Setelah proses penulisan berhasil, tahap *storage* memastikan bahwa data telah tersimpan secara permanen dan siap untuk dilakukan *query* atau analisis lanjutan. Selain itu, pada tahap *export*, data yang tersimpan di MongoDB dapat diekspor ke dalam format JSON atau CSV untuk kebutuhan pelaporan, visualisasi, maupun integrasi dengan sistem lain.

This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International (CC BY-NC 4.0) License
Copyright © 2026 Legawan Perkasa, Tikaridha Hardiani



	A	B	C	D	E	F	G	H	I	J	K	L	M
1	tweet_id	text	created_a	lang	retweet_c	reply_cou	like_count	quote_cou	hashtags	mentions	collected_	processed_at	
2	tw_10000(	cut caffeine bp	2025-12-0	en	16	10	64	1	[,",k,e,s,e,h	[,]		24:14.0	
3	tw_10000(	tidur cukup jam	2025-12-2	id	24	20	171	5	[,",h,i,p,e,r,	[,]		24:14.0	
4	tw_10000(	cek tekanan dar	2025-12-1	id	21	34	201	13	[,",t,e,k,a,n	[,]		24:14.0	
5	tw_10000(	obesitas adalah	2025-12-3	id	32	21	136	11	[,",h,i,p,e,r,	[,]		24:14.0	
6	tw_10000(	stres kerja bikin	2025-12-0	id	7	7	47	0	[,",k,e,s,e,h	[,]		24:14.0	
7	tw_10000(	kurangi kafein t	2025-12-0	id	28	16	124	5	[,",t,e,k,a,n	[,]		24:14.0	
8	tw_10000(	work stress kee	2025-12-2	en	12	5	57	4	[,",h,y,p,e,r	[,]		24:14.0	
9	tw_10000(	berhenti merok	2026-01-0	id	33	10	149	14	[,",d,a,r,a,h	[,]		24:14.0	
10	tw_10000(	yoga dan medit	2025-12-2	id	1	0	5	0	[,",t,e,k,a,n	[,]		24:14.0	
11	tw_10000(	dokter bilang ha	2025-12-2	id	17	8	63	4	[,",h,y,p,e,r	[,]		24:14.0	
12	tw_10000:	keluarga ada riv	2026-01-0	id	15	7	70	2	[,",d,a,r,a,h	[,]		24:14.0	
13	tw_10000:	tip min daily exe	2025-12-0	en	15	6	56	4	[,",d,a,r,a,h	[,]		24:14.0	
14	tw_10000:	got hypertensio	2025-12-2	en	4	6	40	3	[,",b,l,o,o,c	[,]		24:14.0	
15	tw_10000:	started eating b	2026-01-0	en	1	2	11	0	[,",h,i,p,e,r,	[,]		24:14.0	
16	tw_10000:	my mom has hy	2025-12-1	en	55	26	215	9	[,",h,y,p,e,r	[,]		24:14.0	
17	tw_10000:	yoga dan medit	2026-01-0	id	29	31	175	16	[,",b,l,o,o,c	[,]		24:14.0	
18	tw_10000:	hindari makana	2026-01-0	id	39	17	210	5	[,",h,i,p,e,r,	[,]		24:14.0	
19	tw_10000:	obesitas adalah	2025-12-1	id	18	4	80	5	[,",h,y,p,e,r	[,]		24:14.0	
20	tw_10000:	dont ignore hyp	2026-01-0	en	10	7	78	2	[,",t,e,k,a,n	[,]		24:14.0	
21	tw_10000:	berhenti merok	2025-12-1	id	35	28	296	28	[,",h,y,p,e,r	[,]		24:14.0	
22	tw_10000:	quit smoking m	2025-12-0	en	21	5	80	7	[,",d,a,r,a,h	[,]		24:14.0	
23	tw_10000:	minum air putih	2026-01-0	id	3	6	32	2	[,",b,l,o,o,c	[,]		24:14.0	
24	tw_10000:	yoga dan medit	2025-12-2	id	24	12	86	2	[,",h,y,p,e,r	[,]		24:14.0	
25	tw_10000:	tidur cukup jam	2025-12-1	id	13	13	105	7	[,",b,l,o,o,c	[,]		24:14.0	
26	tw_10000:	jangan remehka	2025-12-1	id	2	0	14	0	[,",h,i,p,e,r,	[,]		24:14.0	
27	tw_10000:	tidur cukup jam	2025-12-2	id	9	3	71	2	[,",t,e,k,a,n	[,]		24:14.0	

cleaned_tweets_10000_final (1)

Gambar 5. Hasil Data CSV

Gambar 5 menampilkan hasil akhir data *tweet* yang telah diproses dan diekspor ke format CSV, kemudian dibuka menggunakan Microsoft Excel. Setiap baris merepresentasikan satu *tweet* terkait topik hipertensi yang telah melalui tahap pembersihan dan validasi. Dataset memuat kolom utama seperti *tweet_id, created_at, text,* serta metrik interaksi (*reply_count, retweet_count, like_count, quote_count*), dan fitur tambahan seperti *hashtags, mentions, collected_at,* serta *processed_at*, yang menunjukkan bahwa proses *feature extraction* telah berhasil dilakukan. Berdasarkan hasil pengumpulan dan pemrosesan data, sistem berhasil memperoleh dan menyimpan sebanyak 10.000 *tweet*, yang selanjutnya siap digunakan untuk analisis lanjutan.

Berdasarkan hasil validasi performa, sistem memproses 10.000 dokumen dalam total waktu 150 detik (10 siklus *batch* × 15 detik). Dengan demikian, throughput sistem mencapai sekitar 66,7 dokumen per detik atau sekitar 4.000 *tweet* per menit, yang menunjukkan kemampuan sistem dalam menangani aliran data tanpa terjadi bottleneck. Hal ini membuktikan bahwa integrasi Kafka, Spark, dan MongoDB berjalan stabil dalam mendukung pemrosesan *real-time*.

Dari sisi *latency*, sistem memiliki waktu tunda rata-rata 15 detik sejak data diterima dari API hingga tersimpan di MongoDB. Selain itu, nilai *Success Rate* sebesar 100% menunjukkan seluruh 10.000 data yang dikirim oleh Spark berhasil tersimpan tanpa kehilangan maupun duplikasi, berkat penggunaan *tweet_id* sebagai *unique key*. Hasil ini menegaskan bahwa sistem memiliki performa yang cepat, stabil, dan reliabel.

4. Kesimpulan

Penelitian ini berhasil menerapkan Arsitektur Kappa sebagai pendekatan pemrosesan data *real-time* untuk memantau percakapan publik terkait hipertensi di media sosial X. Penerapan arsitektur *Big* Data berbasis *streaming* (*Kappa Architecture*) menggunakan Apache Kafka dan Apache Spark untuk mengolah dan menganalisis data percakapan tentang hipertensi di Media Sosial X secara *real-time* terbukti mampu membangun sistem pemrosesan yang terpadu dan efisien. Dengan memanfaatkan integrasi Apache Kafka sebagai *immutable event log*, Apache Spark *Structured Streaming* sebagai *engine* pemrosesan, dan MongoDB sebagai *serving layer*, sistem mampu membangun satu jalur pemrosesan terpadu tanpa pemisahan *batch* dan *speed layer*. Hasil pengujian menunjukkan bahwa sistem mampu memproses dan menyimpan 10.000 data tweet dengan *throughput* rata-rata 19,23 *tweet* per detik, *latency* sekitar 520 detik, serta *success rate* sebesar 100%, yang menandakan tidak adanya kehilangan maupun duplikasi data selama proses *streaming* berlangsung.

Implementasi Arsitektur Kappa secara keseluruhan terbukti efektif, stabil, dan skalabel dalam menangani data media sosial yang bersifat dinamis dan terus mengalir. Pendekatan ini tidak hanya menyederhanakan arsitektur pemrosesan data, tetapi juga meningkatkan efisiensi, konsistensi, dan

This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International (CC BY-NC 4.0) License
Copyright © 2026 Legawan Perkasa, Tikaridha Hardiani

kemampuan sistem dalam melakukan pemantauan serta analisis isu kesehatan masyarakat secara *real-time*. Dengan demikian, Arsitektur Kappa dapat menjadi solusi yang relevan untuk pengolahan *Big* data berbasis *streaming*, khususnya dalam konteks analisis dan monitoring topik kesehatan seperti hipertensi di media sosial.

References

Abirami T and Dr. Chandrasekar B S. (2024). Kappa and Lambda Architectures for Telecom Big Data Pipelines. *International Journal of Research Publication and Reviews*, *5*(9), 739–743. https://doi.org/10.13140/RG.2.2.16197.56803

Anom, H., Aji, S., & Prasetyo, A. C. (2024). Evaluasi Kinerja Jaringan WiFi Mahasiswa : Analisis Throughput , Delay , Jitter , dan Packet loss. *Jurnal BATIRSI*, *8*(1), 23–27.

Bertin, J., Penka, N., & Debauche, O. (2022). An Optimized Kappa Architecture for IoT Data Management in Smart. *Journal of Ubiquitous Systems & Pervasive Networks*, *17*(2), 59–65. https://doi.org/10.5383/JUSPN.17.02.002

Gede, K., Gede, I. P., Suputra, H., & Gde, I. A. (2024). Pengolahan Big Data Dengan Sharding Database Dan Kappa Architecture Untuk Data Time-Series. *Jurnal Elektronik Ilmu Komputer Udayana*, *13*(1), 43–54. https://doi.org/https://doi.org/10.24843/JLK.2024.v13.i01.p05

Hilmy Farid, Dadang Yusup, C. (2022). Analisis Usability Pada Aplikasi Momby Spa Menggunakan Metode Usability Testing. *Jurnal Ilmiah Wahana Pendidikan*, *8*(14), 155–163. https://doi.org/https://doi.org/10.5281/zenodo.6982246

Maulana, M. R., Fazilatunnisa, A., Febriansyah, M. Y., Muiz, A., & Fauzan, I. (2026). Analisis dan Prediksi Curah Hujan Bulanan Kota Serang Berbasis Apache Spark Menggunakan Dataset BPS Provinsi Banten. *Jurnal Ilmu Komputer Dan Teknik Informatika*, *2*(1), 15–21. https://doi.org/https://doi.org/10.64803/juikti.v2i1.78

Mikola, A., Sari, M., Informasi, T., Informasi, S., Informasi, F. T., Kristen, U., & Wacana, S. (2022). Analisis Sistem Jaringan Berbasis QoS untuk Hot-Spot Di Institut Shanti Bhuana. *JIFOTECH (JOURNAL OF INFORMATION TECHNOLOGY)*, *2*(1), 2–6. https://doi.org/10.46229/jifotech.v2i1.398

Mita Permatasari, T. H. (2025). Implementation Of The K-Nearest Neighbor Algorithm For Low Sodium Food. *Jurnal Sistem Informasi DanTeknologi Informasi*, *7*(3), 867–879. https://doi.org/https://doi.org/10.52005/jursistekni.v7i3.505

Musababa, M. A., Fachrie, M., & Yogyakarta, U. T. (2025). Data Streaming Pipeline Model Using DBSTREAM-Based Online Machine Learning for E-Commerce User Segmentation. *Journal of Applied Informatics and Computing (JAIC)*, *9*(6), 3346–3355. https://doi.org/https://doi.org/10.30871/jaic.v9i6.11522

Nabawi, F. (2022). Jurnal Implementasi Sistem Distribusi Pesan dan Proses Data Secara Real Time dengan Apache Kafka. *Jurnal Teknologi Informatika Dan Komputer*, *8*(1), 173–189. https://doi.org/10.37012/jtik.v8i1.836

Nursinggah, L., Mufizar, T., & Perjuangan, U. (2024). Analisis Sentimen Pengguna Aplikasi X Terhadap Program Makan Siang Gratis. *JITET (Jurnal Informatika Dan Teknik Elektro Terapan)*, *12*(3). https://doi.org/https://doi.org/10.23960/jitet.v12i3.4336

Park, S., & Huh, J. H. (2023). A Study on Big Data Collecting and Utilizing Smart Factory Based Grid Networking Big Data Using Apache Kafka. *IEEE Access*, *11*(September), 96131–96142. https://doi.org/10.1109/ACCESS.2023.3305586

Parmar, T. (2025). Data Architectures and Methods for Fast Track Data Processing Using Hot and Cold Paths. *SSRN Electronic Journal*, *February*. https://doi.org/10.2139/ssrn.5190568

Pradinata, A., Lestari Lokapitasari B, P., & Azis, D. H. (2023). Perancangan Aplikasi E-ticketing Dengan Model Arsitektur Microservice Menggunakan Kafka. *Buletin Sistem Informasi Dan Teknologi Islam*, *4*(3), 286–295. https://doi.org/https://doi.org/10.33096/busiti.v4i3.1806

Puthenpariyarath, S. (2025). REAL-TIME DATA PROCESSING WITH KAFKA VS . PUB / SUB. *International Journal of Data Analytics (IJDA)*, *5*(1), 1–12. https://doi.org/https://doi.org/10.34218/IJDA_05_01_001

Sabrina, D., Iqbal, M., & Suri, N. (2026). Komponen Biaya yang Mempengaruhi Total Cost of Illness pada

This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International (CC BY-NC 4.0) License
Copyright © 2026 Legawan Perkasa, Tikaridha Hardiani

Pasien Hipertensi Rawat Inap: Narrative Review. *Sains Medisina*, *4*(3), 218–223. https://doi.org/10.63004/snsmed.v4i3.902

Studies, M., & Guntupalli, B. (2023). ETL Architecture Patterns: Hub-and-Spoke, Lambda, and More. *International Journal of AI, BigData, Computational and Management Studies*, *4*(3), 61–71. https://doi.org/10.63282/3050-9416.ijaibdcms-v4i3p107

Tri Buana, D. M. (2022). Penggunaan aplikasi tik tok (versi terbaru) dan kreativitas anak. *Jurnal Inovasi*, *16*(12), 34–44. https://doi.org/https://doi.org/10.33557/ji.v16i2.2227

Vaghani Divyeshkumar. (2024). *Hybrid Data Processing Approaches: Combining Batch and Real Time Processing with Spark*. *SSRN 49533*. https://doi.org/https://doi.org/10.2139/ssrn.4953336

Zhou, Z., & Zhou, L. (2024). *applied sciences A Distributed Real-Time Monitoring Scheme for Air Pressure Stream Data Based on Kafka*. *14*(12), 4967. https://doi.org/https://doi.org/10.3390/app14124967

Zulkifli, R. (2025). Analisis Sentimen Real-Time Media Sosial Menggunakan Edge Computing dan Apache Kafka. *Bit-Tech*, *7*(3), 1106–1117. https://doi.org/10.32877/bt.v7i3.2372

This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International (CC BY-NC 4.0) License
Copyright © 2026 Legawan Perkasa, Tikaridha Hardiani