Implementasi Metode *Gradient Descent* untuk Peningkatan Kinerja *Database* pada Sistem Perekaman Kelelahan Mental Berbasis Sinyal *Electroencephalography*

Naufal Rijaldi Saputra[1], Edita Rosana Widasari[2]

Program Studi Teknik Komputer, Fakultas Ilmu Komputer, Universitas Brawijaya
Email: [1]saputrarijaldinaufal@gmail.com, [2]editarosanaw@ub.ac.id

Abstrak

Penelitian mengenai deteksi kelelahan mental melalui pengukuran fisiologis menggunakan sinyal *Electroencephalography* pernah dibahas pada penelitian yang dilakukan oleh Syifa' Hukma Shabiyya pada tahun 2023. Namun, penelitian tersebut memiliki kekurangan ketika digunakan langsung oleh psikolog, terutama psikolog klinis. Hal ini disebabkan oleh ketidakmampuan sistem atau alat hasil penelitian sebelumnya dalam menyimpan riwayat perekaman sinyal. Oleh karena itu, penelitian ini mengusulkan sistem perekaman kelelahan mental dalam bentuk *database* untuk memungkinkan penyimpanan rekam medis pasien dan pengembangan *dashboard* sebagai alat visualisasi data yang memudahkan psikolog klinis menyimpan dan melihat kembali data pasien. Sistem yang diusulkan dapat mendeteksi kelelahan mental dan menyimpan data ke dalam *database*. Metode *Gradient Descent* diimplementasikan melalui proses *self-tuning* untuk mencari nilai optimum waktu eksekusi *query* saat menampilkan data ke *dashboard*. Hasil penelitian menunjukkan proses pengujian parameter akan memperkecil ukuran dari setiap parameternya agar *database* dapat berjalan dengan optimal dengan nilai parameter yang sesuai dan *source* yang digunakan tidak terlalu berlebihan sehingga kinerja dari *database* akan meningkat dan semua proses yang dieksekusi di *database* menjadi lebih efektif. Selain itu, Terdapat 4 nilai optimum parameter yang meminimalkan waktu *query*. Rata-rata waktu eksekusi *query* pada penelitian ini adalah 0,0063 detik dan terjadi peningkatan kinerja *database* sebesar 79% setelah metode *Gradient Descent* diimplementasikan. Berdasarkan kesimpulan, hasil penelitian menunjukkan efektivitas sistem dalam menyimpan data dan mengoptimalkan waktu eksekusi *query* dapat mendukung psikolog klinis dalam menyimpan rekam medis pasien di *database*.

Kata kunci: *Kelelahan Mental, Electroencephalography, Gradient Descent, Database, Dashboard*

Abstract

Research on detecting mental fatigue through physiological measurements using Electroencephalography signal was discussed in research conducted by Syifa' Hukma Shabiyya in 2023. However, this research has shortcomings when used directly by psychologists, especially clinical psychologists. This is caused by the inability of the system or tool resulting from previous research to store a history of signal recording. Therefore, this research proposes a mental fatigue recording system in the form of a database to enable storage of patient medical records and the development of a dashboard as a data visualization tool that makes it easier for clinical psychologists. The proposed system can detect mental fatigue and save the data into a database. The Gradient Descent method is implemented through self-tuning to find the optimum value of query execution time when displaying data on the dashboard. The research results show that the parameter testing process will reduce the size of each parameter so that the database can run optimally with appropriate parameter values and sufficient sources so that the performance of the database will increase and all processes executed in the database become more effective. Apart from that, there are 4 optimum parameter values that minimize query time. The average query execution time in this study was 0.0063 seconds and there was an increase in database performance of 79% after the Gradient Descent method was implemented. Based on the conclusions, the research results show that the system's effectiveness in storing data and optimizing query execution time can support clinical psychologists in storing patient medical records in the database.

Keywords: *Mental Fatigue, Electroencephalography, Gradient Descent, Database, Dashboard*

1. PENDAHULUAN

Kelelahan merupakan kondisi fisiologis yang dapat menurunkan kinerja fisik dan non-fisik seseorang yang disebabkan oleh berbagai faktor seperti terlalu banyak aktivitas, kondisi fisik yang kurang fit, dan kelelahan mental (Romolo, Widasari dan Prasetio, 2022). Kelelahan dibagi menjadi kelelahan mental dan fisik. Kelelahan mental merupakan kondisi psikobiologis yang memengaruhi kinerja kognitif (Fadhilah, 2021). Berdasarkan kode etik Himpunan Psikologi Indonesia (HIMPSI), psikolog klinis memiliki tanggung jawab menyelenggarakan layanan kesehatan jiwa dan mencatat rekam medis sesuai regulasi (HIMPSI, 2010; Kemenkes RI, 2017).

Proses diagnosis kelelahan mental konvensional melibatkan subjektivitas psikolog klinis sehingga dapat ditingkatkan dengan pengukuran fisiologis sebagai validasi. Penelitian menggunakan *Electroencephalography* (EEG) untuk mendeteksi kelelahan mental telah dilakukan pada penelitian (Shabiyya dan Widasari, 2023) akan tetapi belum optimal karena tidak memiliki kemampuan menyimpan data rekam medis. Menurut Pasal 21 Permenkes 45 tahun 2017, psikolog wajib menyimpan rekam medis pasien. Namun, penelitian sebelumnya belum memiliki sistem penyimpanan data secara efektif. Oleh karena itu, penelitian ini merancang sistem *database* untuk rekam medis kelelahan mental dan *dashboard* visualisasi sehingga memungkinkan psikolog klinis mengakses dan memeriksa kembali data pasien.

Dalam proses implementasinya, penelitian ini menggunakan metode *Gradient Descent* melalui proses *self-tuning* untuk mengoptimalkan eksekusi *query* dan parameter *buffer pool database*. Tujuannya adalah mencari nilai optimum waktu eksekusi *query* dan nilai optimum parameter yang dapat meningkatkan kinerja *database* secara keseluruhan. Metode ini diharapkan dapat membantu psikolog klinis mengakses data dengan waktu *query* yang optimal serta menjadikan sistem efektif dan efisien dalam mendukung penanganan pasien kelelahan mental.

2. STUDI LITERATUR

2.1. *Database*

Database atau basis data merupakan kumpulan informasi terstruktur atau data yang biasanya disimpan secara elektronik dalam sistem komputer. *Database* biasanya dikendalikan oleh sistem manajemen *database* (DBMS). Data, DBMS, dan aplikasi yang terhubung dengannya biasanya disebut sebagai sistem *database*, sering disingkat menjadi *database*. Penyimpanan data dalam *database* yang paling umum digunakan saat ini biasanya dimodelkan dalam baris dan kolom dalam serangkaian tabel untuk membuat pemrosesan dan *query* data menjadi efisien. Data kemudian dapat dengan mudah diakses, dikelola, dimodifikasi, diperbarui, dikendalikan, dan diatur. Sebagian besar *database* menggunakan bahasa *query* terstruktur (SQL) untuk menulis dan membaca data.

2.2. *Database Management System* (DBMS)

Database Manajement Sistem (DBMS) merupakan suatu software yang digunakan sebagai perantara bagi *user* untuk membangun sebuah *database*. Selain itu DBMS membantu *user* dalam pemeliharaan dan memonitoring data dalam jumlah besar. DBMS dijalankan menggunakan bahasa *database* yang ditentukan oleh pengembang dalam melakukan interaksi ke *database* (Alexandra, 2017). Terdapat 2 jenis bahasa *database* yang dapat digunakan yaitu *Database Definition Languange* (DDL) dan *Data Manipulation Language* (DML). DDL digunakan untuk membuat tabel baru, membuat indeks, ataupun mengubah tabel. DML digunakan untuk melakukan menipulasi dan pengambilan data pada suatu *database* seperti penambahan data baru ke dalam *database*, menghapus data dari suatu *database*, dan pengubahan data di suatu *database*.

2.3. *Electroencephalography* (EEG)

Electroencephalography (EEG) merupakan suatu teknik pengukuran sinyal yang digunakan untuk mengetahui aktivitas listrik otak yang berbentuk sinyal serta bisa juga untuk mendeteksi adanya kelainan pada otak. Teknik ini bekerja dengan cara merekam aktivitas kelistrikan otak kemudian direpresentasikan ke dalam bentuk gelombang / grafik sinyal. Pada otak, terdapat 4 gelombang utama yang dapat terdeteksi yaitu gelombang *alpha*, *beta*, *delta*, dan *theta*. gelombang *alpha* memiliki frekuensi

8-12 siklus per detik dan hanya akan terpancar ketika manusia dalam keadaan sadar sepenuhnya ataupun saat mata tertutup. Selanjutnya ada gelombang *beta*, gelombang ini memiliki frekuensi 13-30 siklus dan akan terpancar ketika manusia dalam keadaan sadar. Selanjutnya ada gelombang *delta* yang bisa ditemukan ketika manusia sedang tertidur namun pada umumnya gelombang ini ditemukan pada anak kecil. Terakhir ada gelombang *theta*, gelombang *theta* memiliki 4-7 siklus dan terpancar ketika manusia sedang dalam keadaan tidur.

2.4. *Gradient Descent*

Gradient Descent sendiri merupakan algoritma yang dapat digunakan digunakan ketika melatih model pembelajaran mesin (Farsi, Setia Budi dan Primananda, 2021)Penelitian ini akan berfokus kepada sistem perekaman kelelahan mental yang berfungsi untuk menyimpan data sinyal *Electroencephalography* (EEG) di *database* dan dapat memvisualisasikan kembali data yang di simpan di *database* melalui *dashboard*. Metode *Gradient Descent* memiliki peran untuk menemukan nilai bobot yang memberikan nilai keluaran optimum. Metode ini memiliki prisnsip kerja memperkecil nilai fungsi biaya dengan cara melakukan perubahan pada nilai parameter secara bertahap (Syahrian, 2016). Metode *Gradient Descent* dipilih karena pada penelitian kali ini penulis akan menghitung dan mencari nilai optimum dari parameter yang telah ditentukan sehingga metode ini dipilih berdasarkan kecocokan antara kebutuhan dari penelitian dan fungsi dari metode itu sendiri. Metode *Gradient Descent* akan di implementasikan dalam proses perhitungan waktu eksekusi *query* ketika menampilkan data yang ada di dalam *database* ke dalam *dashboard* sehingga dengan digunakannya metode ini penulis dapat mengetahui waktu optimum dari eksekusi *query* di dalam *database*.

2.5. *Buffer Pool Size*

Buffer Pool Size merupakan variabel atau parameter yang terdapat pada *Buffer Pool* dari *database* server MariaDB. *Buffer Pool* sendiri merupakan area di dalam memori utama tempat *cache*, tabel, dan data indeks berada yang memungkinkan data yang sering digunakan diakses secara langsung tanpa melakukan pengambilan data dari penyimpanan (Oracle, 2024). Selain itu, pada MariaDB *database*, *Buffer Pool* merupakan komponen kunci untuk mengomptimalkan *database* tersebut. *Buffer Pool Size* merupakan variabel atau parameter paling penting dari *Buffer Pool* karena parameter tersebut menentukan ukuran *cache* dalam *Buffer Pool* yang berfungsi untuk membantu SQL dapat bekerja secara langsung dengan informasi dalam *cache* di *Buffer Pool* tanpa perlu melakukan akses ke penyimpanan atau *disk* sehingga waktu yang dipelukan untuk melakukan eksekusi *query* akan menjadi lebih cepat karena sistem hanya perlu mengambil data yang ada di *Buffer Pool* tanpa perlu mengambil data ke penyimpanan (MariaDB, 2023).

2.6. *Read Io Threads* dan *Write Io Threads*

Read Io Threads dan *Write Io Threads* berfungsi untuk mengatur jumlah *threads* atau utas untuk melakukan operasi *read* dan *write* pada mesin *InnoDB* dalam *database* MariaDB (MariaDB, 2023). *Write Io Threads* berfungsi untuk mengatur jumlah *threads* yang digunakan untuk menulis data pada penyimpanan di *database*. Sedangkan, *Read Io Threads* berfungsi untuk megatur jumlah *threads* yang digunakan untuk membaca data yang ada di penyimpanan *database*. Parameter ini dipilih karena pada dasarnya SQL atau *query* yang digunakan dalam penelitian ini adalah perintah untuk menuliskan dan membaca data ke dan dari *database* sehingga parameter ini digunakan dengan tujuan agar proses membaca dan menulis data di *database* dapat berjalan dengan baik tanpa hambatan sehingga dapat memaksimalkan kinerja *database*.

2.7. *Io Capacity*

Io Capacity berfungsi untuk membatasi aktivitas I/O untuk tugas latar belakang pada mesin penyimpanan *InnoDB* MariaDB *database* serta berfungsi untuk menggabungkan data dari *buffer insert* dan *flushing pages* (MariaDB, 2023). *Io Capacity* harus diatur ke sekitar jumlah operasi I / O per detik yang dapat ditangani sistem, berdasarkan jenis drive yang digunakan. *Io Capacity* dapat diatur lebih tinggi ketika server mulai membantu dengan beban kerja ekstra dan mengurangi ukurannya untuk penggunaan normal. Nilai *Io Capacity* yang besar akan berdampak pada peningkatan kinerja *database* ketika malakukan pemrosesan data akan tetapi semakin besar *Io Capacity* juga akan berdampak pada berkurangnya efektivitas *caching* karena data yang tersimpan di *buffer* akan semakin cepat terhapus.

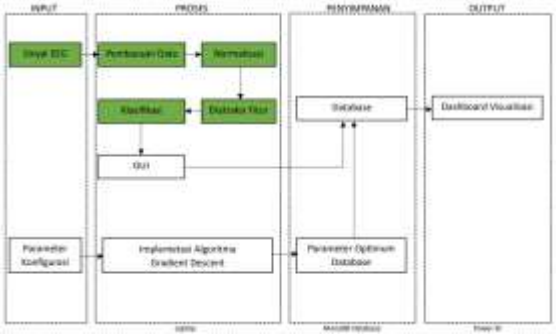
2.8. *Self-Tuning*

Tuning merupakan proses melakukan pengaturan parameter *database* dengan tujuan untuk meningkatkan kinerja dari database sehingga dapat mencapai tujuan yang spesifik (Rahmanto, Budi and Primananda, 2021). Pada penelitian ini, proses *tuning* akan menggunakan metode *Gradient Descent* dalam melakukan perubahan nilai parameter sehingga proses pengaturan parameter akan dilakukan secara otomatis mengikuti perhintungan matematis metode tersebut. Proses *tuning* yang dilakukan secara otomatis menggunakan metode tertentu dinamakan *self-tuning*. *Self-tuning* bertujuan untuk membantu *Database Administrator* (DA) untuk meningkatkan kinerja *database* secara otomatis tanpa campur tangan DA secara langsung ketika melakukan pengaturan parameter *database* pada saat *tuning*.

3. METODE PENELITIAN

3.1. Blok Diagram Sistem

Alur sistem pada penelitian ini dapat dilihat pada Gambar 1 berikut.

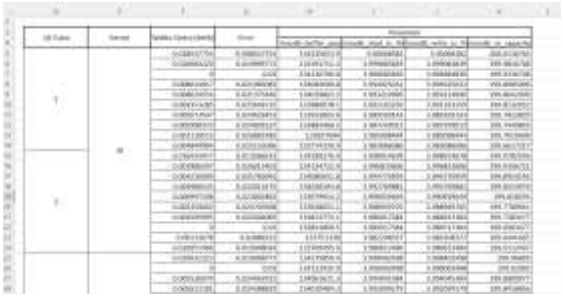


Gambar 1. Blok Diagram Sistem

Berdasarkan Gambar 1. Blok Diagram Sistem di atas, sistem dibagi menjadi 4 bagian yaitu *input*, proses, penyimpanan, dan *output*. Blok diagram sistem yang berwarna hijau telah dbahas pada penelitian (Shabiyya dan Widasari, 2023) dan blok diagram sistem berwarna putih merupakan fokus utama pada penelitian ini. Pada tahapan *input*, dilakukan konfigurasi parameter dengan nilai *default* dari setiap parameter *buffer pool* sebagai nilai awal yang di set untuk melakukan implementasi metode *Gradient Descent*. Pada tahapan proses, dilakukan implementasi GUI dan implementasi metode *Gradient Descent*. Implementasi GUI digunakan pada penelitian ini untuk memilih data yang akan di proses dan disimpan ke dalam *database* sedangkan implementasi metode *Gradient Descent* dilakukan untuk mencari nilai optimum dari parameter *buffer_pool_size*, *read_io_threads*, *write_io_threads*, dan *io_capacity* yang diukur berdasarkan waktu eksekusi *query*. Pada tahapan penyimpanan, nilai optimum dari setiap parameter yang telah ditemukan pada tahapan sebelumnya digunakan dengan cara mengatur nilai setiap parameter *database* menggunakan nilai optimumnya. Kemudian pada tahapan *output*, *dashboard* visualisasi digunakan untuk melihat data yang tersimpan di *database* yang ditampilkan dalam bentuk plot sinyal, informasi pasien, nilai ekstaksi fiturnya, dan status diagnosa mental pasien.

3.2. Teknik Pengumpulan Data

Data dikumpulkan secara langsung setelah *query* dieksekusi dengan syarat *query* yang dijalankan tersebut berhasil diekekusi ketika implementasi metode *Gradient Descent* dilakukan. Pengambilan data diambil dari *Database Management System* (DBMS) yang sama yaitu MariaDB dan menggunakan data uji yang sama. Pada penelitian ini, terdapat beberapa data yang diambil dari pengujian diantaranya adalah waktu *query*, nilai *error*, dan nilai parameter *buffer pool* (*buffer_pool_size, io_capacity, read_io_threads*, dan *write_io_threads*).



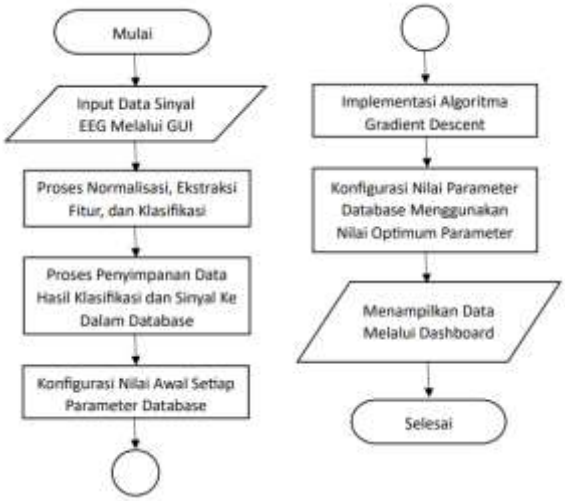
Gambar 2. Data Waktu *Query*, *Error*, dan Parameter *Database*

Gambar 2 di atas merupakan data gabungan dari waktu *query*, nilai *error*, dan nilai parameter *buffer pool*. Data data tersebut diambil ketika pengujian dan implementasi metode *Gradient Descent* dilakukan. Pada penelitian ini, pengambilan data dilakukan melalui dua kali uji coba pada setiap iterasinya.

4. PERANCANGAN DAN IMPLEMENTASI

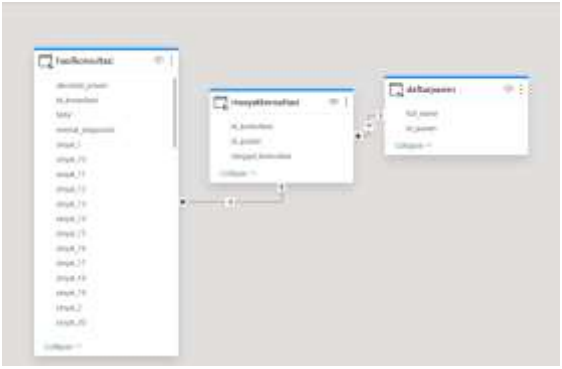
4.1. Perancangan Sistem

Diagram alir perancangan sistem secara keseluruhan dapat dilihat pada Gambar 3 berikut.



Gambar 3. Diagram Alir Perancangan Sistem

Berdasarkan Gambar 3. Diagram Alir Perancangan Sistem di atas, sistem dimulai dengan melakukan input data yang akan di proses melalui GUI MATLAB. Setelah itu, dilakukan proses normalisasi, ekstraksi fitur, dan klasifikasi sinyal EEG untuk mendapatkan diagnosa status mental pasien. Kemudian sinyal hasil normalisasi, nilai fitur, dan diagnosa status mental pasien disimpan ke dalam *database* bersamaan dengan informasi pribadi pasien seperti nama, id pasien, dan informasi pasien lainnya. Setelah itu, dilakukan konfigurasi nilai awal dari parameter sesuai dengan nilai *default* dari setiap parameter yang dilanjutkan dengan implementasi metode *Gradient Descent* dengan tujuan untuk mencari nilai optimum dari setiap parameter *database*. Setelah ditemukan nilai optimum dari setiap parameter, dilakukan konfigurasi ulang parameter *database* menggunakan nilai optimum dari setiap parameternya. Proses terakhir dari sistem ini adalah menampilkan data yang ada di *database* ke dalam *dashboard* visualisasi.

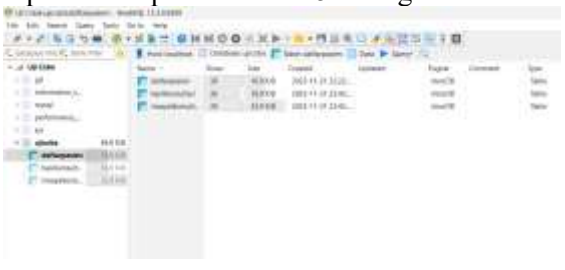


Gambar 4. Perancangan Skema dan Aristektur *Database*

Gambar 4. merupakan skema *database* yang digunakan pada penelitian ini. Skema *database* tersebut akan digunakan untuk menyimpan data rekam medis pasien psikolog pada penelitian ini. Skema *database* di atas menunjukkan bahwa *database* terdiri dari 3 tabel yang saling terhubung yaitu tabel hasil konsultasi, tabel riwayat konsultasi, dan tabel daftar pasien. Masing masing tabel memiliki *primary key* dan *foreign key* untuk menghubungkan tabel dengan tabel lainnya. Pada tabel daftar pasien, terdapat 2 atribut yaitu full_name dan id_pasien sedangkan untuk primary key dari tabel daftar pasien adalah atribut id_pasien. Pada tabel riwayat konsultasi, terdapat 3 atribut yaitu id_konsultasi, id_pasien, dan tanggal_konsultasi sedangkan untuk *primay key* dari tabel ini adalah id_konsultasi dan atribut yang berperan sebagai *foreign key* untuk menghubungkan tabel ini dengan tabel daftar pasien adalah atribut id_pasien. Sedangkan, tabel hasil konsultasi memiliki 64 atribut yang terdiri dari id_konsultasi, MAV, *absolute_power*, *standard deviation*, *mental diagnostic*, dan atribut sinyal 1 sampai sinyal 59. *Primary key* dari tabel ini adalah atribut id_konsultasi yang sekaligus juga menjadi *foreign key* untuk menghubungkan tabel hasil konsultasi dan tabel riwayat konsultasi.

4.2. Implementasi Sistem

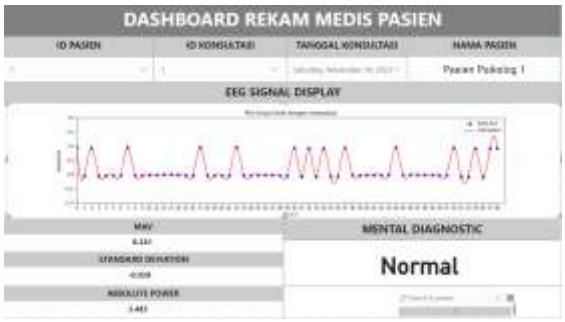
Implementasi sistem dari penelitian ini dapat dilihat pada Gambar 5. sebagai berikut.



Gambar 5. Implementasi Skema dan Arsitektur *Database*

Gambar 5. Implementasi Skema dan Arsitektur *Database* di atas merupakan implementasi *database* yang dilakukan pada MariaDB *database*. *Database* terdiri dari 3 tabel yang saling terhubung yaitu tabel hasil konsultasi, tabel riwayat konsultasi, dan tabel daftar pasien.

Untuk menampilkan data rekam medis pasien yang ada di *database*, diperlukan suatu *interface* yang dapat dioperasikan dengan mudah oleh psikolog. Pada penelitian ini, *Dashboard* di Power BI akan digunakan sebagai *interface* untuk menampilkan data yang tersimpan di *database* karena *dashboard* dibuat sesederhana mungkin agar psikolog dapat dengan mudah mengoperasikannya. *Dashboard* yang akan digunakan oleh psikolog dapat dilihat pada Gambar 6. Implementasi *Dashboard* berikut.



Gambar 6. Implementasi *Dashboard*

Dashboard akan menampilkan data seperti nama pasien, *mental diagnostic*, MAV, AVP, dan SD pasien sesuai dengan data yang ada di *database*. Selain itu, *dashboard* ini juga dapat menampilkan visualisasi dari sinyal EEG yang sudah di plot menggunakan bahasa *Python* dan dilakukan proses interpolasi ketika menampilkan data dengan tujuan untuk mendapatkan hasil ploting sinyal yang lebih *smooth*. Selain itu, *dashboard* juga memiliki 3 filter yang dapat digunakan untuk memilih data yang akan ditampilkan di *dashboard*. Filter tersebut terdiri dari id pasien, id konsultasi, dan tanggal konsultasi.

5. HASIL DAN PEMBAHASAN

5.1. Hasil Pengujian Nilai Parameter Pada Metode *Gradient Descent*

Pengujian dilakukan dengan cara melakukan 2 kali uji coba pada setiap iterasinya. Iterasi pada pengujian ini dilakukan sebanyak 10, 20, 30, dan 40 yang dilakukan secara berurutan dimulai dari iterasi yang paling kecil. Pada pengujian ini, waktu *query* akan didapatkan dalam satuan detik hasil dari perhitungan waktu ketika menjalankan *query* untuk melakukan pengambilan nilai pada tabel hasil konsultasi berdasarkan partisi id pasien. pada parameter *buffer pool* size hasil pengujian akan mendapatkan besar memori yang digunakan pada *database* dengan satuan *byte*. Pada parameter *read io thread, write io threads,* dan *io capacity* hasil dari pengujian akan berupa nilai yang bersifat numerik dan tidak memiliki satuan. Hasil dari pengujian nilai parameter pada metode *Gradient Descent* dapat dilihat pada Tabel 1 berikut.

Tabel 1. Nilai Awal dan Akhir Parameter

Parameter	Nilai Akhir	Nilai Awal
Buffer_pool_size	132208813	134175701
Read_io_threads	3.923	3.998
Write_io_threads	3.923	3.998
Io_capacity	198.467	199.967

Berdasarkan analisis keempat parameter diatas, dapat disimpulkan bahwa keempat parameter memiliki *trend line* menurun yang berarti bahwa proses pengujian parameter akan memperkecil ukuran dari setiap parameternya yang disebabkan oleh *query* yang sangat sederhana dan waktu eksekusi *query* sangat cepat sehingga hasil dari perhitungan metode *Gradient Descent* akan mengubah nilai parameter menjadi lebih kecil ketika jumlah iterasi semakin besar. Hal tersebut dilakukan agar *database* dapat berjalan dengan optimal dengan ukuran atau nilai parameter yang sesuai sehingga *source* yang digunakan tidak terlalu berlebihan sehingga kinerja dari *database* akan meningkat dan semua proses yang dieksekusi di *database* menjadi lebih efektif.

5.2. Hasil Pengujian Nilai Optimum dan Nilai Rata-Rata Parameter *Database*

Pengujian dilakukan sebanyak 4 kali uji coba dengan jumlah iterasi yang digunakan untuk mencari nilai parameter adalah sebanyak 10, 20, 30, dan 40 iterasi. Hal tersebut dilakukan karena maksimum iterasi yang dapat dilakukan oleh sistem adalah 40 sehingga dengan memaksimalkan iterasi maka akan semakin banyak nilai parameter yang dapat digunakan

untuk perhitungan rata rata dan diharapkan dapat membuat perhitungan rata rata menjadi lebih akurat. Pada pengujian ini, nilai optimum parameter diambil dari nilai parameter yang memiliki nilai waktu *query* mendekati nol. Sehingga pada pengujian ini terdapat beberapa parameter optimum yang dapat dilihat pada Tabel 2 berikut.

Tabel 2. Nilai Optimum Parameter

Parameter		
Buffer_pool_size	**Read/Write_io_threads**	**Io_Capacity**
134130797	3.997	199.934
134112936	3.996	199.920
134068860	3.994	199.886
133916265	3.989	199.770

Tabel 3. Nilai Rata Rata Parameter

Parameter		
Buffer_pool_size	**Read/Write_io_threads**	**Io_Capacity**
133448404	3.971	199.413

Empat nilai optimum parameter pada Tabel 2. Di atas dapat digunakan atau di konfigurasikan pada *database* MariaDB agar *database* tersebut bekerja secara optimal dan Tabel 3. Merupakah hasil perhitungan rata rata dari nilai parameter setelah di implementasikan metode *Gradient Descent.*

5.3. Hasil Pengujian Waktu Rata Rata dan Peningkatan Kinerja *Database*

Tabel 4. Waktu Optimum Eksekusi Query

Waktu Optimum Eksekusi Query		
Uji Coba	**Iterasi**	**Waktu Query (Detik)**
1	3	0
2	2	0
3	3	0
4	6	0

Berdasarkan Tabel 4. Waktu Optimum Eksesusi Query diatas, hasil pengujian menunjukkan bahwa waktu eksekusi *query* optimum pada pengujian ini adalah 0 detik pada pengujian 1, pengujian 2, pengujian 3, dan pengujian 4. Walaupun pada kenyataanya, proses eksekusi *query* secepat apapun pasti akan membutuhkan waktu. Waktu hasil *query* 0 disebabkan oleh *query* yang sangat sederhana serta keterbatasan sistem dan perangkat yang digunakan dalam menangkap waktu eksekusi yang sangat cepat sehingga nilai yang terdeteksi menjadi 0. Rata rata waktu eksekusi *query* pada pengujian kali ini dapat dilakukan dengan mengikuti Persamaan 1 sebagai berikut.

Rata-rata waktu eksekusi *query* = $\frac{total\ waktu\ query}{jumlah\ iterasi}$

Rata-rata waktu eksekusi *query* = $\frac{0{,}6267}{100}$

Rata-rata waktu eksekusi *query* = 0,0063 detik

Peningkatan kinerja database database yang dihitung berdasarkan waktu ekseusi *query* pada pengujian kali ini dapat dilakukan dengan mengikuti Persamaan 2 sebagai berikut:

Peningkatan Kinerja = $\frac{Target\ waktu\ eksekusi\ -\ rata\ rata\ waktu\ eksekusi\ query}{Target\ waktu\ eksekusi}\ X\ 100\%$

Peningkatan Kinerja = $\frac{0{,}03-0{,}0063}{0{,}03}$ x 100%

Peningkatan Kinerja = $\frac{0{,}0237}{0{,}03}$ x 100%

Peningkatan Kinerja = 79%

Berdasarkan perhitungan diatas, terjadi peningkatan kinerja *database* sebesar 79% ketika mengeksekusi *query*. Peningkatan tersebut merupakan selisih dari target waktu eksekusi *query* dengan rata rata waktu eksekusi *query* (setelah implementasi metode *Gradient Descent*) kemudian dibagi dengan target waktu eksekusi awal dan dikalikan dengan 100%. Peningkatan kinerja *database* tersebut dihitung setelah mengimplementasikan metode *Gradient Descent*.

6. KESIMPULAN

Nilai 4 parameter database memiliki *trend line* menurun yang terjadi karena pada proses pengujian parameter akan memperkecil ukuran dari setiap parameternya yang disebabkan oleh *query* yang sangat sederhana dan waktu eksekusi *query* yang sangat cepat sehingga hasil dari perhitungan metode *Gradient Descent* akan mengubah nilai parameter menjadi lebih kecil ketika jumlah iterasi semakin besar. Hal tersebut dilakukan agar *database* dapat berjalan dengan optimal dengan ukuran atau nilai parameter yang sesuai dan *source* yang digunakan tidak terlalu berlebihan sehingga kinerja dari *database* akan meningkat dan semua proses yang dieksekusi di *database* menjadi lebih efektif.

Selain itu, terdapat 4 nilai optimum parameter yang diambil dari nilai parameter yang memiliki nilai waktu *query* mendekati nol. Nilai optimum parameter tersebut dapat digunakan atau di konfigurasikan pada *database* MariaDB agar *database* tersebut bekerja secara optimal. Pada pengujian ini juga didapatkan nilai rata rata dari parameter *Innodb_buffer_pool_size* sebesar 133448404 *byte*,

Innodb_read_io_threads sebesar 3,971, *Innodb_write_io_threads* sebesar 3,971, dan *Innodb_io_capacity* sebesar 199,413.

Hasil perhitungan menujukkan bahwa terjadi peningkatan kinerja *database* sebesar 79% setelah metode *Gradient Descent* diimplementasikan dengan target waktu eksekusi *query* 0.03 detik. Selain itu, rata rata waktu eksekusi *query* setelah implementasi metode *Gradient Descet* adalah 0,0063 detik. Hal tersebut menujukkan bahwa rata rata waktu eksekusi *query* lebih cepat 0,0237 detik setelah ditemukan parameter optimum *database* hasil implementasi metode *Gradient Descent*.

7. DAFTAR PUSTAKA

Alexandra, J. (2017, Desember 18). *Database Management System*. Retrieved from School of Information System: https://sis.binus.ac.id/2017/12/18/*database*-management-system/

Fadhilah, R. (2021). Pengaruh Beban Kerja Terhadap Work Fatigue Pada Karyawan PT. Pos Indonesia (Persero) Banda Aceh.

Farsi, Z., Setia Budi, A. dan Primananda, R., 2021. Implementasi Self-Tuning Pada Sistem Database Untuk Meningkatkan Kinerja Query Dengan Menggunakan Metode Gradient Descent. [online] 5(1), pp.271–275. Available at: <http://jptiik.ub.ac.id>

HIMPSI, 2010. Kode Etik Psikologi Indonesia. *Pengurus Pusat Himpunan Psikologi Indonesia*, [online] pp.11–19. Available at: <http://himpsi.or.id/phocadownloadpap/kode-etik-himpsi.pdf>.

Kemenkes RI, 2017. Peraturan Menteri Kesehatan Nomor 45 Pasal 21 tahun 2017 tentang Izin dan Praktik Psikolog Klinis.Jakarta: Mentri Kesehatan Republik Indonesia.

MariaDB, 2023. MariaDB *Server Documentation*. [Online] Available at: https://mariadb.com/kb/en/innodb-buffer-pool/[Diakses 24 Maret 2024].

Oracle, 2024. MySQL 8.0 *Reference Manual*. [Online] Available at: https://dev.mysql.com/doc/refman/8.0/en/innodb-buffer-pool.html [Diakses 24 Maret 2024].

Rahmanto, A., Budi, A.S. and Primananda, R., 2021. Implementasi Self - Tuning Pada Database Dengan Menggunakan Metode Nesterov Accelerated Gradient. 5(5), pp.1907–1913.

Romolo, D., Widasari, E.R. dan Prasetio, B.H., 2022. Analisis Perbandingan Filter Finite Impulse Response, Infinite Impulse Response, dan Discrete Wavelet Transform pada Kondisi Kelelahan Mental berbasis Sinyal Electroenchephalography. *Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer*, 6(9), pp.4580–4585.

Shabiyya, S.H. dan Widasari, E.R., 2023. Implementasi Algoritme Learning Vector Quantization untuk Deteksi Kelelahan Mental berbasis Sinyal Electroencephalography. *Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer*, 7(13).

Syahrian, F., 2016. Perbandingan Metode Optimasi Stochastic Gradient Descent, ADADELTA, dan Adam pada Jaringan Saraf Tiruan dalam Klasifikasi Data Aritmia.