

LAPORAN TUGAS BESAR 3

Penerapan String Matching dan Regular Expression dalam Pembuatan ChatGPT Sederhana



Disusun Oleh:
Kelompok 30

Anggota Kelompok:
Ezra Maringen Christian Mastra Hutagaol - 13521073
Moch. Sofyan Firdaus - 13521083
Addin Munawwar Yusuf - 13521085

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG
BANDUNG
MEI 2023

Bab 1

Deskripsi Tugas

Dalam tugas besar ini, anda diminta untuk membangun sebuah aplikasi ChatGPT sederhana dengan mengaplikasikan pendekatan QA yang paling sederhana tersebut. Pencarian pertanyaan yang paling mirip dengan pertanyaan yang diberikan pengguna dilakukan dengan algoritma pencocokan string Knuth-Morris-Pratt (KMP) dan Boyer-Moore (BM). Regex digunakan untuk menentukan format dari pertanyaan (akan dijelaskan lebih lanjut pada bagian fitur aplikasi). Jika tidak ada satupun pertanyaan pada database yang exact match dengan pertanyaan pengguna melalui algoritma KMP ataupun BM, maka gunakan pertanyaan termirip dengan kesamaan setidaknya 90% Apabila tidak ada pertanyaan yang kemiripannya di atas 90%, maka chatbot akan memberikan maksimum 3 pilihan pertanyaan yang paling mirip untuk dipilih oleh pengguna.

Perhitungan tingkat kemiripan dibebaskan kepada anda asalkan dijelaskan di laporan, namun disarankan menggunakan salah satu dari algoritma Hamming Distance, Levenshtein Distance, ataupun Longest Common Subsequence

Fitur-Fitur Aplikasi:

ChatGPT sederhana yang anda membuat wajib dapat melakukan beberapa fitur/ klasifikasi *query* seperti berikut:

1. Fitur pertanyaan teks (didapat dari database)

Mencocokkan pertanyaan dari input pengguna ke pertanyaan di database menggunakan algoritma **KMP atau BM**.

2. Fitur kalkulator

Pengguna memasukkan input query berupa persamaan matematika. Contohnya adalah $2*5$ atau $5+9*(2+4)$. Operasi cukup Tambah, kurang, kali, bagi, pangkat, kurung.

3. Fitur tanggal

Pengguna memasukkan input berupa tanggal, lalu chatbot akan merespon dengan hari apa di tanggal tersebut. Contohnya adalah 25/08/2023 maka chatbot akan menjawab dengan hari senin.

4. Tambah pertanyaan dan jawaban ke database

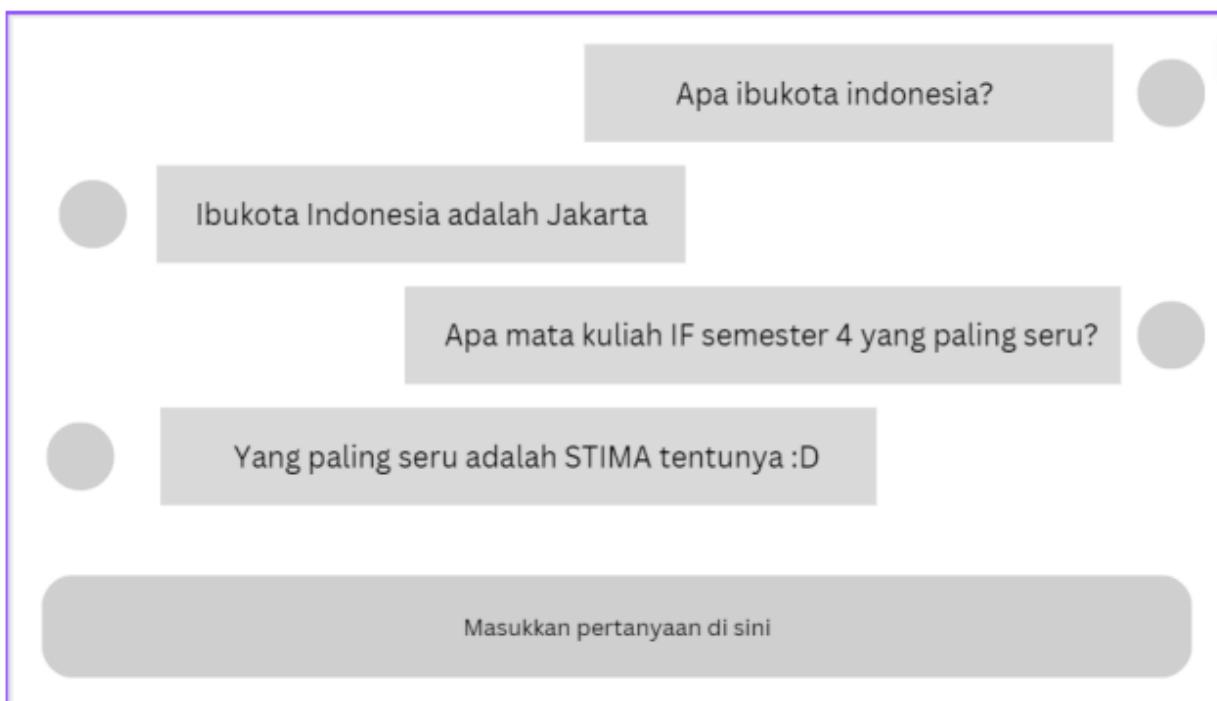
Pengguna dapat menambahkan pertanyaan dan jawabannya sendiri ke database dengan query contoh “Tambahkan pertanyaan xxx dengan jawaban yyy”. Menggunakan algoritma string

matching untuk mencari tahu apakah pertanyaan sudah ada. Apabila sudah, maka jawaban akan diperbaharui

5. Hapus pertanyaan dari database

Pengguna dapat menghapus sebuah pertanyaan dari database dengan query contoh “Hapus pertanyaan xxx”. Menggunakan string algoritma string matching untuk mencari pertanyaan xxx tersebut pada database.

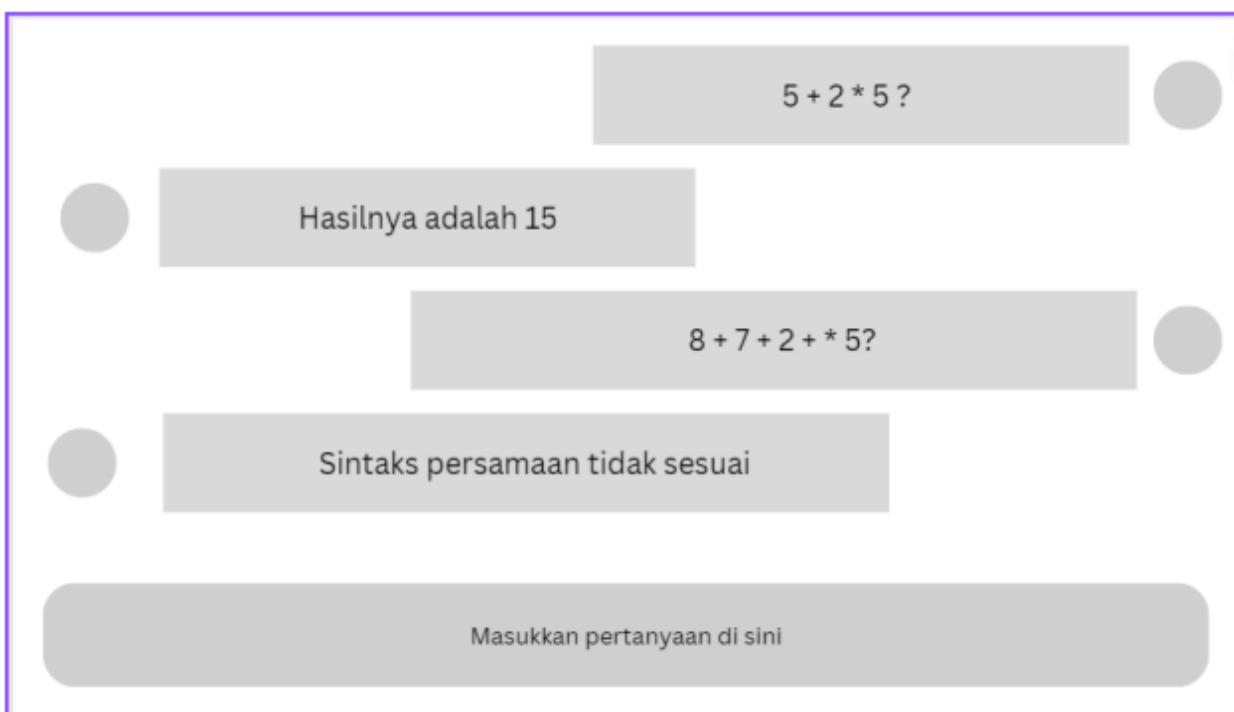
Klasifikasi dilakukan menggunakan regex dan terklasifikasi layaknya bahasa sehari - hari. Algoritma string matching KMP dan BM digunakan untuk klasifikasi query teks. Tersedia toggle untuk memilih algoritma KMP atau BM. Semua pemrosesan respons dilakukan pada sisi backend. Jika ada pertanyaan yang sesuai dengan fitur, maka tampilkan saja “Pertanyaan tidak dapat diproses”. Berikut adalah beberapa contoh ilustrasi sederhana untuk tiap pertanyaannya. (Note: Tidak wajib mengikuti ilustrasi ini, tampilan disamakan dengan chatGPT juga boleh



Gambar 1. Ilustrasi Fitur Pertanyaan teks kasus exact



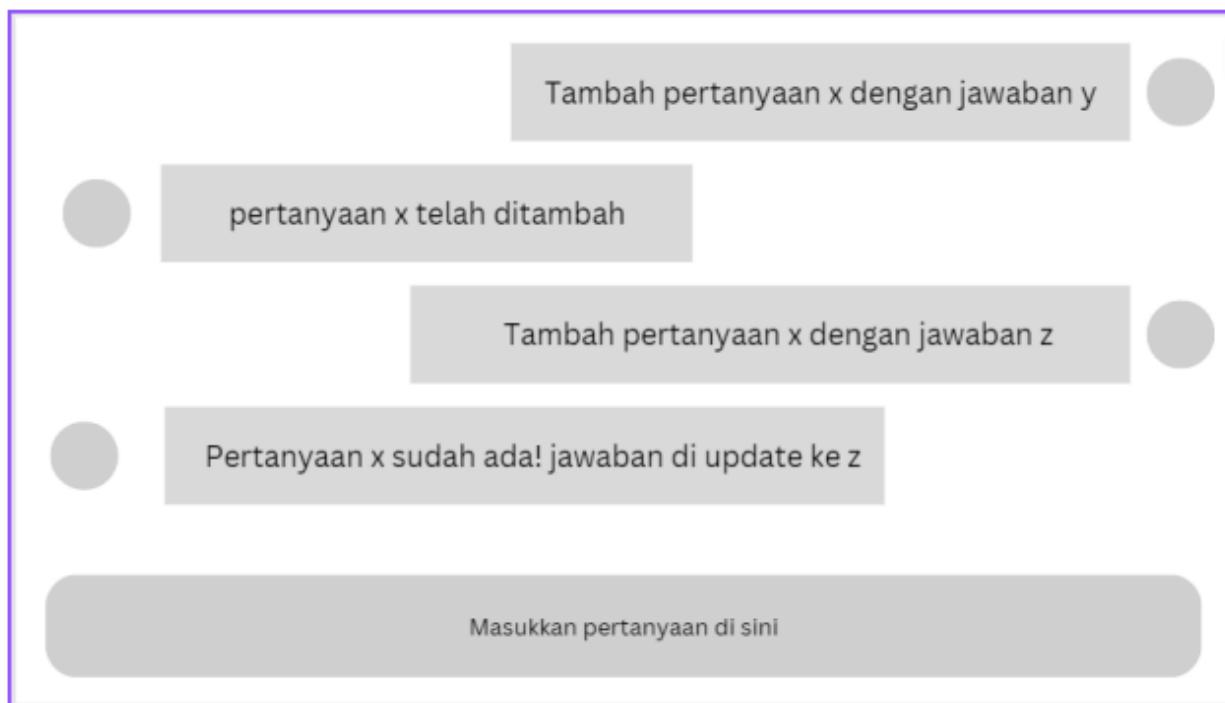
Gambar 2. Ilustrasi Fitur Pertanyaan teks kasus tidak exact



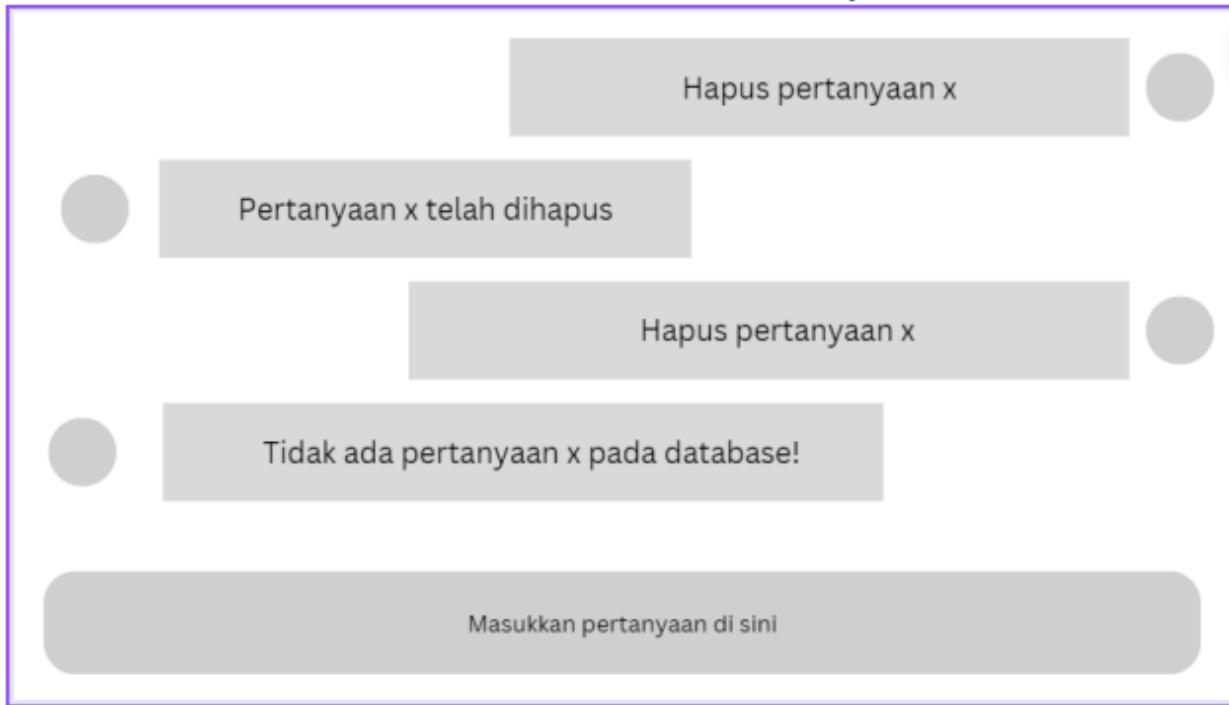
Gambar 3. Ilustrasi Fitur Kalkulator



Gambar 4. Ilustrasi Fitur Tanggal

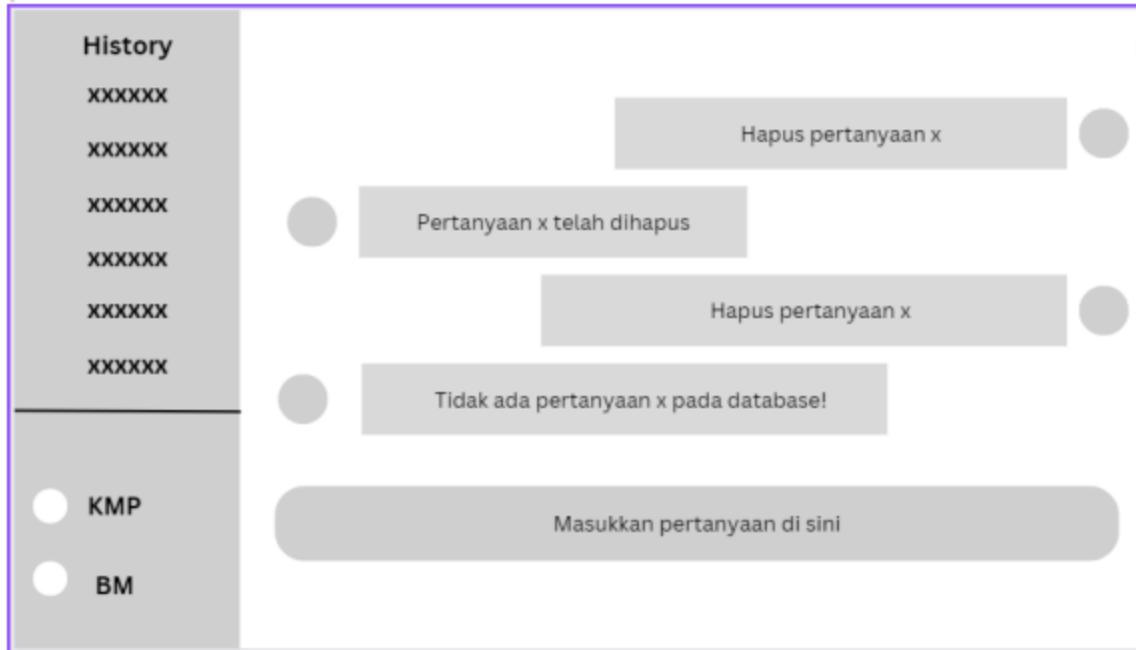


Gambar 5. Ilustrasi Fitur Tambah Pertanyaan



Gambar 6. Ilustrasi Fitur Hapus Pertanyaan

Layaknya ChatGPT, di sebelah kiri disediakan history dari hasil pertanyaan anda. Cukup tampilkan 5-10 pertanyaan terbaru di toolbar kiri. Perhatikan bahwa sistem history disini disamakan dengan chatGPT, sehingga satu history yang diklik menyimpan seluruh pertanyaan pada sesi itu. Apabila history diclick, maka akan merestore seluruh pertanyaan dan jawaban di halaman utama. Contoh ilustrasi keseluruhan:



Gambar 7. Ilustrasi Keseluruhan

Spesifikasi Program:

1. Aplikasi berbasis website dengan pembagian Frontend dan Backend yang jelas.
2. Implementasi Backend wajib menggunakan Node.js / Golang, sedangkan Frontend dibebaskan tetapi disarankan untuk menggunakan React / Next.js / Vue / Angular. Lihat referensi untuk selengkapnya.
3. Penyimpanan data wajib menggunakan basis data (MySQL / PostgreSQL / MongoDB).
4. Algoritma pencocokan string (KMP dan Boyer-Moore) dan Regex wajib diimplementasikan pada sisi Backend aplikasi.
5. Informasi yang wajib disimpan pada basis data:
 - a. Tabel pasangan pertanyaan dan Jawaban
 - b. Tabel history
6. Skema basis data dibebaskan asalkan mencakup setidaknya kedua informasi di atas.
7. Proses string matching pada tugas ini Tidak case sensitive.
8. Pencocokan yang dilakukan adalah dalam satu kesatuan string pertanyaan utuh (misal “Apa ibukota Filipina?”), bukan kata per kata (“apa”, “ibukota”, “Filipina”).

Bab 2

Landsasan Teori

Bab 2.1 Deskripsi singkat algoritma KMP, BM, dan Regex

KMP

Algoritma Knuth-Morris-Pratt adalah algoritma string matching yang dikembangkan oleh Donald E. Knuth pada 1967 dan James H. Morris bersama dengan Vaughan R. Pratt pada tahun 1966. Algoritma ini mirip seperti algoritma brute force, mengecek dari kiri ke kanan, akan tetapi dilakukan pergeseran yang lebih hemat sehingga bisa menghindari perbandingan yang sia-sia

Sebelum pengecekan, algoritma ini memroses pola yang akan dicari terlebih dahulu. Proses ini adalah fungsi pinggiran (border function). Fungsi ini mencari prefix terpanjang dari $p[0..i]$ yang juga termasuk suffix terpanjang dari $p[1..i]$. Dengan menggunakan fungsi ini, jika ada karakter yang berbeda dengan pola ($P[j] \neq T[i]$), maka digunakan fungsi tersebut untuk “lompat”, menghindari pengecekan yang sia-sia. Langkah-langkah algoritma ini adalah: ($P[j] \neq T[i]$), maka digunakan fungsi tersebut untuk “lompat”, menghindari pengecekan yang sia-sia. Langkah-langkah algoritma ini adalah:

1. Algoritma Knuth-Morris-Pratt mulai mencocokkan pattern pada awal teks.
2. Dari kiri ke kanan, algoritma ini akan mencocokkan karakter per karakter pattern dengan karakter di teks yang bersesuaian, sampai salah satu kondisi berikut dipenuhi:
 - Karakter di pattern dan di teks yang dibandingkan tidak cocok (mismatch)
 - Semua karakter di pattern cocok. Kemudian algoritma akan memberitahukan penemuan di posisi ini dan loop berhenti
3. Algoritma kemudian menggeser pattern berdasarkan tabel hasil border function, lalu mengulangi langkah 2 sampai pattern berada di ujung teks.

BM

Algoritma Boyer-Moore adalah algoritma string matching yang dipublikasikan oleh Robert S. Boyer dan J. Strother Moore pada tahun 1977. Algoritma ini dianggap sebagai algoritma yang paling efisien pada aplikasi umum. Algoritma ini menggunakan 2 teknik, the looking glass dan character jump. Teknik the looking glass adalah membandingkan pola dan teks dari akhir pola ke awal pola, dicek secara mundur. Teknik character jump terjadi ketika $T[i] == x$ tapi $P[j]$ tidak sama dengan $T[i]$. Teknik ini ada 3 kasus. Kasus pertama x ada di P , maka P akan bergeser ke kanan sehingga x terakhir di P sejajar dengan $T[i]$. Kasus kedua adalah x ada di P , tapi P harus bergeser ke kiri. Karena tidak boleh geser ke kiri, maka geser P sebanyak 1 karakter ke $T[i+1]$. Kasus 3 terjadi ketika tidak ada x di P , maka geser $P[0]$ ke $T[i+1]$. Langkah-langkah dari algoritma ini adalah sebagai berikut:

1. Algoritma Boyer-Moore mulai mencocokkan pattern pada awal teks.
2. Dari kanan ke kiri (the looking glass), algoritma ini akan mencocokkan karakter per karakter pattern dengan karakter di teks yang bersesuaian, sampai salah satu kondisi berikut dipenuhi:
 - Karakter di pattern dan di teks yang dibandingkan tidak cocok (mismatch).
 - Semua karakter di pattern cocok. Kemudian algoritma akan memberitahukan penemuan di posisi ini dan memberhentikan loop
3. Jika mismatch, gunakan teknik character jump, lalu ulangi langkah ke-2

Regex

Regex merupakan sekumpulan notasi yang dapat digunakan untuk pencocokan string. Regex bisa berupa sebuah teks (string) yang mendefinisikan sebuah pola pencarian sehingga dapat membantu kita untuk melakukan matching (pencocokan), locate (pencarian), dan manipulasi teks. Pencocokan string yang lebih fleksibel tergantung notasi yang digunakan untuk mencocokkan pola dengan teks.

Bab 2.2 Aplikasi web yang dibangun

Aplikasi ChatGPT sederhana yang dibuat, merupakan sebuah aplikasi interaktif yang dapat digunakan layaknya ChatGPT. Pertanyaan - pertanyaan yang diberikan user dapat disimpan pada basis data lalu aplikasi ini juga bisa menjawab pertanyaan pertanyaan user yang sudah disimpan di basis data. Berikut adalah fitur-fitur dari aplikasi:

1. Fitur pertanyaan teks (didapat dari database)

Mencocokkan pertanyaan dari input pengguna ke pertanyaan di database menggunakan algoritma **KMP** atau **BM**.

2. Fitur Kalkulator

Pengguna memasukkan input query berupa persamaan matematika. Contohnya adalah $2*5$ atau $5+9*(2+4)$. Operasi yang bisa dilakukan hanyalah tambah, kurang, kali, bagi, pangkat, kurung.

3. Fitur tanggal

Pengguna memasukkan input berupa tanggal, lalu chatbot akan merespon dengan hari apa di tanggal tersebut. Contohnya adalah pengguna memasukkan input 25/08/2023 maka chatbot akan menjawab dengan hari senin.

4. Tambah pertanyaan dan jawaban ke database

Pengguna dapat menambahkan pertanyaan dan jawabannya sendiri ke database dengan query contoh “Tambahkan pertanyaan **XXX** dengan jawaban **YYY**”. menggunakan algoritma string matching untuk mencari tahu apakah pertanyaan sudah ada. Apabila sudah, maka jawaban akan diperbarui.

5. Hapus pertanyaan dari database

Pengguna dapat menghapus sebuah pertanyaan dari database dengan query contoh “Hapus pertanyaan **XXX**”. Menggunakan string algoritma string matching untuk mencari pertanyaan **XXX** tersebut pada database

Proses pencocokkan string bisa dilakukan dengan algoritma **KMP** atau **BM** sesuai dengan pilihan user..

Bab 3

Analisis Pemecahan Masalah

Bab 3.1 Analisis Pemecahan Masalah

Program akan menerima input berupa string. Dari input tersebut akan digunakan algoritma pencocokan string **Knuth-Morris-Pratt(KMP)** dan **Boyer-Moore(BM)**. Regex digunakan untuk menentukan bahwa input string dari user ingin menggunakan fitur apa yang disediakan oleh aplikasi. Pada algoritma string matching ini digunakan algoritma **Hamming Distance** untuk menghitung tingkat kemiripan. Algoritma **Hamming Distance** adalah algoritma yang digunakan untuk menghitung jarak antara dua string dengan panjang yang sama dalam hal perbedaan karakter. Jarak Hamming antara dua string didefinisikan sebagai jumlah karakter yang berbeda pada posisi yang sama di kedua string tersebut. Algoritma Hamming distance bekerja dengan mengiterasi kedua string secara bersamaan, karakter per karakter, dan memeriksa apakah karakter-karakter tersebut sama atau tidak. Jika karakter-karakter tersebut sama, algoritma tidak melakukan apa-apa dan lanjut ke karakter berikutnya. Namun, jika karakter-karakter tersebut berbeda, algoritma menambahkan 1 pada hitungan jarak Hamming dan lanjut ke karakter berikutnya. Setelah iterasi selesai, algoritma mengembalikan nilai jarak hamming yang dihitung

Bab 3.2 Langkah penyelesaian masalah setiap fitur

1. Fitur Pertanyaan teks (didapat dari database)

Pada fitur ini, program akan memeriksa input terlebih dahulu lalu mencocokkan pertanyaan dari input pengguna ke pertanyaan di database menggunakan algoritma KMP atau BM. Jika pertanyaan yang dimasukkan dari input user ada di database, maka program akan memberikan jawaban dari pertanyaan tersebut

2. Fitur Kalkulator

Pada fitur ini, program akan memeriksa input terlebih dahulu menggunakan algoritma string matching apakah pengguna sedang memasukkan input untuk menggunakan kalkulator atau tidak. Setelah memeriksa input maka program akan melakukan penghitungan kalkulator dan menampilkan kembali jawaban dari pertanyaan tersebut. Pada kalkulator ini operasi yang bisa digunakan hanyalah tambah (+), kali (x), kurang (-), bagi (/), pangkat (^), dan kurung (())

3. Fitur tanggal

Pada fitur ini, program akan memeriksa input terlebih dahulu menggunakan algoritma string matching apakah pengguna sedang memasukkan input untuk menanyakan tanggal atau tidak. Setelah memeriksa input maka program akan merespon dengan hari apa di tanggal tersebut.

4. Tambah pertanyaan dan jawaban ke database

Pada fitur ini, program akan memeriksa input terlebih dahulu menggunakan algoritma string matching apakah pengguna sedang memasukkan input untuk menambahkan pertanyaan dan jawaban ke database atau tidak. Setelah memeriksa input maka program akan menambahkan pertanyaan dan jawaban ke database. Apabila ternyata pertanyaan yang dimasukkan sudah ada, maka jawaban akan diperbaharui

5. Hapus pertanyaan dari database

Pada fitur ini, program akan memeriksa input terlebih dahulu menggunakan algoritma string matching apakah pengguna sedang memasukkan input untuk menghapus sebuah pertanyaan dari database atau tidak. Setelah memeriksa input maka program akan mencari pertanyaan tersebut di dalam database lalu menghapusnya

Bab 3.3 Fitur fungsional dan arsitektur aplikasi web yang dibangun

Terdapat beberapa fitur fungsional yang dibangun pada aplikasi web ini, antara lain :

1. Fitur pertanyaan teks (didapat dari database)
2. Fitur kalkulator
3. Fitur tanggal
4. Tambah pertanyaan dan jawaban ke database
5. Hapus pertanyaan dari database

Secara garis besar, source code dari aplikasi web ini terbagi menjadi 2 yaitu, folder backend dan folder frontend. Implementasi backend dari aplikasi web ini menggunakan bahasa GO sedangkan implementasi frontend menggunakan bahasa Javascript dengan framework SolidJS.

Bab 4

Implementasi dan pengujian

Bab 4.1 Struktur Data

```
└── backend
    ├── algorithm
    ├── api
    ├── database
    ├── migrations
    ├── models
    ├── queries
    └── util
└── frontend
    └── src
        ├── assets
        └── components
```

Gambar 4.1.1 Bagan struktur data

Bab 4.2 Spesifikasi teknis program

Berikut fungsi dan prosedur pada algoritma yang digunakan :

4.1.1 algorithm.go

Nama	Kegunaan
func New() Algorithm	Constructor untuk algorithm
func LowerAndTrim(s *string)	Membuat lowercase dan menghilangkan spasi sebuah string
func Lower(s *string)	Membuat lowercase string
func Trim(s *string)	Menghilangkan spasi dari string
func TrimFrontBack(s *string)	Menghilangkan spasi di depan dan di belakang string saja
func (alg *Algorithm) HammingDistance(s1, s2 String) float64	Melakukan penghitungan Hamming Distance

4.1.2 bm.go

Nama	Kegunaan
func (a Algorithm) BM(text string, pattern string) int	Mengimplementasikan algoritma BM(Bayer-Moore)
func buildLast(pattern string) []int	Mengembalikan array berisi karakter terakhir dari pattern

4.1.3 calculator.go

Nama	Kegunaan
func (a *Algorithm) SolveMath(expr string) (float64, error)	Melakukan perhitungan matematika
func isOperator(token byte) bool	Memeriksa apakah operator atau bukan
func isNumber(token string) bool	Memeriksa apakah bilangan atau bukan
func preprocessInput(input string) string	Memproses input yang masih dalam notasi infix
func tokenize(input string) ([]string, error)	Memisahkan rangkaian karakter menjadi token-token matematika
func shuntingYard(input string) ([]string, error)	Mengubah notasi infix ke postfix
func evaluatePostfix(postfix []string)	Mengevaluasi notasi postfix

4.1.4 classifier.go

Nama	Kegunaan
func (a Algorithm) Classify(text string) int	Mengklasifikasikan text yang diberikan
func ContainsCandidateMathExp(text string) bool	Mengecek apakah string yang diberikan mengandung kandidat ekspresi matematika
func IsUnaryMathExp(text string) bool	Mengecek apakah string yang diberikan sebuah ekspresi matematika unary
func ContainsDate(text string) bool	Memeriksa apakah text yang diberikan mengandung tanggal
func ContainsQAAddRequest(text string) bool	Mengecek apakah text yang diberikan merupakan permintaan menambahkan

	pertanyaan
func ContainsQADeleteRequest(text string) bool	Mengecek apakah text yang diberikan merupakan permintaan menghapus pertanyaan
func ExtractMathExps(text string) []string	Mengekstrak ekspresi matematika yang diberikan teks

4.1.5 date.go

Nama	Kegunaan
func DateToDay(date string) string	Mengkonversi string tanggal menjadi string hari
func ExtractDates(text string) []string	Mengekstrak tanggal dari string

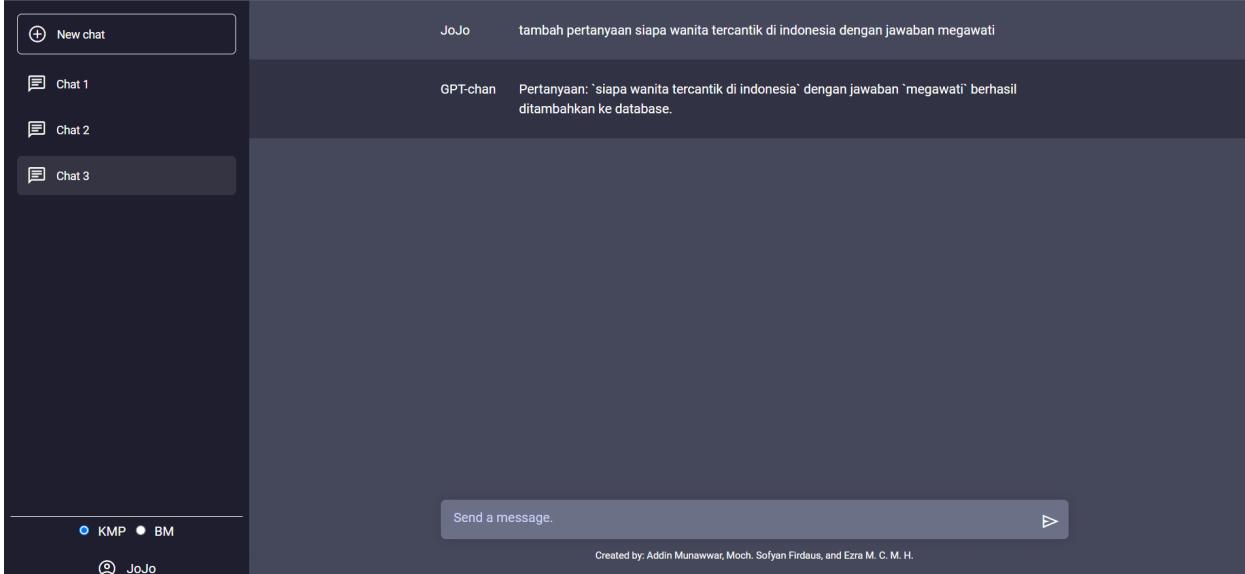
4.1.6

Nama	Kegunaan
func (a Algorithm) KMP(text string, pattern string) int	Mengimplementasikan algoritma Knuth-Morris-Pratt
func computeBorder(pattern string) []int	Fungsi yang membantu KMP mengkomputasikan border array

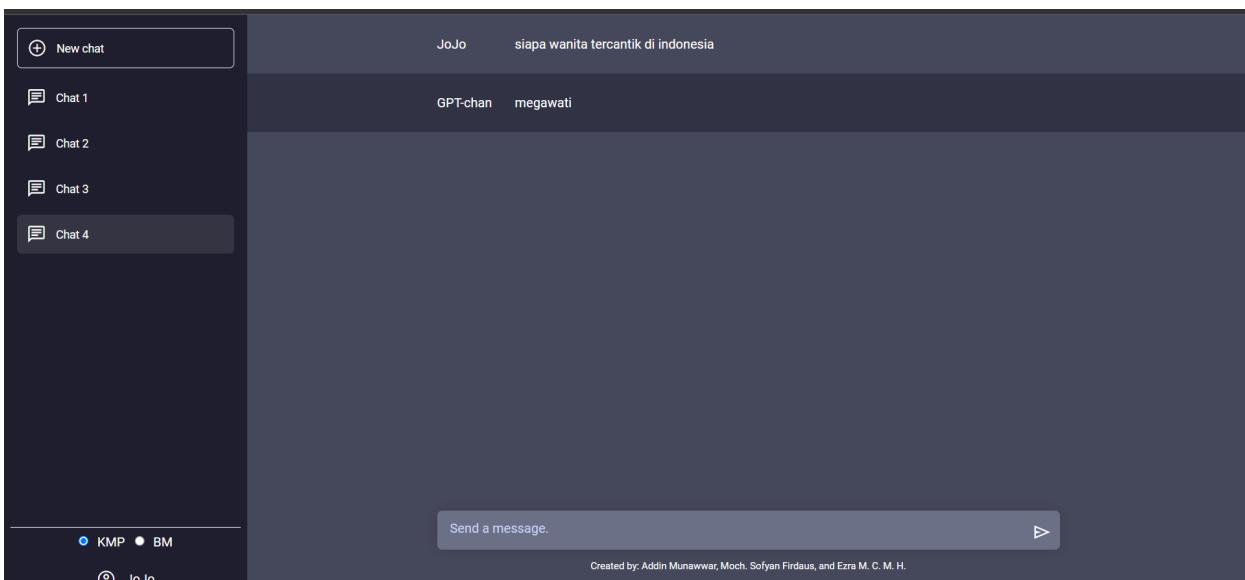
Bab 4.3 Penjelasan tata cara penggunaan program

1. Install semua dependency yang diperlukan
2. Pindah ke directory frontend
3. Ketik npm i pada terminal
4. Ketik npm run dev pada terminal
5. Akan muncul tampilan web layaknya chatgpt, silakan masukkan input yang diinginkan untuk mencoba berbagai fitur yang disediakan

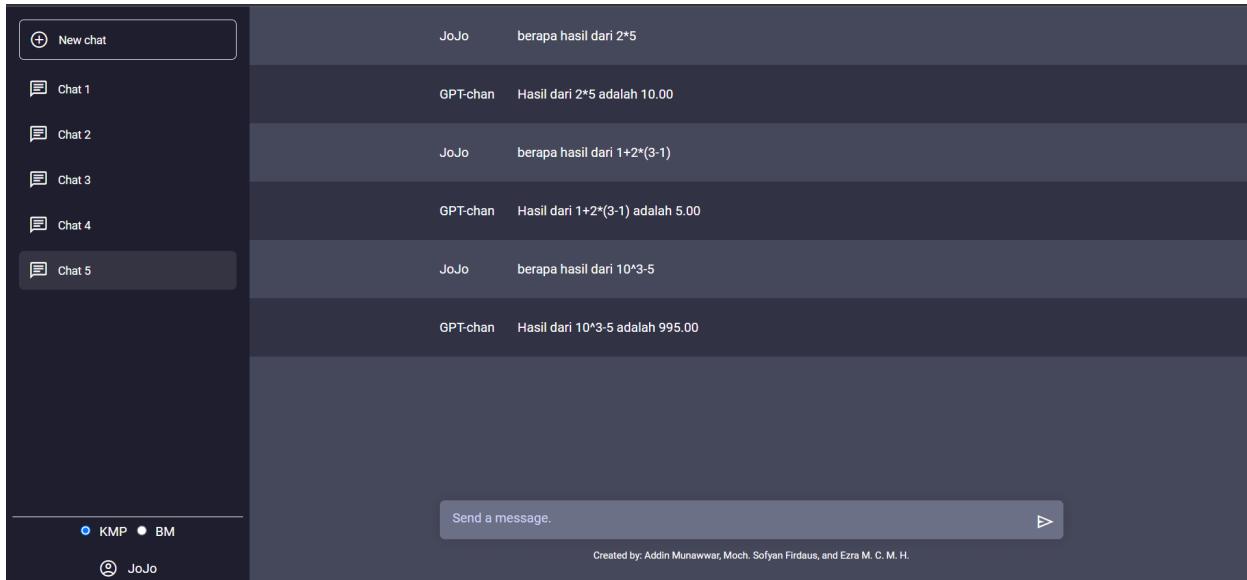
Bab 4.4 Hasil pengujian



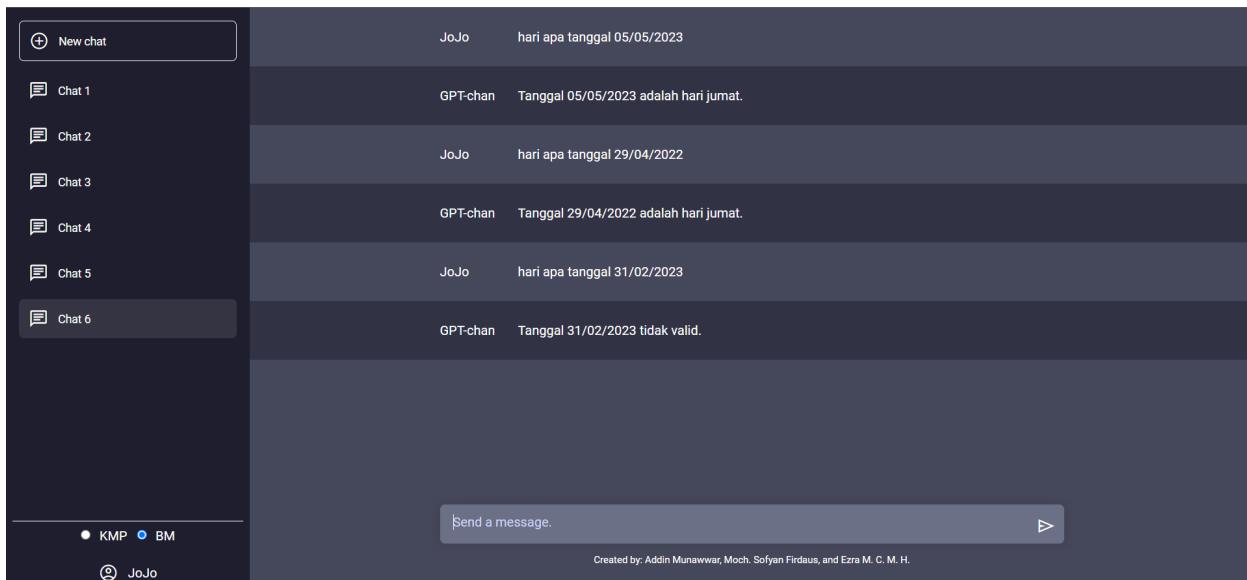
Gambar 4.4.1 menambahkan pertanyaan ke database



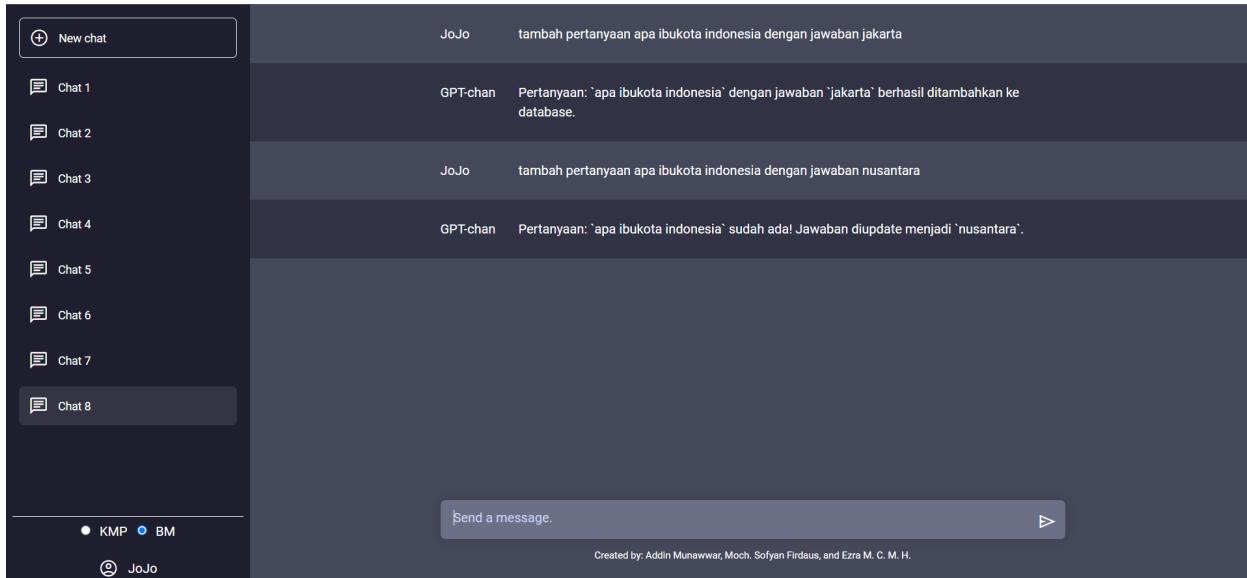
Gambar 4.4.2 menjawab pertanyaan



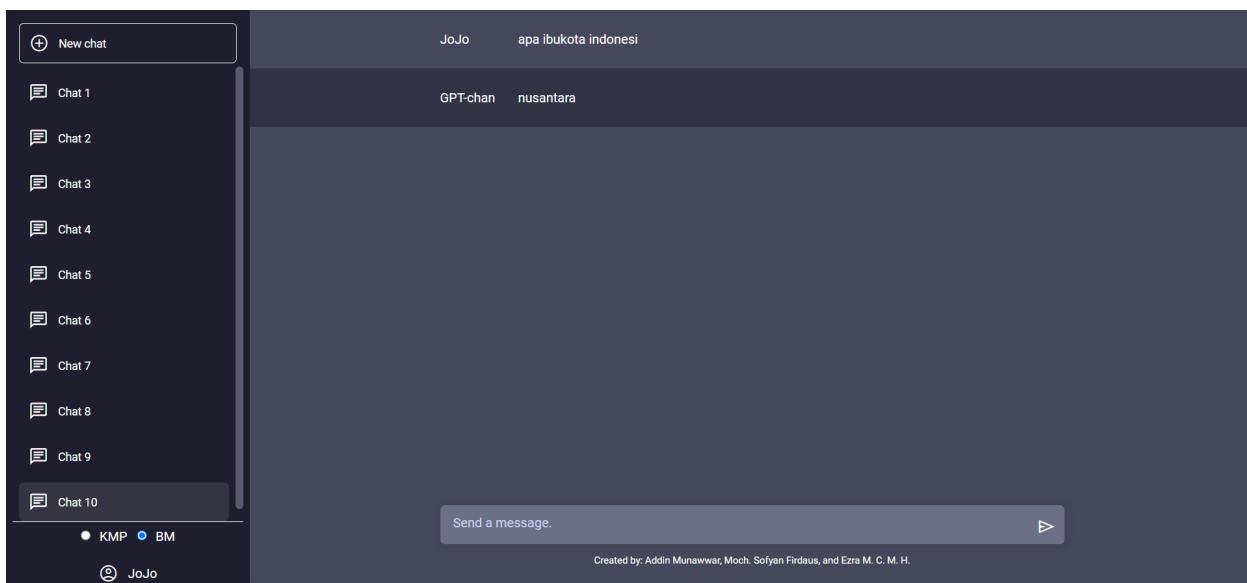
Gambar 4.4.3 Fitur kalkulator



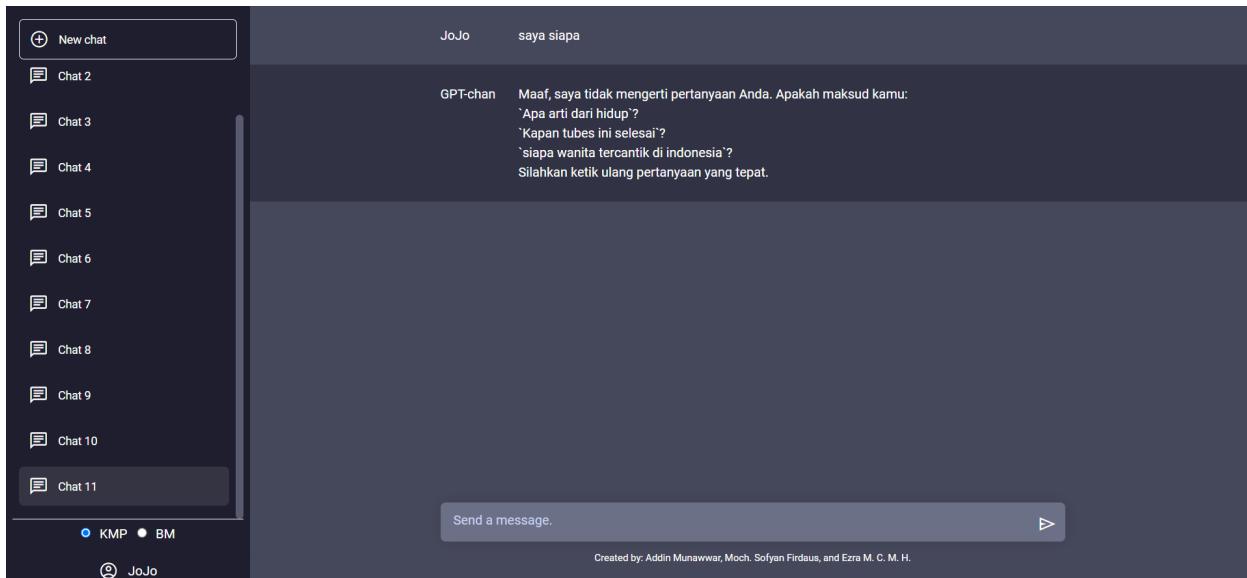
Gambar 4.4.4 Fitur tanggal



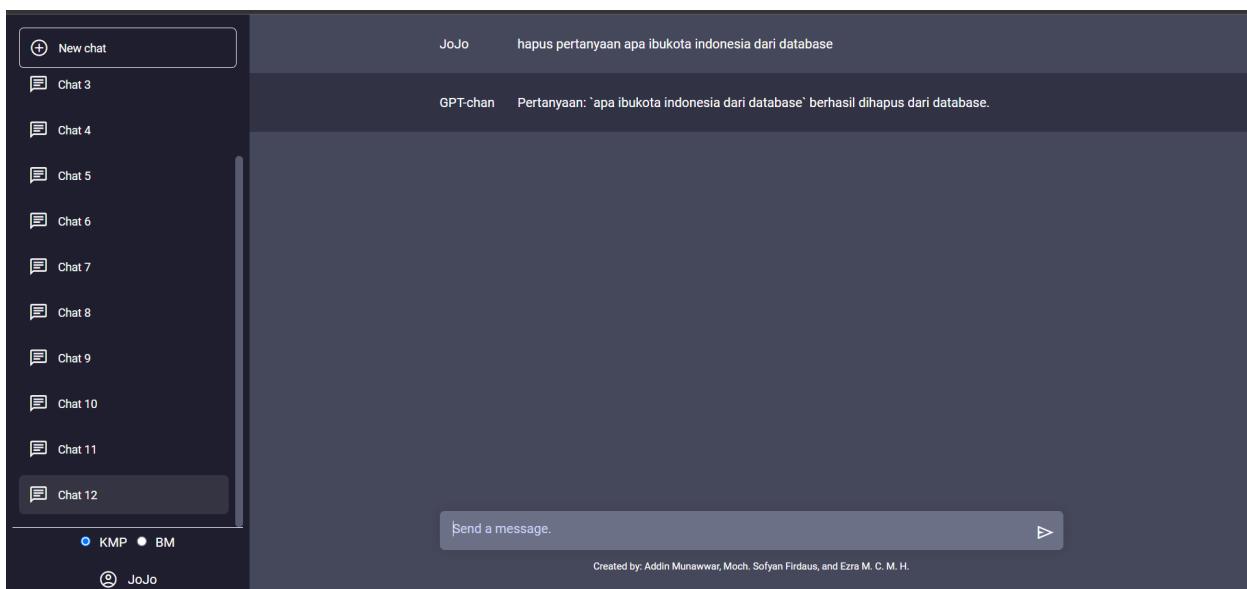
Gambar 4.4.5 update jawaban dari pertanyaan yang telah ada



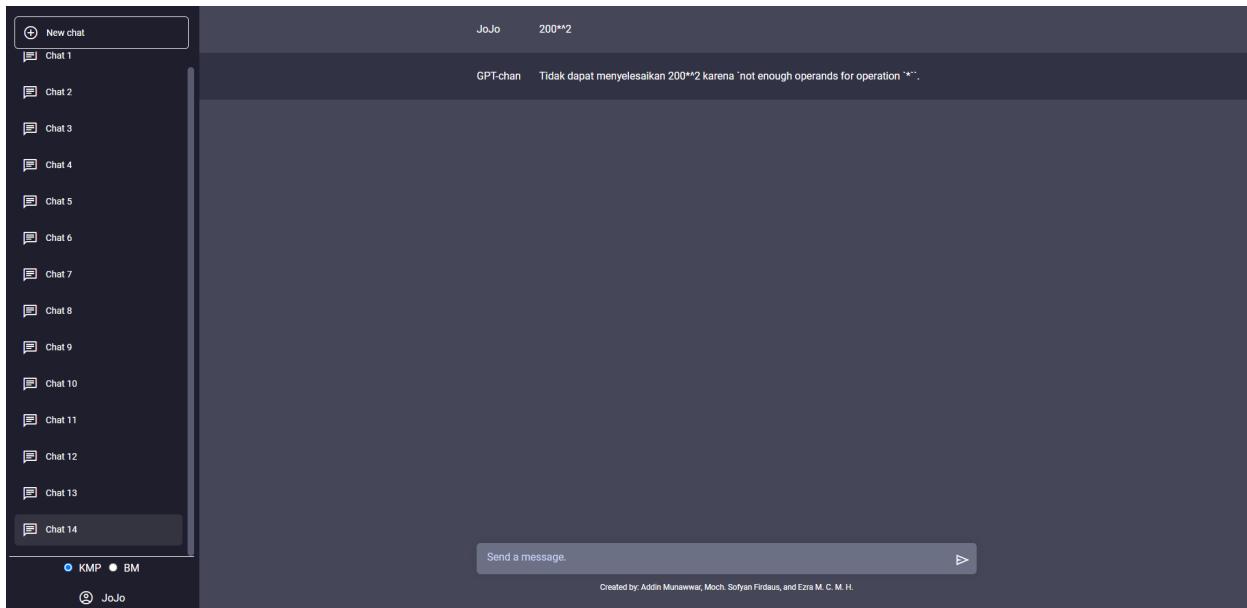
Gambar 4.4.6 Menjawab pertanyaan untuk kasus tidak exact



Gambar 4.4.7 Pertanyaan tidak ada di database



Gambar 4.4.8 Menghapus pertanyaan dari database



Gambar 4.4.9 Fitur kalkulator ketika terdapat kesalahan sintaks

Bab 4.5 Analisis hasil pengujian

Berdasarkan berbagai pengujian yang telah dilakukan pada bagian sebelumnya, dapat disimpulkan bahwa algoritma string matching menggunakan regex, KMP dan BM yang sudah dimplementasikan dapat berjalan dengan baik. Fitur fitur yang disebutkan dalam spesifikasi seperti fitur pertanyaan teks, fitur kalkulator, fitur tanggal, tambah pertanyaan dan jawaban ke database, hapus pertanyaan dari database dapat berjalan dengan baik.

Bab 5

Kesimpulan, saran, dan komentar/refleksi

Bab 5.1 Kesimpulan

Melalui tugas besar 3 mata kuliah IF2211 Strategi Algoritma ini, dapat disimpulkan bahwa implementasi string matching dan regular expression sangatlah luas dan dapat dimanfaatkan untuk berbagai hal, salah satunya yaitu dapat diimplementasikan pada Penerapan String Matching dan Regular Expression dalam pembuatan ChatGPT sederhana.

Bab 5.2 Saran

Untuk tampilan web, dapat dibuat lebih menarik lagi dengan memanfaatkan fitur-fitur lainnya yang telah disediakan oleh SolidJS

Bab 5.3 Komentar/Refleksi

Tugas besar 3 dari mata kuliah IF2211 Strategi Algoritma ini mendorong kami untuk melakukan eksplorasi sendiri untuk mempelajari framework backend dan frontend untuk membangun sebuah website

Link Repository

[moonawar/Tubes3_gpt-chan \(github.com\)](https://github.com/moonawar/Tubes3_gpt-chan)