LAPORAN TUGAS BESAR 3

Penerapan String Matching dan Regular Expression dalam Pembuatan ChatGPT Sederhana

Disusun Oleh:

Kelompok 24

Anggota Kelompok:

Tobias Natalio Sianipar - 13521096

Haidar Hamda– 13521105

Ammar Rasyad Chaeroel - 13521136

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

BANDUNG

MARET 2023

Bab 1
Deskripsi Tugas

Dalam tugas besar 3 ini, dibangun sebuah aplikasi ChatGPT sederhana dengan mengaplikasikan pendekatan QA yang paling sederhana. Pencarian pertanayan yang paling mirip dengan pertanyaan yang diberikan pengguna dilakukan dengan algoritma pencocokan string **Knuth-Morris-Pratt (KMP)** dan **Boyer-Moore (BM)**. Regex (regular expression) digunakan untuk menentukan format dari pertanyaan. Jika tidak ada satupun pertanyaan pada database yang *exact match* dengan pertanyaan pengguna melalui algoritma KMP ataupun BM, maka digunakan pertanyaan termirip dengna kesamaan setidaknya 90%. Apabila tidak ada pertanyaan yang kemiripannya di atas 90%, mkaa chatbot akan memberikan maksimum 3 pilihan pertanyaan yang paling mirip untuk dipilih pengguna.

Fitur-Fitur Aplikasi:

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

1. Fitur pertanyaan teks

   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

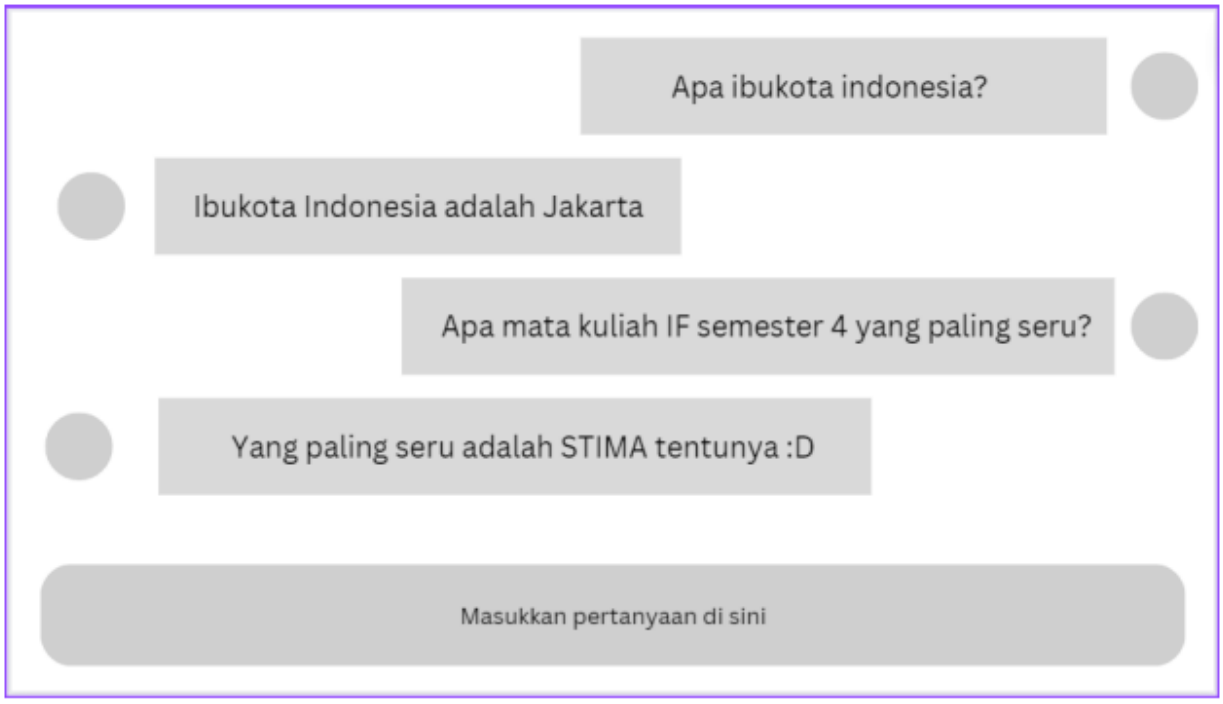
   Pengguna dapat menambahkan pertanyaan dan jawabannya sendiri ke database dengan query contoh “Tambahkan pertanyaan xxx dengan jawaban yyy”. Menggunakan algoritma string matching untuk mencari tahu apakah pertanyaan sudah ada. Apabila sudah, maka jawaban akan diperbaharui.

5. Hapus pertanyaan dari database

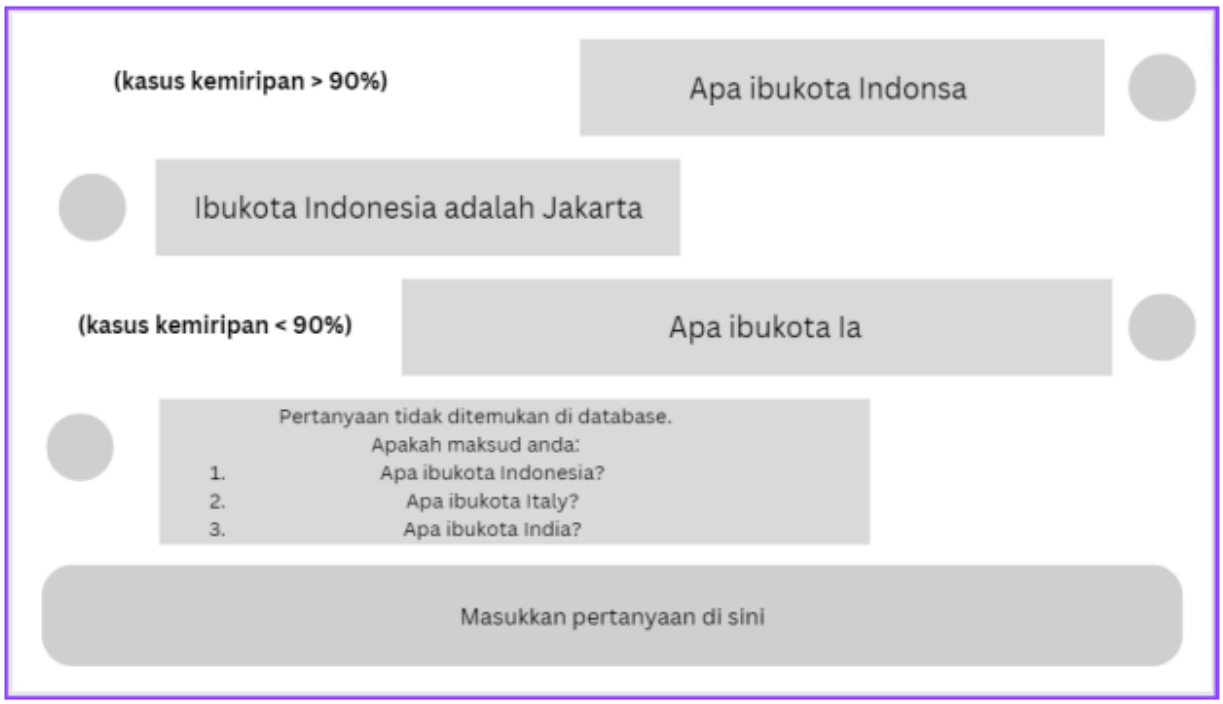
   Pengguna dapat menghapus sebuah pertanyaan dari database dengan query contoh “Hapus pertanyaan xxx”. Menggunakan algoritma string matching untuk mencari pertanyaan xxx tersebut pada database.

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 tidak dikenal, maka tampilkan saja “Pertanyaan tidak dapat diproses”.

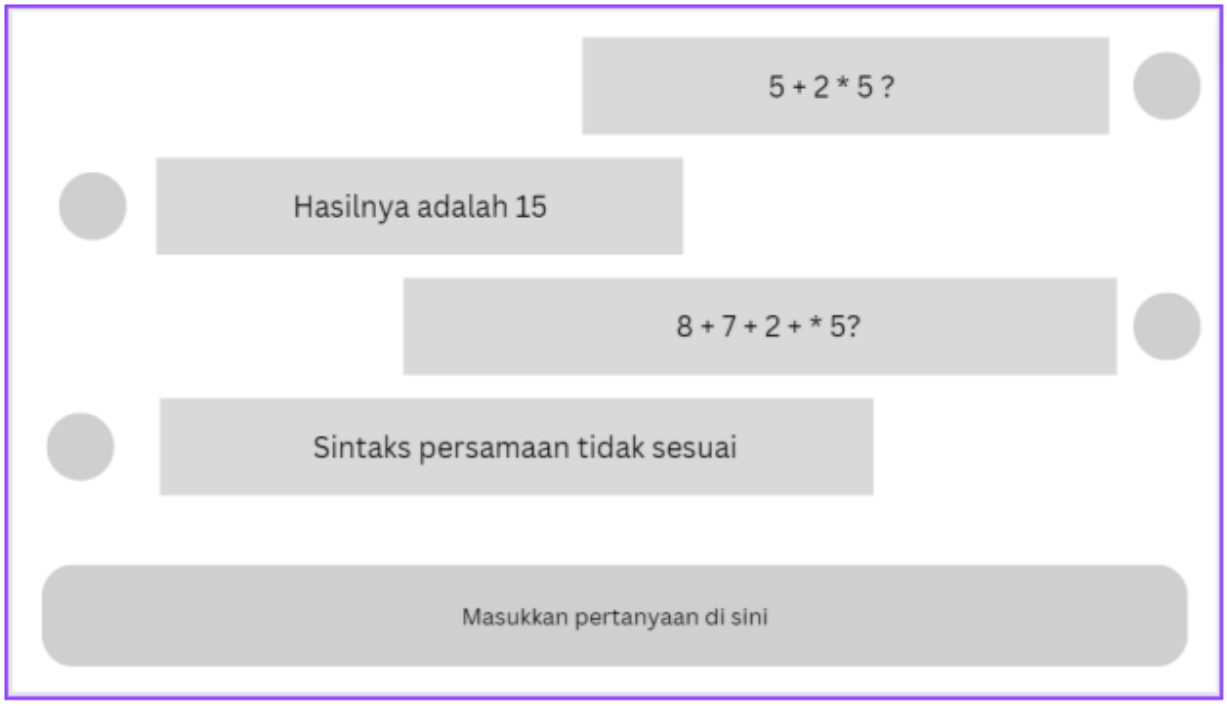
Contoh:



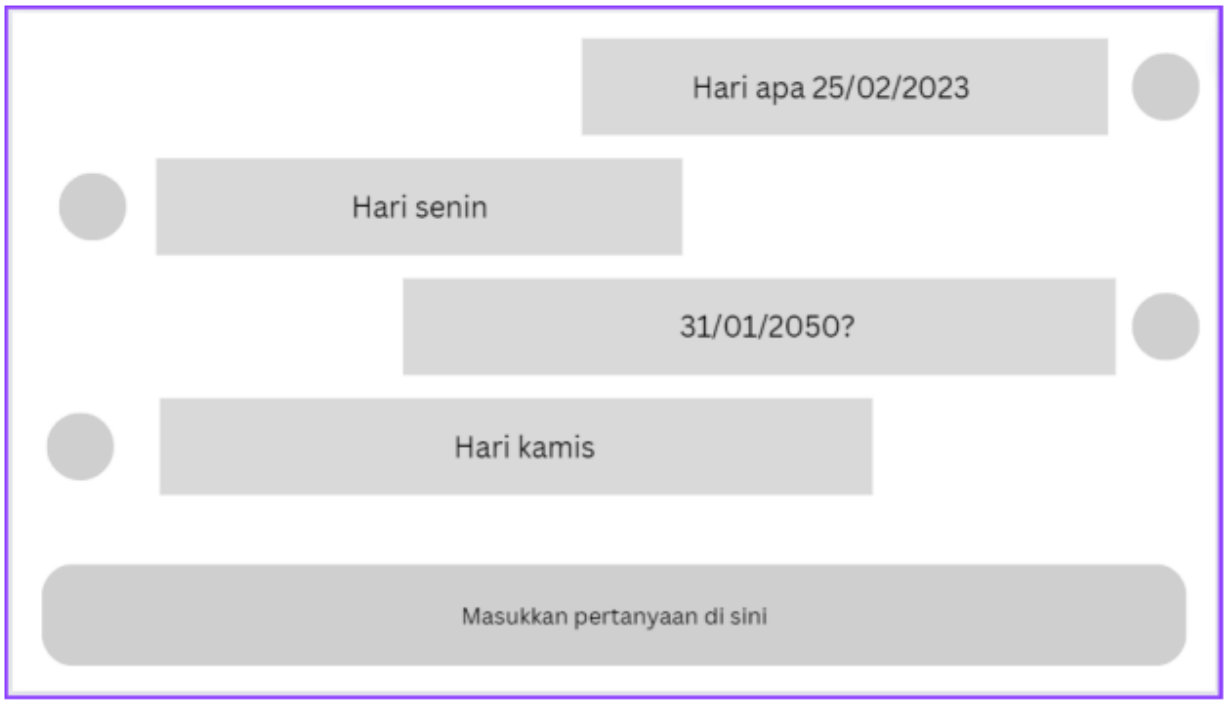
Gambar 2. Ilustrasi Fitur Pertanyaan teks kasus exact



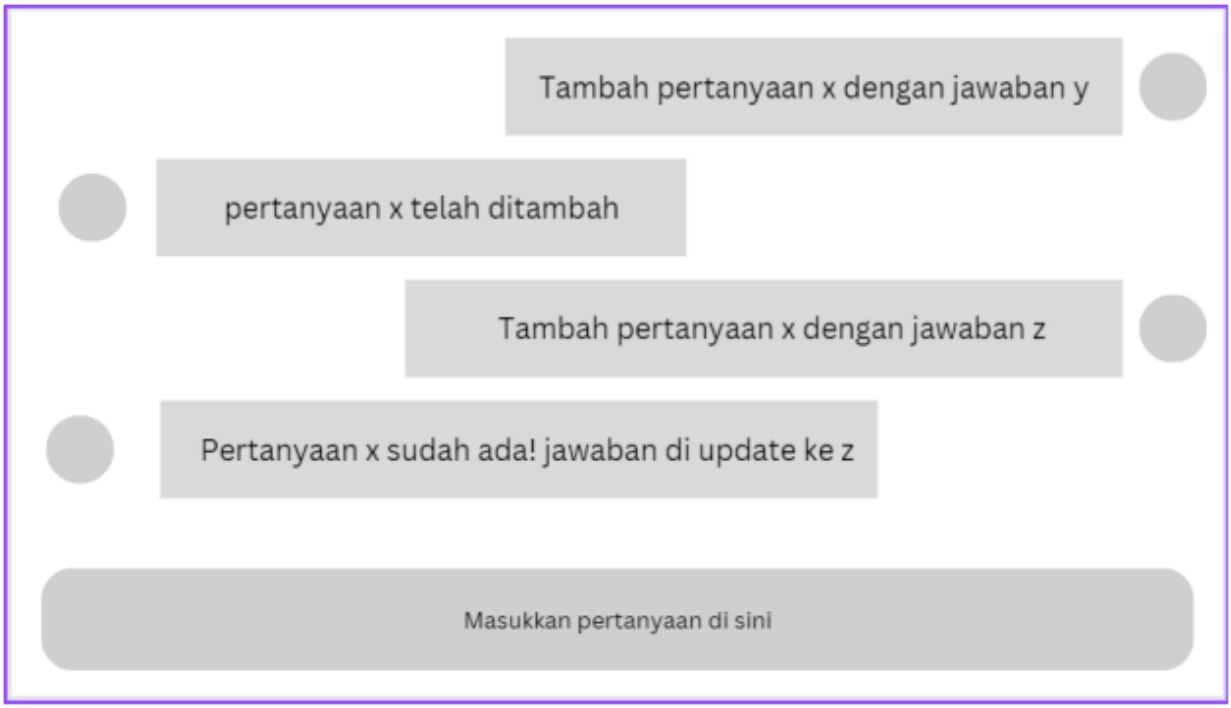
Gambar 3. Ilustrasi Fitur Pertanyaan teks kasus tidak exact



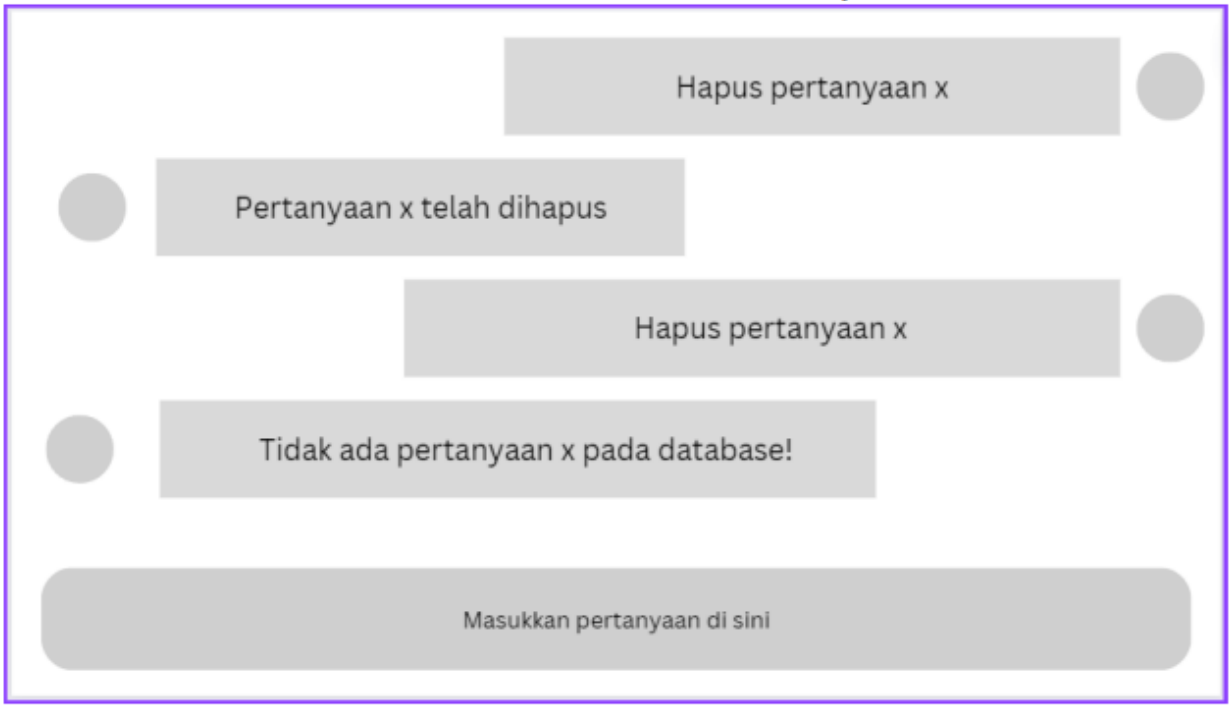
Gambar 4. Ilustrasi Fitur Kalkulator



Gambar 5. Ilustrasi Fitur Tanggal



Gambar 6. Ilustrasi Fitur Tambah Pertanyaan



Gambar 7. 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.

Spesifikasi Program:

1. Aplikasi berbasis website dengan pembagian Frontend dan Backend yang jelas.
2. Implementasi Backend **wajib** menggunakan Node.js / Golang, sedangan Frontend dibebaskan tetapi **disarankan** untuk menggunakan React / Next.js / Vue / Angular.
3. Penyimpanan data **wajib** menggunakan basis data
4. Algoritma pencocokan string dan regex **wajib** diimplementasikan pada sisi Backend.
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 satu kesatuan string pertanyaan utuh (misal “Apa ibukota Filipina?” bukan kata per kata “apa”, “ibukota”, “Filipina”).

Bab 2

Landasan Teori

2.1 Knuth-Morris-Pratt (KMP) algorithm

Knuth-Morris-Prath (KMP) merupakan algoritma extract string matching yang dikembangkan oleh Donald Ervin Knuth, J.H.Morris, dan V.R. Pratt. Algoritma ini melakukan pencocokan dari kiri ke kanan. Jika ditemukan perbedaan antara text dan pattern P pada P[j], sehingga T[i] ≠ P[j], algoritma akan menggeser pattern sebesar panjang dari prefix terbesar P[0..j-1] yang juga menjadi suffix pada P[1..j-1]

Algoritma KMP memiliki kompleksitas waktu O(n+m) di mana n adalah panjang text dan m adalah panjang pattern. Waktu yang dibutuhkan untuk menghitung batas adalah O(m) dan waktu yang dibutuhkan untuk memeriksa kecocokan text dan pattern adalah O(n). Algoritma ini cocok digunakan untuk mencari pattern pada suatu text yang besar.

2.2 Boyer-Moore (BM) algorithm

Boyer-Moore (BM) merupakan algoritma extract string matching yang dikembangkan oleh Robert Boyer dan J Strother Moore. Algoritma ini melakukan pencocokan dari kanan ke kiri. Algoritma ini menggunakan 2 teknik, yaitu:

1. Looking-glass technique

Mencari P di T dengan bergerak mundur melalui P dengan mulai dari akhir

2. Character-jump technique

Ketika ditemukan perbedaan pada T[i]==x, karakter pada pattern P[j] tidak sama dengan T[i]

Algoritma ini melakukan pergeseran berdasarkan 3 kasus:

1. Jika P terdapat x, maka coba untuk geser P ke kanan menyesuaikan posisi kemunculan terakhir x pada P dengan posisi x pada T[i]
2. Jika P terdapat x tetapi tidak bisa menggeser ke kemunculan terakhir x, maka geser P ke kanan 1 karakter ke T[i+1]
3. Jika kasus 1 dan 2 tidak bisa dilakukan, maka geser P ke kanan sampai posisi P[0] sesuai dengan T[i+1]

2.3 Regular Expression (Regex)

Regular expression (Regex) adalah kumpulan karakter yang digunakan untuk membuat pola yang dapat digunakan untuk pencarian dan validasi text. Regex terdiri dari sejumlah karakter yang merepresentasikan pola tertentu seperti huruf, angka, atau tanda baca. Pengguna dapat mengombinasikan karakter dan simbol regex untuk mencali pola yang spesifik pada text.

Penggunaan regex juga memiliki kelemahan, sebagai contoh, karena regex memiliki opsi yang banyak, pengguna memerlukan waktu untuk membuat, menguji, dan memperbaiki pola. Jika regex tidak dibuat dengan baik, dapat menyebabkan berkurangnya efisiensi dalam penggunaan nya.

2.4 Deskripsi Singkat Program

Dalam pembuatan aplikasi ChatGPT Sederhana ini, digunakan algoritma KMP, BM, dan regex untuk menangani pertanyaan pengguna. Pengguna dapat mengirimkan pertanyaan dan sistem mengklasifikasikan pertanyaan dari user dengan regex. Setelah diklasifikasi, program akan mencari pertanyaan pada database dengan algoritma sesuai pilihan pengguna (KMP atau BM). Jika pertanyaan tidak ada di database, program akan menjawab dengan pertanyaan dekat jarak nya berdasarkan algoritma levenshtein distance jika kemiripan pertanyaan user dan database memiliki kemiripan lebih dari 90%. Jika tidak ada pertanyaan yang tingkat kemiripan nya lebih dari 90%, program akan memberikan 3 pertanyaan termirip.

Bab 3

Analisis Pemecahan Masalah

3.1 Langkah-Langkah Pemecahan Masalah

Program akan menerima *prompt* dari user, kemudian akan diproses oleh backend. User dapat memilih untuk menggunakan algoritma KMP atau BM. Sesuai dengan pilihan pengguna, *prompt* akan diproses menggunakan regex. Jika ditemukan pola tertentu, maka KMP atau BM akan digunakan untuk mengecek *prompt* dengan database. Algoritma tersebut ditambah dengan Levenshtein distance akan menentukan seberapa mirip *prompt* dengan pertanyaan di database.

3.2 Fitur dan Arsitektur Aplikasi

Pada *backend*, digunakan bahasa pemrograman Go (vanilla; tanpa framework). Lalu pada *frontend* digunakan bahasa pemrograman JavaScript dengan framework React.

Aplikasi memiliki fitur-fitur seperti menanyakan hari apa pada sebuah tanggal, menambahkan pertanyaan ke *database*, menghitung *arithmetic expression* sederhana, dan mengupdate jawaban dari pertanyaan yang sudah tercatat di *database.*

Aplikasi juga memiliki fitur untuk memilih algoritma apa yang digunakan untuk string matching. User bisa memilih untuk menggunakan KMP atau BM pada tombol yang tersedia di sebelah tombol Send.

Gambar 3.1 – Message box dan Pilihan Algoritma

Bab 4

Implementasi dan Pengujian

4.1 Struktur Data, Fungsi, dan Prosedur

4.1.1 Backend

4.1.1.1 Sql_Connection

File *Sql_Connection* memiliki beberapa function yang menjadi antarmuka program dengan database SQL menggunakan DBMS MariaDB. Terdapat *Create_Connection()*, operasi CRD pada tabel history, dan operasi CRUD pada tabel pertanyaan.

4.1.1.2 Algorithm

File *Algorithm* memiliki function algoritma perhitungan levenshtein distance, *helper functions* untuk *ProcessQuestion(),* function *ProcessQuestion()* itu sendiri.

4.1.1.3 KMP

File KMP berisi fungsi pencocokan string KMP dan border function KMP.

4.1.1.4 BM

File BM berisi fungsi pencocokan string BM dan *helper function* untuk algoritma BM.

4.1.1.5 Calculator

File calculator berisi fungsi kalkulator yang menghitung ekspresi matematika berbentuk infix dan fungsi pembantu.

4.1.1.6 Test

File “Test” berfungsi sebagai jembatan antar *backend* dengan *frontend* agar program bisa berjalan. File ini berisi fungsi-fungsi yang berhubungan dengan HTTP dan hubungan ke *database.*

4.1.2 Frontend

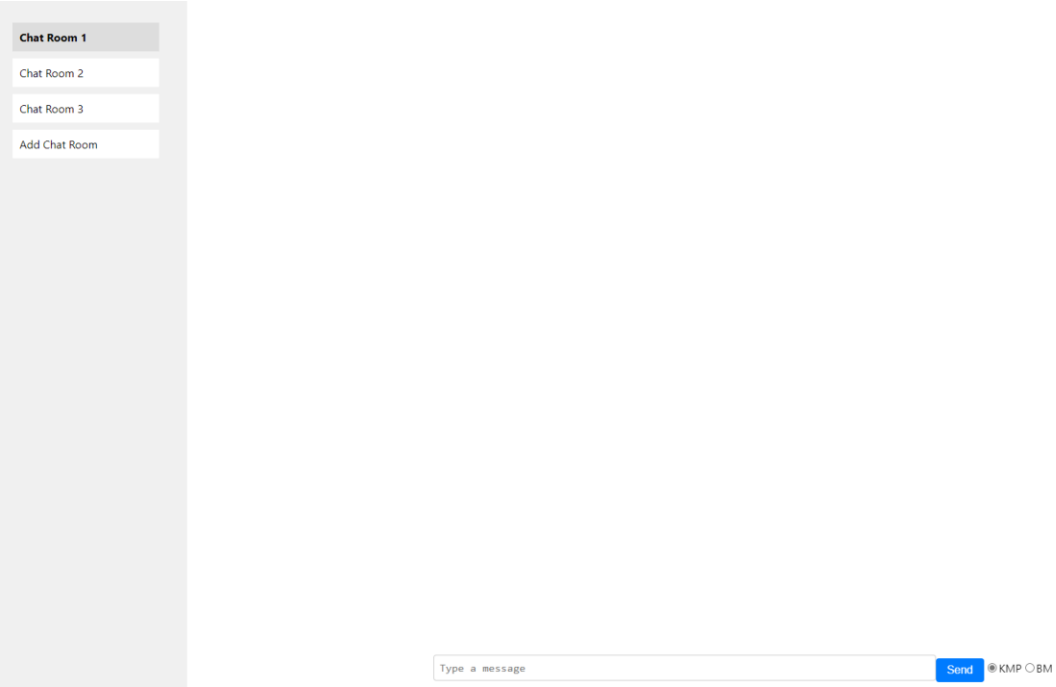
4.1.2.1 App.js

File App.js berfungsi untuk menginisiasi side bar dan main container untuk suatu chat room.

4.1.2.2 ChatRoom.js

File ChatRoom.js berfungsi untuk menentukan id chat room serta melakukan *http request* pada server backend. Berbagai macam request yang dilakukan adalah request untuk mendapatkan jawaban dari database, request untuk menginisiasi chat history pada chat room dengan id 1, dan request untuk menyimpan history seluruh chat room pada 1 table “history” pada database.

4.2 Tata Cara Penggunaan Program



Gambar 4.1 – Interface Program

Pada sebelah kiri *page* terdapat pilihan *chat room* 1 sampai 3, dan user bisa menambahkan *chat room* baru. Lalu di bawah terdapat *text box* untuk input user. User bisa menginput pertanyaan seperti menghitung ekspresi matematika, menanyakan hari apa pada tanggal yang user ingin ketahui, menambahkan atau menghapuskan pertanyaan beserta jawabannya dari database.

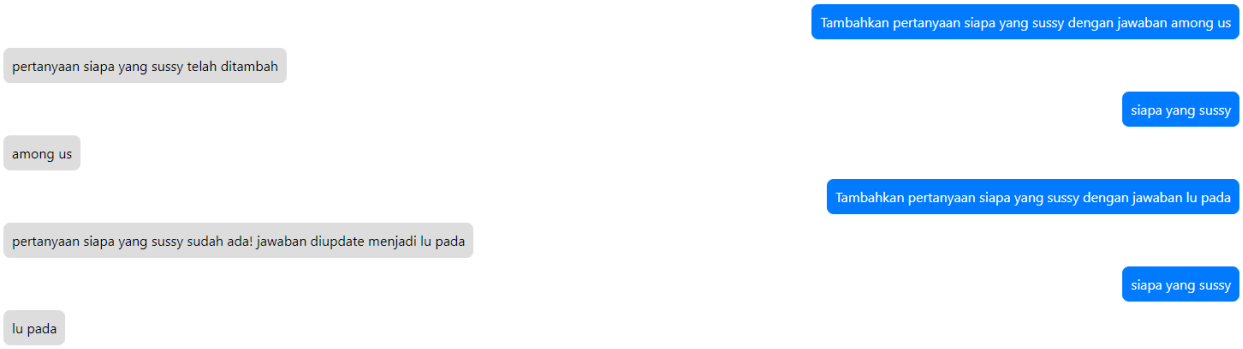
Jika sudah ditambahkan ke database, maka user dapat menanyakan pertanyaan tersebut. Dengan menggunakan algoritma yang terpilih (KMP atau BM), program akan mengcocokan input dari user dengan pertanyaan di database. Jika tidak ada yang 100% cocok, maka akan dipilih jawaban dari database dengan kemiripan di atas 90%. Jika tidak ada yang kemiripannya di atas 90%, maka program akan menampilkan 3 pilihan pertanyaan yang paling mirip ke user.

4.3 Hasil Pengujian

4.3.1 Pengujian 1

4.3.2 Pengujian 2

4.3.3 Pengujian 3



4.3.4 Pengujian 4

4.3.5 Pengujian 5

4.4 Analisis Hasil Pengujian

Pada pengujian 1 pertanyaan hitung [ekspresi matematika] dijawab dengan hasil kalkulasi nya. Setelah hitung, ditanyakan pertanyaan “Matkul wajib terseru smestr 4”, program menjawab langsung dengan “stima” karena pertanyaan yang tersimpan dalam database “Matkul wajib terseru smester 4” yang mana memiliki kemiripan lebih dari 90% dari pertanyaan.

Pada pengujian 2, ketika ditanyakan “matkul wajib terseru”, program menjawab dengan menampilkan 3 pertanyaan termirip karena tidak ada pertanyaan yang tersimpan dalam database yang memiliki kemiripan lebih dari 90% dari yang ditanyakan.

Pada pengujian 3, ditambahkakn pertanyaan “siapa yang sussy” dengan jawaban “among us” setelah itu, jawaban pertanyaan tersebut diubah menjadi “lu pada”.

Pada pengujian 4, ditanyakan hari pada suatu tanggal.

Pada pengujian 5, pertanyaan “siapa yang sussy dihapus”, setelah dihapus, jika ditanyakan kembali “siapa yang sussy” akan menampilkan 3 pertanyaan termirip karena tidak ada pertanyaan dengan kemiripan lebih dari 90%.

Bab 5

Kesimpulan dan Saran

Berdasarkan tugas ini kita dapat menyimpulkan bahwa algoritma **KMP** dan **BM** dapat digunakan untuk membuat sebuah chatbot sederhana. Walaupun tidak sefleksibel ChatGPT (yang menggunakan *Artificial Intelligence*), kedua algoritma tersebut cukup untuk melakukan string matching sederhana.

Saran-saran yang dapat kami berikan adalah:

a. Fitur-fitur dapat diperjelaskan lagi

b. Basis data dapat dibuat lebih spesifik lagi, seperti menyimpan apa saja untuk fitur apa saja

Daftar Pustaka

- Pencocokan string (String matching/pattern matching)
- LEVENSHTEIN ALGORITHM
- Regex cheat sheet

Lampiran

Link Repository: https://github.com/ammarasyad/Tubes3_13521090

Link Youtube: https://youtu.be/GRHbPscX6tE