

**Tugas Besar III IF2211 Strategi Algoritma
Semester II Tahun 2022/2023
Penerapan String Matching dan Regular Expression dalam
Pembuatan ChatGPT Sederhana**



Disusun Oleh

Kelompok 40 botman

13521110 Yanuar Sano Nur Rasyid

13521112 Rayhan Hanif Maulana Pradana

13521173 Dewana Gustavus Haraka Otang

**TEKNIK INFORMATIKA
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG
2023**

BAB I

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.

Spesifikasi Program yang dibuat:

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 II

LANDASAN TEORI

2.1. Persoalan String Matching

Persoalan String Matching adalah persoalan untuk mencari kemunculan sebuah string pattern dalam string text. Persoalan String Matching terbagi menjadi 2 jenis yaitu Exact String Matching dan Approximate String Matching. Exact String Matching akan mencari seluruh indeks pertama kemunculan pattern dalam text apabila terdapat kemunculan, contoh algoritma untuk menyelesaikan persoalan Exact String Matching adalah KMP (algoritma Knuth-Morris-Pratt), dan BM (algoritma Boyer-Moore). Approximate String Matching adalah persoalan untuk menentukan kemunculan pattern dalam text namun dengan sedikit toleransi untuk perbedaan kata, contoh algoritma untuk menyelesaikan Approximate String Matching adalah Levenshtein Edit Distance, dan Hamming Edit Distance.

2.2. Algoritma KMP

Algoritma KMP (Knuth-Morris-Pratt) merupakan algoritma yang umumnya digunakan untuk mencari sebuah substring dari sebuah string/teks yang memiliki pola sama dengan string/teks yang akan diuji. Algoritma KMP menggunakan sebuah *lookup-table* untuk pola string yang diberikan yang diberi nama LPS. Tabel LPS sendiri adalah tabel yang mengandung nilai berupa integer, dimana nilai-nilai integer tersebut dimiliki oleh semua karakter yang ada pada pola string yang diberikan. Nilai-nilai integer tersebut merepresentasikan indeks dari sebuah karakter yang substring-nya matching dengan karakter atau substring yang sedang dibaca. Algoritma KMP yang diimplementasikan tidak menggunakan *back-tracking*.

Akan diasumsikan indeks awal dari sebuah string/teks yang ingin di-match, yaitu target string dan indeks awal dari pola string yang diberikan, yaitu pattern string keduanya bernilai 1. Nilai i merupakan indeks awal dari target string dan nilai j adalah 0. Nilai i akan digunakan untuk mengiterasi target string dan nilai j akan digunakan untuk mengecek pattern string.

Iterasi nilai i hingga akhir target string. Jika karakter i dan j+1 match, maka lanjutkan iterasi. Apabila karakter pada i dan j+1 tidak match/*mismatch*, maka cek dulu apakah j telah bernilai 0. Jika iya, maka lanjutkan iterasi i. Apabila tidak, cek tabel LPS dari karakter j+1 yang bersesuaian. Pindahlah *assign* nilai j dengan integer yang ada pada tabel LPS yang disebutkan dan lanjutkan matching.

2.3. Algoritma BM

Algoritma BM (Boyer-Moore) merupakan algoritma yang umumnya digunakan untuk mencari sebuah substring dari sebuah string/teks yang memiliki pola sama dengan string/teks yang akan diuji. Algoritma BM menggunakan sebuah *lookup-table*, yaitu BMT (Bad Match Table).

BMT dapat direpresentasikan dengan *map* atau *hash* yang mengandung *key* yang berupa sebuah karakter pada string pattern dan *value* yang berupa sebuah integer. Integer untuk setiap karakter dihitung menggunakan rumus :

$$\text{value} = \max(1, \text{panjang pattern} - \text{indeks karakter pada pattern} - 1)$$

Akan ditambahkan juga *key* berupa simbol *asterisk*(*) dengan *value* panjang pattern. Algoritma BM dimulai dengan membandingkan karakter terakhir pada pattern string dengan karakter pada target string yang bersesuaian. Apabila match, lanjutkan perbandingan dengan mundur. Jika ditemukan mismatch, cek *value* BMT dengan *key* yang berupa karakter pada target string yang sedang dibandingkan. Pindah atau *shift* pattern string sejauh *value* yang bersangkutan dan lanjutkan matching.

2.4. Algoritma Levenshtein distance

algoritma Levenshtein adalah algoritma untuk menyelesaikan persoalan Approximate String Matching dengan menggunakan pendekatan Dynamic Programming. Algoritma Levenshtein distance menghitung kemiripan dua buah string dengan cara menghitung berapa langkah minimum untuk mengubah string yang satu menjadi string lainnya apabila disediakan kumpulan operasi insert, delete, dan substitution. Algoritma Levenshtein distance akan menghitung operasi minimum dengan memakai algoritma rekursif dengan algoritma sebagai berikut,

$$dp[a][b] = dp[a-1][b-1], \text{ jika } text[a] = pattern[b]$$

$$1 + \min(dp[a-1][b], dp[a][b-1], dp[a-1][b-1]), \text{ jika } text[a] \neq pattern[b]$$

a, jika b = 0
b, jika a = 0

Operasi minimum yang dibutuhkan dapat dihitung dengan menghitung nilai $dp[|string|][|pattern|]$. Persentase kemiripan dua buah string dapat dihitung dengan cara $(1 - Levenshtein_distance(pattern, text) / \max(|pattern|, |text|)) * 100 \%$.

2.5. Regex

Regex (Regular Expression) adalah kumpulan karakter yang dapat menjelaskan pola untuk mencocokkan sebuah string. Regex biasa digunakan untuk melakukan String Matching dengan cara memutuskan apabila memenuhi pola yang diberikan, sehingga string yang dianggap cocok bisa lebih dari satu string.

Beberapa contoh regex

- “`^\d+$`” akan mencocokkan dengan string yang hanya terdiri dari angka.
- “`^[a-zA-Z]+$`” akan mencocokkan dengan string yang hanya terdiri dari huruf kecil atau huruf besar.
- “`^\w+@[a-zA-Z_]+?\.[a-zA-Z]{2,3}$`” akan mencocokkan dengan string yang berbentuk format email.

2.6. Pembuatan aplikasi web

Aplikasi web adalah program yang dijalankan di dalam web browser. Aplikasi web dapat berupa halaman web statis atau dinamis yang hanya menampilkan informasi tertentu tergantung dari pembuat web. Aplikasi web biasanya terdiri dari dua bagian utama, yaitu frontend dan backend. Frontend adalah bagian antarmuka pengguna seperti tombol, formulir, dan lainnya. Sementara itu, backend adalah bagian yang berjalan di server, dan bertanggung jawab untuk memproses permintaan dari frontend dan mengirimkan data kembali ke browser.

Salah satu bahasa pemrograman populer yang digunakan untuk membuat *backend* adalah Golang. Golang merupakan *statically-type high level programming language* yang

didesain Google. Salah satu *library* yang tersedia untuk membuat *framework* web pada golang adalah Echo. Echo merupakan *framework* cepat dan minimalis untuk pembuatan API dan aplikasi web. Echo didesain untuk memudahkan proses pembuatan aplikasi web dengan menyediakan fitur seperti *routing*, *middleware*, dan *serving static file*.

Salah satu bahasa pemrograman populer yang digunakan untuk membuat *frontend* adalah Javascript. Javascript merupakan *weakly-type high level programming language* yang sering digunakan sebagai scripting language untuk mengatur alur keberjalanan program web sehingga menjadi lebih dinamis. Beberapa library yang juga sering dipakai untuk membantu pengembangan aplikasi web dengan javascript adalah *Typescript* dan *React*. Typescript merupakan bahasa pemrograman yang merupakan *superset* dari javascript dengan menghadirkan konsep *strongly-type annotation* sehingga memudahkan programmer ketika mengembangkan aplikasi. React merupakan library javascript yang digunakan untuk merancang *User Interface* pada aplikasi web dengan menggunakan konsep *component-based* yang memisahkan tampilan dengan logika dari suatu aplikasi.

BAB III

ANALISIS PEMECAHAN MASALAH

3.1. Langkah penyelesaian masalah Regex

1. Program akan menerima input chat message dari user
2. Program akan menjalankan pola Regex dan menentukan pertanyaan tipe apakah chat yang diberikan user
3. Apabila chat yang diberikan merupakan operasi matematis atau date, maka program akan langsung menampilkan hasil perhitungan operasi matematis atau menampilkan hari apa tanggal yang diberikan
4. Apabila chat yang diberikan merupakan perintah untuk menghapus atau menambah data, maka program akan melanjutkan chat user kepada pengelola database untuk nantinya menjalankan program untuk mengolah database
5. Apabila chat yang diberikan merupakan pertanyaan, maka program akan melanjutkan data chat kepada algoritma Exact String Matching untuk mencari jawaban pertanyaan di dalam database

3.2. Langkah penyelesaian masalah Exact String Matching

1. Ketika fungsi Exact String Matching dijalankan, fungsi akan menerima input chat user, data pertanyaan dan jawaban dari database, dan algoritma Exact String Matching yang ingin dipakai user
2. Untuk setiap pertanyaan yang berada dalam database, cek apakah ada pertanyaan yang exact match dengan menggunakan algoritma KMP ataupun BM.
3. Apabila hanya terdapat satu pertanyaan pada database yang exact match dengan input chat user, maka return pertanyaan tersebut.
4. Apabila ada lebih dari satu pertanyaan pada database yang exact match dengan input chat user, maka bandingkan pertanyaan-pertanyaan tersebut berdasarkan nilai kemiripan terhadap input chat user dengan menggunakan Algoritma Levenshtein Distance.
5. Return sebuah pertanyaan dengan nilai kemiripan tertinggi.

3.3. Langkah penyelesaian masalah Approximate String Matching

1. Apabila pada penyelesaian Exact String Matching tidak ditemukan hasil, maka program akan menyelesaikan masalah dengan pendekatan Approximate String Matching
2. Untuk setiap pertanyaan yang berada dalam database hitung berapa persen kemiripan pertanyaan dengan chat user, perhitungan dilakukan dengan algoritma Levenshtein distance
3. Apabila terdapat pertanyaan dalam database yang memiliki kemiripan lebih dari 90%, maka pertanyaan tersebut akan dianggap pertanyaan yang ingin user tanyakan, dan pertanyaan yang memiliki kemiripan tertinggi akan ditampilkan jawabannya kepada user

4. Apabila tidak terdapat pertanyaan dalam database yang memiliki kemiripan lebih dari 90%, maka fungsi akan menampilkan 3 buah pertanyaan dengan kemiripan tertinggi untuk menjadi saran kepada user agar bertanya kembali dengan pertanyaan yang lebih tepat

3.4. Langkah penyelesaian masalah fitur fungsional dan arsitektur aplikasi web yang dibangun

Pembuatan aplikasi web menggunakan *design pattern* MVC atau Model-View-Controller. Desain ini membagi komponen-komponen aplikasi menjadi tiga bagian yaitu Model sebagai representasi data dan *logic* dari aplikasi seperti memverifikasi data yang digunakan. Kemudian, View yang berfungsi sebagai *interface* dengan user atau pengguna sehingga user dapat berinteraksi dengan aplikasi, contohnya adalah tombol dan input teks. Terakhir, Controller yaitu komponen yang berfungsi sebagai penghubung antara View dengan Model sehingga interaksi pengguna dengan View akan merubah Model yang ada pada aplikasi. Detail lebih lanjut dari desain adalah seperti berikut.

1. Backend

- a. Model

Model dibuat pada *packages* models juga controller. *Packages* models berisi data *struct* yang mengatur bentuk data yang digunakan sesuai dengan tipenya masing-masing seperti query model yang berisi data pertanyaan dan jawaban juga chatlog model yang berisi data mengenai riwayat dari percakapan.

Model tersebut akan dipakai untuk mengakses *database* sesuai dengan model masing-masing. Untuk pengaksesannya, dibuat API untuk melakukan operasi CRUD pada basis data sesuai dengan models-nya pada *packages* controller.

- b. Controller

Penghubung antara Model dengan View dibuat pada *packages* controller. Pada file *frontend_controller.go* terdapat fungsi-fungsi yang menerima *request* dari View yang kemudian akan diproses dan dilanjutkan dengan pemanggilan API model yang sesuai.

2. Frontend

- a. View

View dibuat pada folder Chat dan akan menampilkan data yang diambil dari database kepada user. Model yang telah didefinisikan pada *packages* models ditulis kembali dalam bahasa typescript pada sub-folder *BackendInterface*. Data yang diambil dari database didapat dengan menggunakan controller untuk melakukan request data seluruh Chat Log dan seluruh isi chatnya. kemudian frontend menyediakan view kepada user untuk melakukan beberapa aksi seperti merename Chat Log, menghapus Chat Log, menambah Chat Log, dan juga meminta jawaban input dari pertanyaan user yang nantinya akan diteruskan oleh View kepada Controller untuk mengolah data yang diberikan.

BAB IV IMPLEMENTASI DAN PENGUJIAN

4.1. Penjelasan struktur data yang dipakai

| Struktur Data | Pemakaian |
|---------------|---|
| Stack | Struktur data stack digunakan pada fungsi Calculator yang kegunaannya adalah untuk mengetahui urutan perhitungan untuk operasi yang mengandung karakter tanda kurung. Stack juga dipakai untuk mengetahui nilai yang telah dihitung oleh perhitungan aritmatika sejauh ini. |
| Map | Struktur data map digunakan untuk menampung pasangan data pertanyaan dan jawaban yang diberikan query sehingga apabila ingin mengetahui jawaban dari pertanyaan x dapat dilakukan dengan mudah dengan cara melihat nilai pada indeks x pada map. |
| List | Struktur data list digunakan untuk menampung beberapa string pertanyaan dalam database yang bisa terbilang mirip dengan pertanyaan yang ditanyakan user, dan nantinya list akan diurutkan berdasarkan tingkat kemiripan yang tertinggi hingga terkecil. |

4.2. Penjelasan fungsi yang dibuat

File stringMatching.go

| Fungsi | Penjelasan |
|--|---|
| <pre>func generateLPS(pattern string) []int</pre> | Fungsi generate LPS akan menerima string berupa sebuah pattern yang akan dibuat P-table/LPS-nya. LPS direpresentasikan dengan sebuah list/slice integer yang menjadi return-value. |
| <pre>func KmpExact(pattern string, text string) bool</pre> | Akan mengecek apakah kedua string, yaitu pattern yang berupa input dari user dan text yang merupakan sebuah teks dari database, benar-benar sama persis (exact) dengan Algoritma KMP. |
| <pre>func Kmp(pattern string, text string) []int</pre> | Akan mengecek apakah terdapat sebuah substring dari text yang sama dengan pattern dengan Algoritma KMP. Akan mengembalikan sebuah list/slice of integer yang merupakan indeks-indeks dari |

| | |
|--|---|
| | substring yang matching. |
| <pre>func generateBMT(pattern string) map[byte]int</pre> | Fungsi ini akan menerima string berupa sebuah pattern yang akan dibuat BMT (Bad Match Table)-nya. BMT menjadi return-value yang berupa sebuah map dengan byte (sebuah karakter) dengan value integer yang dihitung dengan Algoritma. |
| <pre>func bmMatchPattern(pattern string, text string, k *int) bool</pre> | Akan mengecek secara mundur string dari pattern dan string dari text. Memiliki return-value boolean yang merepresentasikan apakah pattern dengan sebuah substring dari text matching. Nilai k merupakan pointer ke sebuah nilai yang ada di fungsi Algoritma Boyer-Moore utama. Nilai tersebut adalah indeks dari text yang berbanding dengan indeks terakhir dari pattern. |
| <pre>func BoyerMooreExact(pattern string, text string) bool</pre> | Akan mengecek apakah kedua string, yaitu pattern yang berupa input dari user dan text yang merupakan sebuah teks dari database, benar-benar sama persis (exact) dengan Algoritma Boyer-Moore. |
| <pre>func BoyerMoore(pattern string, text string) []int</pre> | Akan mengecek apakah terdapat sebuah substring dari text yang sama dengan pattern dengan Algoritma Boyer-Moore. Akan mengembalikan sebuah list/slice of integer yang merupakan indeks-indeks dari substring yang matching. |
| <pre>func generateMatchedIdx(pattern string, text string, idx int) []int</pre> | Akan membuat sebuah list/slice of integer yang menyimpan indeks dari substring text yang match dengan pattern. Parameter indeks merupakan indeks yang menunjuk ke karakter terakhir dari substring pada text yang match dengan pattern. |
| <pre>func min(nums ...int) int</pre> | Mengecek sebuah integer dengan nilai minimum dari sekelompok (list/slice) integer. |
| <pre>func levenstheinDistance(pattern string, text string) int</pre> | Mengecek berapa jumlah dari tiga operasi (removal, insertion, substitution) yang dibutuhkan agar string pattern sama dengan string text. Menggunakan algoritma Levenshtein Distance. |
| <pre>func CalculateSimilarity(pattern string, text string) float64</pre> | Menghitung kemiripan antara pattern dan text dengan referensi berapa jumlah dari tiga operasi yang dibutuhkan pada Algoritma Levenshtein Distance. Kemiripan berada di antara 0-100. |

File day.go

| Fungsi | Penjelasan |
|--|---|
| <pre>func parseDate(date string) [3]int</pre> | Menerima input sebuah string yang merupakan tanggal valid dalam format DD/MM/YYYY dan nantinya akan mengembalikan array berisi 3 integer yang nilainya adalah nilai hari, bulan, tahun dalam bentuk integer. |
| <pre>func toJulian(date string) int</pre> | Menghitung sebuah date ke integer berupa Julian Day Number, yaitu sebuah integer yang ditetapkan untuk “whole solar day” dalam kalender/hari Julian yang dimulai dari noon Universal Time. |
| <pre>func createDays() map[int]string</pre> | Akan membuat dan mengembalikan array berisi 7 integer yang isinya merupakan nama hari dalam bahasa indonesia dimulai dari hari senin. |
| <pre>func createReverseDays() map[int]string</pre> | Akan membuat dan mengembalikan array berisi 7 integer yang isinya merupakan nama hari dalam bahasa indonesia dengan urutan terbalik dimana indeks pertama berisi hari senin dan indeks selanjutnya merupakan hari minggu sampai Selasa. |
| <pre>func CalculateDay(date string) string</pre> | Menerima input string tanggal dalam format DD/MM/YYYY dan akan mengembalikan sebuah string yang menjelaskan nama hari pada tanggal tersebut. |

File stack.go

| Fungsi | Penjelasan |
|---|---|
| <pre>func (s *Stack) Pop() (value interface{})</pre> | Akan mengembalikan nilai elemen teratas pada stack sekaligus menghapus elemen teratas pada stack. |
| <pre>func (s *Stack) Peek() (value interface{})</pre> | Akan mengembalikan nilai elemen teratas pada stack tanpa menghapusnya, |
| <pre>func (s *Stack) Len() int</pre> | Akan mengembalikan nilai yang merupakan panjang dari stack. |
| <pre>func (s *Stack) IsEmpty() bool</pre> | Akan mengembalikan nilai true apabila stack kosong, dan mengembalikan nilai false apabila stack tidak kosong. |

File calculator.go

| Fungsi | Penjelasan |
|--|--|
| <code>func precedence(operator byte) int</code> | Menerima input sebuah karakter operator dan akan menentukan nilai precedence operator tersebut. Nilai-nilai precedence adalah 1 untuk '+' dan '-', 2 untuk '*' dan '/', 3 untuk '^', dan -1 apabila bukan merupakan operator yang valid. |
| <code>func isOperator(char byte) bool</code> | Menerima input sebuah karakter dan akan menentukan apakah karakter tersebut adalah karakter operator valid, karakter operator valid terdiri atas '+', '-', '*', '/', '^'. |
| <code>func isOperatorStr(str string) bool</code> | Menerima input sebuah string dan akan menentukan apakah karakter tersebut adalah string operator yang valid, string operator yang valid terdiri atas "+", "-", "*", "/", "^". |
| <code>func isBeginEndOperator(infix string) bool</code> | Mengecek apakah sebuah ekspresi memiliki operator pada awal dan akhir ekspresi infix. |
| <code>func infixToPostfix(infix string) []string</code> | Mengubah ekspresi infix ke ekspresi postfix, dengan postfix sebagai return-value yang direpresentasikan dengan list/slice of string. |
| <code>func operate(op1 float64, op2 float64, operator string) float64</code> | Mengoperasikan operand op1 dengan operand op2 dengan menggunakan operator. Return-value berupa floating point number. |
| <code>func Calculator(expression string) string</code> | Menghitung hasil persamaan postfix. Hasil di return sebagai string agar dapat digunakan dengan mudah dalam implementasi chatbot. |

File regex.go

| Fungsi | Penjelasan |
|---|---|
| <code>func WhichFeature(question string) int</code> | Menerima input sebuah string yang merupakan chat dari user dan akan menentukan masuk kedalam pertanyaan tipe berapakah input chat yang diberikan user. |
| <code>func GetQueries(questions string) []string</code> | Menerima string questions yang merupakan pertanyaan dari user kemudian memeriksa apakah ada kata pemisah sebagai penanda ada lebih dari satu pertanyaan. Jika ada akan dikembalikan list pertanyaan user yang sudah terpisah. |

| | |
|--|--|
| <pre>func ExtractExpressionFour(question string) [2]string</pre> | <p>Mengekstrak sebuah substring dari string question dengan memanfaatkan regex dan trim. Digunakan khusus untuk fitur keempat dan list/slice of string yang merupakan return-value adalah pertanyaan dan jawaban yang diekstrak.</p> |
| <pre>func ExtractExpression(question string) string</pre> | <p>Mengekstrak sebuah substring dari string question dengan memanfaatkan regex dan trim. Digunakan selain untuk fitur keempat dan list/slice of string yang merupakan return-value adalah ekspresi yang diekstrak (digunakan untuk implementasi logic chatbot)</p> |

File chatLogic.go

| Fungsi | Penjelasan |
|---|--|
| <pre>func checkExactPattern(pattern string, text string, stringMatchingAlgo string) bool</pre> | <p>Merupakan controller yang akan menentukan fungsi apakah yang perlu dipanggil dari input dan konfigurasi algoritma yang diberikan user.</p> |
| <pre>func stringMatchingLogic(pattern string, data map[string]string, stringMatchingAlgo string) ([]string, bool)</pre> | <p>Merupakan fungsi yang akan melakukan pattern/string matching dengan semua algoritma yang telah dibuat, dengan urutan prioritas exact matching (KMP dan BM) kemudian matching dengan kemiripan (Levenshtein Distance). Menerima sebuah pattern dari input user dan map berupa semua pertanyaan yang ada di database dengan value jawaban dari pertanyaan yang bersangkutan. Parameter stringMatchingAlgo merupakan jeni Algoritma yang ingin digunakan.</p> |
| <pre>func GetAnswer(questions string, data map[string]string, stringMatchAlgo string) string</pre> | <p>Mengiterasi terhadap semua pertanyaan yang dimasukkan oleh user dan memiliki return-value berupa string jawaban-jawaban dari pertanyaan yang bersangkutan.</p> |
| <pre>func ChatLogic(question string, data map[string]string, stringMatchingAlgo string) string</pre> | <p>Merupakan implementasi dari logika yang digunakan oleh chatbot. Akan mengembalikan nilai berupa string yang merupakan jawaban/message yang diberikan kepada pengguna. Menggunakan parameter question yang merupakan input dari user, data yang merupakan sebuah map dengan key berupa pertanyaan yang ada di database dengan value jawaban dari pertanyaan yang bersangkutan. Parameter stringMatchingAlgo merupakan jeni Algoritma yang ingin digunakan.</p> |

4.3. Penjelasan prosedur yang dibuat

File stack.go

| Prosedur | Penjelasan |
|--|--|
| <pre>func (s *Stack) Push(value interface{})</pre> | Akan menambahkan elemen yang diberikan ke dalam stack. |

File calculator.go

| Prosedur | Penjelasan |
|---|---|
| <pre>func writeToPostfix(postfix *[]string, currentString *strings.Builder)</pre> | Menuliskan sebuah string kepada ekspresi postfix. |

File chatLogic.go

| Prosedur | Penjelasan |
|---|--|
| <pre>func sortSimilarity(result *[]string, resultSimilarity *[]int)</pre> | Akan melakukan pengurutan terhadap pasangan array string pertanyaan dalam database dan array integer similarity dengan pertanyaan yang diberikan user, dan akan diurutkan berdasarkan nilai kemiripan dari yang terbesar sampai yang terkecil. |
| <pre>func addQuestion(add models.Query)</pre> | Prosedur yang digunakan sebagai interface untuk menambahkan sebuah pasangan pertanyaan dan jawaban baru atau meng-update jawaban apabila pertanyaan sudah ada ke dalam database. |
| <pre>func deleteQuestion(question models.Query)</pre> | Prosedur yang digunakan sebagai interface untuk menghapus sebuah pertanyaan dari database. |

4.4. Cara menggunakan program

Untuk menjalankan program, lakukan instalasi program *prerequisite* terlebih dahulu yaitu file .env yang berisi API key untuk basis data, golang untuk menjalankan *backend* dan node js untuk menjalankan *frontend*, serta jaringan internet yang stabil. Setelah semua *prerequisite* sudah ter-install atau tersedia itu jalankan

```
npm run dependency
```

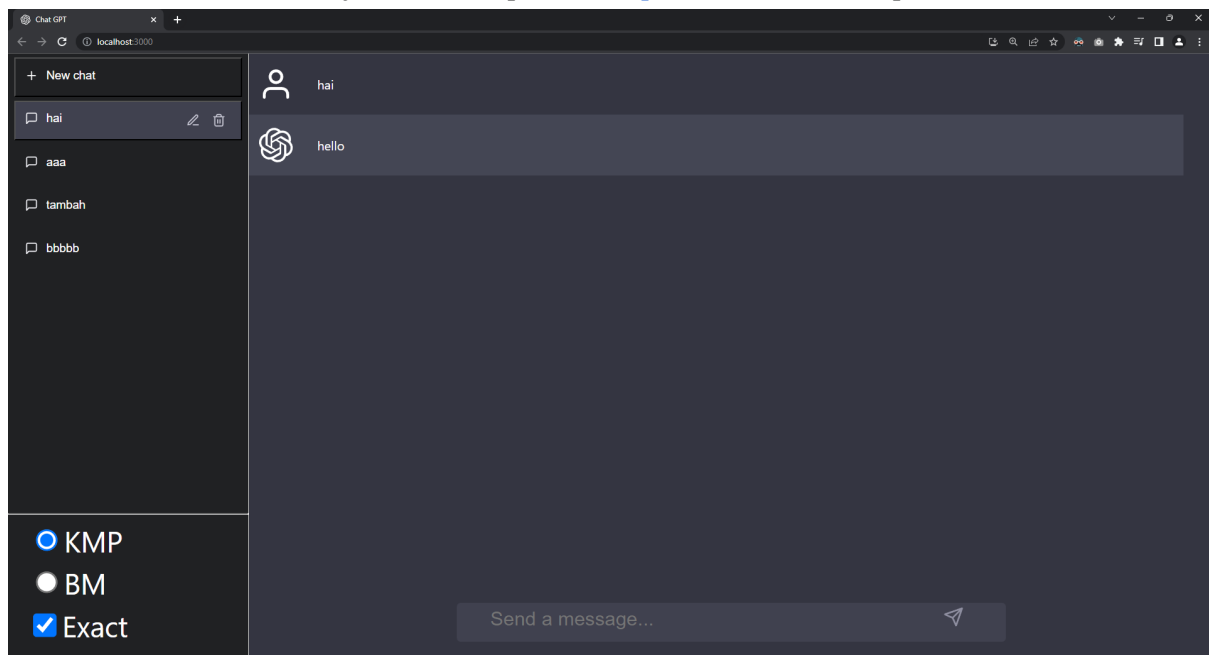
untuk menginstall semua *dependency* yang dibutuhkan. Langkah awal menjalankan program adalah menyalakan server dengan *command*

```
npm run go
```

kemudian jalankan *frontend* dengan *command*.

```
npm run web
```

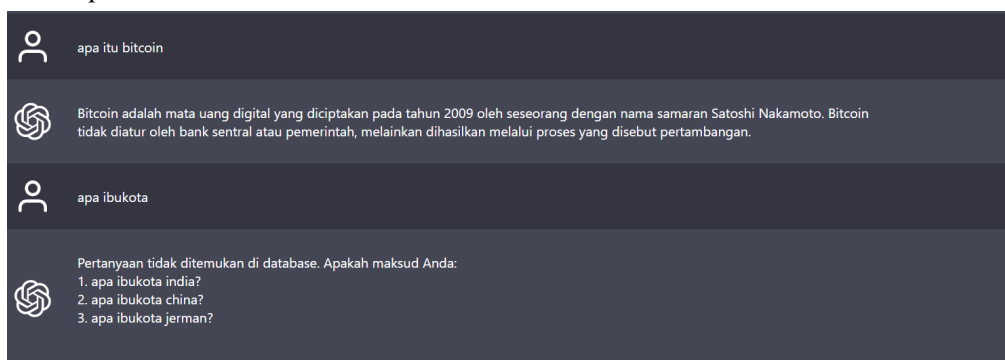
Command tersebut akan menjalankan web pada url <http://localhost:3000/> seperti berikut.



Program juga memiliki beberapa fitur sebagai berikut.

1. Fitur pertanyaan teks (didapat dari database)

Menanyakan kepada program sebuah pertanyaan dan menjawabnya jika pertanyaan tersebut berada pada basis data.



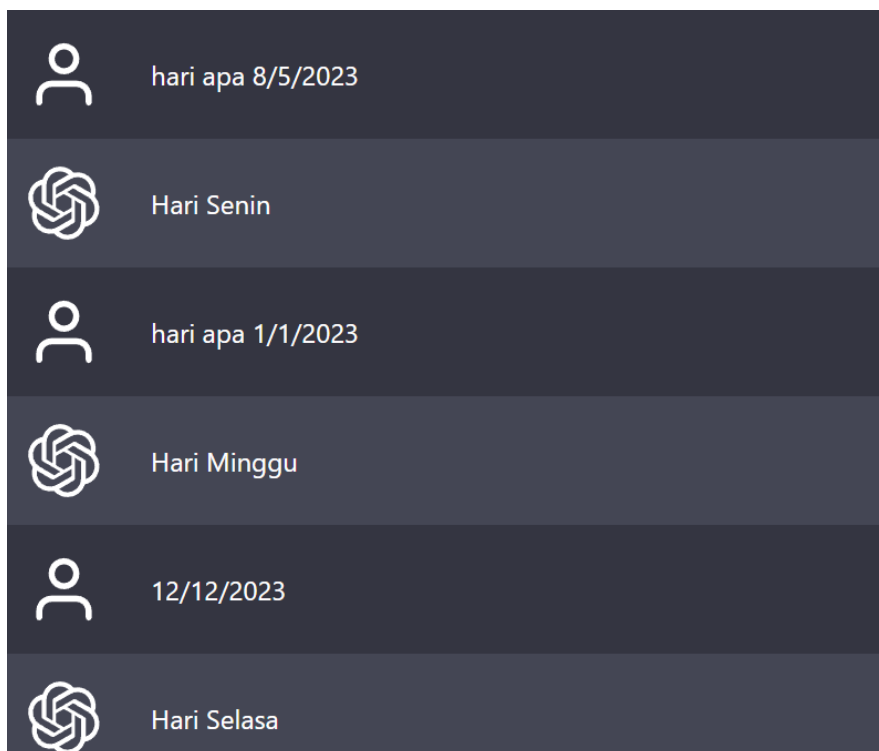
2. Fitur kalkulator

Program dapat menghitung operasi matematika yang ditanyakan. Permintaan harus dimulai dengan kata “hitung”.



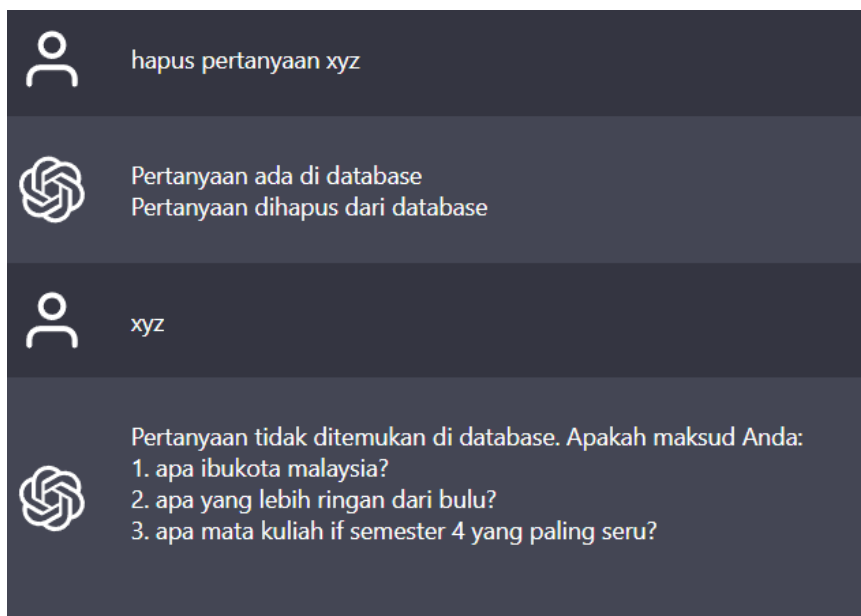
3. Fitur tanggal

Program dapat memberikan hari dari tanggal yang dimasukkan dengan format DD/MM/YYYY.



4. Fitur penambahan dan penghapusan pertanyaan

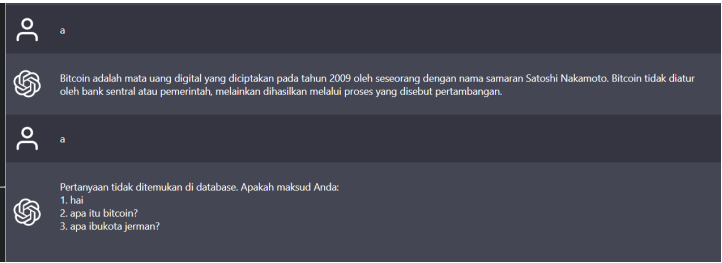
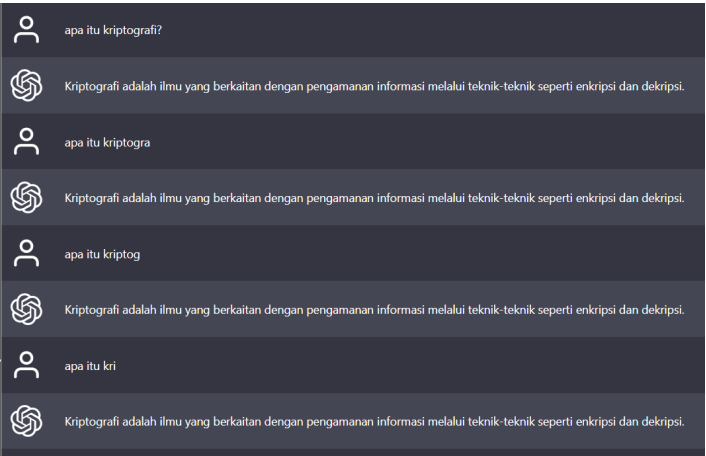
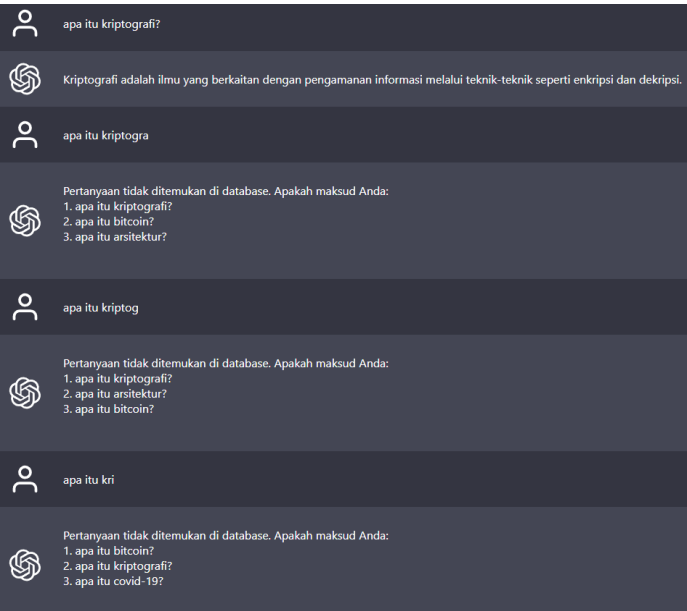
Pertanyaan dapat ditambah atau dihapus dari *database* dengan format *query* “tambahkan pertanyaan <pertanyaan> dengan jawaban <jawaban>” untuk menambahkan pertanyaan dan “hapus pertanyaan <pertanyaan>” untuk menghapus pertanyaan dari *database*.













Selain itu, program memiliki berapa opsi untuk mengolah input dari pengguna seperti algoritma KMP, BM, serta memilih algoritma yang *exact* atau tidak.

4.5. Hasil pengujian


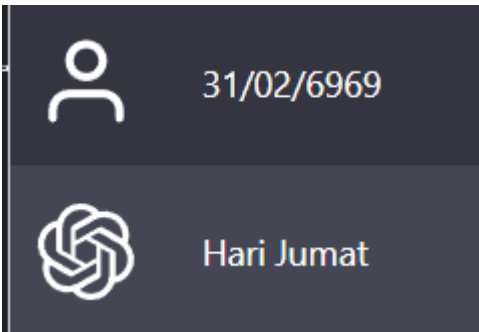
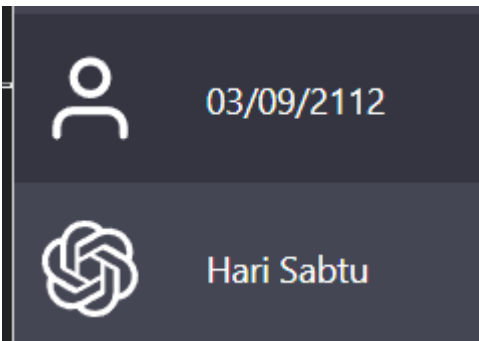
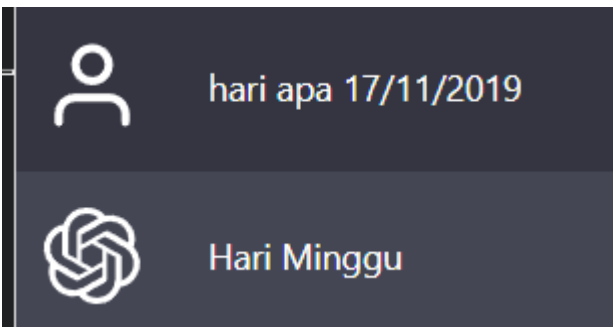
1. Pengujian pertanyaan teks



| Pertanyaan | Hasil |
|---|--|
| a |  <p>The screenshot shows a chat window with a dark theme. The user asks 'a'. The assistant responds with a definition of Bitcoin: 'Bitcoin adalah mata uang digital yang diciptakan pada tahun 2009 oleh seseorang dengan nama samaran Satoshi Nakamoto. Bitcoin tidak diatur oleh bank sentral atau pemerintah, melainkan dihasilkan melalui proses yang disebut pertambangan.' The user then asks 'a' again. The assistant responds with a message: 'Pertanyaan tidak ditemukan di database. Apakah maksud Anda: 1. hai 2. apa itu bitcoin? 3. apa ibukota jerman?'</p> |
| apa itu kriptografi? apa itu kriptogra apa itu kriptog apa itu kri |  <p>The screenshot shows a chat window with a dark theme. The user asks 'apa itu kriptografi?'. The assistant responds: 'Kriptografi adalah ilmu yang berkaitan dengan pengamanan informasi melalui teknik-teknik seperti enkripsi dan dekripsi.' The user then asks 'apa itu kriptogra'. The assistant responds: 'Kriptografi adalah ilmu yang berkaitan dengan pengamanan informasi melalui teknik-teknik seperti enkripsi dan dekripsi.' The user then asks 'apa itu kriptog'. The assistant responds: 'Kriptografi adalah ilmu yang berkaitan dengan pengamanan informasi melalui teknik-teknik seperti enkripsi dan dekripsi.' The user then asks 'apa itu kri'. The assistant responds: 'Kriptografi adalah ilmu yang berkaitan dengan pengamanan informasi melalui teknik-teknik seperti enkripsi dan dekripsi.'</p> |
| apa itu kriptografi? apa itu kriptogra apa itu kriptog apa itu kri |  <p>The screenshot shows a chat window with a dark theme. The user asks 'apa itu kriptografi?'. The assistant responds: 'Kriptografi adalah ilmu yang berkaitan dengan pengamanan informasi melalui teknik-teknik seperti enkripsi dan dekripsi.' The user then asks 'apa itu kriptogra'. The assistant responds: 'Pertanyaan tidak ditemukan di database. Apakah maksud Anda: 1. apa itu kriptografi? 2. apa itu bitcoin? 3. apa itu arsitektur?'. The user then asks 'apa itu kriptog'. The assistant responds: 'Pertanyaan tidak ditemukan di database. Apakah maksud Anda: 1. apa itu kriptografi? 2. apa itu arsitektur? 3. apa itu bitcoin?'. The user then asks 'apa itu kri'. The assistant responds: 'Pertanyaan tidak ditemukan di database. Apakah maksud Anda: 1. apa itu bitcoin? 2. apa itu kriptografi? 3. apa itu covid-19?'</p> |

2. Pengujian fitur kalkulator











| Pertanyaan | Hasil |
|--|--|
| Hitung $((12 + 7) / 3) * (8 - 4) + (10^3) - ((5 * 2) / 4)$ |  Hitung $((12 + 7) / 3) * (8 - 4) + (10^3) - ((5 * 2) / 4)$  Hasilnya adalah 1022.8333333333333 |
| Hitung $3.2 * (4 + 5.010) - 12 / 3$ |  Hitung $3.2 * (4 + 5.010) - 12 / 3$  Hasilnya adalah 24.832 |
| Hitung $+ 3.2 * (4 + 5.010) - 12 / 3$ |  Hitung $+ 3.2 * (4 + 5.010) - 12 / 3$  Sintaks persamaan tidak sesuai |
| Hitung $a + b * (c^d - e)^{(f + g * h) - i}$ |  Hitung $a + b * (c^d - e)^{(f + g * h) - i}$  Sintaks persamaan tidak sesuai |
| Hitung $4 + 5 * (2 - 7)$ |  Hitung $4 + 5 * (2 - 7)$  Sintaks persamaan tidak sesuai |

















3. Pengujian fitur tanggal

| Pertanyaan | Hasil |
|---------------------|--|
| Hari apa 17/08/1945 |  |
| 31/02/6969 |  |
| 03/09/2112 |  |
| hari apa 17/11/2019 |  |

| | |
|--------------------|--|
| hari apa -1/2/2012 |  hari apa -1/2/2012  Pertanyaan tidak ditemukan di database. Apakah maksud Anda: 1. bagaimana kerja printer? 2. apa ibukota perancis? 3. apa ibukota malaysia? |
|--------------------|--|

4. Pengujian fitur penambahan dan penghapusan pertanyaan

| Pertanyaan | Hasil |
|--|---|
| Tambahkan pertanyaan apa 1 + 1 dengan jawaban 3 dan hapus pertanyaan apa 1 + 1 |  Tambahkan pertanyaan apa 1 + 1 dengan jawaban 3  Pertanyaan belum ada di database Pertanyaan ditambah ke database  apa 1 + 1  3  hapus pertanyaan apa 1 + 1  Pertanyaan ada di database Pertanyaan dihapus dari database  apa 1 + 1  Pertanyaan tidak ditemukan di database. Apakah maksud Anda: 1. hitung 10 + 2 2. apa itu covid-19? 3. apa itu bitcoin?  hapus pertanyaan apa 1 + 1  Pertanyaan tidak ada di database |

| | |
|---|---|
| <p>Tambahkan pertanyaan hitung $10 + 2$ dengan jawaban nol dan hapus pertanyaan hitung $10 + 2$</p> | <div data-bbox="662 210 1385 656">  Tambahkan pertanyaan hitung $10 + 2$ dengan jawaban nol  Pertanyaan belum ada di database Pertanyaan ditambah ke database  hitung $10 + 2$  Hasilnya adalah 12 </div> <div data-bbox="662 667 1262 1247">  hapus pertanyaan hitung $10 + 2$  Pertanyaan ada di database Pertanyaan dihapus dari database  hitung $10 + 2$  Hasilnya adalah 12 </div> |
| <p>tambahkan pertanyaan a a - a / a z a dengan jawaban b b c dan hapus pertanyaan a a - a / a z a</p> | <div data-bbox="662 1285 1385 2011">  tambahkan pertanyaan a a - a / a z a dengan jawaban b b c  Pertanyaan belum ada di database Pertanyaan ditambah ke database  a a - a / a z a  b b c  hapus pertanyaan a a - a / a z a  Pertanyaan ada di database Pertanyaan dihapus dari database  a a - a / a z a  Pertanyaan tidak ditemukan di database. Apakah maksud Anda: 1. apa yang dimaksud dengan robotika? 2. mengapa ikan tidak suka bermain kartu? 3. apa mata kuliah if semester 4 yang paling seru? </div> |

| | |
|--|--|
| | |
|--|--|

4.6. Analisis hasil pengujian

1. Analisis pertanyaan teks

Algoritma string matching yang digunakan memiliki 2 mode yaitu pencocokan dengan substring dan pencocokan exact. Perbedaan dapat dilihat dari hasil dari uji pertama. Perbedaan hasil didapatkan karena untuk pencocokan dengan substring karakter a terdapat dalam pertanyaan “apa itu bitcoin?” karakter a muncul pada urutan awal sehingga terdapat match. Untuk mode exact, pertanyaan dengan karakter a tidak memiliki *pattern* yang sama persis dengan pertanyaan yang ada sehingga menghasilkan daftar pertanyaan yang paling mirip dengan input user. Perbedaan tersebut juga terlihat dari uji kedua dan ketiga. Untuk uji kedua dengan mode substring mendapatkan jawaban lebih sering jika dibandingkan dengan mode exact.

2. Analisis kalkulator

Kalkulator dapat digunakan dengan menuliskan kata kunci hitung yang diikuti dengan ekspresi matematika/aritmatika yang ingin digunakan. Hasil perhitungan akan langsung dijawab oleh chatbot. Chatbot akan menjawab “Sintaks persamaan tidak sesuai” apabila ekspresi matematika/aritmatika tidak sesuai aturan. Dalam hasil pengujian pada bagian 4.5. nomor 2 (Pengujian fitur kalkulator), didapat kesalahan terhadap peletakan/penggunaan operator, kesalahan terhadap peletakan/penggunaan kurung, dan kesalahan penggunaan operand yang seharusnya berupa angka.

3. Analisis tanggal

Pertanyaan mengenai tanggal yang diterima oleh aplikasi sesuai dengan format tanggal yaitu tanggal 1-31, bulan 1-12, dan tahun 0-... . Akan tetapi, walaupun hasilnya sudah benar, restriksi dari nilai untuk format tersebut khususnya tanggal belum sesuai dengan nilai tanggal yang sebenarnya seperti pada bulan Februari yang tidak memiliki tanggal 31.

4. Analisis penambahan dan penghapusan pertanyaan

Penambahan dan penghapusan pertanyaan dari basis data melalui aplikasi sudah berjalan cukup baik. Aplikasi dapat menerima input tidak huruf saja dan sudah sensitif dengan *whitespace* yang dimasukkan. Akan tetapi, jika pertanyaan yang dimasukkan oleh user memiliki arti lain yang prioritasnya lebih tinggi seperti fitur kalkulator maka program akan mengeluarkan hasil perhitungan fitur tersebut walaupun pertanyaan baru tersebut sudah dimasukkan ke dalam basis data.

BAB V

KESIMPULAN, SARAN, DAN REFLEKSI

5.1. Kesimpulan

Algoritma string matching dengan menggunakan RegEx, Algoritma KMP (Knuth-Morris-Pratt) dan BM (Boyer-Moore) untuk penentuan string yang *exact match*, serta Algoritma Levenshtein Distance untuk aproksimasi kemiripan string berguna dalam pembuatan ChatBot yang *semi-intelligent*. ChatBot akan menerima pertanyaan dari user dan menjawab pertanyaan tersebut semampunya. ChatBot akan menggunakan RegEx untuk menentukan tipe apa pertanyaan tersebut, kemudian menggunakan Algoritma KMP, BM, kemudian Levenshtein Distance untuk menentukan jawaban yang sesuai dari “memorinya” atau dalam kata lain dari database.

5.2. Saran

Dalam spesifikasi dapat dijelaskan lebih lanjut mengenai arti dari *exact match*, apakah berupa substring yang *match* dengan pola string yang ingin digunakan, atau diharuskan bahwa pola string yang digunakan tepat sama dengan string yang dibandingkan.

5.3. Refleksi

ChatBot dapat ditingkatkan “kecerdasannya” dengan menggunakan *constraint* yang dapat ditentukan. RegEx juga dapat ditambahkan untuk menentukan lebih banyak kasus input user.

DAFTAR PUSTAKA

<https://www.geeksforgeeks.org/applications-of-string-matching-algorithms/>, diakses 5 Mei 2023

Munir, Rinaldi. 2021. “Pencocokan String (String/Pattern Matching)”. <https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Pencocokan-string-2021.pdf>, diakses 5 Mei 2023.

Khodra, Masayu Leylia. 2019. “String Matching dengan Regular Expression”. <https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2018-2019/String-Matching-dengan-Regex-2019.pdf>, diakses 5 Mei 2023.

LAMPIRAN

Link github : https://github.com/yansans/Tubes3_13521110