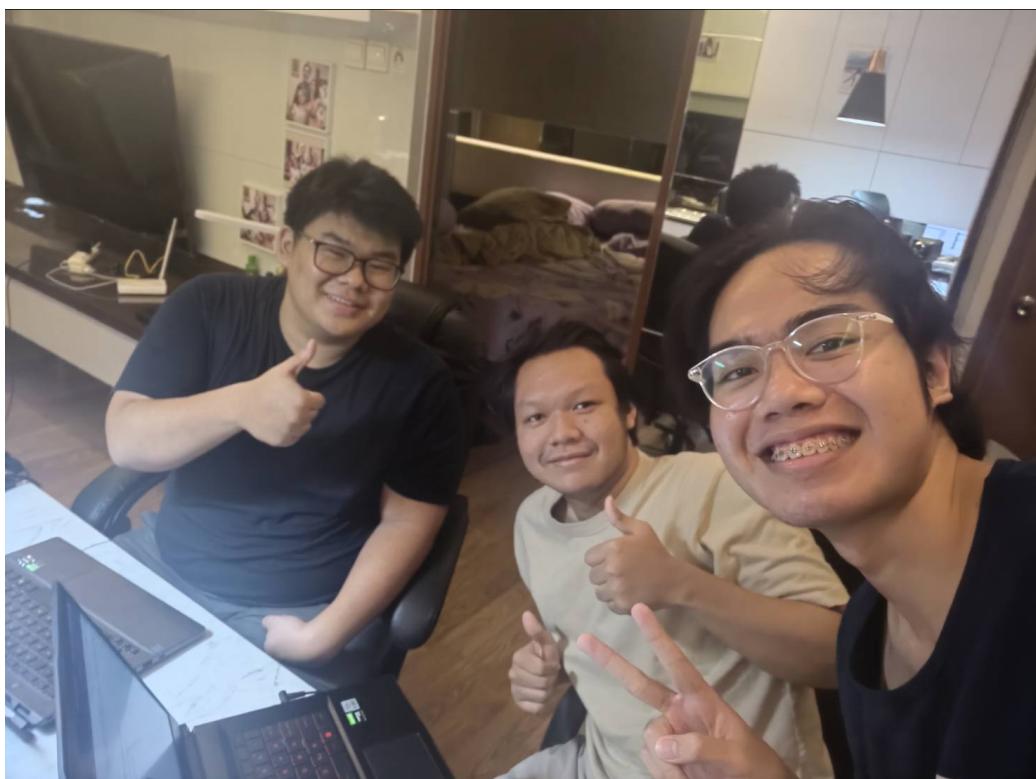


# **Laporan Tugas Besar 3 IF2211 Strategi Algoritma**

**Penerapan String Matching dan Regular Expression**

**dalam Pembuatan ChatGPT Sederhana**



**Disusun oleh  
Kelompok CHATime**

**Yobel Dean Christopher** **13521067**

**Hobert Anthony Jonatan** **13521079**

**Shidqi Indy Izhari** **13521097**

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

## **DAFTAR ISI**

<b>BAB I.....</b>	<b>3</b>
<b>BAB II.....</b>	<b>4</b>
<b>BAB III.....</b>	<b>5</b>
<b>BAB IV.....</b>	<b>6</b>
<b>BAB V.....</b>	<b>7</b>
<b>DAFTAR PUSTAKA.....</b>	<b>8</b>

# BAB I

## DESKRIPSI TUGAS

### 1.1 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.

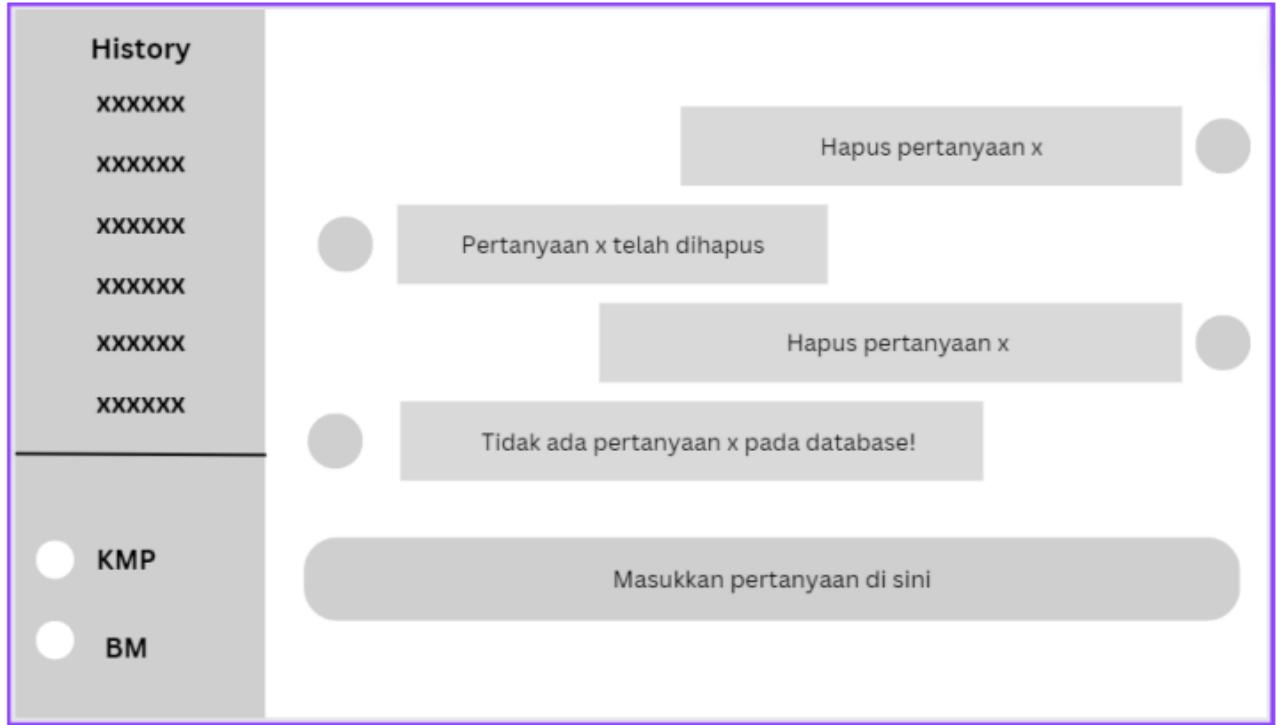
Perhitungan tingkat kemiripan dibebaskan kepada anda asalkan dijelaskan di laporan, namun disarankan menggunakan salah satu dari algoritma Hamming Distance, Levenshtein Distance, ataupun Longest Common Subsequence.

#### 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”.



Gambar 1.1 Ilustrasi Keseluruhan Program

## BAB II

# LANDASAN TEORI

### 2.1 Algoritma Knuth-Morris-Pratt (KMP)

Algoritma Knuth-Morris-Pratt (KMP) adalah algoritma pencocokan string yang digunakan untuk mencari kecocokan pola dalam sebuah teks atau string. Algoritma ini melakukan pencocokan string dari kiri ke kanan sama seperti pada algoritma *brute-force* tetapi dengan pendekatan heuristik yang lebih cerdas sehingga membuatnya lebih efisien.

Algoritma KMP menggunakan tabel fungsi prefiks yang menyimpan informasi tentang panjang prefiks terpanjang dari pola pencarian yang juga merupakan sufiks dari prefiks pola pencarian tersebut. Tabel ini digunakan untuk mengurangi pengulangan pencocokan pada posisi yang telah dicocokkan sebelumnya, dan secara efektif mempercepat pencarian kecocokan pola dalam teks.

Algoritma KMP bekerja dengan menggeser pola pencarian ke kanan hingga kecocokan ditemukan atau seluruh teks telah dipindai. Setiap kali ada kesalahan dalam pencocokan, algoritma KMP memperbarui posisi pola pencarian dengan memanfaatkan informasi pada tabel fungsi prefiks.

Keuntungan dari algoritma KMP adalah kemampuannya untuk mencari kecocokan pola dalam teks dengan waktu yang lebih cepat dibandingkan algoritma Naive, terutama pada kasus teks yang panjang dan pola pencarian yang pendek. Selain itu, algoritma KMP juga cukup sederhana dan mudah dipahami. Namun, kelemahan dari algoritma KMP adalah penggunaannya membutuhkan pengalokasian tabel fungsi prefiks, yang memakan ruang memori yang cukup besar pada teks yang sangat panjang atau pola pencarian yang sangat besar. Selain itu, algoritma KMP juga mungkin kurang efisien pada kasus pola pencarian yang memiliki banyak kemiripan dengan awalan atau akhiran kata dalam teks, yang dapat menghasilkan pengulangan pencocokan yang tidak perlu.

### 2.2 Algoritma Boyer-Moore

Algoritma Boyer-Moore adalah algoritma yang digunakan untuk mencari kecocokan atau pola dalam sebuah teks atau string, kegunaannya serupa dengan KMP, tetapi Algoritma ini berbeda dari algoritma pencocokan string lainnya seperti algoritma *brute-force* dan Knuth-Morris-Pratt (KMP) karena ia menggunakan pendekatan yang lebih cerdas dan efisien untuk pencocokan string.

Algoritma Boyer-Moore bekerja dengan mengurutkan dan memproses pola pencarian dari kanan ke kiri dan kemudian memindai teks dengan menggeser pola pencarian ke kanan. Hal ini memungkinkan algoritma ini untuk melakukan pencocokan secara cepat, bahkan dalam kasus string yang sangat panjang dan rumit.

Salah satu komponen penting dari algoritma Boyer-Moore adalah penggunaan tabel heuristik, seperti tabel *bad character* dan tabel *good suffix*. Tabel *bad character* menyimpan informasi tentang posisi terakhir dari setiap karakter dalam pola pencarian, sehingga ketika

ada kesalahan pencocokan pada karakter tertentu dalam teks, algoritma dapat dengan cepat memindahkan pola pencarian ke posisi yang tepat berdasarkan informasi pada tabel ini. Sementara tabel *good suffix* digunakan untuk memindahkan pola pencarian ke posisi yang tepat jika terjadi kesalahan pada susunan karakter yang tepat dalam pola pencarian.

Keuntungan dari algoritma Boyer-Moore adalah kemampuannya untuk memotong sejumlah besar karakter yang tidak sesuai dalam teks, sehingga meminimalkan jumlah perbandingan yang harus dilakukan. Hal ini membuat algoritma ini sangat efisien dalam melakukan pencocokan string, terutama pada teks yang sangat panjang. Namun, kelemahan dari algoritma ini adalah ketika pola pencarian berisi banyak karakter yang sama atau memiliki susunan yang sama dalam jumlah yang banyak, maka algoritma ini tidak begitu efektif. Hal ini terjadi karena algoritma ini hanya dapat memeriksa beberapa karakter pada setiap langkah, sehingga jika ada banyak karakter yang sama dalam pola pencarian, ia akan memerlukan lebih banyak langkah untuk menemukan kecocokan.

## 2.3 Regular Expression

Regular Expression (RegEx) adalah sebuah sintaks yang digunakan untuk mencocokkan pola atau pattern dalam sebuah teks atau string. Dalam RegEx, sebuah pola dapat diartikan sebagai kumpulan karakter yang dipilih untuk dicocokkan pada sebuah teks.

RegEx digunakan dalam berbagai aplikasi yang membutuhkan pencocokan pola pada sebuah teks, seperti pengolahan teks, pengambilan data dari file teks, dan pengujian validitas data input. RegEx juga digunakan dalam beberapa bahasa pemrograman, seperti Perl, Python, dan Java, sebagai bagian dari fitur String Processing. Dalam aplikasi web yang dibangun pada tugas besar 3 ini, RegEx digunakan untuk mengenali pola pada kalimat yang dimasukkan oleh *user* untuk dicocokkan dengan pola-pola yang dapat digunakan untuk mengakses fitur pada aplikasi web.

## 2.4 Pengembangan Aplikasi Web

Pengembangan aplikasi web pada tugas besar ini dipecah menjadi 2 bagian besar, yaitu *frontend* dan *backend*. *frontend* merupakan bagian yang bertanggung jawab terkait *User Interface* untuk aplikasi web ini, sedangkan *backend* merupakan bagian yang bertanggung jawab untuk mengelola algoritma, logika penggerjaan, dan *database* terkait aplikasi web yang dibangun. Teknologi yang digunakan untuk *frontend* adalah HTML, CSS, Javascript, dan React, sedangkan teknologi yang digunakan untuk *backend* adalah NodeJS, ExpressJS, dan MongoDB. *Tech Stack* yang digunakan untuk membangun aplikasi web ini merupakan salah satu *stack* yang sedang cukup populer juga sekarang, yaitu MERN (MongoDB, Express, React, Node). Bagian *frontend* dan *backend* akan berkomunikasi melalui API agar data masukan dari *user* pada bagian *frontend* dapat diteruskan kepada bagian *backend* serta bagian *backend* dapat mengirim kembali jawaban dari permintaan *user*.

## **BAB III**

### **ANALISIS PEMECAHAN MASALAH**

#### **3.1 Langkah Penyelesaian Masalah Tiap Fitur**

##### **3.1.1 Penyelesaian Fitur Pertanyaan Teks**

Fitur utama ini diselesaikan dengan menggunakan dua buah algoritma utama; KMP dan BM. Kedua algoritma tersebut merupakan algoritma string matching. Pencarian jawaban dari sebuah pertanyaan akan dicari dalam sebuah database.

Algoritma Knuth-Morris-Pratt (KMP) digunakan untuk mencari pola dalam sebuah teks dalam fungsi kmp. Fungsi kmp menerima dua parameter, yaitu text dan word. Fungsi ini mengembalikan indeks pertama dari kemunculan pola word dalam teks text, atau -1 jika tidak ditemukan. Fungsi kmp menggunakan fungsi buildPatternTable untuk membangun tabel pola untuk word. Tabel pola ini digunakan untuk mengoptimalkan pencarian pola dalam teks. Fungsi buildPatternTable menggunakan pendekatan prefix-suffix untuk membangun tabel pola. Fungsi ini memeriksa setiap karakter dalam word dan mencatat panjang maksimum dari awalan yang sama dengan akhiran dari substring word yang berakhir pada karakter tersebut. Tabel pola ini kemudian digunakan oleh kmp untuk menghitung indeks pertama dari kemunculan pola dalam teks. Fungsi kmp bekerja dengan memeriksa setiap karakter dalam teks text secara berurutan. Jika karakter tersebut sama dengan karakter pada indeks yang sama dalam word, maka fungsi kmp memeriksa karakter berikutnya. Jika seluruh karakter dalam word ditemukan secara berurutan dalam text, maka fungsi kmp mengembalikan indeks pertama dari kemunculan pola dalam text. Jika tidak, fungsi kmp menggunakan tabel pola untuk menggeser indeks pencarian text dan word agar mencocokkan karakter yang tidak cocok dalam pola.

Selanjutnya, adalah algoritma Boyer-Moore (BM) yang digunakan untuk mencari pola dalam sebuah teks. Fungsi bm menerima dua parameter, yaitu str dan pattern. Fungsi ini mengembalikan indeks pertama dari kemunculan pola pattern dalam teks str, atau -1 jika tidak ditemukan. Fungsi bm menggunakan fungsi buildBadMatchTable untuk membangun tabel pencocokan buruk untuk pattern. Tabel pencocokan buruk ini digunakan untuk mengoptimalkan pencarian pola dalam teks. Fungsi buildBadMatchTable membangun tabel dengan menghitung jarak dari kemunculan terakhir suatu karakter dalam pattern menuju ujung kanan pattern. Jarak ini digunakan untuk menentukan seberapa jauh pattern harus digeser ke kanan setiap kali karakter yang tidak cocok dalam teks str ditemukan. Fungsi bm bekerja dengan memeriksa setiap karakter dalam teks str secara berurutan. Jika karakter tersebut sama dengan karakter pada indeks yang sama dalam pattern, maka fungsi bm memeriksa karakter berikutnya. Jika seluruh karakter dalam pattern ditemukan secara berurutan dalam str, maka fungsi bm mengembalikan indeks pertama dari kemunculan pola dalam str. Jika tidak, fungsi bm menggunakan tabel pencocokan buruk untuk menggeser indeks pencarian str dan pattern agar mencocokkan karakter yang tidak cocok dalam pola. Jika tidak ada karakter dalam tabel pencocokan buruk yang cocok dengan karakter yang tidak cocok dalam pola, fungsi bm menggeser pattern sejauh satu karakter ke kanan dan melanjutkan pencarian.

### **3.1.2 Penyelesaian Fitur Kalkulator**

Fitur ini diselesaikan menggunakan sebuah fungsi calculate yang digunakan untuk menghitung hasil dari suatu persamaan matematika yang diinputkan dalam bentuk string. Fungsi ini menerima satu parameter expression, yang berisi persamaan matematika yang akan dihitung. Fungsi calculate bekerja dengan memisahkan operand dan operator dalam persamaan, kemudian memprosesnya secara berurutan dengan memperhatikan prioritas operator. Fungsi ini juga dapat menangani tanda kurung dalam persamaan, sehingga dapat menghitung persamaan yang kompleks. Fungsi calculate memanfaatkan fungsi evaluate untuk menghitung hasil dari dua operand dengan satu operator. Persamaan matematika yang dihitung harus memiliki format yang sesuai, jika tidak maka fungsi akan mengembalikan pesan "Sintaks persamaan tidak sesuai".

### **3.1.3 Penyelesaian Fitur Tanggal**

Fitur Tanggal diselesaikan melalui fungsi getDayOfWeek yang digunakan untuk mengembalikan nama hari dalam bahasa Indonesia berdasarkan tanggal yang diinputkan. Fungsi ini menerima satu parameter dateString, yang berisi tanggal dalam format "dd/mm/yyyy". Fungsi getDayOfWeek bekerja dengan memisahkan tanggal, bulan, dan tahun dari dateString, kemudian menghitung jumlah hari sejak tanggal referensi (1 Januari 1900) dengan memperhatikan apakah tahun tersebut kabisat atau bukan. Setelah itu, fungsi getDayOfWeek menggunakan modulus untuk menentukan hari dalam bahasa Indonesia berdasarkan jumlah hari yang telah dihitung. Fungsi getDayOfWeek juga menggunakan array daysInMonth untuk menentukan jumlah hari dalam setiap bulan pada tahun kabisat atau bukan kabisat, serta array daysOfWeek untuk menentukan nama hari dalam bahasa Indonesia. Fungsi tambahan stringToDate digunakan untuk mengubah format tanggal dari "dd mmmm yyyy" menjadi "dd/mm/yyyy". Fungsi ini memisahkan tanggal, bulan, dan tahun dari input string, kemudian mengubah bulan dalam format teks menjadi angka sesuai dengan format "mm" dalam tanggal. Fungsi ini kemudian mengembalikan tanggal baru dalam format "dd/mm/yyyy".

### **3.1.4 Penyelesaian Fitur Tambah Pertanyaan dan Jawaban ke Database**

Fitur ini diselesaikan dengan fungsi insertQuestion yang digunakan untuk memasukkan atau mengupdate sebuah pertanyaan dan jawabannya dalam sebuah database MongoDB. Fungsi ini menerima lima parameter, yaitu client, dbName, collectionName, question, dan answer. Fungsi insertQuestion bekerja dengan menggunakan fungsi isQuestionInDB untuk memeriksa apakah pertanyaan tersebut sudah ada dalam database. Jika pertanyaan sudah ada, fungsi insertQuestion akan mengupdate jawaban pertanyaan tersebut. Jika tidak, fungsi insertQuestion akan memasukkan pertanyaan dan jawaban baru ke dalam database. Fungsi insertQuestion menggunakan fungsi updateOne dan insertOne dari MongoDB untuk mengupdate atau memasukkan data ke dalam database. Fungsi ini mengembalikan pesan yang sesuai dengan hasil operasi yang dilakukan. Jika operasi berhasil, fungsi insertQuestion akan mengembalikan pesan sukses. Jika gagal, fungsi insertQuestion akan mengembalikan pesan kesalahan.

### **3.1.5 Penyelesaian Fitur Hapus Pertanyaan dari Database**

Fitur ini diselesaikan dengan sebuah fungsi deleteQuestion yang digunakan untuk menghapus sebuah pertanyaan dari sebuah database MongoDB. Fungsi ini menerima empat parameter, yaitu client, dbName, collectionName, dan question. Fungsi deleteQuestion bekerja dengan menggunakan fungsi deleteOne dari MongoDB untuk menghapus data dari database. Fungsi ini mengembalikan pesan yang sesuai dengan hasil operasi yang dilakukan.

Jika operasi berhasil, fungsi `deleteQuestion` akan mengembalikan pesan sukses. Jika pertanyaan tidak ditemukan dalam database, fungsi `deleteQuestion` akan mengembalikan pesan bahwa pertanyaan tersebut tidak ditemukan. Jika operasi gagal, fungsi `deleteQuestion` akan mengembalikan pesan kesalahan.

## 3.2 Fitur Fungsional Pada Aplikasi Web

### 3.2.1 Fitur Pertanyaan Teks

Fitur pertanyaan teks digunakan untuk menggunakan masukan *user* sebagai pola yang harus dicocokan dengan data yang ada dalam *database*, yaitu kumpulan pertanyaan yang telah disimpan terlebih dahulu dalam *database*. Pencocokan pola kalimat dilakukan dengan menggunakan algoritma Boyer-Moore atau KMP, sesuai dengan pilihan *user*. Apabila terdapat *exact match* dengan pertanyaan yang ada dalam *database* maka akan diambil jawaban dari pertanyaan tersebut di *database* sebagai jawaban yang akan diberikan kepada *user*, sedangkan jika tidak terjadi *exact match*, maka akan dicari pertanyaan pada *database* dengan persentase kecocokan minimal 90% dengan pola yang dicari, jika ada maka jawaban dari pertanyaan tersebutlah yang dikembalikan sebagai jawaban akhir, sedangkan jika masih tidak ditemukan, maka akan ditampilkan 3 pertanyaan dari *database* yang memiliki persentase kecocokan tertinggi agar *user* dapat memilih sendiri pertanyaan mana yang kira-kira dia maksudkan.

### 3.2.2 Fitur Kalkulator

Fitur kalkulator digunakan untuk menjawab masukan *user* yang dikenali sebagai ekspresi matematika yang dapat dihitung nilai akhirnya. Namun, operasi yang dapat dilakukan hanya terbatas pada operasi tambah, kurang, kali, bagi, pangkat, dan kurung.

### 3.2.3 Fitur Tanggal

Fitur tanggal digunakan untuk menjawab masukan *user* yang dikenali sebagai format tanggal yang valid untuk penanggalan, dalam hal ini format yang digunakan oleh chatbot kami adalah “DD/MM/YYYY”, format tanggal selain ini tidak dapat dikenali sebagai format tanggal yang valid. Dari tanggal yang diberikan, chatbot kemudian dapat menjawab hari apakah tanggal tersebut.

### 3.2.4 Fitur Tambah Pertanyaan dan Jawaban ke Database

Fitur ini digunakan untuk menambahkan pertanyaan baru ke *database* beserta jawaban dari pertanyaan tersebut, fitur ini akan dipanggil ketika masukan dari *user* dikenali oleh fungsi regex sebagai format untuk menambahkan pertanyaan beserta jawaban ke *database*. Sebelum menambahkan pertanyaan dan jawaban, akan diperiksa terlebih dahulu apakah pertanyaan serupa telah ada dalam database atau tidak, jika pertanyaan tersebut sudah ada dalam *database* maka akan diperiksa kembali apakah jawaban yang ditambahkan untuk pertanyaan tersebut sama dengan jawaban yang telah ada pada *database* atau tidak, jika sama maka akan dikembalikan pesan bahwa pertanyaan dan jawaban yang sama telah ada dalam *database*, sedangkan jika jawabannya tidak sama dengan jawaban yang ada dalam *database*, maka jawaban untuk pertanyaan tersebut akan di *update* menjadi jawaban yang dimasukan pengguna. Jika pertanyaan tersebut belum ada pada *database*, maka pertanyaan beserta jawaban dari *user* akan ditambahkan ke dalam *database*.

### 3.2.5 Fitur Hapus Pertanyaan dari Database

Fitur ini digunakan untuk menghapus pertanyaan dari *database*, fitur ini akan dipanggil ketika masukkan dari *user* dikenali oleh fungsi regex sebagai format untuk menghapus

pertanyaan dari database. Akan diperiksa terlebih dahulu apakah pertanyaan yang ingin dihapus oleh *user* ada di dalam *database* atau tidak, jika ada maka pertanyaan dan jawaban dari pertanyaan tersebut akan dihapus dari *database*, sedangkan jika tidak maka akan dikembalikan pesan bahwa pertanyaan tersebut tidak ada dalam *database*.

### **3.3 Arsitektur Aplikasi Web**

#### **3.3.1 Frontend**

Frontend pada website ini menggunakan React dari JavaScript. ReactJS adalah sebuah library JavaScript yang digunakan untuk membangun antarmuka pengguna (user interface) yang interaktif dan dinamis pada aplikasi web. ReactJS dikembangkan oleh Facebook dan dirilis pada tahun 2013. ReactJS menggunakan pendekatan komponen (component-based) dalam pembangunan antarmuka pengguna. Setiap komponen pada ReactJS dapat digunakan ulang dan dapat disusun bersama untuk membentuk aplikasi yang kompleks. File penghubung utama antara frontend dengan backend adalah App.js dan SendChat.js

#### **3.3.2 Backend**

Backend pada website ini menggunakan Node dan Express dari JavaScript. Node.js adalah sebuah platform untuk menjalankan JavaScript di sisi server yang memungkinkan pengembang untuk membuat aplikasi web berkinerja tinggi dan skala besar dengan mudah. Node.js menggunakan model non-blocking I/O yang membuatnya sangat cepat dan efisien dalam mengelola permintaan yang datang. Express.js adalah sebuah framework web untuk Node.js yang digunakan untuk membangun aplikasi web RESTful dengan mudah dan cepat. Express.js menyediakan berbagai fitur seperti middleware, routing, dan template engine untuk mempermudah pengembangan aplikasi web. Dengan menggunakan Express.js, pengembang dapat membuat aplikasi web dengan cepat dan mudah, serta dapat mengelola permintaan dan respons dengan mudah melalui struktur middleware yang fleksibel. Keduanya sering digunakan bersama-sama untuk membangun aplikasi web modern dan skalabel. File penghubung utama antara frontend dengan backend adalah server.js dan router.js. Selain itu, terdapat juga database pada bagian backend pada file app.js yang diupload pada cloud MongoDB.

## BAB IV

# IMPLEMENTASI DAN PENGUJIAN

### 4.1 Spesifikasi Teknis Program

#### 1. bm.js

Struktur data yang digunakan dalam kode ini adalah objek. Objek digunakan untuk menyimpan tabel pencocokan buruk (bad match table) yang dibangun dengan fungsi buildBadMatchTable. Tabel ini berisi jarak dari kemunculan terakhir suatu karakter dalam pattern menuju ujung kanan pattern.

Fungsi buildBadMatchTable digunakan untuk membangun tabel pencocokan buruk. Fungsi ini menerima satu parameter, yaitu str, dan mengembalikan sebuah objek yang berisi tabel pencocokan buruk.

Fungsi bm digunakan untuk mencari pola dalam sebuah teks menggunakan algoritma Boyer-Moore. Fungsi ini menerima dua parameter, yaitu str dan pattern, dan mengembalikan indeks pertama dari kemunculan pola pattern dalam teks str, atau -1 jika tidak ditemukan.

Prosedur yang dilakukan dalam fungsi **bm** adalah sebagai berikut:

- Membangun tabel pencocokan buruk menggunakan fungsi buildBadMatchTable.
- Menginisialisasi variabel offset dengan nilai 0.
- Menghitung indeks terakhir dari pattern dan menyimpannya dalam variabel patternLastIndex.
- Menghitung nilai maksimum offset yang dapat dicapai dan menyimpannya dalam variabel maxOffset.
- Melakukan iterasi selama offset masih kurang dari atau sama dengan maxOffset.
- Melakukan scanning pada teks str dari indeks offset hingga mencapai indeks offset + patternLastIndex.
- Jika seluruh karakter pada pattern cocok dengan karakter pada str pada indeks yang sesuai, maka pola ditemukan dan indeks pertama pola tersebut di dalam str dikembalikan.
- Jika terdapat karakter pada str yang tidak cocok dengan karakter pada pattern, maka dilakukan pencocokan buruk.
- Jika karakter pada str tidak terdapat dalam tabel pencocokan buruk, maka offset ditambahkan dengan 1.
- Jika karakter pada str terdapat dalam tabel pencocokan buruk, maka offset ditambahkan dengan jarak yang sesuai pada tabel pencocokan buruk.
- Jika pola tidak ditemukan setelah iterasi selesai, maka -1 dikembalikan.

#### 2. kmp.js

Struktur data yang digunakan dalam kode ini adalah array. Array digunakan untuk menyimpan tabel pola (pattern table) yang dibangun dengan fungsi buildPatternTable. Tabel ini berisi panjang pola terpanjang yang merupakan awalan dan akhiran dari sebuah sub-pola dalam word.

Fungsi buildPatternTable digunakan untuk membangun tabel pola yang akan digunakan dalam algoritma Knuth-Morris-Pratt (KMP). Fungsi ini menerima satu parameter, yaitu word, dan mengembalikan sebuah array yang berisi tabel pola.

Fungsi kmp digunakan untuk mencari pola dalam sebuah teks menggunakan algoritma KMP. Fungsi ini menerima dua parameter, yaitu text dan word, dan mengembalikan indeks pertama dari kemunculan pola word dalam teks text, atau -1 jika tidak ditemukan.

Prosedur yang dilakukan dalam fungsi kmp adalah sebagai berikut:

- Mengecek apakah word memiliki panjang 0. Jika iya, maka fungsi akan mengembalikan 0.
- Menginisialisasi variabel textIndex dan wordIndex dengan nilai 0.
- Membangun tabel pola menggunakan fungsi buildPatternTable.
- Melakukan iterasi selama textIndex kurang dari panjang text.
- Jika karakter pada text cocok dengan karakter pada word pada indeks yang sesuai, maka dilakukan pembandingan pada karakter selanjutnya.
- Jika seluruh karakter pada word cocok dengan karakter pada text, maka pola ditemukan dan indeks pertama pola tersebut di dalam text dikembalikan.
- Jika terdapat karakter pada text yang tidak cocok dengan karakter pada word, maka dilakukan penyesuaian wordIndex dengan tabel pola yang sesuai.
- Jika karakter pada text tidak terdapat dalam tabel pola, maka textIndex ditambahkan dengan 1.
- Jika pola tidak ditemukan setelah iterasi selesai, maka -1 dikembalikan.

### 3. calcu.js

Struktur data yang digunakan dalam kode tersebut adalah array. Array digunakan untuk menyimpan operand dan operator dalam tumpukan, serta objek priority yang menyimpan urutan prioritas operator.

Fungsi calculate digunakan untuk menghitung nilai dari suatu ekspresi matematika dalam bentuk string. Fungsi ini menerima satu parameter, yaitu expression, dan mengembalikan hasil perhitungan dari ekspresi tersebut.

Prosedur yang dilakukan dalam fungsi calculate adalah sebagai berikut:

- Mendefinisikan fungsi evaluate untuk menghitung nilai antara dua angka dengan satu operator.
- Mendefinisikan array operands dan operators untuk menampung operand dan operator dalam tumpukan.
- Mendefinisikan fungsi processStack untuk memproses tumpukan operand dan operator.
- Mendefinisikan objek priority untuk menentukan urutan prioritas operator.
- Melakukan iterasi pada setiap karakter dalam expression.
- Jika karakter adalah angka, maka karakter tersebut akan ditambahkan ke dalam tumpukan operand.
- Jika karakter adalah tanda kurung buka, maka karakter tersebut akan ditambahkan ke dalam tumpukan operator.
- Jika karakter adalah tanda kurung tutup, maka tumpukan operator akan diproses hingga menemukan tanda kurung buka terdekat.
- Jika karakter adalah operator, maka tumpukan operator yang memiliki prioritas lebih tinggi akan diproses terlebih dahulu.
- Setelah iterasi selesai, tumpukan operator yang tersisa akan diproses.
- Hasil akhir adalah nilai dalam tumpukan operand. Jika tumpukan operand kosong, maka akan dikembalikan pesan "Sintaks persamaan tidak sesuai.".

#### 4. regex.js

Struktur data yang digunakan dalam kode tersebut adalah konstanta. Konstanta digunakan untuk menyimpan nilai tetap yang merepresentasikan fitur-fitur yang dapat dikenali oleh fungsi classifyString.

Fungsi classifyString digunakan untuk mengklasifikasikan input string ke dalam salah satu kategori fitur yang telah ditentukan. Fungsi ini menerima satu parameter, yaitu inputString, dan mengembalikan nilai kategori fitur yang sesuai.

Prosedur yang dilakukan dalam fungsi classifyString adalah sebagai berikut:

- Mendefinisikan konstanta untuk setiap fitur yang dikenali.
- Menggunakan regular expression untuk mengenali pola pada input string yang sesuai dengan fitur-fitur tertentu.
- Jika input string cocok dengan pola "add" atau "Tambahkan pertanyaan", maka kategori fitur yang dikembalikan adalah FITUR\_TAMBAH\_PERTANYAAN.
- Jika input string cocok dengan pola "delete", "remove", atau "Hapus pertanyaan", maka kategori fitur yang dikembalikan adalah FITUR\_HAPUS\_PERTANYAAN.
- Jika input string cocok dengan pola untuk tanggal dengan format "DD/MM/YYYY" atau "DD month YYYY", maka kategori fitur yang dikembalikan adalah FITUR\_TANGGAL.
- Jika input string cocok dengan pola untuk pertanyaan seperti "what", "where", "when", "why", "how", "apa", "dimana", "kapan", "kenapa", atau "bagaimana", maka kategori fitur yang dikembalikan adalah FITUR\_PERTANYAAN.
- Jika input string cocok dengan pola untuk ekspresi matematika, maka kategori fitur yang dikembalikan adalah FITUR\_KALKULATOR.
- Jika input string tidak cocok dengan pola-pola yang telah ditentukan, maka kategori fitur yang dikembalikan adalah UNKNOWN.

#### 5. tanggal.js

Struktur data yang digunakan dalam kode tersebut adalah array. Array digunakan untuk menyimpan data dalam bentuk urutan, seperti jumlah hari pada setiap bulan dan nama hari dalam bahasa Indonesia.

Fungsi getDayOfWeek digunakan untuk mendapatkan nama hari dalam bahasa Indonesia dari suatu tanggal yang diberikan dalam format "DD/MM/YYYY". Fungsi ini menerima satu parameter, yaitu dateString, dan mengembalikan nama hari dalam bahasa Indonesia.

Prosedur yang dilakukan dalam fungsi getDayOfWeek adalah sebagai berikut:

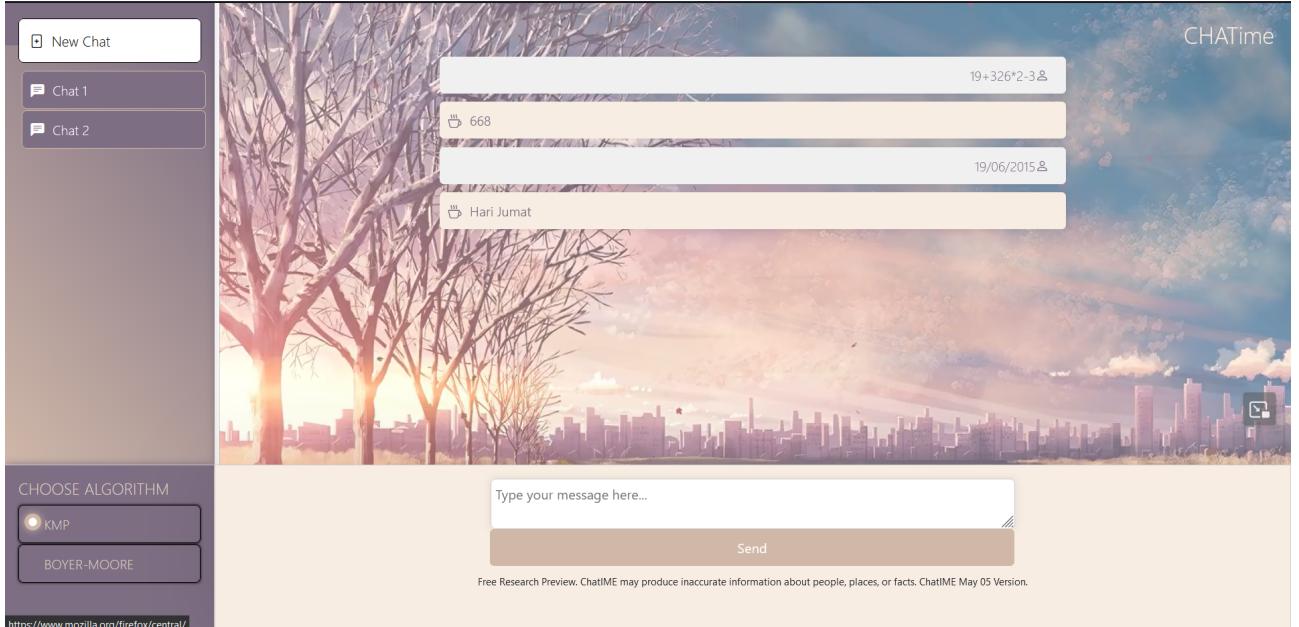
- Memisahkan tanggal, bulan, dan tahun dari dateString.
- Mendefinisikan array daysInMonth untuk menentukan jumlah hari pada setiap bulan pada tahun kabisat dan bukan kabisat.
- Mendefinisikan array daysOfWeek untuk menentukan nama hari dalam bahasa Indonesia.
- Menentukan jumlah hari sejak 1 Januari 1900 (tanggal referensi) berdasarkan tanggal, bulan, dan tahun yang diberikan.
- Menentukan hari dalam bahasa Indonesia berdasarkan jumlah hari tersebut.

## 4.2 Tata Cara Penggunaan Program

### 1. Clone repository ke dalam local file

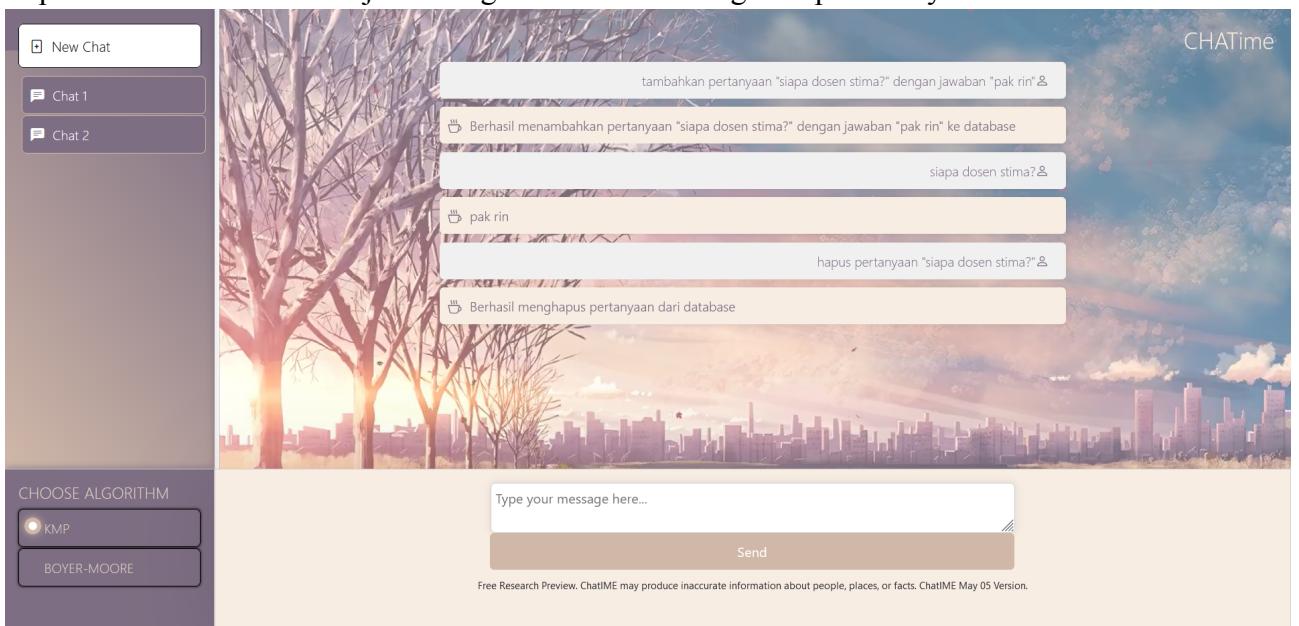
2. Buka dua buah terminal terpisah
3. Ubah directory terminal pertama menjadi Tubes3\_13521067/src/node-backend dan jalankan perintah “node server.js”
4. Ubah directory terminal kedua menjadi Tubes3\_13521067/src/react-website dan jalankan perintah “npm start”
5. Website akan terbuka pada localhost:3000

### 4.3 Hasil Pengujian dan Analisis



**Gambar 4.3.1 Hasil percobaan fitur kalkulator dan tanggal**

Dapat dilihat kedua fitur berjalan dengan baik sesuai dengan keperluannya.



**Gambar 4.3.2 Hasil percobaan fitur tambah pertanyaan, menanyakan sesuatu, dan hapus pertanyaan**

Dapat dilihat ketiga fitur berjalan dengan baik sesuai dengan keperluannya dan juga database yang terupdate secara global.

# BAB V

## KESIMPULAN DAN SARAN

### 5.1 Kesimpulan

Algoritma Knuth-Morris-Pratt (KMP) dan Boyer-Moore merupakan algoritma yang dapat digunakan untuk melakukan pencocokan pola atau string. Kedua algoritma ini lebih efisien dibanding algoritma *brute-force* karena menggunakan pendekatan heuristik yang membuat pencarian atau pencocokan menjadi lebih efisien. Selain itu, ada juga Regular Expression yang merupakan sebuah sintaks yang dapat digunakan untuk mengenali kalimat dengan pola-pola tertentu sesuai yang ingin kita cari agar dapat mengelompokan kalimat sesuai dengan pola yang kita tetapkan.

### 5.2 Saran

Berikut adalah beberapa saran untuk tugas besar ini :

- Sebaiknya waktu penggerjaan sedikit diperpanjang karena mengingat bertabrakan dengan libur lebaran yang membuat mahasiswa terlelap dalam euforia liburan dan masih bersama dengan keluarga sehingga sulit mencair waktu untuk mengerjakan tugas dengan efektif.
- Sebaiknya deadline tidak terlalu dekat dengan jadwal ujian

### 5.3 Komentar Terkait Tugas Besar

Komentar kelompok kami terkait tugas besar ini, yaitu penggerjaannya cukup seru karena mahasiswa dipaksa untuk mempelajari pengembangan aplikasi web menggunakan teknologi-teknologi terbaru yang juga sedang populer sekarang, hasil dari aplikasi web yang dibuat juga dapat dilihat dengan baik dan dapat dikenang sebagai salah satu tugas besar yang paling bermanfaat dan *memorable*.

### 5.4 Refleksi

Berikut adalah refleksi dari kelompok kami terkait penggerjaan tugas besar :

- Seharusnya penggerjaan dimulai lebih awal, karena teknologi yang digunakan sepenuhnya baru bagi kami dan tidak ada anggota kelompok yang memiliki pengalaman dalam pengembangan aplikasi web sebelumnya.
- Seharusnya mempelajari garis besar secara keseluruhan terlebih dahulu sebelum mulai mengerjakan agar sudah jelas apa yang harus dikerjakan dan bagaimana proses penyatuannya nanti.

## **DAFTAR PUSTAKA**

***Link Repository :*** [https://github.com/yobeldc/Tubes3\\_13521067](https://github.com/yobeldc/Tubes3_13521067)

***Link Video :*** [Video demo](#)