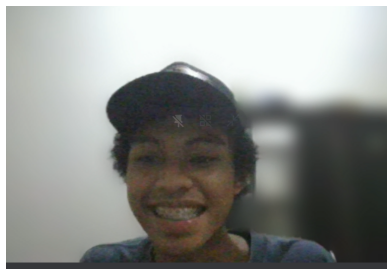


LAPORAN TUGAS BESAR 3
IF2211 STRATEGI ALGORITMA
Penerapan String Matching dan Regular Expression
dalam DNA Pattern Matching



Disusun oleh:

| | |
|-------------------------------|----------|
| Daniel Salim | 13520008 |
| Ken Kalang Al Qalyubi | 13520010 |
| Muhammad Akyas David Al Aleey | 13520011 |

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG
SEMESTER II TAHUN 2020/2021

DAFTAR ISI

| | |
|--|-----------|
| BAB 1 | 3 |
| DESKRIPSI TUGAS | 3 |
| 1.1 Latar Belakang | 3 |
| 1.2 Deskripsi Tugas | 3 |
| 1.3 Fitur-Fitur Aplikasi | 4 |
| 1.4 Spesifikasi Program | 7 |
| BAB 2 | 9 |
| LANDASAN TEORI | 9 |
| 2.1 Deskripsi singkat algoritma KMP, BM, dan Regex | 9 |
| 2.2 Penjelasan singkat mengenai Aplikasi Web yang dibangun | 12 |
| BAB 3 | 13 |
| ANALISIS PEMECAHAN MASALAH | 13 |
| 3.1 Langkah-langkah Pemecahan Masalah | 13 |
| 3.2 Fitur fungsional dan arsitektur aplikasi web yang dibangun | 14 |
| BAB 4 | 16 |
| IMPLEMENTASI DAN PENGUJIAN | 16 |
| 4.1 Spesifikasi Teknis Program | 16 |
| 4.2 Penjelasan dan Tata Cara Penggunaan Program | 18 |
| 4.3 Hasil Pengujian | 19 |
| 4.4 Analisis Hasil Pengujian | 21 |
| BAB 5 | 22 |
| KESIMPULAN DAN SARAN | 22 |
| 5.1 Kesimpulan | 22 |
| 5.2 Saran | 22 |
| DAFTAR PUSTAKA | 23 |
| LAMPIRAN | 24 |
| Repository Github | 24 |
| Video Demonstrasi | 24 |

BAB 1

DESKRIPSI TUGAS

1.1 Latar Belakang

Manusia umumnya memiliki 46 kromosom di dalam setiap selnya. Kromosom-kromosom tersebut tersusun dari DNA (deoxyribonucleic acid) atau asam deoksiribonukleat. DNA tersusun atas dua zat basa purin, yaitu Adenin (A) dan Guanin (G), serta dua zat basa pirimidin, yaitu sitosin (C) dan timin (T). Masing-masing purin akan berikatan dengan satu pirimidin. DNA merupakan materi genetik yang menentukan sifat dan karakteristik seseorang, seperti warna kulit, mata, rambut, dan bentuk wajah. Ketika seseorang memiliki kelainan genetik atau DNA, misalnya karena penyakit keturunan atau karena faktor lainnya, ia bisa mengalami penyakit tertentu. Oleh karena itu, tes DNA penting untuk dilakukan untuk mengetahui struktur genetik di dalam tubuh seseorang serta mendeteksi kelainan genetik. Ada berbagai jenis tes DNA yang dapat dilakukan, seperti uji pra implantasi, uji pra kelahiran, uji pembawa atau carrier testing, uji forensik, dan DNA sequence analysis.

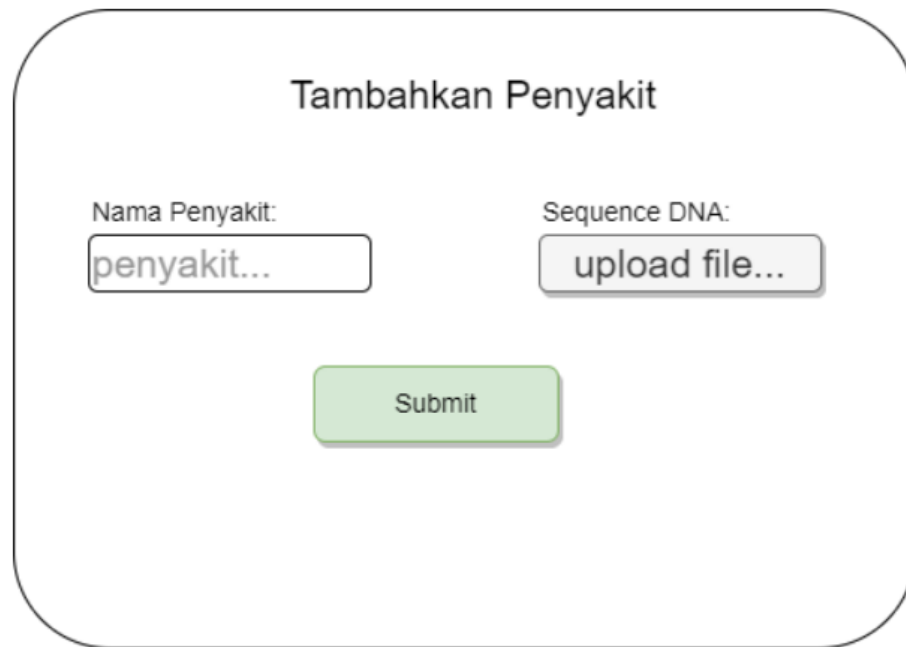
Salah satu jenis tes DNA yang sangat berkaitan dengan dunia bioinformatika adalah DNA sequence analysis. DNA sequence analysis adalah sebuah cara yang dapat digunakan untuk memprediksi berbagai macam penyakit yang tersimpan pada database berdasarkan urutan sekuens DNA-nya. Sebuah sekuens DNA adalah suatu representasi string of nucleotides yang disimpan pada suatu rantai DNA, sebagai contoh: ATTCGTAAGTAAAGTTA. Teknik pattern matching memegang peranan penting untuk dapat menganalisis sekuens DNA yang sangat panjang dalam waktu singkat. Oleh karena itu, mahasiswa Teknik Informatika berniat untuk membuat suatu aplikasi web berupa DNA Sequence Matching yang menerapkan algoritma String Matching dan Regular Expression untuk membantu penyedia jasa kesehatan dalam memprediksi penyakit pasien. Hasil prediksi juga dapat ditampilkan dalam tabel dan dilengkapi dengan kolom pencarian untuk membantu admin dalam melakukan filtering dan pencarian.

1.2 Deskripsi Tugas

Dalam tugas besar ini, anda diminta untuk membangun sebuah aplikasi DNA Pattern Matching. Dengan memanfaatkan algoritma String Matching dan Regular Expression yang telah anda pelajari di kelas IF2211 Strategi Algoritma, anda diharapkan dapat membangun sebuah aplikasi interaktif untuk mendeteksi apakah seorang pasien mempunyai penyakit genetik tertentu. Hasil prediksi tersebut dapat disimpan pada basis data untuk kemudian dapat ditampilkan berdasarkan query pencarian.

1.3 Fitur-Fitur Aplikasi

1. Aplikasi dapat menerima input penyakit baru berupa nama penyakit dan sequence DNA-nya (dan dimasukkan ke dalam database).
 - a. Implementasi input sequence DNA dalam bentuk file.
 - b. Dilakukan sanitasi input menggunakan regex untuk memastikan bahwa masukan merupakan sequence DNA yang valid (tidak boleh ada huruf kecil, tidak boleh ada huruf selain AGCT, dan tidak ada spasi).
 - c. Contoh input penyakit:



The image shows a web form titled "Tambahkan Penyakit" (Add Disease). It contains two input fields: "Nama Penyakit:" (Disease Name) with a placeholder text "penyakit..." and "Sequence DNA:" with a placeholder text "upload file...". Below these fields is a green "Submit" button.

Gambar 1. Ilustrasi Input Penyakit

2. Aplikasi dapat memprediksi seseorang menderita penyakit tertentu berdasarkan sequence DNA-nya.
 - a. Tes DNA dilakukan dengan menerima input nama pengguna, sequence DNA pengguna, dan nama penyakit yang diuji. Asumsi sequence DNA pengguna > sequence DNA penyakit.
 - b. Dilakukan sanitasi input menggunakan regex untuk memastikan bahwa masukan merupakan sequence DNA yang valid (tidak boleh ada huruf kecil, tidak boleh ada huruf selain AGCT, tidak ada spasi, dll).
 - c. Pencocokan sequence DNA dilakukan dengan menggunakan algoritma string matching.
 - d. Hasil dari tes DNA berupa tanggal tes, nama pengguna, nama penyakit yang diuji, dan status hasil tes. Contoh: 1 April 2022 - Mhs IF - HIV - False.
 - e. Semua komponen hasil tes ini dapat ditampilkan pada halaman web (refer ke poin 3 pada "Fitur-Fitur Aplikasi") dan disimpan pada sebuah tabel database.
 - f. Contoh tampilan web:

Gambar 2. Ilustrasi Prediksi

3. Aplikasi memiliki halaman yang menampilkan urutan hasil prediksi dengan kolom pencarian di dalamnya. Kolom pencarian bekerja sebagai filter dalam menampilkan hasil.
 - a. Kolom pencarian dapat menerima masukan dengan struktur: , contoh “13 April 2022 HIV”. Format penanggalan dibebaskan, jika bisa menerima >1 format lebih baik.
 - b. Kolom pencarian dapat menerima masukan hanya tanggal ataupun hanya nama penyakit. Fitur ini diimplementasikan menggunakan regex.
 - c. Contoh ilustrasi:
 - i. Masukan tanggal dan nama penyakit

Gambar 3. Ilustrasi Interaksi 1

ii. Masukan hanya tanggal

The screenshot shows a web form with a title "Masukan hanya tanggal". Below the title is a date input field containing "13 April 2022". Below this is a list of six data entries, each in a separate box:

1. 13 April 2022 - Fulan - Diabetes - True.
2. 13 April 2022 - Kamal - Sinusitis - False.
3. 13 April 2022 - Entah - Down Syndrome - False.
4. 13 April 2022 - Jamal - Polio - True.
5. 13 April 2022 - Yubai - TBC - True.
6. 13 April 2022 - Hika - Hepatitis A - False.

Gambar 4. Ilustrasi Interaksi 2

iii. Masukan hanya nama penyakit

The screenshot shows a web form with a title "Masukan hanya nama penyakit". Below the title is a disease input field containing "HIV". Below this is a list of six data entries, each in a separate box:

1. 13 April 2022 - Fulan - HIV - True.
2. 14 April 2022 - Kamal - HIV - False.
3. 15 April 2022 - Entah - HIV - False.
4. 16 April 2022 - Jamal - HIV - True.
5. 17 April 2022 - Yubai - HIV - True.
6. 18 April 2022 - Hika - HIV - False.

Gambar 5. Ilustrasi Interaksi 3

4. (Bonus) Menghitung tingkat kemiripan DNA pengguna dengan DNA penyakit pada tes DNA.
 - a. Ketika melakukan tes DNA, terdapat persentase kemiripan DNA dalam hasil tes. Contoh hasil tes: 1 April 2022 - Mhs IF - HIV - 75% - False.
 - b. Perhitungan tingkat kemiripan dapat dilakukan dengan menggunakan Hamming distance, Levenshtein distance, LCS, atau algoritma lainnya (dapat dijelaskan dalam laporan).
 - c. Tingkat kemiripan DNA dengan nilai lebih dari atau sama dengan 80% dikategorikan sebagai True. Perlu diperhatikan mengimplementasikan atau tidak mengimplementasikan bonus ini tetap dilakukan pengecekan string matching terlebih dahulu.
 - d. Contoh tampilan:

Tes DNA

Nama Pengguna:

Sequence DNA:

Prediksi Penyakit:

Hasil Tes

<Tanggal> - <pengguna> - <penyakit> - <similarity> - <True/False>

Gambar 5. Ilustrasi Bonus

1.4 Spesifikasi Program

1. Aplikasi berbasis website dengan pembagian Frontend dan Backend yang jelas.
2. Implementasi Backend wajib menggunakan Node.js / Golang, sedangkan Frontend 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) wajib diimplementasikan pada sisi Backend aplikasi.
5. Informasi yang wajib disimpan pada basis data:
 - a. Jenis Penyakit:
 - Nama Penyakit.
 - Rantai DNA Penyusun.
 - b. Hasil Prediksi:
 - Tanggal prediksi
 - Nama pasien

- Penyakit prediksi
 - Status terprediksi
6. Jika mengerjakan bonus tingkat kemiripan DNA, simpan hasil tingkat kemiripan tersebut pada basis data.

BAB 2

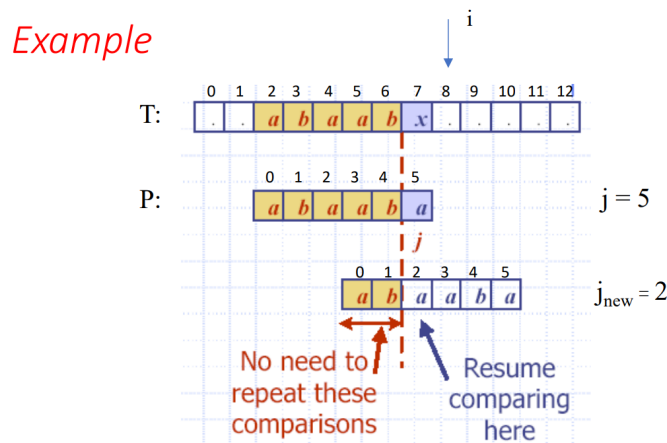
LANDASAN TEORI

2.1 Deskripsi singkat algoritma KMP, BM, dan Regex

String Matching adalah algoritma pencocokan string yang bekerja dengan mencari lokasi pertama suatu *pattern* dalam suatu teks. Misalkan pencocokan antara string pattern ABAC pada string teks ABADABACDE. Dengan algoritma string matching akan dikeluarkan hasil 4 (indeks ke-4) atau huruf ke-5. Dalam Tugas Besar ini, diimplementasikan dua algoritma string matching yaitu, Knuth-Morris-Pratt (KMP) dan Boyer-Moore.

1. Knuth-Morris-Pratt (KMP)

Algoritma KMP bekerja seperti algoritma Brute Force dalam string matching tetapi pergeseran *pattern* dalam proses pengecekan lebih efektif dan efisien dibandingkan Brute Force. Misalkan saat proses perbandingan *pattern* dengan *text*, ditemukan ketidaksesuaian pada indeks ke- j dari *pattern*. Untuk pergeseran *pattern*, sebelumnya dicek terlebih dahulu prefix dan suffix dari *pattern* tersebut, dimana prefix dilihat dari indeks 0 hingga $j-1$ sedangkan suffix pada indeks 1 hingga $j-1$. Pada pengecekan prefix dan suffix dari *pattern* tersebut, akan dicari pasangan prefix dan suffix yang sama dan terbesar / terpanjang. Setelah didapatkan, maka *pattern* akan diberikan pergeseran sebanyak j dikurangi dengan panjang prefix-suffix tersebut. Lalu setelah digeser, pengecekan dilakukan dimulai dari indeks ke- i dari *pattern*, dengan i adalah panjang prefix-suffix tersebut.



Gambar 2.1. Ilustrasi Algoritma KMP

Sumber :

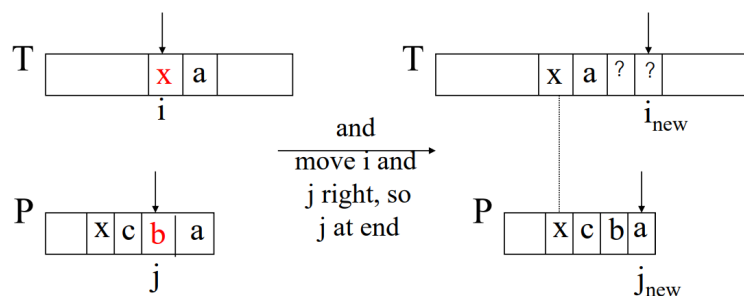
<https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Pencocokan-string-2021.pdf>

Mengapa hal ini dilakukan ? Karena saat mengambil prefix dan suffix yang sama, maka setelah pergeseran posisi prefix yang baru akan berada pada posisi suffix yang lama sehingga tidak perlu dicek lagi karena mereka adalah hal yang sama dan sudah sesuai dengan *text*.

2. Boyer-Moore(BM)

Proses *string matching* pada algoritma BM didasarkan pada 2 teknik, yaitu *The looking-glass technique* dan *The character-jump technique*. Pada teknik *The looking-glass*, perbandingan pada *pattern* dilakukan dimulai dari posisi paling belakang dan maju hingga posisi awal *pattern* (apabila sudah tepat). Pada teknik *The character-jump*, apabila terdapat ketidaksesuaian pada *pattern* dan *text* dengan karakter pada indeks ke-*i* pada *text* adalah *x* dan karakter pada indeks ke-*j* dari *pattern* adalah tidak sama dengan *x*, maka terdapat 3 kasus pergeseran *pattern* yang dilakukan satu persatu dan berurut.

Pada kasus 1, apabila pada *pattern* setelah indeks ke-*j* terdapat *x* maka *pattern* akan digeser hingga huruf *x* terkiri (apabila lebih dari satu) sejajar dengan *x* pada indeks ke-*i* dari *text*.

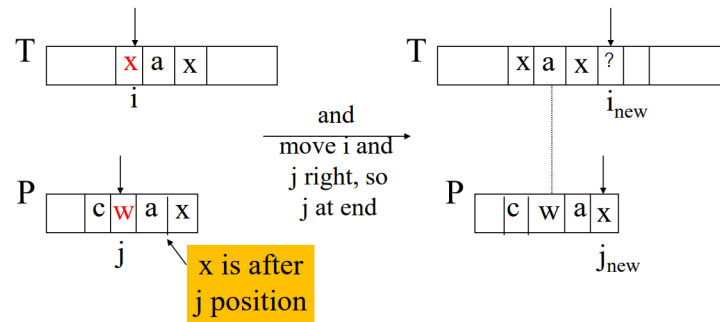


Gambar 2.2. Ilustrasi case 1

Sumber :

<https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Pencocokan-string-2021.pdf>

Pada kasus 2, apabila pada *pattern* hanya terdapat x disebelah kanan dari indeks ke-j dari *pattern*(sudah dilewati / dicek), maka dilakukan pergeseran *pattern* hingga posisi indeks ke-j dari *pattern* sejajar dengan posisi indeks ke- i+1 dari *text*.

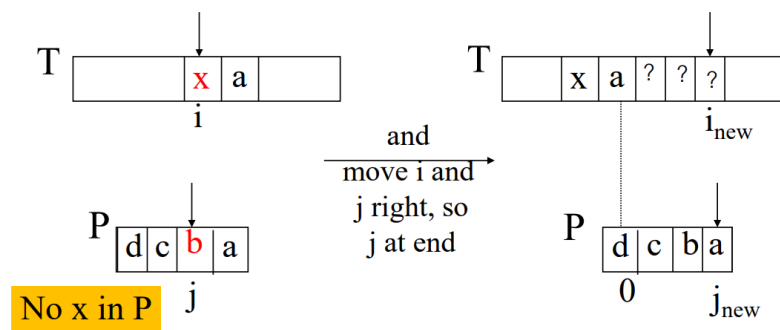


Gambar 2.3. Ilustrasi case 2

Sumber :

<https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Pencocokan-string-2021.pdf>

Pada kasus 3, apabila kasus 1 dan 2 tidak berlaku (tidak terdapat x pada *pattern*) maka geser *pattern* hingga posisi indeks ke-0 dari *pattern* sejajar dengan posisi indeks ke-i+1 dari *text*.



Gambar 2.4. Ilustrasi case 3

Sumber :

<https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Pencocokan-string-2021.pdf>

3. Regex

Regex dimanfaatkan dalam beberapa bagian yaitu pertama pada bagian input *sequence* DNA dan juga kolom pencarian riwayat. Pada bagian input *sequence* DNA, regex dimanfaatkan untuk melakukan sanitasi / pengecekan pada *sequence* dimana *sequence* DNA hanya bisa berisi karakter AGCT, dalam huruf besar dan tanpa spasi. Pada bagian kolom pencarian riwayat, regex digunakan untuk menampilkan hasil pencarian riwayat berdasarkan 3 input yang bisa diterima, yaitu hanya tanggal, hanya nama penyakit dan keduanya.

2.2 Penjelasan singkat mengenai Aplikasi Web yang dibangun

Dalam membangun Aplikasi Web pada Tubes ini, kami menggunakan ReactJs yaitu suatu pustaka atau *library* yang merupakan produk besutan Facebook. Teknologi ini dibuat khusus untuk membuat *user interface* pada website, bahkan dapat juga untuk *mobile*, dengan menggunakan React Native. Kebanyakan pengembang menyebutnya sebagai suatu kerangka, padahal aslinya React adalah *library* yang digunakan dalam pembuatan UI.

React menyajikan Virtual DOM yang diketahui cukup cepat. Para pengembang web saat ini banyak menggunakan React karena dapat membuat serta mendesain tampilan yang simple bagi tiap tingkatan pada web yang sedang dikembangkan. Ada beberapa fitur-fitur unggulan dari *library* ini, seperti komponen dapat dibuat secara *encapsulated*, sehingga dapat membuat rangkaian UI yang lain dengan dasar kemampuan komponen tersebut. Selain itu, fitur baru yang nantinya ingin ditambahkan juga dapat dengan mudah diaplikasikan tanpa harus mengganti kode sebelumnya, dimana React dapat bekerja menggunakan *library* Node JS ataupun *mobile apps* yang menggunakan React Native. Dengan dasar bahasanya yang menggunakan JavaScript, React dapat dipadukan dengan kerangka JavaScript lainnya, seperti AngularJS dan MVC.

BAB 3

ANALISIS PEMECAHAN MASALAH

3.1 Langkah-langkah Pemecahan Masalah

1. Menerima input penyakit baru berupa nama penyakit dan sequence DNA-nya (dan dimasukkan ke dalam database)
 - Menerima *input sequence* DNA dalam bentuk file.
 - Melakukan sanitasi untuk *input* menggunakan regex (melakukan pengecekan apakah *input* sudah benar, yaitu tidak ada huruf kecil, tidak ada huruf selain AGCT dan tidak ada spasi).
 - Apabila hasil sanitasi tidak sesuai dengan yang diinginkan, maka menampilkan pemberitahuan bahwa input ditolak.
 - Apabila lolos sanitasi, maka *input sequence* akan dimasukkan ke dalam *database* dengan nama penyakit sesuai dengan yang dimasukkan.
2. Memprediksi seseorang menderita penyakit tertentu berdasarkan sequence DNA-nya
 - Menerima *input* dari pengguna berupa nama pengguna, nama penyakit yang akan diuji, *input sequence* DNA yang akan diuji dalam bentuk file dan algoritma pengecekan yang diinginkan.
 - Dilakukan sanitasi untuk *input* menggunakan regex (sebagaimana halnya seperti pada nomor 1 - input penyakit baru).
 - Apabila hasil sanitasi tidak sesuai dengan yang diinginkan, maka menampilkan pemberitahuan bahwa input ditolak.
 - Apabila lolos sanitasi, maka *input sequence* akan diuji menggunakan algoritma *string matching* yang dipilih terhadap *sequence* DNA penyakit pada database yang sudah ditentukan pengguna untuk diuji. Algoritma *string matching* yang dapat dipilih adalah algoritma Knuth-Morris-Pratt (KMP) dan Boyer-Moore (BM).
 - Setelah dilakukan pencocokan, maka aplikasi akan mengeluarkan *output* kepada pengguna berupa rangkaian kalimat yang terdiri atas ‘tanggal tes - nama pengguna - penyakit yang diuji - hasil tes’, hasil tes akan berupa *True / False* berdasarkan hasil dari pencocokan.

- *Output* dari aplikasi juga memberikan tingkat kemiripan DNA pengguna dengan DNA penyakit yang juga bisa mempengaruhi *output* pada bagian hasil tes (**Bonus dikerjakan**), lebih lanjut dapat dilihat pada **nomor 4** di bawah.
 - *Output* yang dikeluarkan juga akan disimpan pada *database*.
3. Memiliki halaman yang menampilkan urutan hasil prediksi dengan kolom pencarian di dalamnya. Kolom pencarian bekerja sebagai filter dalam menampilkan hasil.
- Mengecek *input* dari pengguna menggunakan regex untuk mencari dan menampilkan hasil yang sesuai dengan pencarian pada *database*.
 - Aplikasi dapat menerima *input* berupa nama penyakit dan tanggal tes, hanya nama penyakit, dan hanya tanggal tes.
 - Apabila hasil pencocokan dengan regex sesuai dengan salah satu dari 3 pola *input* di atas, maka aplikasi akan melakukan pencarian pada *database* berdasarkan input yang diterima dan menampilkannya.
 - Apabila *input* yang dimasukkan tidak terdapat pada *database* atau input tidak sesuai dengan pencocokan regex, maka aplikasi tidak akan menampilkan apa-apa

3.2 Fitur fungsional dan arsitektur aplikasi web yang dibangun

1. Fitur Fungsional dari aplikasi web yang dibangun
 - Menerima *input* penyakit baru dan memasukkannya ke dalam *database*
 - Melakukan pengecekan *input sequence* DNA (sanitasi)
 - Melakukan prediksi penyakit seseorang berdasarkan *input sequence* DNA-nya dengan algoritma *string matching* dengan pertimbangan persentase kemiripan (bonus)
 - Dalam tes prediksi dapat memilih algoritma *string matching* yang ingin digunakan
 - Menyimpan seluruh hasil tes ke dalam *database*
 - Menampilkan hasil pencarian dari hasil tes yang disimpan pada *database* berdasarkan *input* dari pengguna (3 jenis : nama penyakit saja, tanggal saja, atau keduanya)

2. Arsitektur aplikasi web yang dibangun

- **Frontend**

Frontend dari aplikasi menggunakan library ReactJS

- **Backend**

Backend dari aplikasi dibangun dengan Node.js, untuk keperluan DBMS (*Database Management System*) digunakan cloud Atlas dari MongoDB.

BAB 4

IMPLEMENTASI DAN PENGUJIAN

4.1 Spesifikasi Teknis Program

Pada program ini, program dijalankan dengan dua terminal yang berbeda dengan terminal pertama sebagai backend sekaligus baseURL untuk menjalankan fungsi pada frontend sebagai terminal kedua.

Tata cara pengguna

- Struktur data

Struktur data yang digunakan pada pengerjaan tugas besar ini ada beberapa yaitu bentuk *struct* pada DBMS MongoDB seperti berikut :

1. Pada *struct* cekDna

```
{
    Pengguna string `json:"nama" binding:"required"`
    Dna       string `json:"dna" binding:"required"`
    Sakit     string `json:"sakit" binding:"required"`
    Tanggal   string `json:"date" binding:"required"`
    Status    string `json:"status"`
}
```

2. Pada *struct* pencarian

```
{
    Tanggal string `json:"date"`
    Nama     string `json:"nama"`
    Sakit    string `json:"sakit"`
    Status   string `json:"status"`
}
```

3. Pada *struct* Penyakit

```
{
    Penyakit string `json:"penyakit"`
    Sequence string `json:"sequence"`
}
```


- Fungsi dan Program yang dibangun

| No | Fungsi dan Prosedur | Keterangan |
|----|---------------------------------------|--|
| 1. | func postHandler(c echo.Context) | Fungsi untuk menerima masukan berupa nama, dna, sakit, date, dan boolean kmp dari frontend, kemudian memproses masukan menggunakan fungsi string matching baik dengan kmp maupun booyer moore. Kemudian fungsi akan merespons data dengan format JSON. |
| 2. | func addPenyakit(c echo.Context) | Menerima masukan dari frontend kemudian mengupload data masukan ke dalam database |
| 3. | func pencarianHandler(c echo.Context) | Menerima masukan data sesuai frontend kemudian mencari data yang sesuai pada database, lalu merespons kepada frontend data yang didapat dari database. |
| 4. | KMP(pattern, text) | Algoritma <i>string matching</i> Knuth-Morris-Pratt yang akan mencari dan mengembalikan posisi pattern pada text, apabila tidak ada maka mengembalikan -1 |
| 5. | BorderFunction(pattern, M, startIdx) | Mengolah pattern untuk menyimpan ukuran prefix terbesar yang juga merupakan suffixnya terhadap pattern itu sendiri untuk tiap posisi kemungkinan ketidaksamaan. Disimpan pada sebuah array |
| 6. | BooyerMoore(pattern, text) | Algoritma Pencarian Booyer-Moore yang akan mencari dan mengembalikan true jika pattern berhasil ditemukan dalam text, apabila tidak ada maka mengembalikan false |
| 7. | GetLastOccurence(pattern) | Algoritma Booyer-Moore memproses terlebih dahulu pattern dan mengembalikan index okurensi terakhir jika ditemukan atau mengembalikan -1 jika tidak ditemukan |

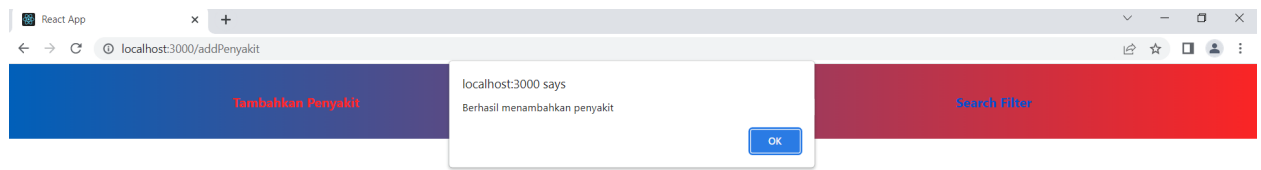
| | | |
|----|-------------------|---|
| 8. | DNAValidator(dna) | Fungsi untuk memproses menyanitasi input DNA pada saat sebelum memasuki proses pencarian dengan menggunakan Regex |
|----|-------------------|---|

4.2 Penjelasan dan Tata Cara Penggunaan Program

Langkah-langkah untuk menjalankan program yang kami buat :

1. Siapkan dua terminal untuk menjalankan backend dan juga frontend.
2. Setelah dijalankan lakukan input yang tersedia pada page frontend.
3. Pada halaman ini, pengguna diminta mengisi form yang ada untuk nama pengguna, *input file sequence* DNA, nama penyakit yang ingin diprediksi dan pilihan algoritma pengecekan yang diinginkan.
4. Apabila *sequence* DNA yang diberikan oleh pengguna sudah tepat format isinya (lolos hasil regex) dan menekan tombol submit, maka program akan menampilkan hasil prediksi dari *sequence DNA* yang diinput beserta tingkat kemiripannya.
5. Apabila *sequence* DNA yang diberikan oleh pengguna tidak lulus regex maka program akan mengeluarkan peringatan bahwa *input* ditolak.
6. Pengguna juga dapat mengakses page untuk menambah penyakit dimana pengguna dapat memasukkan nama dan *sequence* DNA dari sebuah penyakit yang akan dimasukkan ke dalam *database*.
7. Page berikutnya adalah page untuk mencari riwayat hasil tes prediksi yang sudah disimpan pada *database*, untuk pencarian pengguna dapat mencari dengan tiga jenis *keyword* yaitu tanggal saja, nama penyakit saja dan tanggal beserta nama penyakit. Program hanya akan menampilkan riwayat berdasarkan ketiga jenis *keyword* tersebut.

4.3 Hasil Pengujian



Tambahkan Penyakit

Nama Penyakit :

HIV

Sequence DNA :

Choose File testcase.txt

Submit



Menambahkan penyakit dengan nama HIV dan sequence dna dengan file txt, data berhasil ditambahkan pada database.

TES DNA

Nama Pengguna :

andi

Sequence DNA :

Choose File testcase.txt

Prediksi Penyakit :

HIV

KMP



Submit

Hasil

29/4/2022 - andi - HIV - True - AGTCGTA

Test DNA dengan algoritma KMP hasil true jika sequence sama

TES DNA

| | | |
|-----------------------------------|---|----------------------------------|
| Nama Pengguna : | Sequence DNA : | Prediksi Penyakit : |
| <input type="text" value="andi"/> | <input type="text" value="Choose File"/> testcase.txt | <input type="text" value="HIV"/> |
| | KMP | |
| | <input checked="" type="checkbox"/> | |
| | <input type="button" value="Submit"/> | |
| | Hasil | |

29/4/2022 - andi - HIV - False - AGTCGTC

Test DNA dengan algoritma KMP dengan hasil False karena sequence dna berbeda

Cari Hasil

| | | |
|---|----------------------|---------------------------------------|
| Masukkan Tanggal: | Masukkan Penyakit : | |
| <input type="text" value="2022-04-29"/> | <input type="text"/> | <input type="button" value="Submit"/> |
| *Format (YYYY-MM-DD) | | |

Hasil Pencarian

2022-04-29-alqalyubi-diabetes-False
2022-04-29-alqalyubi-diabetes-False
2022-04-29-alqalyubi-sariawan-True
2022-04-29-daniel-sariawan-True
2022-04-29-akyas-diabetes-False
2022-04-29-andi-HIV-True
2022-04-29-andi-HIV-True

Pencarian dengan masukkan tanggal

Cari Hasil

Masukkan Tanggal:

*Format (YYYY-MM-DD)

Masukkan Penyakit :

Submit

Hasil Pencarian

2022-04-29-andi-HIV-True

2022-04-29-andi-HIV-True

Pencarian dengan masukan penyakit

4.4 Analisis Hasil Pengujian

Berdasarkan hasil pengujian, seluruh permasalahan dan fitur yang dibutuhkan dapat diselesaikan dan mengimplementasikan. Aplikasi dapat menerima input nama dan *sequence* DNA penyakit yang selanjutnya akan dimasukkan ke dalam database. Selain itu, aplikasi ini juga dapat melakukan sanitasi *input sequence* DNA untuk memastikan format string tersebut valid. Aplikasi dapat melakukan pengetesan DNA terhadap suatu penyakit yang pengguna masukkan. Aplikasi ini mengimplementasikan algoritma *string matching* berupa Knuth-Morris-Pratt (KMP) dan Boyer-Moore(BM). Aplikasi ini juga akan mencatat seluruh tes DNA pengguna terhadap suatu penyakit ke dalam *database* dan dapat ditampilkan pada search filter kepada pengguna berdasarkan *input* pencarian yang diinginkan pengguna.

BAB 5

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Pada Tugas Besar kali ini, kami memperoleh pengetahuan bahwa algoritma pencocokan string dengan metode Knuth-Morris-Pratt (KMP) dan Boyer-Moore(BM) dapat dengan efektif dan efisien digunakan untuk pembuatan suatu program terkait DNA *pattern matching*. Algoritma tersebut dirasa cocok untuk menyelesaikan persoalan ini karena *sequence* DNA sendiri tersusun dari 4 buah huruf yang penyusunannya bervariasi. Program ini juga mengimplementasikan algoritma regex yang melakukan proses sanitasi input berupa masukan tanggal, nama penyakit, dan *sequence* DNA untuk memastikan bahwa masukan tersebut valid.

5.2 Saran

Saran yang dapat kami berikan terkait keberjalanan tugas kedepannya yaitu membuat tampilan website menjadi lebih menarik, responsif, dan kalau bisa di deploy. Selain itu, pengerjaan tugas sebaiknya dimulai sejak awal agar tidak terdapat kekurangan yang diakibatkan oleh keterbatasan waktu.

DAFTAR PUSTAKA

Munir, Rinaldi. (2022). Pencocokan String.

<https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Pencocokan-string-2021.pdf>

(Materi string matching Booyer Moore dan KMP)

Pencocokan string dengan Regular Expression (Regex), Rinaldi Munir

<https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2018-2019/String-Matching-dengan-Regex-2019.pdf>

LAMPIRAN

Repository Github

https://github.com/kenkalang/Tubes3_13520008