

Laporan Tugas Besar III
IF2211 Strategi Algoritma
Penerapan String Matching dan Regular Expression dalam
DNA Pattern Matching

Oleh:

13520032 - Fadil Fauzani

13520052 - Gregorius Moses Marevson

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG
SEMESTER II 2021/2022

Daftar Isi

Daftar Isi	2
BAB I	3
Deskripsi Tugas	3
BAB II	9
Landasan Teori	9
BAB III	11
Analisis Pemecahan Masalah	11
BAB IV	13
Implementasi dan pengujian	13
BAB V	17
Kesimpulan dan Saran	17
Daftar Pustaka	18

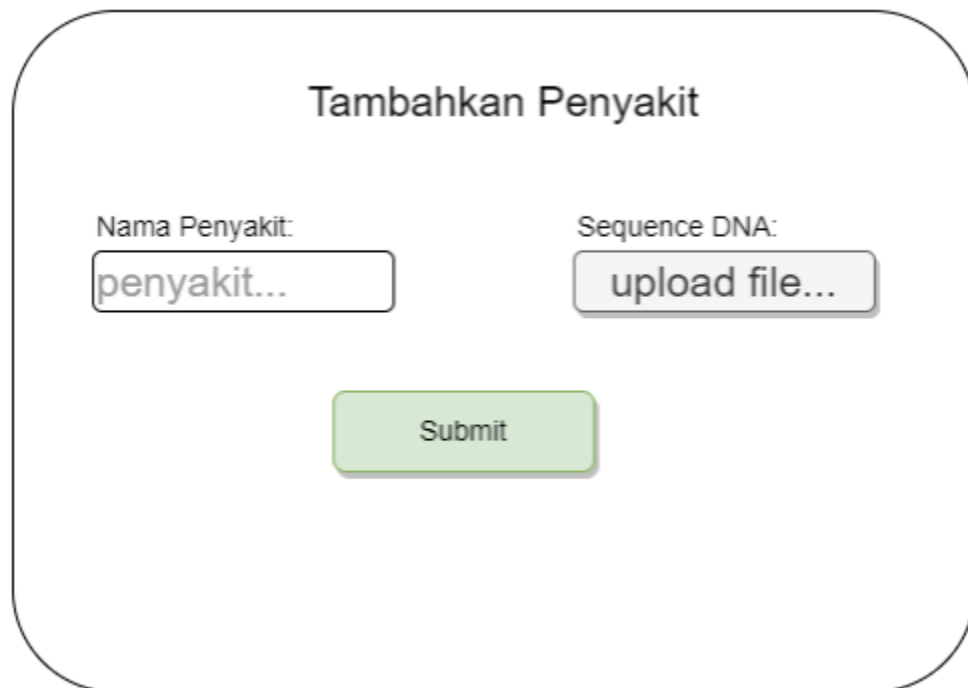
BAB I

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.

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 2. 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:

Tes DNA

Nama Pengguna:

Sequence DNA:

Prediksi Penyakit:

Hasil Tes

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

Gambar 3. 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

13 April 2022 HIV

1. 13 April 2022 - Fulan - HIV - True.

2. 13 April 2022 - Kamal - HIV - False.

3. 13 April 2022 - Entah - HIV - False.

4. 13 April 2022 - Jamal - HIV - True.

5. 13 April 2022 - Yubai - HIV - True.

6. 13 April 2022 - Hika - HIV - False.

Gambar 4. Ilustrasi Interaksi 1

ii. Masukan hanya tanggal

13 April 2022

1. 13 April 2022 - Fulan - Diabetes - True.

2. 13 April 2022 - Kamal - Sinusitis - False.

3. 13 April 2022 - Entah - Down Syndrome - False.

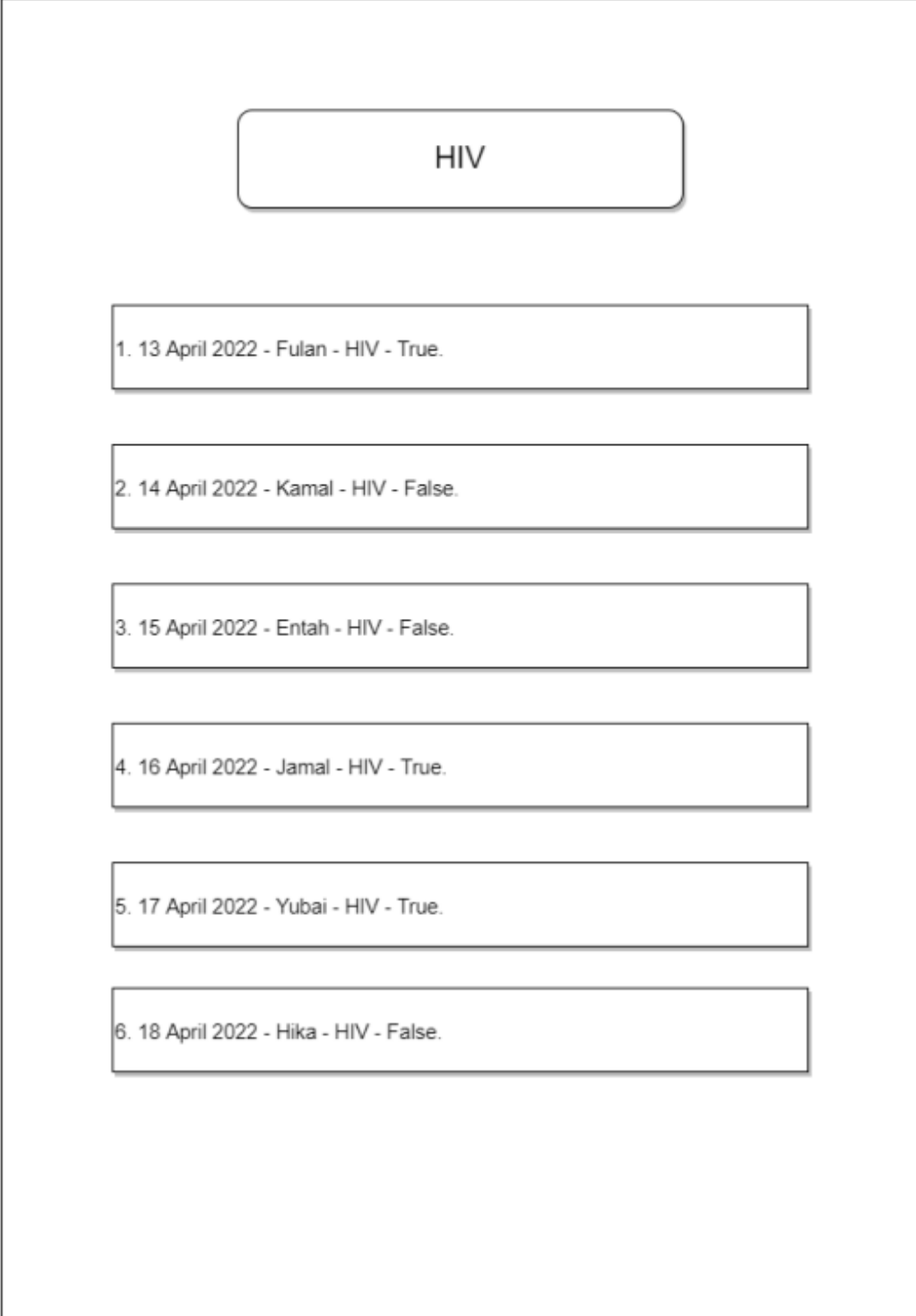
4. 13 April 2022 - Jamal - Polio - True.

5. 13 April 2022 - Yubai - TBC - True.

6. 13 April 2022 - Hika - Hepatitis A - False.

Gambar 5. Ilustrasi Interaksi 2

iii. Masukan hanya nama penyakit



The screenshot shows a web application interface. At the top, there is a rounded rectangular button labeled "HIV". Below this button, there is a list of six test results, each displayed in a rectangular box. The results are as follows:

- 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 6. 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:

Spesifikasi Program:

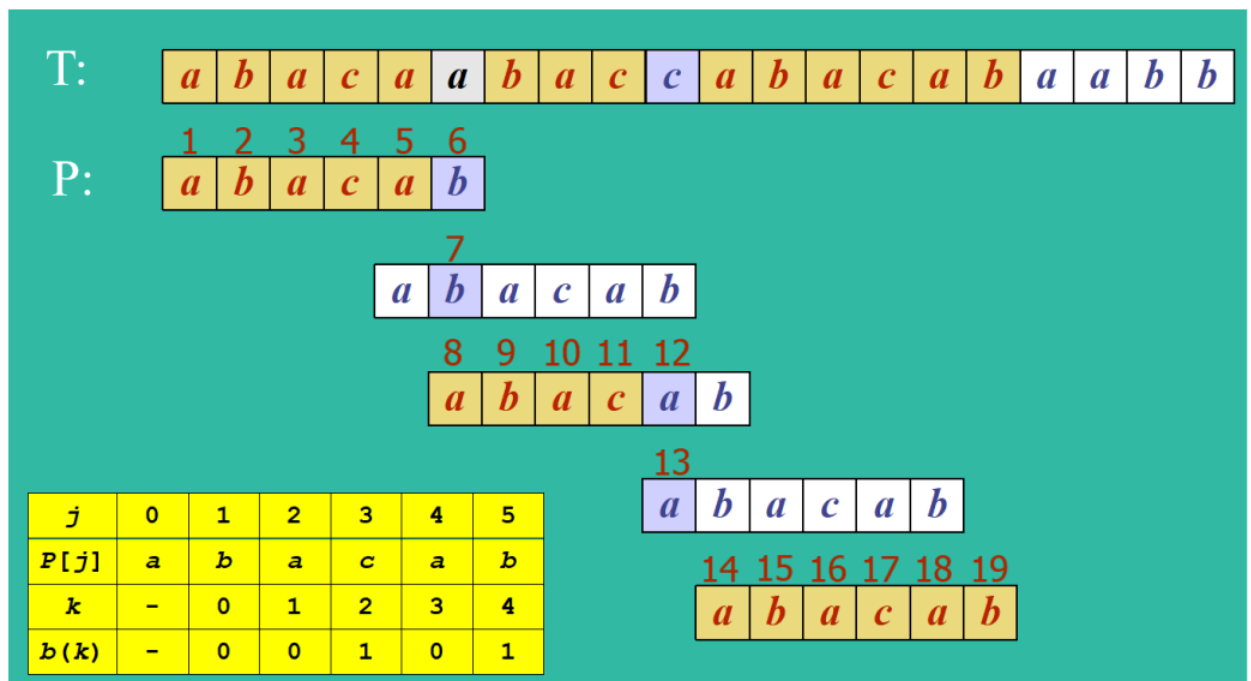
1. Aplikasi berbasis website dengan pembagian Frontend dan Backend yang jelas.
2. Implementasi Backend wajib menggunakan Node.js / Golang, sedangkan Frontend disarankan untuk menggunakan React / Next.js / Vue / Angular. Lihat referensi untuk selengkapnya.
3. Penyimpanan data wajib menggunakan basis data (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 II

Landasan Teori

2.1. Knuth-Morris-Pratt Algorithm (KMP Algorithm)

Pengecekan String dengan KMP mirip brute-force bedanya perpindahan pengecekan dilakukan berdasarkan tabel pergeseran yang telah ditentukan berdasarkan pattern yang dicari. Tabel pergeseran dibuat berdasarkan panjang prefix terbesar pattern[0..k] yang juga merupakan suffix untuk pattern [1..k] dengan k indeks pada tabel pergeseran.



Gambar Contoh penggunaan KMP Algorithm

Penjelasan:

1. Pada pengetesan pertama ditemukan ketidakcocokan pada pattern[5] dan T[5] (mulai dari 0). Maka indeks pada T tidak bergeser, yang bergeser indeks pengecekan pada pattern yaitu ke $b(5) = 1$. Maka untuk pengecekan selanjutnya dimulai pada T[5] dan P[1]
2. Pada Pengecekan selanjutnya, ditemukan ketidakcocokan pada T[9] dan P[4] maka indeks pengecekan P bergeser ke $b(4) = 0$. Maka pengecekan selanjutnya dimulai pada T[9] dan P[0]
3. Pada Pengecekan selanjutnya, ditemukan ketidakcocokan pada T[13] dan P[7] karena $b(7)$ tidak terdefinisi maka dimulai ke indeks 0. Maka pengecekan selanjutnya dimulai pada T[13] dan P[0]
4. Pada pengecekan selanjutnya, tidak ditemukan ketidakcocokan hingga P[5] maka pattern ditemukan pada Text.

2.2. Boyer-Moore Algorithm

Boyer-Moore Algorithm adalah algoritma pencarian string yang didasarkan pada 2 teknik, yaitu: looking-glass technique dan character-jump technique. Looking-glass technique artinya mencari Pattern pada Text dengan cara bergerak mundur mulai dari ujung belakang Pattern. Character-jump technique adalah teknik yang dilakukan ketika dijumpai ketidaksesuaian karakter. Misalkan terjadi mismatch pada karakter 'X' dalam Teks. Terdapat 3 kemungkinan kasus pada character-jump technique yaitu:

1. Jika karakter 'X' terdapat di sebelah kiri Pattern yang sudah dibaca. Geser Pattern ke kanan sehingga karakter 'X' pada Pattern sejajar dengan Text.
2. Jika karakter 'X' hanya terdapat di sebelah kanan Pattern yang sudah dibaca. Geser Pattern satu karakter ke kanan.
3. Jika karakter 'X' tidak terdapat dalam Pattern. Geser Pattern melewati karakter 'X' pada teks tersebut.

Algoritma ini cocok dipergunakan untuk pencarian dengan alphabet besar, tetapi lambat apabila dipergunakan untuk alphabet yang kecil

2.3. Regular Expression

Regular Expression (Regex) adalah sekuens karakter yang menspesifikasikan suatu pola pencarian dari suatu teks. Regular Expression biasanya digunakan pada algoritma pencarian string, seperti untuk melakukan pencarian pattern ataupun validasi input. Fitur pencarian dengan regex, sudah banyak disupport oleh bahasa pemrograman multi-purpose seperti: python, C, C++, Java, dan Javascript

2.4. PostgreSQL

PostgreSQL (dikenal juga dengan Postgres) adalah Relational Database Management System (RDBMS) open source yang menekankan ekstensibilitas dan kesesuaian dengan SQL. Awalnya dinamai POSTGRES yang menunjukkan asalnya dari Ingres. Pada tahun 1996, proyek ini dinamai PostgreSQL yang menunjukkan support terhadap SQL.

2.5. NPM

npm (Numpy Package Manager) adalah package manager untuk bahasa pemrograman Javascript. npm adalah package manager default untuk Javascript runtime environment (Node.js). npm terdiri atas command line client (npm) dan database online berisi package public maupun private-berbayar (disebut npm registry). Registry diakses via client dan package yang tersedia bisa dicari melalui website npm (<https://www.npmjs.com/>). Package manager dan registry dikelola oleh npm, Inc.

BAB III

Analisis Pemecahan Masalah

3.1. Langkah Penyelesaian masalah Fitur 1

Untuk menyelesaikan permasalahan yang telah disebutkan di deskripsi persoalan, dilakukan beberapa langkah sebagai berikut:

1. Membuat form yang berisi textbox untuk mengisi nama penyakit dan sebuah tombol upload untuk mengupload file berupa .txt
2. Membuat tombol submit yang akan mensubmit isi form dan memproses input tersebut
3. Dalam prosesnya, perlu ada validasi isi filenya apakah sesuai dengan DNA (hanya berisi A T C G)
4. Jika Valid, diprosesnya akan melakukan penambahan data pada database sesuai data input
5. Jika proses sudah selesai, maka pada tampilan diberikan pemberitahuan tentang status proses (berhasil atau tidak).

3.2. Langkah Penyelesaian masalah Fitur 2

Untuk menyelesaikan permasalahan yang telah disebutkan di deskripsi persoalan, dilakukan beberapa langkah sebagai berikut:

1. Membuat form yang berisi textbox untuk mengisi nama pengguna, sebuah dropdown untuk memilih penyakit mana yang akan dites, dan sebuah tombol upload untuk mengupload file berupa .txt yang akan menjadi DNA pengguna
2. Membuat tombol submit yang akan mensubmit isi form dan memproses input tersebut
3. Dalam prosesnya, perlu ada validasi isi filenya apakah sesuai dengan DNA (hanya berisi A T C G)
4. Jika Valid, diprosesnya akan melakukan penambahan data pada database sesuai data input
5. Jika proses sudah selesai, maka pada tampilan diberikan pemberitahuan tentang status proses (berhasil atau tidak). Dan menampilkan hasil test ke layar

3.3. Langkah Penyelesaian masalah Fitur 3

Untuk menyelesaikan permasalahan yang telah disebutkan di deskripsi persoalan, dilakukan beberapa langkah sebagai berikut:

1. Membuat form yang berisi textbox yang akan menjadi acuan pencarian
2. Setiap ada perubahan di textbox tersebut maka akan terjadi perubahan pada data (karena query nya berubah). Karena itu diperlukan proses meload data yang akan di-invoke setiap textbox diubah isinya.

3. Proses tersebut akan mengambil tanggal jika sesuai dengan format penanggalan. Setelah itu proses akan mengambil data yang tanggalnya sesuai (jika ada) dan memiliki nama mirip dengan isi textbox yang bukan tanggal.
4. Setelah proses selesai, hasil pencarian akan ditampilkan.

BAB IV

Implementasi dan pengujian

4.1. Spesifikasi Teknis Program

Program dibuat dengan menggunakan React.js sebagai Front-end dan Express.js sebagai Back-end. Pada Front-end terdapat beberapa komponen yaitu:

1. Form
Berisi halaman untuk menginput test DNA.
2. Formpenyakit
Berisi halaman untuk menginput penyakit baru.
3. History
Berisi halaman untuk melihat semua history test DNA.

Pada Backend terdapat beberapa Request Method, yaitu

1. GET /penyakit
Untuk mengambil semua list penyakit pada database
2. POST /penyakit
Untuk menambah data penyakit ke database
3. POST /penyakit-hist
Untuk menambah data test DNA ke database
4. GET /penyakit-hist
Untuk mengambil semua list penyakit pada database
5. POST /penyakit-histo
Untuk mengambil list penyakit pada database yang sesuai dengan request

Pada Backend juga terdapat fungsi tambahan untuk membantu pemrosesan data, yaitu

1. Getter
Untuk melakukan pencarian pada database
2. Kmp
Untuk melakukan test KMP Algorithm
3. Bya
Untuk melakukan test Boyer-Moore Algorithm
4. String
Untuk melakukan pencocokan pada string
5. regex
Untuk melakukan pencocokan regex pada string

4.2. Tata Cara Penggunaan Program

Untuk menjalankan program ini pastikan untuk menginstall Node Packet Manager (NPM) atau alternatifnya dan PostgreSQL. Disini akan dicontohkan dalam NPM.

Start Server Backend:

1. Ganti directory ke directory back-end, dengan command “cd be” pada terminal
2. Install Dependencies pada back-end dengan command “npm install” pada terminal
3. Jika sudah, bisa start server dengan command “npm start” pada terminal

Start Server Frontend:

1. Ganti directory ke directory front-end, dengan command “cd fe” pada terminal
2. Install Dependencies pada front-end dengan command “npm install” pada terminal
3. Jika sudah, bisa start server dengan command “npm start” pada terminal

Start server database pada PostgreSQL:

1. install PostgreSQL
2. setup password asdfgh jika diganti maka ganti password di be/routes/database.js
3. buat database data_stima jika diganti maka ganti database di be/routes/database.js
4. buat table dengan query berikut

```
CREATE TABLE penyakit( id serial PRIMARY KEY, nama VARCHAR (255), rantai
text );
CREATE TABLE HISTORY_DNA( id serial PRIMARY KEY, nama_pengguna
VARCHAR (255) NOT NULL, nama_penyakit VARCHAR (255) NOT NULL,
tanggal_test DATE DEFAULT CURRENT_DATE, hasil BOOLEAN NOT NULL );
```

5. pastikan user, host, database, password, dan port sesuai dengan konfigurasi di PostgreSQL anda!

4.3. Hasil Pengujian

Menguji Menambahkan Penyakit:

1. Mencoba menginput penyakit sakit kepala dengan rantai DNA =
“AAAAAAAAAAAAAAAAAAAAAAAAAAAAA”

- [Home](#)
- [Penyakit](#)
- [Test Penyakit](#)
- [History Test](#)

Name:

2. Data Berhasil ditambahkan

- [Home](#)
- [Penyakit](#)
- [Test Penyakit](#)
- [History Test](#)

Name:

localhost:3000 says
Berhasil

3. Penyakit bisa dipilih pada Tes penyakit

- Name:
- Penyakit**
- Select Penyakit
- Select Penyakit
- takut tambah dewasa
- malaria
- ganteng maksimal
- ganteng doang
- Sakit Kepala**

- Mencoba melakukan test penyakit pada fadil dengan rantai DNA = “ATGCGATTCTGTAAGTAAAGTTAATTCGTAACTAGTAAGTTAATTCGTAACTA GTAAGTTAATTCGTAACTAGTAAGTTAAA” pada penyakti ganteng maksimal
- Test berhasil dilakukan

- Name:

Penyakit

3. Data telah masuk ke history tes

-

29 April 2022-Fadil-ganteng maksimal-true

Menguji Searching pada History Test:

1. Melakukan pencarian tes dengan penyakit malaria

malaria

29 April 2022-fadil-malaria-true

29 April 2022-orang-malaria-true

4.4. Analisis Pengujian

Dari berbagai pengujian, proses Pattern matching menggunakan KMP atau pun Boyer-Moore selalu menghasilkan hasil yang sama, jadi dengan metode apapun jika benar maka hasilnya akan tetap sama. Jika pada fitur 1 dan 2 tidak mengupload file DNA maka program akan Hang. dan pada fitur 3 pencarian dilakukan dengan Case sensitive, jadi pencarian “Malaria” dan “malaria” akan menghasilkan list yang berbeda

BAB V

Kesimpulan dan Saran

5.1. Kesimpulan

Adapun kesimpulan dari pelaksanaan tugas besar ini adalah sebagai berikut:

1. Algoritma KMP dan Boyer-Moore cocok dipergunakan untuk pencarian string dalam jumlah yang besar
2. Regular Expression bisa dipergunakan dalam melakukan validasi input dan pencarian string berformat spesifik

5.2. Saran

Saran yang dapat kami berikan untuk tugas besar 3 IF2211 Strategi Algoritma Semester 2 2021/2022 meliputi:

1. Membuat Tampilan aplikasi menjadi lebih menarik dan bagus.
2. Membuat aplikasi bisa dipakai di internet (saat ini hanya bisa dipakai di localhost)

Daftar Pustaka

Munir, Rinaldi. "Pencocokan string (String matching/pattern matching)." *Informatika*, 2022, Diakses online dari <https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Pencocokan-string-2021.pdf>. Pada 27 Maret 2020.

Masayu Leylia Khodra. "Pencocokan string (String matching/pattern matching)." *Informatika*, 2019, Diakses online dari <https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2018-2019/String-Matching-dengan-Regex-2019.pdf>. Pada 27 Maret 2020.