

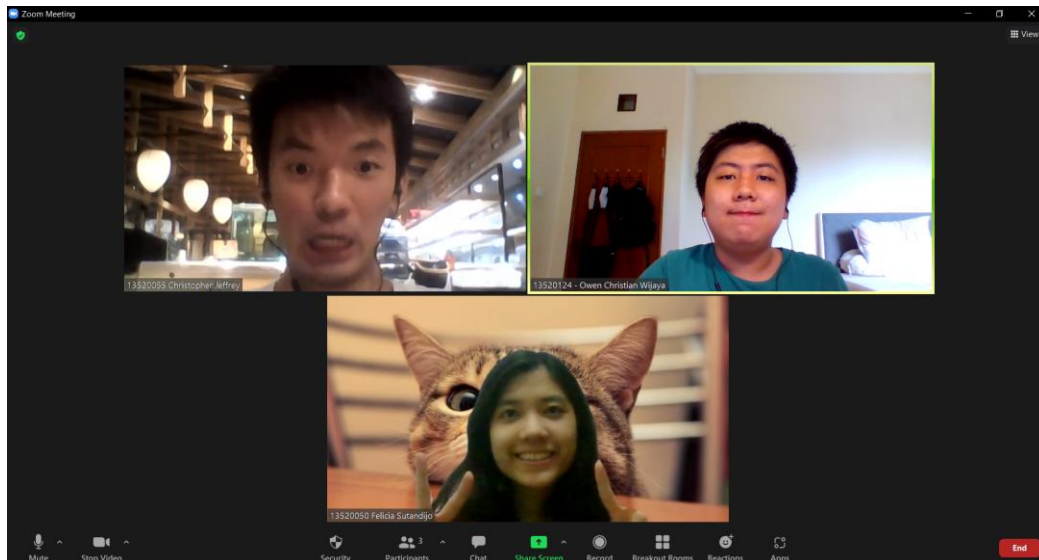
TUGAS BESAR III
Penerapan String Matching dan Regular Expression dalam DNA Pattern
Matching

LAPORAN

Diajukan sebagai salah satu tugas mata kuliah
IF2211 Strategi Algoritma pada Semester II
Tahun Akademik 2021 - 2022
DNA Pattern Matching

Oleh

Felicia Sutandijo	13520050
Christopher Jeffrey	13520055
Owen Christian Wijaya	13520124



PROGRAM STUDI TEKNIK INFORMATIKA
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG
BANDUNG
2022

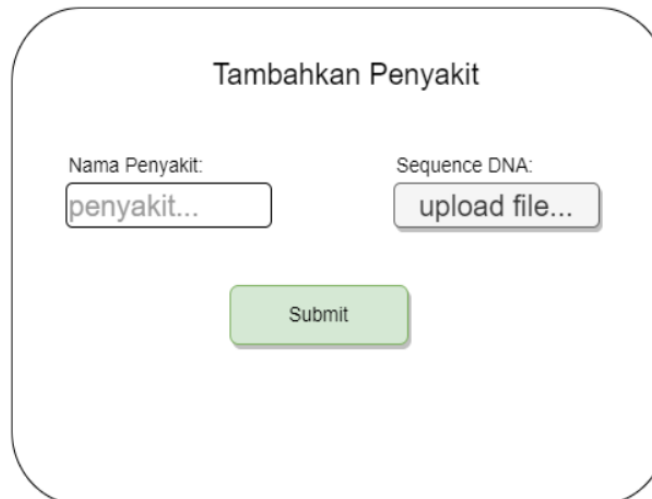
DAFTAR ISI

BAB 1: DESKRIPSI TUGAS	2
BAB 2: LANDASAN TEORI	6
2.1 Algoritma Knuth-Morris-Pratt	6
2.2 Algoritma Boyer-Moore	6
2.3 Regular Expression	7
2.4 Penjelasan Singkat tentang Aplikasi	8
BAB 3: ANALISIS PEMECAHAN MASALAH	8
3.1 Langkah Penyelesaian Masalah Setiap Fitur	8
3.2 Fitur Fungsional dan Arsitektur Web.....	9
BAB 4: IMPLEMENTASI DAN PENGUJIAN	10
4.1 Spesifikasi Teknis Program	10
4.1.1 Front End	10
4.1.2 Back End.....	11
4.2 Tata Cara Menggunakan Program	12
4.2.1 Cara Menggunakan Website	12
4.2.1.3 Pencarian Riwayat Pencocokan DNA.....	14
4.3 Hasil Pengujian	15
4.3.1 Pengujian Pencocokan DNA.....	15
4.3.2 Pengujian Input DNA.....	15
4.3.3 Pengujian Search Riwayat Pencocokan DNA.....	16
4.4 Analisis Hasil Pengujian	17
4.4.1 Pengujian Pencocokan DNA.....	17
4.4.2 Pengujian Input DNA.....	17
4.4.3 Pengujian Search Riwayat Pencocokan DNA.....	17
BAB 5: KESIMPULAN	18
5.1 Kesimpulan	18
5.2 Saran	19
5.3 Refleksi	19
LAMPIRAN.....	20
DAFTAR PUSTAKA	20

BAB 1: DESKRIPSI TUGAS

Dalam tugas besar ini, kami diminta untuk membangun sebuah aplikasi DNA Pattern Matching. Dengan memanfaatkan algoritma String Matching dan Regular Expression yang telah kami pelajari di kelas IF2211 Strategi Algoritma, kami 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. 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". It has two input fields: "Nama Penyakit:" with a placeholder text "penyakit..." and "Sequence DNA:" with a placeholder text "upload file...". Below these fields is a green "Submit" button.

Gambar 1.1. 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:

Tes DNA

Nama Pengguna:

Sequence DNA:

Prediksi Penyakit:

Hasil Tes

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

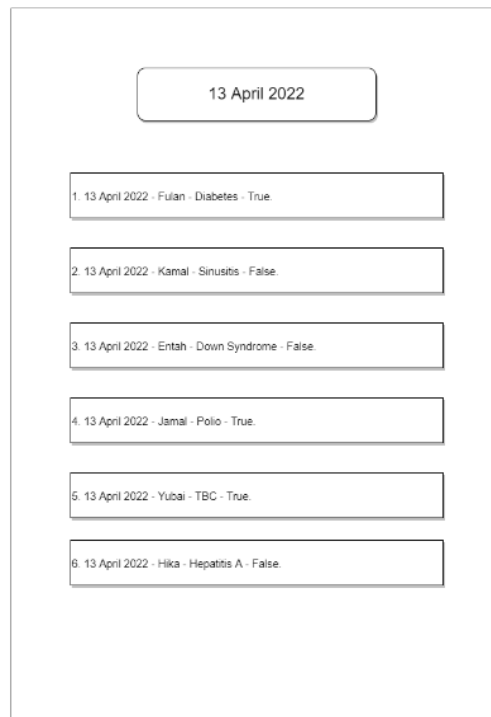
Gambar 1.2. Ilustrasi Prediksi

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: <tanggal_prediksi><spasi><nama_penyakit>, 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

1. 13 April 2022 - Fulan - HIV - True.
2. 13 April 2022 - Kamol - 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 1.3. Ilustrasi Interaksi 1

ii. Masukkan hanya tanggal

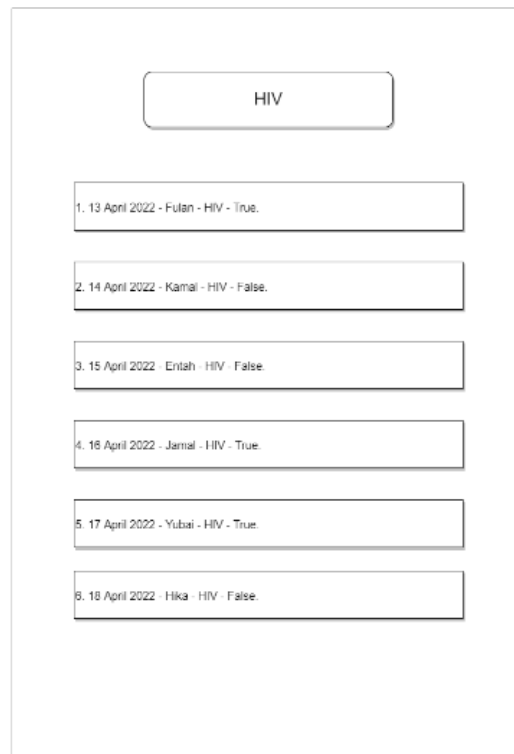


13 April 2022

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 1.4. Ilustrasi Interaksi 2

iii. Masukkan hanya nama penyakit



HIV

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 1.5. Ilustrasi Interaksi 3

4. (Bonus) Menghitung tingkat kemiripan DNA pengguna dengan DNA penyakit pada tes DNA
- Ketika melakukan tes DNA, terdapat persentase kemiripan DNA dalam hasil tes. Contoh hasil tes: 1 April 2022 - Mhs IF - HIV - 75% - False
 - Perhitungan tingkat kemiripan dapat dilakukan dengan menggunakan Hamming distance, Levenshtein distance, LCS, atau algoritma lainnya (dapat dijelaskan dalam laporan).
 - 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.
 - Contoh tampilan:

Tes DNA

Nama Pengguna:

Sequence DNA:

Prediksi Penyakit:

Hasil Tes

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

Gambar 1.6. Ilustrasi Interaksi 3

BAB 2: LANDASAN TEORI

2.1 Algoritma Knuth-Morris-Pratt

Algoritma Knuth-Morris-Pratt (lebih umum dan di dokumen ini dirujuk sebagai KMP) adalah sebuah algoritma pencocokan *string* dengan urutan pencocokan kiri-ke-kanan. Secara sekilas, algoritma ini mempunyai kesamaan dengan algoritma *brute-force* dalam hal pencocokan karakter satu-per-satu dengan patokan teks. Akan tetapi, algoritma ini mempunyai satu kelebihan dibandingkan algoritma *brute-force*.

Algoritma *brute-force* membandingkan string dengan mencocokkan setiap kemungkinan dari subteks yang ada dengan pola yang hendak dicocokkan. Oleh karena itu, apabila terdapat ketidakcocokan, algoritma *brute-force* akan melakukan pergeseran sebesar satu karakter ke kanan. Akan tetapi, algoritma KMP melakukan pencocokan subteks awal pattern dengan subteks akhir pattern. Apabila terdapat ketidakcocokan, algoritma KMP akan melakukan pergeseran sebesar ukuran prefiks yang mempunyai kecocokan dengan sufiks.

Perhitungan ukuran prefiks ini dilakukan dengan *border function* yang menentukan ukuran yang sesuai untuk skenario ketidakcocokan. Penggunaan *border function* diimplementasikan dalam fungsi `computeFailure`, dimana fungsi akan menerima pola yang ingin dicocokkan dan mengembalikan *array* berisi ukuran prefiks yang cocok dengan sufiks untuk skenario ketidakcocokan di tiap karakter. *Array* ini akan digunakan di algoritma utama sebagai patokan pergeseran karakter yang perlu dilakukan.

Dalam tugas besar ini, nilai yang dikembalikan adalah panjang kecocokan maksimum. Untuk algoritma KMP, nilai yang dikembalikan adalah nilai kecocokan dari depan. Apabila terdapat kecocokan lebih dari 80%, maka hasil dianggap terdeteksi..

2.2 Algoritma Boyer-Moore

Algoritma Boyer-Moore (dirujuk sebagai algoritma BM dalam dokumen ini) adalah algoritma pencocokan string dengan pencocokan dari kanan-ke-kiri. Algoritma BM melakukan pengecekan secara mundur dengan melakukan perhitungan *last occurrence* (mirip dengan perhitungan *border function* pada algoritma KMP namun dilakukan pencocokan posisi terakhir suatu karakter dalam suatu pattern).

Pencocokan dilakukan dari bagian paling kanan string teks secara mundur. Menggunakan variabel iterasi, kecocokan ditemukan apabila nilai variabel iterasi tersebut 0. Namun jika tidak, akan dilakukan pengecekan terus secara mundur. Teknik ini disebut *looking-glass technique*, di mana program akan melakukan pengecekan karakter satu-per-satu secara mundur selama karakter di pattern sama dengan karakter di teks. Jika ditemukan ketidakcocokan, maka akan dilakukan pergeseran berdasarkan *last occurrence array* yang telah dibuat sebelumnya. Pencocokan akan dilakukan terus menerus.

Dalam tugas besar ini, nilai yang dikembalikan adalah panjang kecocokan maksimum. Untuk algoritma BM, nilai yang dikembalikan adalah nilai maksimum kecocokan dari belakang, dibandingkan dengan nilai kecocokan dari depan untuk algoritma KMP.

2.3 Regular Expression

Regular Expression adalah sebuah teknik atau cara untuk menentukan keberadaan suatu pattern atau pola, pada sebuah teks. Dengan regular Expression (regex), kita bisa menentukan ada tidaknya suatu pattern pada sebuah string yang lebih panjang atau sama dengan pattern tersebut, baik hanya muncul sekali ataupun berkali-kali. Regex di bangun dari konvensi, jadi regex memiliki sintaks yang universal.

expression	matches...
abc	abc (that exact character sequence, but anywhere in the string)
^abc	abc at the <i>beginning</i> of the string
abc\$	abc at the <i>end</i> of the string
a b	either of a and b
^abc abc\$	the string abc at the beginning or at the end of the string
ab{2,4}c	an a followed by two, three or four b's followed by a c
ab{2,}c	an a followed by at least two b's followed by a c
ab*c	an a followed by any number (zero or more) of b's followed by a c
ab+c	an a followed by one or more b's followed by a c
ab?c	an a followed by an optional b followed by a c; that is, either abc or ac
a.c	an a followed by any single character (not newline) followed by a c
a\\.c	a.c exactly
[abc]	any one of a, b and c
[Aa]bc	either of Abc and abc
[abc]+	any (nonempty) string of a's, b's and c's (such as a, abba, acbacacaa)
[^abc]+	any (nonempty) string which does <i>not</i> contain any of a, b and c (such as defg)
\\d\\d	any two decimal digits, such as 42; same as \\d{2}
\\w+	a "word": a nonempty sequence of alphanumeric characters and low lines (underscores), such as foo and 12bar8 and foo_1
100\\s*mk	the strings 100 and mk optionally separated by any amount of white space (spaces, tabs, newlines)
abc\\b	abc when followed by a word boundary (e.g. in abc! but not in abcd)
perl\\B	perl when <i>not</i> followed by a word boundary (e.g. in perlert but not in perl stuff)

Format dari regex. Foto diatas bukanlah seluruh format dari regex, melainkan sebagian, hanya untuk memberikan contoh.

Pada implementasi kali ini, regex digunakan untuk memenuhi dua buah kebutuhan. Kebutuhan pertama adalah sanitasi input DNA. DNA input yang diberikan harus memiliki format ATGC (case sensitive). Pattern regex yang digunakan untuk mengetahui kevalidan input DNA adalah berikut

^[CAGT]+\$

Artinya input berisi satu atau lebih, dari antara keempat karakter 'C', 'A', 'G', 'T'. Jika tidak memenuhi pattern tersebut, dapat disimpulkan bahwa input string tidak valid.

Penggunaan kedua adalah untuk membaca input dari query search, dan menentukan bagian yang merupakan tanggal, dan bagian yang merupakan nama penyakit. Input dapat berbentuk <tanggal> <nama penyakit>, atau <tanggal>, atau <penyakit>. Misalnya '12 April 2022 HIV'.

Regex digunakan untuk menentukan bagian tanggal, dan bagian penyakit. Pertama digunakan regex berikut

(?i)\\d{1,2}

(januari|februari|Maret|April|mei|juni|july|agustus|september|oktober|november|desember|january|february|march|may|june|july|august|october|december) \\d{4}

untuk mendapatkan bagian dari query search yang berbentuk tanggal.

Selanjutnya digunakan dua buah regex terhadap query string untuk mendapatkan penyakit. Regex pertama,

[^(0-9)]+

digunakan untuk mendapatkan seluruh penyakit yang bukan merupakan angka(dengan asumsi nama penyakit tidak memiliki angka). Kita sebut hasil dari penghilangan angka ini sebagai penyakitKotor. Selanjutnya digunakan regex


```
(?i)(januari|februari|Maret|April|mei|juni|july|agustus|september|oktober|november|desember|january|february|  
march|may|june|july|august|october|december)
```

untuk mendapatkan bagian dari penyakitKotor yang bukan merupakan nama bulan(dengan asumsi nama penyakit yang valid tidak memiliki nama bulan). Dengan dua buah regex tersebut, akan didapatkan nama penyakit yang sudah valid.

Contoh inputnya adalah '12 April 2022 HIV', akan menghasilkan tanggal '12 April 2022' dan penyakit 'HIV'

Langkah selanjutnya adalah forming tanggal menjadi bentuk 'YYYY-MM-DD', untuk menyesuaikan format yang disimpan di database. Langkah yang dilakukan mendapatkan kata 'April' (jika mengikuti contoh diatas) terlebih dahulu, lalu menggunakan regex lagi untuk mengubahnya menjadi angka bulan (04). Regex yang digunakan adalah

```
(?i)(januari|january)  
(?i)(februari|february)  
(?i)(maret|march)  
(?i)(april|april)  
(?i)(mei|may)  
(?i)(juni|june)  
(?i)(juli|july)  
(?i)(agustus|august)  
(?i)(september|september)  
(?i)(oktober|october)  
(?i)(november|november)  
(?i)(desember|december)
```

Jika kata bulan cocok dengan salah satu dari format diatas, akan diubah menjadi angka bulan yang bersesuaian.

Hasil akhirnya adalah, untuk input '12 April 2022 HIV', tanggal '2022-04-12' dan penyakit 'HIV'.

2.4 Penjelasan Singkat tentang Aplikasi

Aplikasi yang dibuat adalah sebuah aplikasi web yang mampu melakukan pencocokan string dan pencarian riwayat pencocokan sebelumnya. Aplikasi ini dibuat menggunakan *bundling* Vite.js berdasarkan React.js dan *library backend* Echo. Aplikasi dapat dijalankan secara lokal atau diakses secara daring di <https://deoxyde.netlify.app/>.

BAB 3: ANALISIS PEMECAHAN MASALAH

3.1 Langkah Penyelesaian Masalah Setiap Fitur

A. Deteksi Penyakit

Fitur deteksi penyakit dilakukan dengan tahapan berikut:

1. Pengguna memasukkan nama pengguna, mengunggah rantai DNA pengguna dalam format .txt, memilih jenis penyakit yang hendak dideteksi, dan memilih metode pencocokan (Boyer-Moore atau KMP)
2. Pengguna menekan tombol submit, dan program akan mengambil data dan memasukkan dalam suatu dataform. Penyakit terjamin terdaftar pada *database* karena menggunakan *selection box* yang menggunakan data dari *database*.
3. Dataform akan dikirimkan ke server (<http://localhost:1323/api/match>). Data rantai DNA akan diperiksa validasinya menggunakan validasi RegEx. Apabila data tidak valid, maka akan dikirim balik sinyal Bad Request
4. Apabila data valid, data akan diproses menggunakan algoritma yang dipilih. Algoritma akan mengembalikan angka kecocokan tertinggi (100% berarti terdapat subset pada DNA yang cocok dengan panjang pattern). Sebuah penyakit dipertimbangkan terdeteksi apabila angka kecocokan di atas 80%.
5. Program akan mengembalikan *response* berupa data hasil pengecekan ke *website* dan *website* akan menampilkan hasil pencarian. Selain itu, program akan menambahkan riwayat penyakit ke dalam *database*.

B. Pencarian Riwayat

1. Pengguna memasukkan entri dalam format <Tanggal><Spasi><Nama_Penyakit>, <Tanggal>, atau <Nama_Penyakit>
2. Entri akan dikirimkan ke server. Server akan melakukan *parsing* menggunakan RegEx untuk memisahkan tanggal dan nama penyakit.
3. Program akan melakukan pengambilan dari database untuk riwayat dengan tanggal atau nama penyakit tertentu.
4. Program akan mengembalikan *array* ke *website* dan *website* akan menampilkan riwayat penyakit dalam bentuk tabel

C. Penambahan Penyakit

1. Program akan menerima rantai DNA pengguna dan nama penyakit yang diinginkan (mirip seperti deteksi penyakit)
2. Program akan mengirimkan dataform berisi nama dan rantai DNA ke server. Program akan melakukan validasi rantai DNA dan pengecekan apakah ada penyakit dengan nama yang sama.
3. Apabila string DNA valid dan penyakit belum ada sebelumnya, penyakit akan ditambahkan ke dalam *database*.
4. Penyakit yang baru ditambahkan dapat langsung digunakan di *website* karena *website* meng-update data pada *selection box* secara dinamis (melakukan pengambilan data tiap loading).

3.2 Fitur Fungsional dan Arsitektur Web

Terdapat beberapa fitur fungsional dalam web:

- Fitur deteksi penyakit menggunakan *drag-and-drop uploader* dan pemilihan penyakit menggunakan *selection box*
- Fitur pencarian riwayat penyakit berdasarkan tanggal dan nama penyakit
- Fitur penambahan penyakit berdasarkan rantai DNA serta nama

- Fitur validasi string yang dimasukkan. Apabila string DNA tidak sesuai, maka *server* akan mengembalikan *bad response*.

Arsitektur Web menggunakan *framework* React.js yang diimplementasikan dalam *bundling* Vite dan *UI framework* ChakraUI. *Client* Axios digunakan untuk pengiriman *request* dan penerimaan *response* dari *frontend*, sementara manajemen routing API untuk *backend* dilakukan dengan pustaka Echo dari Golang. Algoritma pencocokan dan *parsing* diterapkan dalam bahasa Golang. Penyimpanan data dilakukan secara global menggunakan fitur *context* dari ReactJS.

Untuk *local server*, pustaka Echo akan melakukan inisialisasi server di <http://localhost:1323>, dengan empat buah API. API yang dibuat menerima sebuah *dataform* dari Axios dan mengembalikan response JSON kembali ke *frontend*. Berikut adalah daftar API yang didaftarkan:

- 'api/alldiseases', API untuk mendapatkan daftar penyakit yang terdaftar di *database*. Data ini akan digunakan di *selection box* pada bagian pemilihan penyakit
- 'api/match', API untuk melakukan pengecekan. API ini menerima *data form* dan melakukan pencocokan *string* menggunakan algoritma yang diinginkan. API akan mengembalikan *response* berupa data hasil pengujian dan melakukan *insertion* data baru ke *database*.
- 'api/search', API untuk melakukan pencarian riwayat penyakit. API ini menerima data form, melakukan *parsing* menggunakan RegEx untuk mendapatkan tanggal dan penyakit, dan mengembalikan daftar penyakit yang terdaftar atas tanggal dan penyakit tertentu.
- 'api/add', API yang dipanggil untuk menambahkan penyakit baru ke *database*. API ini menerima *dataform* dan melakukan validasi apakah penyakit sudah ada sebelumnya dan string DNA valid. Apabila DNA tidak valid atau penyakit sudah ada, API akan mengembalikan response 400 (bad request), sedangkan apabila kondisi terpenuhi, API akan mengembalikan response 200 dan menambahkan penyakit baru. Penyakit ini dapat digunakan langsung setelah penambahan.

BAB 4: IMPLEMENTASI DAN PENGUJIAN

4.1 Spesifikasi Teknis Program

4.1.1 Front End

Pada bagian Front End, digunakan *framework* React. Untuk mempermudah pembuatan UI, digunakan Chakra UI. Untuk router, digunakan React Router.

Struktur dari FE bisa dibagi tiga bagian besar,

4.1.1.1 Main

Bagian ini berisi fungsi utama yang akan dipanggil. Disini digunakan untuk meletakkan Provider yang digunakan dan Router, dan akan dipanggil App.

Di dalam App, terdapat component yang akan selalu muncul di setiap halaman, yaitu Navigation dan juga salah satu halaman (berdasarkan URL saat ini). Chakra UI juga membutuhkan sebuah 'pembungkus', dan hal tersebut diletakkan di dalam App.

4.1.1.2 Folder Page

Berisi halaman yang akan dipanggil oleh App. Terdapat empat buah halaman, yaitu

1. Home
Berisi fitur utama, yaitu mencocokkan DNA dengan pattern DNA penyakit yang tersedia.
2. InputDisease
Berfungsi sebagai halaman untuk memasukkan disease baru ke dalam database.
3. Search
Digunakan untuk melakukan pencarian pencocokan DNA yang pernah dilakukan, berdasarkan nama penyakit dan tanggal.
4. About
Berisi keterangan mengenai website yang dibuat

4.1.1.3 Folder Component

Untuk meningkatkan modularitas, 'komponen' dari website dapat dibagi menjadi fungsi sendiri. Hal ini sangat membantu untuk meningkatkan abstraksi, terutama jika komponen tersebut sering digunakan. Komponen yang terdapat disini adalah Navigation, FileUploader, dan Provider.

4.1.2 Back End

Back End pada website ini berguna untuk menyimpan data penyakit, data riwayat pencocokan, melakukan pencocokan dan pengecekan input yang diberikan dari FrontEnd. Digunakan REST API, karena cara tersebut cukup universal digunakan pada backend di masa pembuatan website ini, sehingga tersedia banyak sumber informasi untuk membantu pembuatan REST API pada Go.

Back End website ini dibangun menggunakan bahasa Go, dan framework Echo untuk mempermudah pembuatan REST API di Go. Sistem backend yang dibuat dapat dibagi menjadi tiga bagian,

4.1.2.1 Main

Pada bagian ini, server lokal akan dijalankan yang akan memiliki beberapa endpoint, yang masing-masing memiliki fungsi utamanya. Sebelum hal tersebut dilakukan, akan dipanggil sebuah fungsi untuk melakukan pembuatan database yang akan diisi dengan beberapa penyakit.

4.1.2.2 Endpoint

Terdapat beberapa endpoint yang dibuat pada backend, yaitu

1. /api/alldiseases dengan method GET. digunakan untuk mendapatkan seluruh penyakit yang tersedia di backend
2. /api/match dengan method POST.
Digunakan untuk mencocokkan DNA dengan pattern DNA penyakit dan memberikan result sesuai hasil pencocokan tersebut. Algoritma KMP, BM, dan Regex digunakan pada bagian ini.
3. /api/search dengan method POST

Digunakan untuk memberikan result data yang diberikan sesuai dengan query search yang diberikan pada bagian body request. Algoritma Regex digunakan pada bagian ini

4. /api/add dengan method PUT.

Digunakan untuk menambahkan penyakit baru ke dalam database. Algoritma Regex digunakan pada bagian ini.

4.1.2.3 Algo

Terdapat tiga buah algoritma yang telah dijelaskan pada bab dua. Tiap algoritma tersebut diimplementasikan di backend, dalam fungsinya masing-masing. Untuk KMP, terdapat pada file algo_KMP.go, BM terdapat pada file algo_BM.go, dan regex terdapat pada algo_parseSearch.go

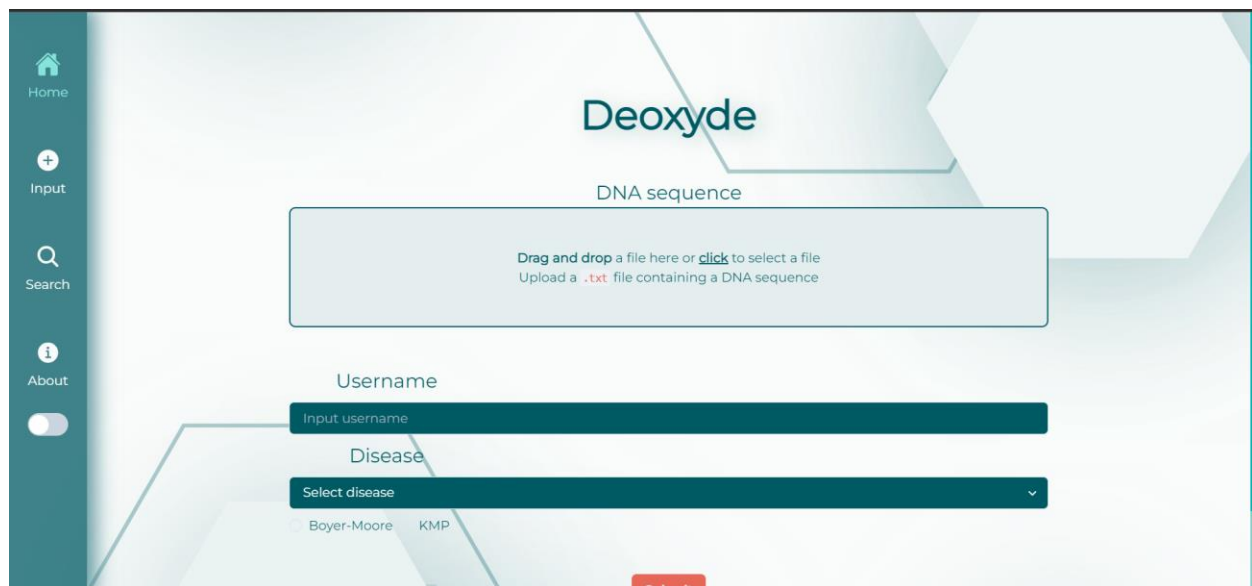
4.2 Tata Cara Menggunakan Program

Program dapat dijalankan secara lokal maupun online. Untuk menjalankan program secara online, cukup buka <https://deoxyde.netlify.app/>. Untuk menjalankan secara lokal, ikuti cara yang berada pada https://github.com/clumsy/Tubes3_13520050/blob/main/README.md

4.2.1 Cara Menggunakan Website

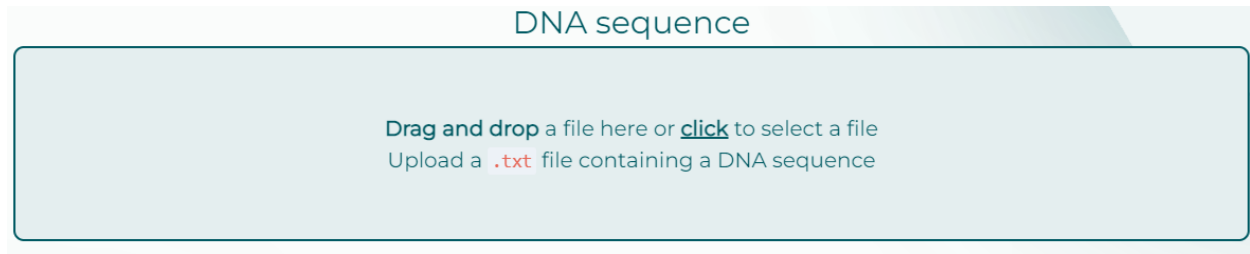
4.2.1.1 Pencocokan DNA

Halaman utama website adalah sebagai berikut.

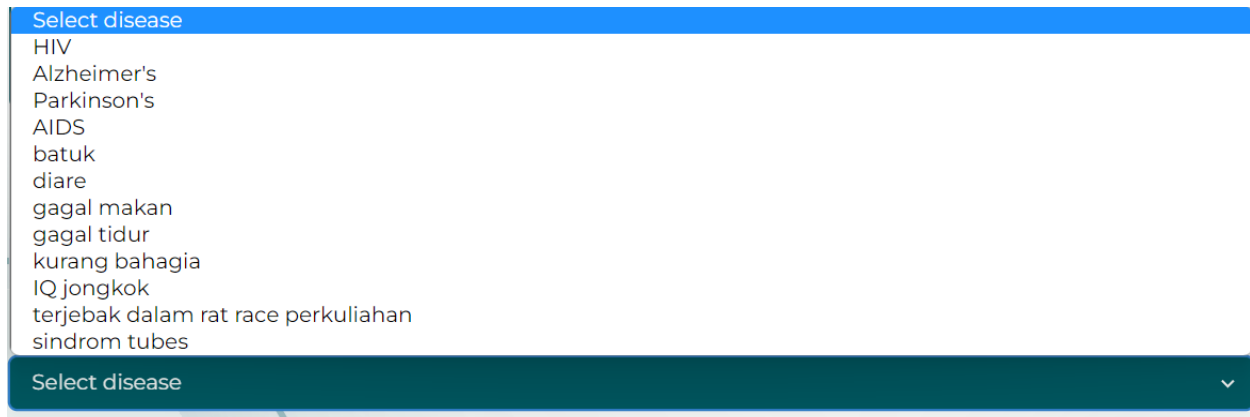


The screenshot shows the Deoxyde website interface. On the left is a dark teal sidebar with icons and labels for 'Home', 'Input', 'Search', 'About', and a toggle switch. The main content area has a light blue background with the 'Deoxyde' logo at the top. Below the logo is a 'DNA sequence' section with a large light blue box containing the text: 'Drag and drop a file here or click to select a file. Upload a .txt file containing a DNA sequence'. Below this is a 'Username' section with a dark teal input field labeled 'Input username'. Underneath is a 'Disease' section with a dark teal dropdown menu labeled 'Select disease'. At the bottom of the main area are two radio buttons labeled 'Boyer-Moore' and 'KMP'. A red 'Submit' button is located at the bottom right of the interface.

Untuk melakukan pencocokan, masukkan file ke bagian secara drag and drop, atau dengan mengklik kotak tersebut.



Selanjutnya masukkan username, untuk identifikasi pengguna yang melakukan pencocokan, dan pilih penyakit diantara pilihan yang tersedia



Terakhir pilih algoritma yang ingin digunakan.



Lalu klik submit. Apabila hasil submisi valid, akan muncul hasil pengujian di bawah. Akan tetapi, apabila ada kesalahan input (teks invalid atau ada field yang tidak terisi), akan dikirimkan alert di website.

4.2.1.2 Input Pattern DNA Penyakit

Pada bagian navigation di kiri website, pilih 'Input'. Akan muncul halaman seperti berikut.



Masukkan pattern DNA penyakit yang ingin diinginkan dengan cara melakukan drag and drop atau klik kotak tersebut. Selanjutnya masukkan nama penyakit, dan klik submit. Penyakit akan ditambahkan ke dalam *database* apabila teks valid dan tidak ada penyakit dengan rantai atau nama yang sama.

DNA sequence

Drag and drop a file here or [click](#) to select a file
Upload a **.txt** file containing a DNA sequence

4.2.1.3 Pencarian Riwayat Pencocokan DNA

Pada navigation di bagian kiri website, pilih Search. Akan tampil halaman sebagai berikut

Selanjutnya masukkan query search dengan format <tanggal> <nama penyakit> atau, <tanggal>, atau <nama penyakit>. Lalu klik submit. Akan muncul riwayat pencocokan DNA sesuai dengan query search yang diberikan.

ID PREDIKSI	TANGGAL	NAMA	PENYAKIT	STATUS	KESAMAAN
46	2022-04-28	TESTPAGI	KURANG BAHAGIA	TERDETEKSI	83%
47	2022-04-28	TESTPAGI	KURANG BAHAGIA	TERDETEKSI	83%
48	2022-04-28	OWEN	KURANG BAHAGIA	TERDETEKSI	83%
49	2022-04-28	OWEN	IQ JONGKOK	TERDETEKSI	80%
50	2022-04-28	OWEN	IQ JONGKOK	TERDETEKSI	80%
51	2022-04-28	TEST2	AIDS	TERDETEKSI	83%
52	2022-04-28	TEST2	AIDS	TERDETEKSI	83%
53	2022-04-28	TEST2	AIDS	TERDETEKSI	83%
54	2022-04-28	TEST2	AIDS	TERDETEKSI	83%
55	2022-04-28	TEST2	AIDS	TERDETEKSI	83%

4.3 Hasil Pengujian

4.3.1 Pengujian Pencocokan DNA

1. Pengujian Pertama

The screenshot shows the application interface for the first test. At the top, there is a 'DNA sequence' section with a text box containing 'AAACACAT' and a message 'Drag and drop a file here or click to change file'. Below this, the 'Username' field is filled with 'Toni'. The 'Disease' dropdown menu is set to 'gagal makan'. The 'Boyer-Moore' algorithm is selected. A red 'Submit' button is visible. At the bottom, the test results are displayed: '2022-04-28 - Toni - gagal makan - Tidak Terdeteksi - 28%'.

2. Pengujian Kedua

The screenshot shows the application interface for the second test. The 'DNA sequence' section is the same. The 'Username' field is filled with 'Toni'. The 'Disease' dropdown menu is set to 'IQ jongkok'. The 'Boyer-Moore' algorithm is selected. A red 'Submit' button is visible. At the bottom, the test results are displayed: '2022-04-28 - Toni - IQ jongkok - Terdeteksi - 100%'.

4.3.2 Pengujian Input DNA

1. Pengujian Pertama

The screenshot shows the application interface for the first test in the 'Input New Disease' section. A modal window at the top says 'decryde.netlify.app says Please make sure your test is valid and disease hasn't existed before!'. Below this, the 'DNA sequence' section contains 'ACTBADADF'. The 'Disease-name' field is filled with 'gagal'. A red 'Submit' button is visible.

2. Pengujian Kedua

deoxyde.netlify.app says
Successfully added new data!

OK

Input New Disease

DNA sequence

Drag and drop a file here or [click](#) to change file

Uploaded: dna.txt

ATGCATGC

Disease name

kurang bobok

Submit

4.3.3 Pengujian Search Riwayat Pencocokan DNA

1. Pengujian Pertama

Search History

28 april 2022

Submit

ID PREDIKSI	TANGGAL	NAMA	PENYAKIT	STATUS
46	2022-04-28	TESTPAGI	KURANG BAHAGIA	TERDETEKSI
47	2022-04-28	TESTPAGI	KURANG BAHAGIA	TERDETEKSI
48	2022-04-28	OWEN	KURANG BAHAGIA	TERDETEKSI
49	2022-04-28	OWEN	IQ JONGKOK	TERDETEKSI
50	2022-04-28	OWEN	IQ JONGKOK	TERDETEKSI
51	2022-04-28	TEST2	AIDS	TERDETEKSI
52	2022-04-28	TEST2	AIDS	TERDETEKSI
53	2022-04-28	TEST2	AIDS	TERDETEKSI
54	2022-04-28	TEST2	AIDS	TERDETEKSI
55	2022-04-28	TEST2	AIDS	TERDETEKSI

2. Pengujian Kedua

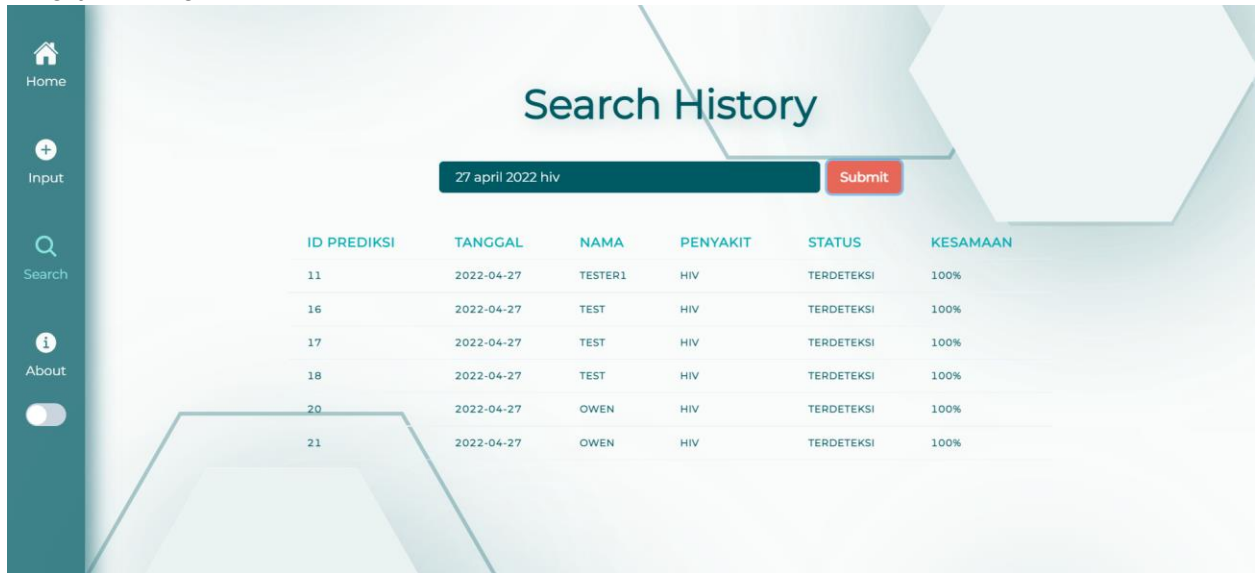
Search History

IQ JONGKOK

Submit

ID PREDIKSI	TANGGAL	NAMA	PENYAKIT	STATUS	KESAMAAN
3	2022-04-24	DNAASDF	IQ JONGKOK	TIDAK TERDETEKSI	0%
4	2022-04-24	USER GANTENG	IQ JONGKOK	TERDETEKSI	100%
49	2022-04-28	OWEN	IQ JONGKOK	TERDETEKSI	80%
50	2022-04-28	OWEN	IQ JONGKOK	TERDETEKSI	80%
61	2022-04-28	TONI	IQ JONGKOK	TERDETEKSI	100%
62	2022-04-28	TONI	IQ JONGKOK	TERDETEKSI	100%

3. Pengujian Ketiga



ID PREDIKSI	TANGGAL	NAMA	PENYAKIT	STATUS	KESAMAAN
11	2022-04-27	TESTER1	HIV	TERDETEKSI	100%
16	2022-04-27	TEST	HIV	TERDETEKSI	100%
17	2022-04-27	TEST	HIV	TERDETEKSI	100%
18	2022-04-27	TEST	HIV	TERDETEKSI	100%
20	2022-04-27	OWEN	HIV	TERDETEKSI	100%
21	2022-04-27	OWEN	HIV	TERDETEKSI	100%

4.4 Analisis Hasil Pengujian

4.4.1 Pengujian Pencocokan DNA

1. Pengujian Pertama

DNA Textnya adalah AACACAT, sedangkan DNA penyakitnya adalah TACAGAT. Dengan menggunakan metode BM, terdapat kecocokan ada dua karakter terakhir, dari keseluruhan pattern penyakit tujuh. Sehingga didapatkan kecocokan 2/7 yaitu 28% jika dibulatkan kebawah.

2. Pengujian Kedua

DNA Textnya adalah AACACAT, sedangkan DNA penyakitnya adalah CACAT. Dengan menggunakan metode KMP, terdapat kecocokan pada seluruh karakter pattern, sehingga didapatkan hasil 100%

4.4.2 Pengujian Input DNA

1. Pengujian Pertama

Data yang diinputkan ada yang bukan berupa ATGC, sehingga tidak bisa diinputkan ke dalam database

2. Pengujian Kedua

Data yang diinputkan sesuai dengan format yang diharuskan, sehingga berhasil diinputkan ke dalam database

4.4.3 Pengujian Search Riwayat Pencocokan DNA

1. Pengujian Pertama

query yang dimasukkan adalah '28 april 2022', dan ditampilkan data pencocokan yang dilakukan pada tanggal 28 april 2022

2. Pengujian Kedua

Query yang dimasukkan adalah 'IQ Jongkok' dan ditampilkan riwayat pencocokan dengan penyakit IQ Jongkok

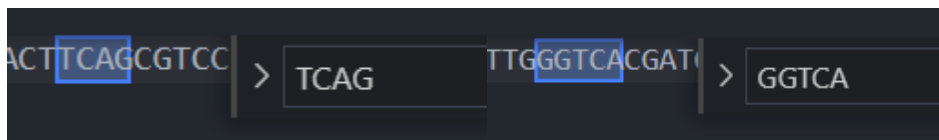
3. Pengujian Ketiga

Query yang dimasukkan adalah '27 april 2022 hiv' dan ditampilkan riwayat pencocokan yang dilakukan pada 27 april 2022 dan dicocokkan dengan penyakit hiv

Perlu diperhatikan bahwa penggunaan algoritma akan berpengaruh terhadap perhitungan persentase. Misalkan pada percobaan berikut:



Kedua percobaan menggunakan berkas dna_test_sample_1.txt dan rantai Parkinson's (GGTCAG). Percobaan di bagian kiri menggunakan algoritma KMP dan menghasilkan persentase 83%, sementara percobaan di bagian kanan menggunakan algoritma Boyer-Moore dan menghasilkan persentase 66%. Hal tersebut terjadi karena pencarian dengan arah terbalik, di mana kecocokan tertinggi menggunakan algoritma KMP adalah 5 karakter (GGTCA), sedangkan kecocokan tertinggi menggunakan algoritma Boyer-Moore adalah 4 karakter (TCAG). Hal ini disebabkan pencocokan karakter dengan algoritma KMP dilakukan dari depan ke belakang, sedangkan pencocokan dengan algoritma Boyer-Moore dilakukan dari belakang ke depan.



BAB 5: KESIMPULAN

5.1 Kesimpulan

Pada tugas besar ini, algoritma Boyer-Moore dan KMP berhasil diterapkan dalam DNA Pattern Matching. Kedua algoritma berfungsi dengan baik dalam mencocokkan sekuens DNA masukan dengan sekuens DNA penyakit yang terdapat di database, dan menghasilkan keluaran yang diinginkan. Program juga dapat menerima masukan penyakit baru untuk disimpan di database, serta melakukan pencarian *search history* dari tes-tes DNA yang telah dilakukan.

5.2 Saran

Untuk tugas besar ini, akan lebih baik bila *hosting backend* difasilitasi oleh ITB karena akan memudahkan dalam pencarian layanan *hosting* yang sesuai secara gratis. Selain itu, pembuatan *frontend* disarankan memakai *framework* yang dapat membantu, seperti React dan ChakraUI. Disarankan juga untuk sudah mempeertimbangkan *deployment* apabila ada rencana untuk melakukan *deployment*. Kreativitas juga dianjurkan dalam membuat tugas besar ini agar eksplorasi terhadap algoritma dan *web development* dapat berkembang.

5.3 Refleksi

Pengerjaan tugas besar merupakan sesuatu yang harus dilakukan dari jauh hari. Seperti tugas-tugas lainnya, penting untuk mengerjakan tugas sedikit demi sedikit hingga selesai, dan tidak menumpuk di akhir. Selain itu, pola pikir yang terbuka dan minat belajar yang tinggi juga sangat mendukung dalam pengerjaan tugas besar ini.

LAMPIRAN

Link website

<https://deoxyde.netlify.app/>

Link repository Github

https://github.com/clumsyyyy/Tubes3_13520050

Link video demo

<https://youtu.be/X4qeS8bLK0s>

DAFTAR PUSTAKA

- Salindia Perkuliahan IF2211 Strategi Algoritma 2020/2021 – Pencocokan String
(<https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Pencocokan-string-2021.pdf>)