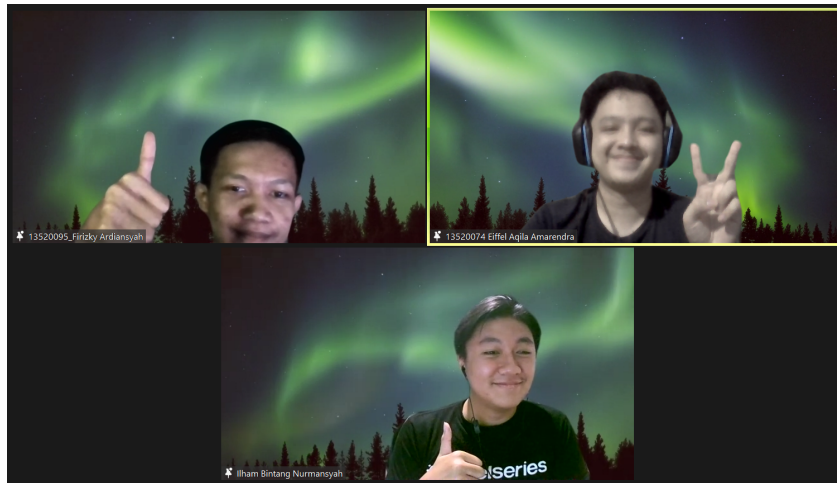


LAPORAN TUGAS BESAR 3
IF2211 STRATEGI ALGORITMA

**PENERAPAN *STRING MATCHING* DAN
REGULAR EXPRESSION DALAM
DNA *PATTERN MATCHING***



Dipersiapkan oleh:

Kelompok Rucika Wavin (18)

| | |
|--------------------------|----------|
| Eiffel Aqila Amarendra | 13520074 |
| Firizky Ardiansyah | 13520095 |
| Ilham Bintang Nurmansyah | 13520102 |

Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung

2022

DAFTAR ISI

| | |
|--|-----------------|
| DAFTAR ISI | <i>i</i> |
| BAB I | 1 |
| BAB II | 7 |
| Algoritma KMP (Knuth Morris Pratt) | 7 |
| Algoritma BM (Boyer-Moore) | 7 |
| Regular Expression | 7 |
| Penjelasan Mengenai Aplikasi Web yang Dibangun | 7 |
| BAB III | 9 |
| Langkah-Langkah Pemecahan Masalah | 9 |
| Fitur Fungsional Web yang Dibangun | 9 |
| Arsitektur Web yang Dibangun | 9 |
| BAB IV | 10 |
| Spesifikasi Teknis Program | 10 |
| Struktur Data | 10 |
| Fungsi dan Prosedur | 10 |
| Tata Cara Penggunaan Program | 11 |
| Interface Program | 11 |
| Fitur Program | 13 |
| Cara Menjalankan Program | 14 |
| Hasil Pengujian | 14 |
| Analisis Hasil Pengujian | 14 |
| BAB V | 16 |
| Kesimpulan | 16 |
| Saran | 16 |
| Komentar/Refleksi | 16 |
| DAFTAR PUSTAKA | 17 |
| LAMPIRAN | 18 |

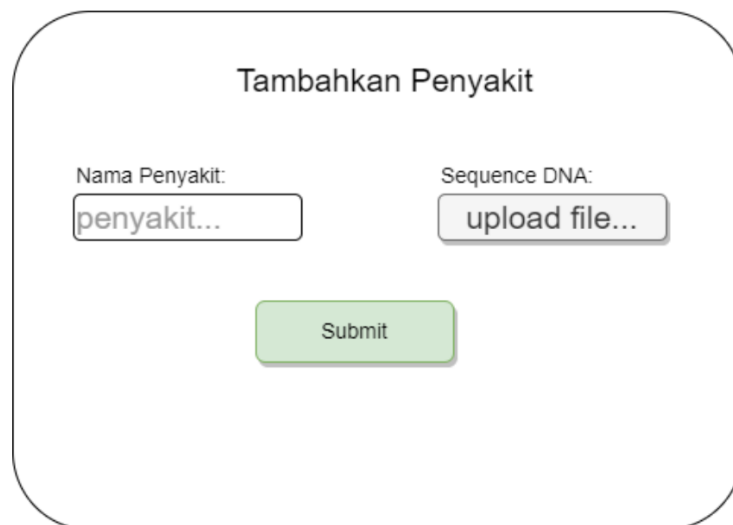
BAB I

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". It contains 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. 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.

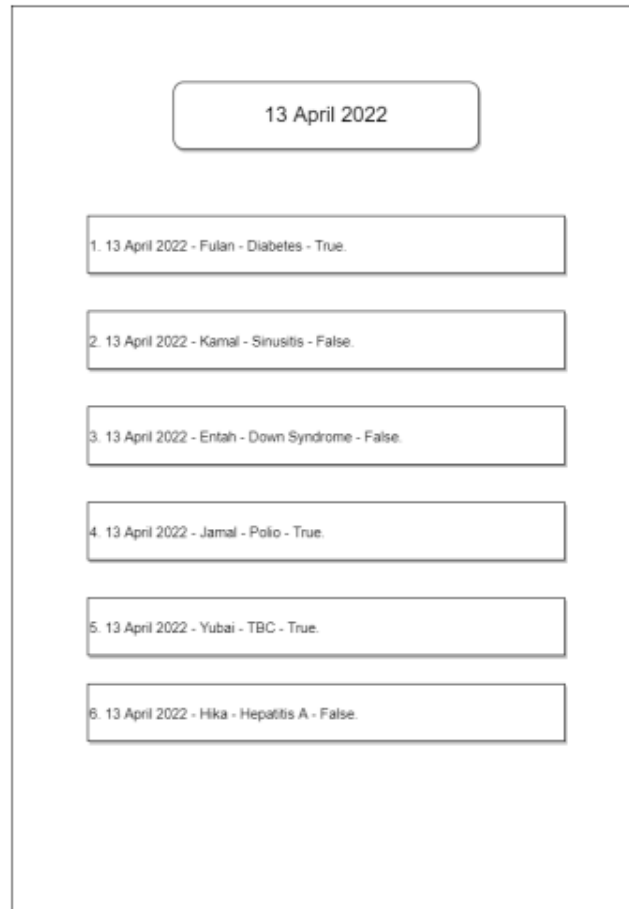
- a. Kolom pencarian dapat menerima masukan dengan struktur: <tanggal_prediksi><spasi><nama_penyakit>, contoh “13 April 2022 HIV”. Format penanggalan dibebaskan, jika bisa menerima >1 format lebih baik.
- b. Kolom pencarian dapat menerima masukan hanya tanggal ataupun hanya nama penyakit. Fitur ini diimplementasikan menggunakan regex.
- c. Contoh ilustrasi:
 - i. Masukan tanggal dan nama penyakit

The image shows a search interface. At the top, there is a search bar containing the text "13 April 2022 HIV". Below the search bar, there is a list of 6 results, each displayed in a separate box. The results are as follows:

| No. | Search Result |
|-----|--------------------------------------|
| 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. |

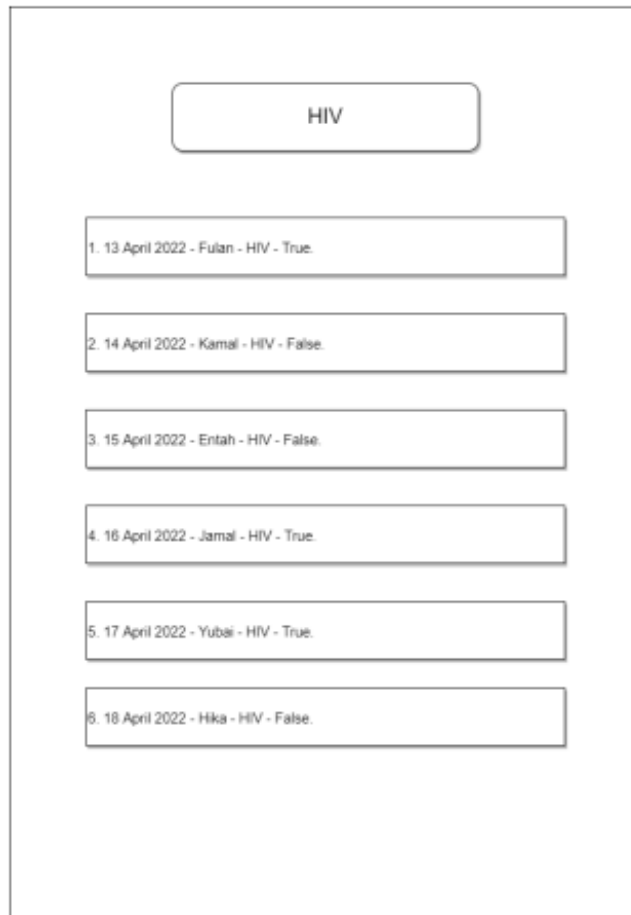
Gambar 3. Ilustrasi Interaksi 1

- ii. Masukan hanya tanggal



Gambar 5. Ilustrasi Interaksi 2

- iii. Masukkan hanya nama penyakit



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:

Tes DNA

Nama Pengguna:
<pengguna>

Sequence DNA:
upload file...

Prediksi Penyakit:
<penyakit>

Submit

Hasil Tes

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

Gambar 7. Ilustrasi Bonus

BAB II

LANDASAN TEORI

2.1. Algoritma KMP (Knuth Morris Pratt)

Algoritma KMP adalah algoritma pencocokan string (*string matching*) yang melakukan pencocokan *pattern* sebuah string dari kiri kekanan seperti melakukan *string matching* menggunakan algoritma *brute force*. Hal yang membedakan dari algoritma *brute force* adalah pergeseran *pattern*nya lebih pintar dibandingkan dengan algoritma *brute force*. Jika ketika dilakukan pencocokan terdapat karakter yang tidak cocok, maka algoritma KMP akan melakukan pergeseran terbesar untuk menghindari perbandingan yang tidak perlu. Perbandingan yang tidak perlu disini artinya adalah bila awal string tidak cocok, berarti bisa melompat hingga beberapa karakter agar tidak membandingkan karakter yang sudah pasti tidak sama.

2.2. Algoritma BM (Boyer-Moore)

Algoritma Boyer-Moore adalah algoritma pencocokan string (*string matching*) yang terdiri atas dua mekanisme umum, antara lain teknik *the looking-glass*, yaitu pencocokan antara *pattern P* dan teks *T* dengan memulai dari indeks terakhir tetapi pemeriksaan terhadap *T* dimulai dari indeks terkecil, serta teknik *the character-jump*, yaitu melakukan lompatan jika pada pencocokan terdapat *mismatch*.

2.3. Regular Expression

Regular Expression (Regex) adalah algoritma pencocokan string (*string matching*) dengan memeriksa kesesuaian dengan sebuah pola tertentu. Teks (*T*) merupakan string yang memiliki panjang n karakter, sedangkan *pattern (P)* merupakan string dengan panjang m karakter dengan panjang m lebih kecil daripada n .

2.4. Penjelasan Mengenai Aplikasi Web yang Dibangun

Aplikasi berbasis web merupakan aplikasi multifungsi yang dapat diakses melalui internet. Terdapat berbagai jenis aplikasi berbasis web, antara lain web media sosial, jual beli dan bisnis, informasi dan berita, dan lain-lain. Dalam tugas besar ini, aplikasi web

yang kami bangun memiliki fungsi untuk mendeteksi serta memprediksi keberadaan penyakit yang ada pada seorang pengguna berdasarkan *sequence* DNA-nya.

Aplikasi web interaktif yang kami kembangkan menggunakan bahasa HTML, CSS, serta JavaScript untuk bagian *frontend* serta Go untuk bagian *backend*. Selain itu, kami juga memanfaatkan framework atau library dari JavaScript, yakni Vue JS untuk membangun tampilan (*interface*) pada website sehingga tampak interaktif bagi pengguna.

BAB III

ANALISIS PEMECAHAN MASALAH

3.1. Langkah-Langkah Pemecahan Masalah

Kami membagi permasalahan menjadi 3 buah masalah, yaitu algoritma string matching, frontend, dan backend. Untuk frontend kami memutuskan menggunakan vue js karena terdapat referensi yang cukup baik untuk digunakan. Untuk backend kami menggunakan go karena referensi yang sama seperti vue js. Setelah tampilan pada frontend selesai, kami mencoba menghubungkan frontend dengan backend. Backend berhasil terhubung kepada frontend dengan menggunakan library go-gin. Lalu kami membuat database pada Azure, dan menghubungkan backend kami dengan database. Terakhir kami menggunakan algoritma yang sudah dibuat untuk melakukan string matching yang didapat dari file input pada frontend, dikirim pada backend, dimasukkan pada database, lalu hasilnya dikembalikan pada frontend lagi.

3.2. Fitur Fungsional Web yang Dibangun

Terdapat beberapa fitur yang dibangun pada web kami, antara lain

1. Fitur untuk menerima input penyakit baru berupa nama penyakit dan sequence DNA-nya serta memasukkan input tersebut ke dalam basis data
2. Fitur untuk memprediksi penyakit yang diderita oleh seorang pengguna berdasarkan sequence DNA-nya serta menghitung tingkat kemiripan DNA pengguna dengan DNA penyakit pada tes DNA
3. Fitur untuk menampilkan hasil prediksi dengan kolom pencarian di dalamnya yang bekerja sebagai *filter* serta menerima masukan tanggal dan/atau nama penyakit.

3.3. Arsitektur Web yang Dibangun

Web kami dibangun dengan arsitektur, sebagai berikut

1. Frontend: HTML/CSS/JS
2. Backend: Go
3. Framework: Vue JS

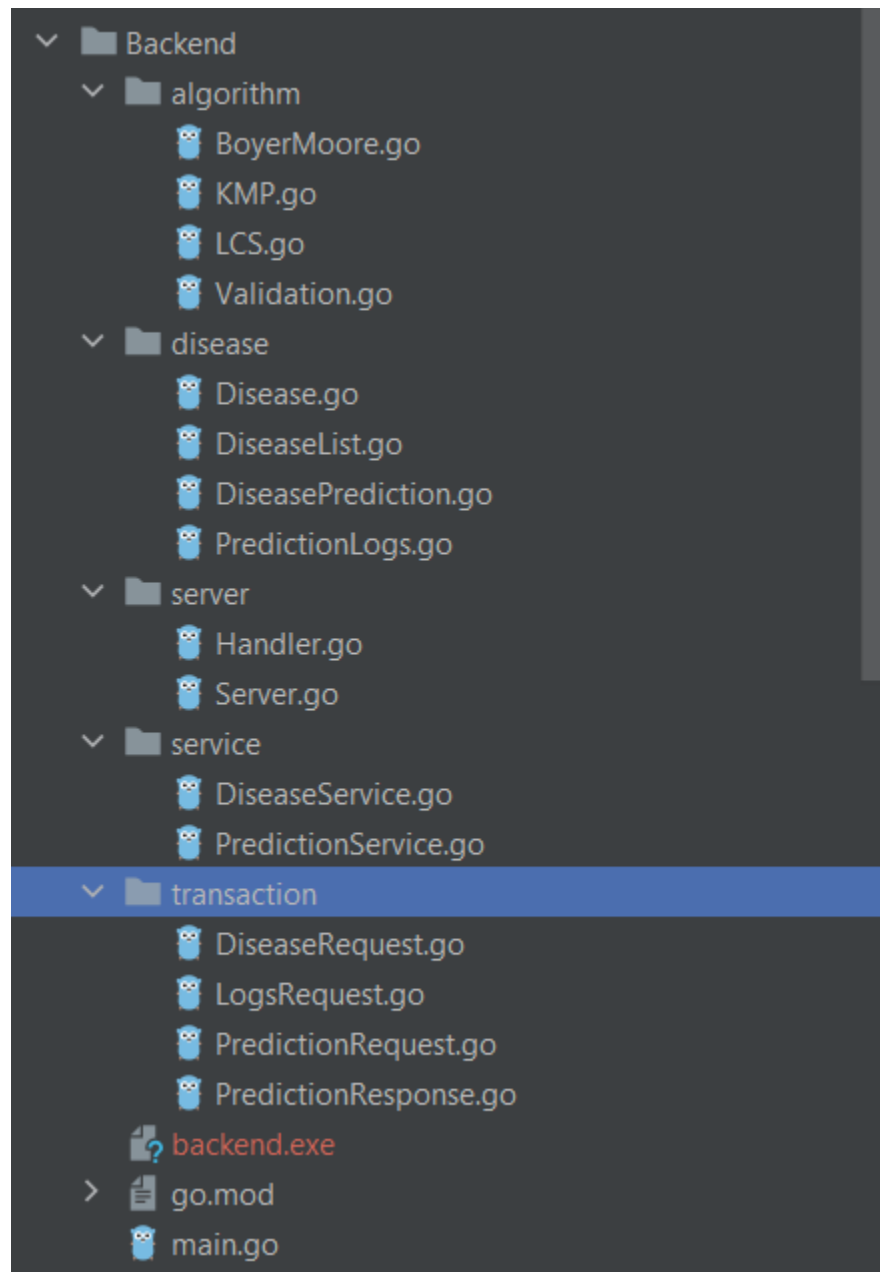
BAB IV

IMPLEMENTASI DAN PENGUJIAN

4.1. Spesifikasi Teknis Program

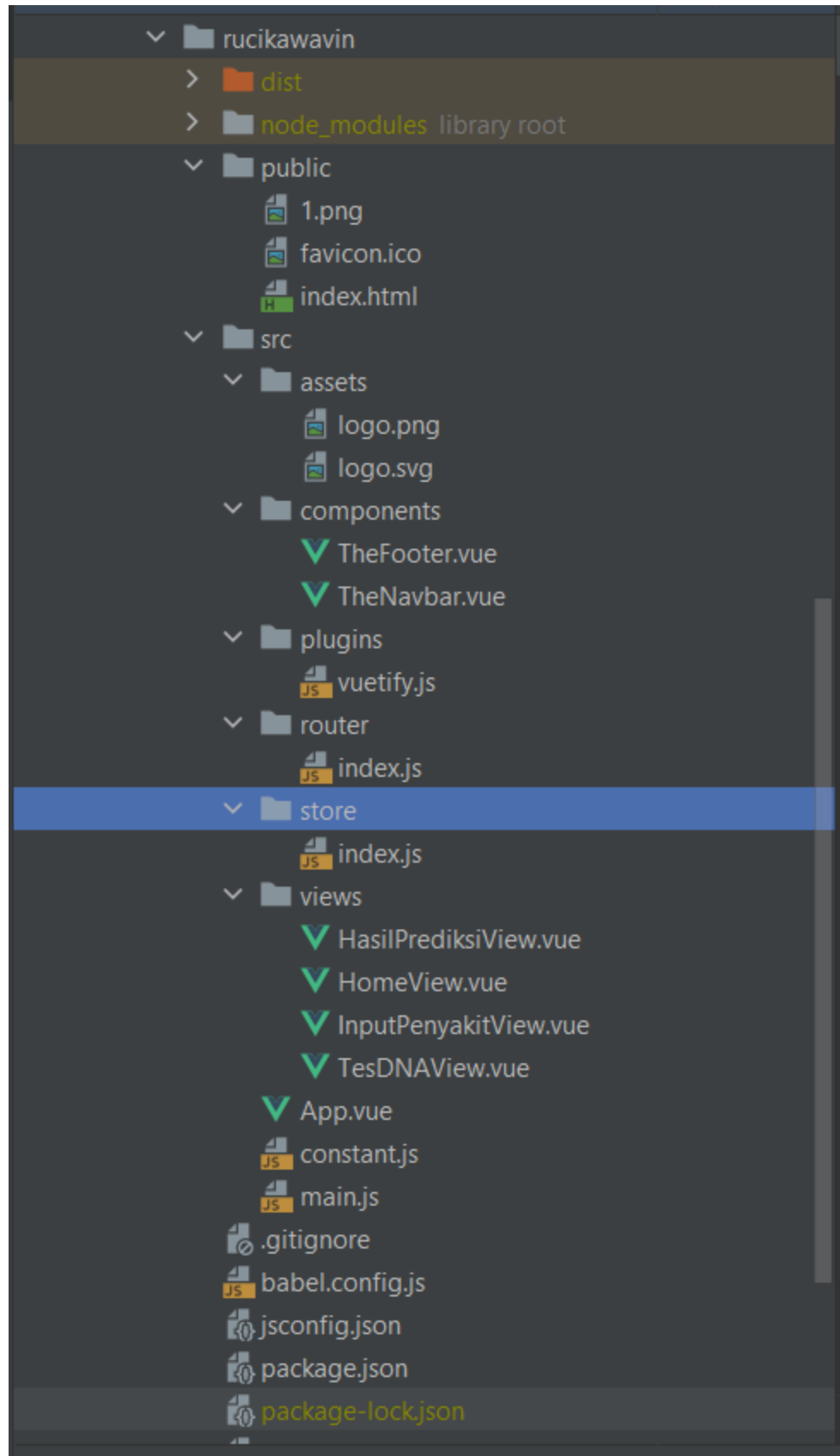
4.1.1. Struktur Folder

Struktur proyek kami meliputi bagian *Backend* dan *Frontend*. Berikut adalah struktur direktori dari *Backend*.



Pada bagian algorithm tersedia library *string matching* dan regex, bagian *Disease* adalah entitas basis data, bagian server berisi komunikasi antara *backend* dan *frontend*, bagian *service* berisi *wrapper* database, dan bagian *transaction* berisi entitas pesan antar *frontend* dan *backend*.

Adapun struktur direkteri *frontend* sebagai berikut.



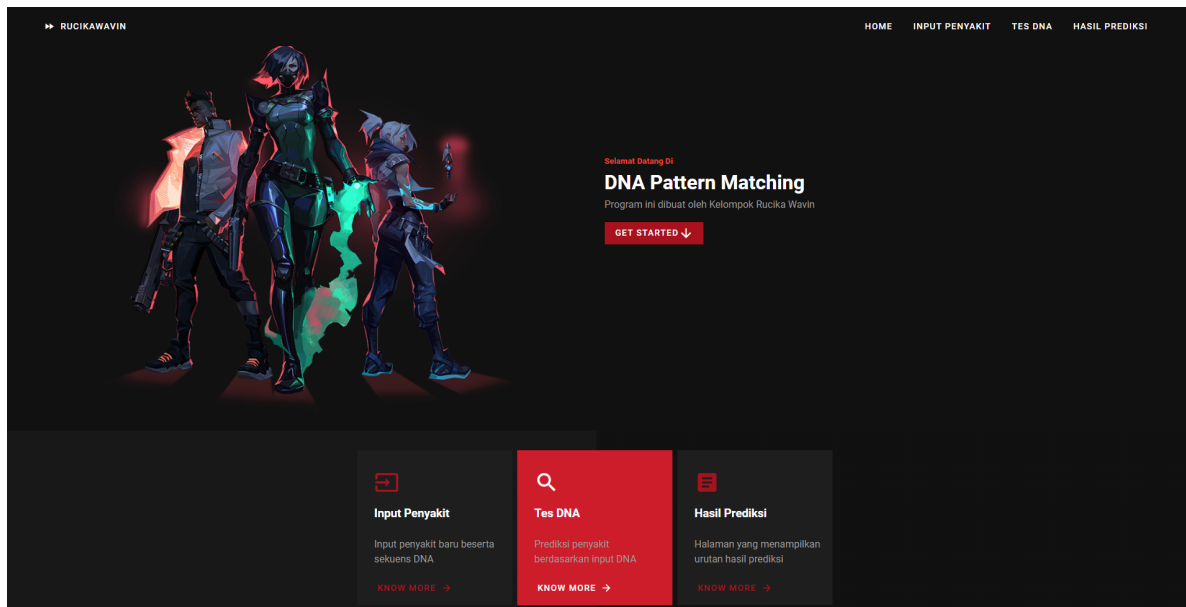
Pada bagian *views* berisi *main program* dari *frontend*, sisanya merupakan *dependency* dan pustaka tambahan dari *frontend*.

4.1.2. Fungsi dan Prosedur

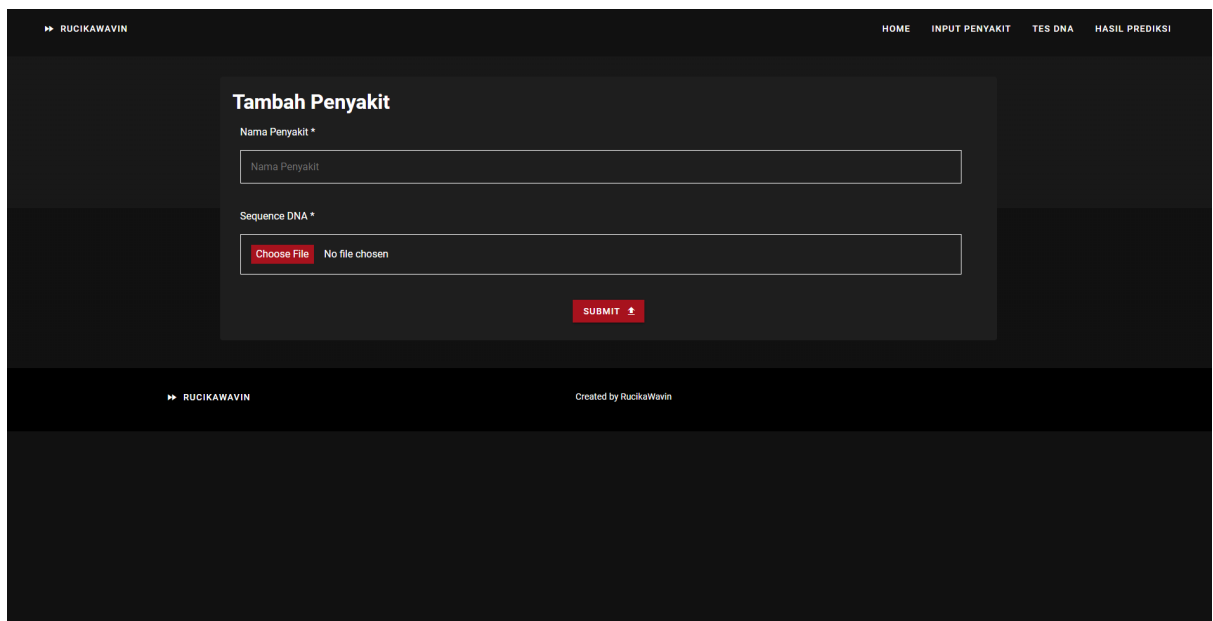
| No | Fungsi | Penjelasan |
|----|---|--|
| 1 | findLPS(virus *string, m int, lps[]) | fungsi ini digunakan untuk menemukan <i>longest proper suffix</i> pada pola virus |
| 2 | KMPSearch(virus string, human string, flag *string) | fungsi ini digunakan untuk melakukan pencocokkan string virus dan human, didalamnya sudah terdapat fungsi findLPS untuk mencari suffix terpanjang dari virus |
| 3 | badCharHeuristic(str string size int, badchar [] int) | fungsi ini digunakan untuk preprocessing fungsi boyer moore, mencari badCharHeuristic. |
| 4 | BoyerMooreSearch(human string, virus string, flag * string) | fungsi ini digunakan untuk melakukan pencocokkan string virus dan human menggunakan algoritma boyer moore |
| 5 | LCS(virus, human string) | fungsi ini digunakan untuk mencari substring terpanjang yang terdapat pada virus yang cocok dengan human. |
| 6 | SimilarityLevel(virus, human string) | fungsi ini digunakan untuk menghasilkan angka kemiripan virus dengan human |
| 7 | ValidateDNA(src string) | fungsi ini digunakan untuk menfilter virus yang di input menggunakan regex |
| 8 | ValidateName(src string) | fungsi ini digunakan untuk menfilter nama yang di input menggunakan regex |
| 9 | ParseQuery(query string) | fungsi ini digunakan untuk memparsing query pencarian pada website |
| 10 | ValidateAlgorithm(src string) | fungsi ini digunakan untuk memvalidasi apakah menggunakan boyer moore atau kmp |

4.2. Tata Cara Penggunaan Program

4.2.1. Interface Program



Gambar 8. Halaman Utama



Gambar 9. Halaman Input Penyakit

Gambar 10. Halaman Tes DNA

Gambar 11. Halaman Hasil Prediksi

4.2.2. Fitur Program

Program kami dibagi menjadi empat halaman, yaitu halaman dashboard, halaman input penyakit, halaman tes DNA, dan halaman hasil prediksi. Pada bagian dashboard, berisi informasi terkait *website* dan navigasi menuju halaman lainnya yang tersedia.

Pada bagian input penyakit terdapat fitur untuk menambahkan penyakit baru yang ada di database. Jika masukan tidak valid atau penyakit sudah tersedia, akan muncul pesan error dan masukan dibatalkan.

Pada bagian tes DNA terdapat fitur pencocokkan sekuens DNA pengguna untuk memprediksi keberadaan sebuah penyakit berdasarkan sekuens DNA pengguna dan sekuens DNA penyakit yang sudah tersimpan di basis data. Sekuens DNA yang tidak valid atau DNA yang tidak ditemukan akan memunculkan pesan kesalahan.

Pada bagian hasil prediksi pengguna dapat memberikan *query* pencarian berdasarkan tanggal dan nama penyakit prediksi. Halaman akan menunjukkan *log* hasil prediksi berdasarkan tanggal dan/atau nama penyakit prediksi yang dimasukkan pengguna.

4.2.3. Cara Menjalankan Program

Berikut adalah langkah kompilasi program dan menjalankan program.

1. Buka direktori frontend (rucikawavin) menggunakan perintah berikut.

```
cd .\src\Frontend\rucikawavin\
```

2. Kemudian jalankan perintah berikut.

```
> npm install
```

3. Proses instalasi akan dijalankan, tunggu beberapa saat hingga proses selesai. Setelah selesai jalankan perintah berikut.

```
npm run build
```

4. Proses *build* akan memakan waktu beberapa waktu lagi. Setelah selesai build, buka direktori dist hasil *build* pada langkah sebelumnya. Folder dist akan berisi index.html yang perlu diedit agar bisa di-load oleh backend.

5. Cari *tag* bernama *script* dengan parameter *type* = “module”. Ubah pada kedua sisi tag menjadi *text/javascript*.

```
type="module" src="/js/chunk-vendors.e884b29e.js"></script><script defer="defer" type="module"
```

6. Simpan hasil perubahan.
7. Buka direktori backend pada terminal dengan menggunakan perintah berikut.

```
> cd ../../Backend\
```

8. Lakukan *build* pada backend menggunakan perintah.

```
go build
```

9. Tunggu beberapa saat, setelah proses *build*, jalankan aplikasi menggunakan perintah berikut.

```
go run main.go
```

10. Jika proses berakhir akan muncul pesan berikut.

```
[GIN-debug] [WARNING] Running in "debug" mode. Switch to "release" mode in production.
- using env:   export GIN_MODE=release
- using code:  gin.SetMode(gin.ReleaseMode)

[GIN-debug] POST   /inputpenyakit      --> backend/server.(*Server).PostDisease-fm (3 handlers)
[GIN-debug] POST   /tesdna             --> backend/server.(*Server).PostPrediction-fm (3 handlers)
[GIN-debug] POST   /hasilprediksi      --> backend/server.(*Server).PostLogs-fm (3 handlers)
[GIN-debug] Loaded HTML Templates (2):
- 
- index.html

[GIN-debug] [WARNING] You trusted all proxies, this is NOT safe. We recommend you to set a value.
Please check https://pkg.go.dev/github.com/gin-gonic/gin#readme-don-t-trust-all-proxies for details.
[GIN-debug] Environment variable PORT is undefined. Using port :8080 by default
[GIN-debug] Listening and serving HTTP on :8080
```

11. *Website* dapat dibuka sesuai dengan port yang ditunjukkan pada keluaran. Buka pada browser halaman berikut.

```
localhost:8080
```

4.3. Hasil Pengujian

4.3.1. Algoritma KMP Cocok

The screenshot shows a web application interface for DNA testing. The header includes the logo 'RUCIKAWAVIN' and navigation links: 'HOME', 'INPUT PENYAKIT', 'TES DNA', and 'HASIL PREDIKSI'. The main form is titled 'Tes DNA' and contains the following fields:

- Nama Pengguna ***: A text input field containing 'Ilham Bintang'.
- Sequence DNA ***: A file upload area with a 'Choose File' button and the text 'No file chosen'.
- Penyakit ***: A text input field containing 'Covid'.
- Algoritma ***: A radio button group with two options: 'KMP' (selected) and 'Boyer Moore'.

Below the form is a red 'SUBMIT' button. Underneath, the 'Hasil Tes' section displays the result: '29 April 2022 - Ilham Bintang - Covid - 100% - true'.

Gambar 12. Algoritma KMP Cocok

4.3.2. Algoritma KMP Tidak Cocok

The screenshot shows the same web application interface as Gambar 12, but with different input data. The header and navigation links are identical. The 'Tes DNA' form contains:

- Nama Pengguna ***: A text input field containing 'Hambin'.
- Sequence DNA ***: A file upload area with a 'Choose File' button and the text 'No file chosen'.
- Penyakit ***: A text input field containing 'Covid 2'.
- Algoritma ***: A radio button group with two options: 'KMP' (selected) and 'Boyer Moore'.

Below the form is a red 'SUBMIT' button. Underneath, the 'Hasil Tes' section displays the result: '29 April 2022 - Hambin - Covid 2 - 72% - false'.

Gambar 13. Algoritma KMP Tidak Cocok

4.3.3. Algoritma Boyer Moore Cocok

The screenshot shows a web application interface for DNA testing. The header includes a logo 'RUCIKAWAYIN' and navigation links: 'HOME', 'INPUT PENYAKIT', 'TES DNA', and 'HASIL PREDIKSI'. The main form is titled 'Tes DNA' and contains the following fields:

- Nama Pengguna ***: A text input field containing 'Hamblin'.
- Sequence DNA ***: A file upload area with a 'Choose File' button and the text 'No file chosen'.
- Penyakit ***: A text input field containing 'Covid'.
- Algoritma ***: A radio button group with two options: 'KMP' and 'Boyer Moore'. The 'Boyer Moore' option is selected.

Below the form is a red 'SUBMIT' button. Underneath, the 'Hasil Tes' section displays the result: '29 April 2022 - Hamblin - Covid - 100% - true'.

Gambar 14. Algoritma Boyer Moore Cocok

4.3.4. Algoritma Boyer Moore Tidak Cocok

This screenshot shows the same web application interface as Gambar 14, but with different input data. The 'Tes DNA' form contains:

- Nama Pengguna ***: A text input field containing 'Hamblin'.
- Sequence DNA ***: A file upload area with a 'Choose File' button and the text 'No file chosen'.
- Penyakit ***: A text input field containing 'Covid 2'.
- Algoritma ***: A radio button group with two options: 'KMP' and 'Boyer Moore'. The 'Boyer Moore' option is selected.

Below the form is a red 'SUBMIT' button. Underneath, the 'Hasil Tes' section displays the result: '29 April 2022 - Hamblin - Covid 2 - 72% - false'.

Gambar 15. Algoritma Boyer Moore Tidak Cocok

4.4. Analisis Hasil Pengujian

Berdasarkan hasil pengujian yang diuji pada bagian sebelumnya, *website* yang sudah dibentuk, sudah memenuhi spesifikasi yang diberikan. Semua kasus uji sudah sesuai dengan yang diharapkan. Hal yang bisa ditingkatkan dari *website* ini misalnya adalah dengan menambah fitur DNA yang sudah ditambahkan.

BAB V

KESIMPULAN DAN SARAN

5.1. Kesimpulan

Algoritma Knuth Morris Pratt (KMP), Boyer-Moore (BM), serta *Regular Expression* (Regex) merupakan algoritma yang dapat digunakan dalam pencocokan string (*string matching*). Dengan memanfaatkan algoritma tersebut, kelompok kami berhasil membangun sebuah aplikasi web interaktif sederhana untuk mendeteksi dan memprediksi keberadaan penyakit genetik tertentu pada seorang pengguna berdasarkan *sequence* DNA-nya.

5.2. Saran

Saran sebaiknya pahami framework yang akan digunakan, pahami cara menghubungkan backend dan frontend, jangan mengerjakan tugas ini mepet mepet deadline.

5.3. Komentar/Refleksi

Tugas besar 3 Strategi Algoritma ini sangat membantu kami dalam memahami penerapan dan aplikasi algoritma-algoritma pencocokan String di dalam fungsi-fungsi yang sederhana. Selain itu, tugas besar ini juga sangat membantu kami untuk mempelajari pengembangan aplikasi berbasis web, mulai dari pengembangan frontend, backend, dan basis data. Meskipun demikian, program yang kami bangun pun masih belum sempurna dan masih dapat dikembangkan lagi di kemudian hari.

DAFTAR PUSTAKA

- Munir, Rinaldi. (2021). Pencocokan string (String matching/pattern matching). Institut Teknologi Bandung. <https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Pencocokan-string-2021.pdf>. Diakses pada 11 April 2022.
- Munir, Rinaldi. (2021). Pencocokan string dengan Regular Expression (Regex). Institut Teknologi Bandung. <https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2018-2019/String-Matching-dengan-Regex-2019.pdf>. Diakses pada 11 April 2022.

LAMPIRAN

Link Github:

https://github.com/firizky29/Tubes3_13520074

Link Youtube:

<https://youtu.be/ztiaxHAR6hQ>