

Laporan Tugas Besar 3 IF 2211

Strategi Algoritma

Penerapan String Matching dan Regular Expression dalam DNA Pattern Matching



Disusun Oleh:

Match DNA bukan Match Tinder

Gede Prasadha Bhawarnawa 13520004

Angelica Winasta Sinisuka 13520097

Rizky Akbar Asmaran 13520111

Institut Teknologi Bandung

Bandung

2022

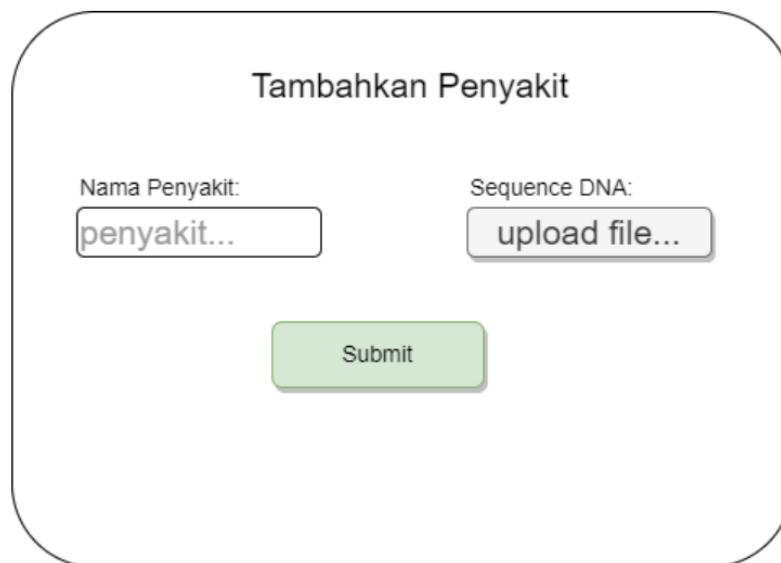
BAB I

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.

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>

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.
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.
5. Spesifikasi Program:
- a. Aplikasi berbasis website dengan pembagian Frontend dan Backend yang jelas.
 - b. Implementasi Backend wajib menggunakan Node.js / Golang, sedangkan Frontend disarankan untuk menggunakan React / Next.js / Vue / Angular. Lihat referensi untuk selengkapnya.
 - c. Penyimpanan data wajib menggunakan basis data (MySQL / PostgreSQL / MongoDB).
 - d. Algoritma pencocokan string (KMP dan Boyer-Moore) wajib diimplementasikan pada sisi Backend aplikasi.
 - e. 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.
 - f. Jika mengerjakan bonus tingkat kemiripan DNA, simpan hasil tingkat kemiripan tersebut pada basis data.

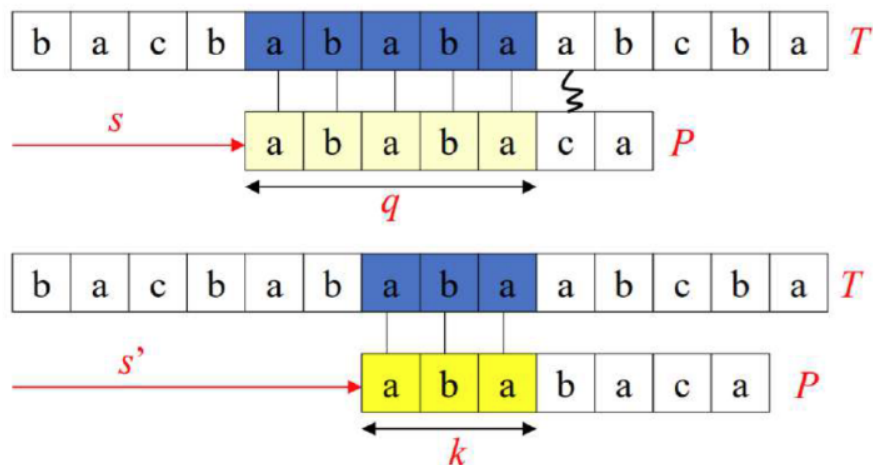
BAB II

LANDASAN TEORI

2.1 Algoritma KMP

Algoritma KMP (Knuth-Morris-Pratt) adalah algoritma untuk mencari pola dalam suatu teks dalam urutan dari kiri ke kanan (hampir mirip seperti algoritma *brute force*). Akan tetapi, Algoritma KMP ini memiliki pola yang lebih cerdas dari pada algoritma *brute force*. Algoritma KMP ini melakukan pre-proses pada pola tersebut untuk menemukan kecocokan awalan pola dengan pola itu sendiri. *Border Function* didefinisikan sebagai ukuran prefiks terbesar dari $P[0..m]$ yang juga merupakan sufiks dari $P[1..m]$. Algoritma KMP ini memiliki kompleksitas waktu sebesar $O(n + m)$.

Berikut merupakan Ilustrasi pengerjaan algoritma Knuth-Morris-Pratt:



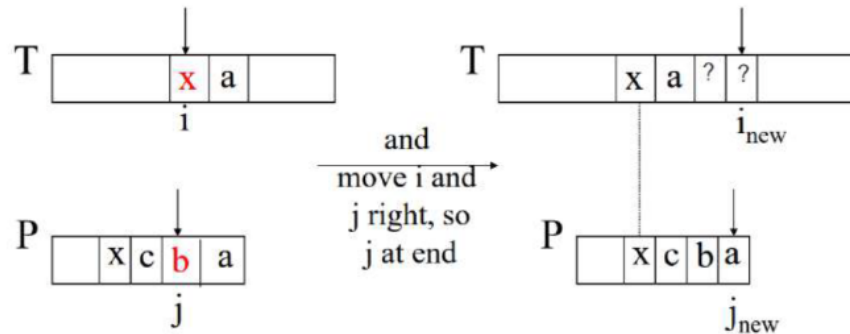
2.2 Algoritma Boyer-Moore

Algoritma BM (Boyer-Moore) merupakan salah satu algoritma yang digunakan dalam pencarian string atau *string matching*. Algoritma ini didasarkan pada dua teknik, yaitu *looking-glass* dan *character-jump*. Pada teknik *looking-glass*, mencari P dalam T dengan bergerak mundur melalui P yang dimulai dari ujungnya (indeks terakhir *pattern*). Sementara, pada teknik *character jump*, merupakan teknik yang diaplikasikan ketika terjadi *mismatch* antara $\text{teks}[i]$ dan $\text{pattern}[j]$. Pada teknik ini terdapat tiga kasus yang mungkin terjadi. Antara lain:

Misal x adalah posisi terjadinya *mismatch* karakter, indeks i merupakan indeks terjadinya *mismatch* karakter pada teks, dan indeks j merupakan indeks terjadinya *mismatch* karakter pada pattern.

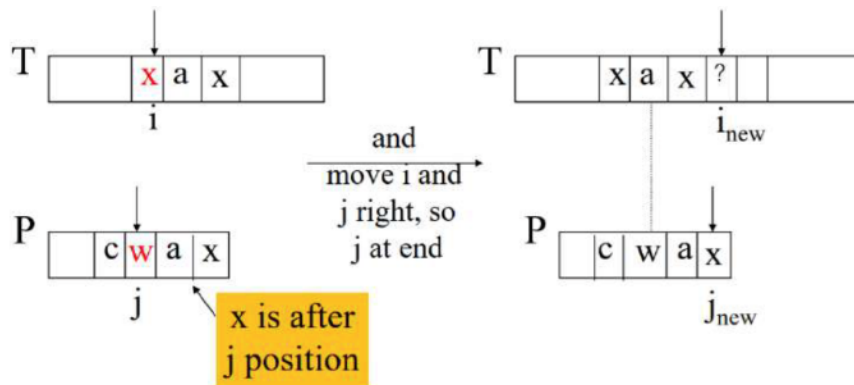
a. Kasus 1

Apabila pattern mengandung x sebelum indeks j , maka lakukan pergeseran kekanan untuk mencocokkan x pada pattern dengan $\text{Teks}[i]$.



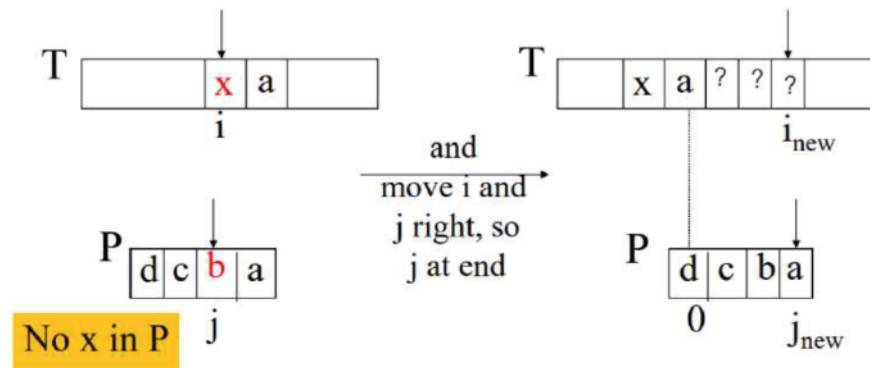
b. Kasus 2

Apabila pattern mengandung x setelah indeks j , sehingga tidak mungkin melakukan pergeseran ke kanan untuk pencocokan x pada pattern dengan $\text{Teks}[i]$. Maka geser pattern ke kanan 1 karakter menuju $\text{Teks}[i+1]$.



c. Kasus 3

Selain kasus 1 dan kasus 2, geser pattern untuk mencocokkan $\text{pattern}[0]$ dengan $\text{teks}[i+1]$.



2.3 Regex

Regex atau *Regular Expression* adalah deretan suatu karakter spesial yang mendefinisikan sebuah pola dalam pencarian teks. Umumnya, pola Regular Expression ini digunakan untuk proses pencarian dan juga “*find and replace*” yang dioperasikan pada String. Selain itu, *Regular Expression* juga dapat digunakan untuk memvalidasi sebuah masukan atau input.

2.4 Aplikasi Web

Website merupakan suatu pages yang dikode menggunakan HTML (Hypertext Markup Language dan CSS (Cascading StyleSheets) untuk memberikan style atau warna pada html yang dasar. Terdapat 2 jenis website, yaitu static dan dynamic. Static website yang disimpan pada server dalam format yang akan dikirim client browser dan biasanya website ini memberikan informasi yang sama ke semua pengunjung, tetapi dapat diupdate secara berkala. Contoh dari jenis website ini adalah blog, brochure website, dan yang lain-lain. Kemudian terdapat website dynamic, yaitu website yang berubah terus menerus. Dynamic website bisa dicapai menggunakan HTML form, dan menyimpan serta membaca kembali browser cookie, atau menghasilkan halaman baru berdasarkan klik dari pelanggan. Dynamic HTML menggunakan JavaScript untuk mengkonstruksi web browser untuk memodifikasi isi dari halamannya.

BAB III

ANALISIS PEMECAHAN MASALAH

3.1 Langkah Penyelesaian Masalah Setiap Fitur

3.1.1 Menerima Input Penyakit Baru

1. Mengambil input dari text box nama penyakit dan file yang berupa dna
2. Isi file dibaca kemudian dilakukan sanitasi input menggunakan fungsi `validasiInput` melalui regex
3. Jika hasil benar, maka akan disimpan ke dalam database.
4. Jika hasil salah akan mengeluarkan textbox salah dan tidak akan direkam ke dalam database.

3.1.2 Memprediksi Penyakit Seseorang menurut sequence DNA-nya

1. Mengambil database berupa file JSON, lalu isi data tersebut dibaca yang berupa pola DNA.
2. Memasukkan pola DNA tersebut kedalam fungsi *bm* atau *KMP*
3. Dari fungsi tersebut akan dicek apakah pola DNA dari orang tersebut apakah terdapat pola yang sama atau tidak.
4. Jika sama, maka akan menghasilkan TRUE, jika tidak maka akan menghasilkan FALSE

3.1.3 Menampilkan Hasil Prediksi

1. Penampilan hasil prediksi akan dilakukan setelah user memasukkan input kedalam prediksi penyakit. Kemudian akan diolah sesuai dengan langkah 3.1.2.
2. Jika sudah benar, akan disimpan ke dalam database kemudian difetch oleh page `DnaResults` yang kemudian akan menampilkan hasil prediksi.

3.1.4 Menghitung Kemiripan DNA pengguna (bonus)

1. Mengambil database berupa file JSON, lalu isi data tersebut dibaca yang berupa pola DNA.
2. Memasukkan pola DNA tersebut kedalam fungsi *similarity*
3. Dari fungsi tersebut akan dihitung presentase kemiripan dari pola DNA orang dengan pola DNA penyakit. Lalu akan disimpan kedalam database dan ditampilkan kedalam program.

3.2 Fitur Fungsional dan Arsitektur Aplikasi Web yang Dibangun

3.2.1 Fitur Fungsional

- a. Menerima Input Penyakit Baru
- b. Filter Input dari User
- c. Memprediksi Penyakit dari Sequence DNA
- d. Menampilkan Hasil Prediksi
- e. Menghitung Kemiripan DNA dari Masukkan Pengguna

3.2.2 Arsitektur Aplikasi Web

Arsitektur aplikasi web dibangun dari 2 bagian, yaitu backend dan frontend.

a. Frontend

Web dibagi menjadi 2 bagian, yaitu front end dan back end. Framework yang digunakan untuk frontend adalah nextjs dan menggunakan sebagian library dari react dan next. Situs dibangun menjadi 4 page, yaitu homepage, form input dna, form search dna, dan result pencocokan dna. Untuk membangun sebuah page digunakan komponen. Terdapat beberapa komponen, yaitu komponen untuk homepage, komponen untuk menampung satu informasi hasil pencocokan, komponen untuk menampung beberapa hasil pencocokan, komponen form menerima masukkan file penyakit, dan komponen form untuk menerima masukkan pencocokkan penyakit. CSS dibagi menjadi 2, yaitu untuk form dan list.

b. Backend

Backend dari web pattern matching dibangun menggunakan bahasa javascript dan menggunakan database MongoDB. Database terdiri dari atribut objek, nama penyakit, dan sequence dnanya. Buat query testing orang memiliki databasenya tersendiri, dan search query berdasarkan tanggal dan penyakit akan diambil melalui database test.

BAB IV

IMPLEMENTASI DAN PENGUJIAN

4.1 Spesifikasi Teknis Program

4.1.1 Struktur Data

Struktur data yang digunakan adalah JSON (Javascript Object Notation) untuk pengambilan dan pengiriman data ke basis data. JSON untuk pengiriman data dalam bentuk array jika hasilnya banyak. Contohnya adalah sebagai berikut:

```
const dummyHasil = [ {  
    Id: '1',  
    Tanggal: '25 April 2022',  
    Nama: 'Rizky',  
    Penyakit: 'HIV',  
    Prediksi: 'FALSE'  
}];
```

4.1.2 Fungsi

- a. **bm**
Fungsi ini menggunakan algoritma boyer moore yang mengembalikan integer. Apabila ketemu, maka hasil sesuai dengan letak posisi, kemudian apabila tidak ketemu, fungsi mengembalikan nilai -1.
- b. **Km**
Fungsi ini menggunakan algoritma Algoritma Knuth- Morris-Pratt yang mengembalikan integer. Apabila ketemu, maka hasil sesuai dengan letak posisi, kemudian apabila tidak ketemu, fungsi mengembalikan nilai -1.
- c. **validateInput**
Fungsi ini digunakan untuk mensanitasi input memasukkan file penyakit. Apabila tidak gagal tes, maka akan return boolean False. Apabila berhasil, akan return True.
- d. **validateSearch**
Fungsi ini digunakan untuk mensanitasi input search query dari search. Apabila tidak gagal tes, maka akan return boolean False. Apabila berhasil, akan return True.

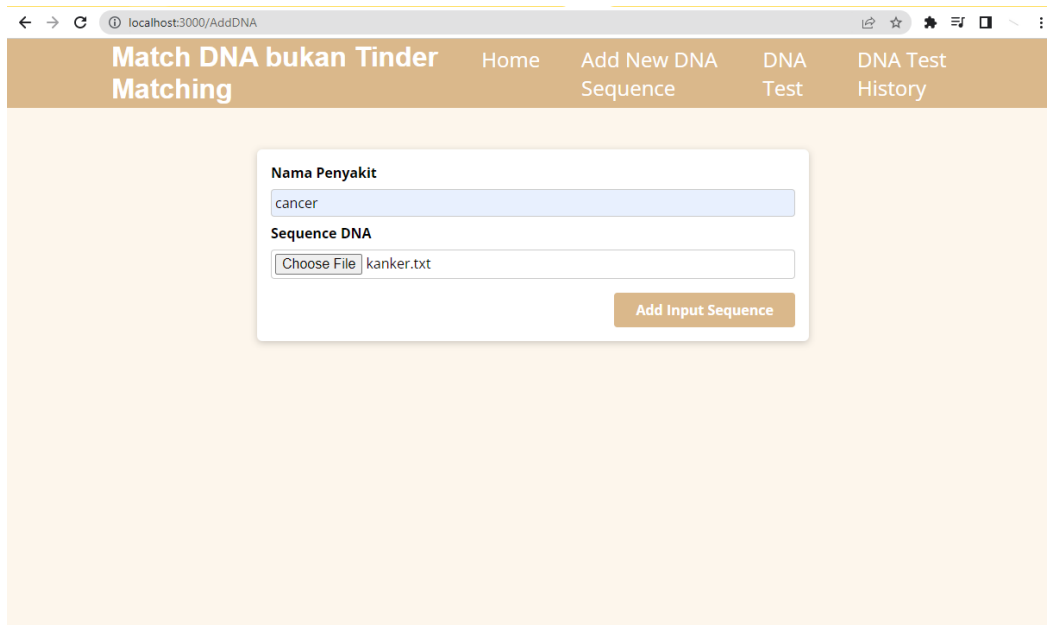
4.1.3 Prosedur

Pada program ini kami tidak menggunakan prosedur

4.2 Tata Cara Penggunaan Program

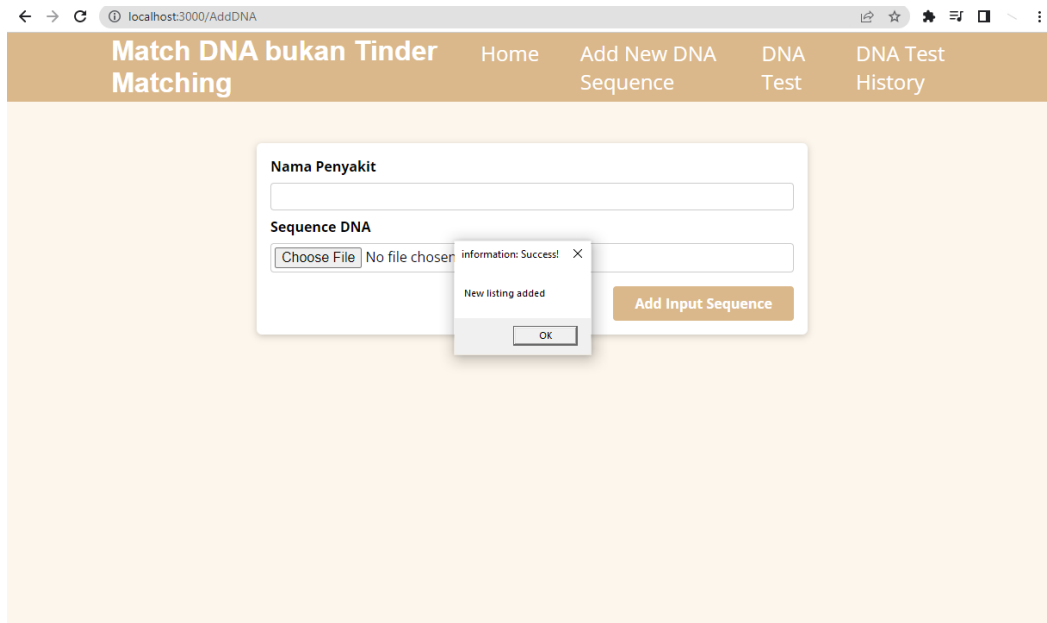
- Pengguna download repository, kemudian pada root directory folder, ketik dalam terminal atau command line "npm install next" kemudian "npm install". Lalu dapat dijalankan menggunakan command "npm run".
- Jika Sudah masuk halaman, pilih tab yang diinginkan kemudian masukkan input sesuai dengan permintaan dari form.
- Respon dari website akan dilihat dilihat dari message box jika sudah berhasil atau tidak dalam mensubmit hasil form yang telah diisi.
- Jika pengguna memasukkan informasi untuk matching, maka akan ditambahkan ke dalam form hasil dari query pencarian yang mendaftar ke bawah.
- Jika pengguna memasukkan informasi untuk input penyakit, maka akan ditampilkan jika sudah berhasil terekam atau tidak.

4.3 Hasil Pengujian

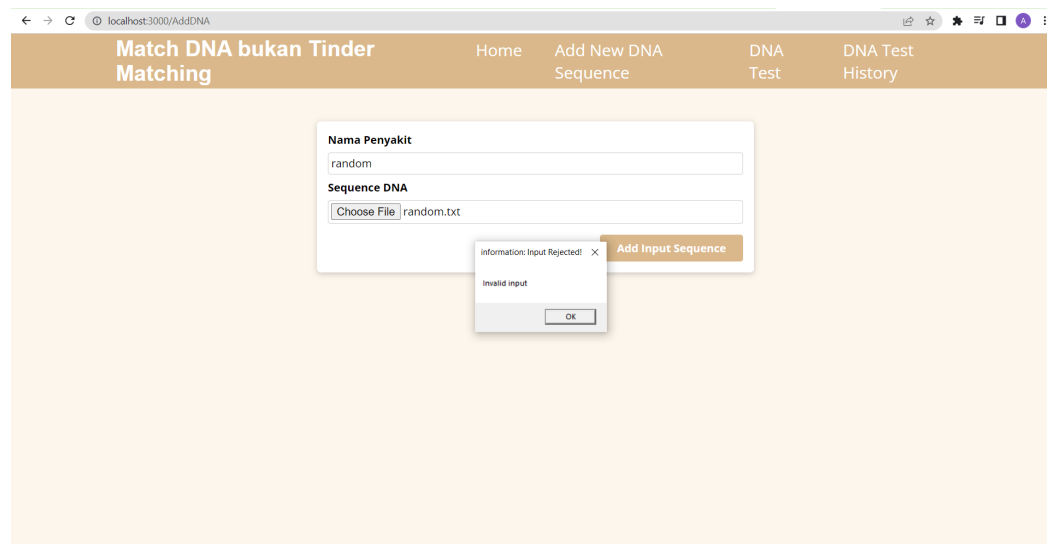


The screenshot displays a web application titled "Match DNA bukan Tinder Matching". The browser's address bar indicates the URL is "localhost:3000/AddDNA". The application's navigation menu includes "Home", "Add New DNA Sequence", "DNA Test", and "DNA Test History". The main content area features a form for adding a new DNA sequence. This form is divided into two sections: "Nama Penyakit" (Disease Name) with a text input field containing the word "cancer", and "Sequence DNA" with a "Choose File" button and a text input field containing "kanker.txt". An "Add Input Sequence" button is positioned at the bottom right of the form.

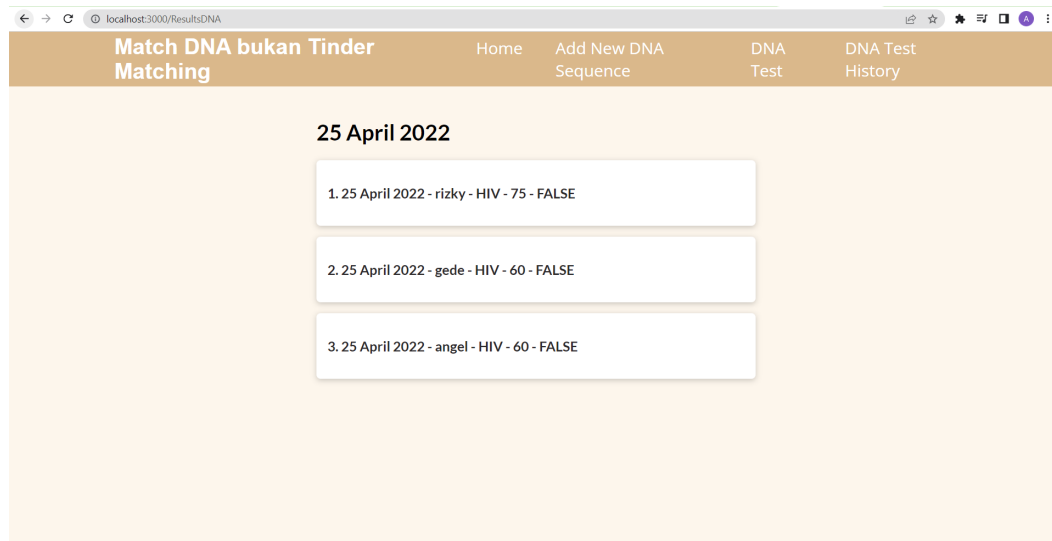
Gambar 4.1 Menambahkan Penyakit ke Database



Gambar 4.2 Menambahkan Penyakit ke Database Benar



Gambar 4.3.1 Sanitasi Input File Salah



Gambar 4.3.2 Hasil Result Query Pencarian

4.4 Analisis Hasil Pengujian

Algoritma *String Matching* yang dibuat berhasil menemukan pola dari DNA yang cocok pada *query*. Fungsi *Levenshtein distance* yang digunakan juga dapat berhasil menghasilkan nilai kemiripan antara pola DNA sehingga program dapat menampilkan persentase kemiripan antara Pola DNA. Program juga dapat menerima input yang akan dimasukkan ke dalam database seperti file txt DNA. Program dapat menampilkan hasil prediksi, menampilkan hasil perhitungan kemiripan DNA. Oleh karena itu program yang dibuat sudah dapat memenuhi semua kriteria fungsionalitasnya.

BAB V

KESIMPULAN, SARAN, DAN REFLEKSI

1. Kesimpulan

Berikut adalah beberapa kesimpulan yang kami dapatkan dari pengerjaan tugas besar Mata Kuliah IF2211 Strategi Algoritma:

- String Matching dapat dilakukan dengan menggunakan beberapa macam algoritma, salah satunya adalah algoritma KMP dan Boyer-Moore.
- Konsep String Matching dapat diimplementasikan pada pencocokan suatu Sequence DNA pada suatu penyakit tertentu.
- Konsep Regex dapat digunakan untuk mencari kumpulan kata atau karakter yang memiliki pola, misalnya pada kasus ini adalah Pola DNA

2. Saran

Berikut adalah beberapa saran yang kami dapatkan dari pengerjaan tugas besar Mata Kuliah IF2211 Strategi Algoritma:

- Dapat mempelajari node.js dengan lebih baik sebelum mengerjakan tugas besar ini
- Mengenali framework Next.js dengan lebih baik sebelum mengerjakan tugas besar ini

Link Github : <https://github.com/LordGedelicious/DNA-Pattern-Matching-Website.git>

Daftar Pustaka

Khodra, M. L. (2019). String Matching dengan Regular Expression. Homepage Rinaldi Munir. <https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2018-2019/String-Matching-dengan-Regex-2019.pdf>

Knuth, D. E., Morris, J. H., & Pratt, V. R. (1974). FAST PATTERN MATCHING IN STRINGS*.

Munir, R. (2021). Pencocokan String. Homepage Rinaldi Munir. <https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Pencocokan-string-2021.pdf>