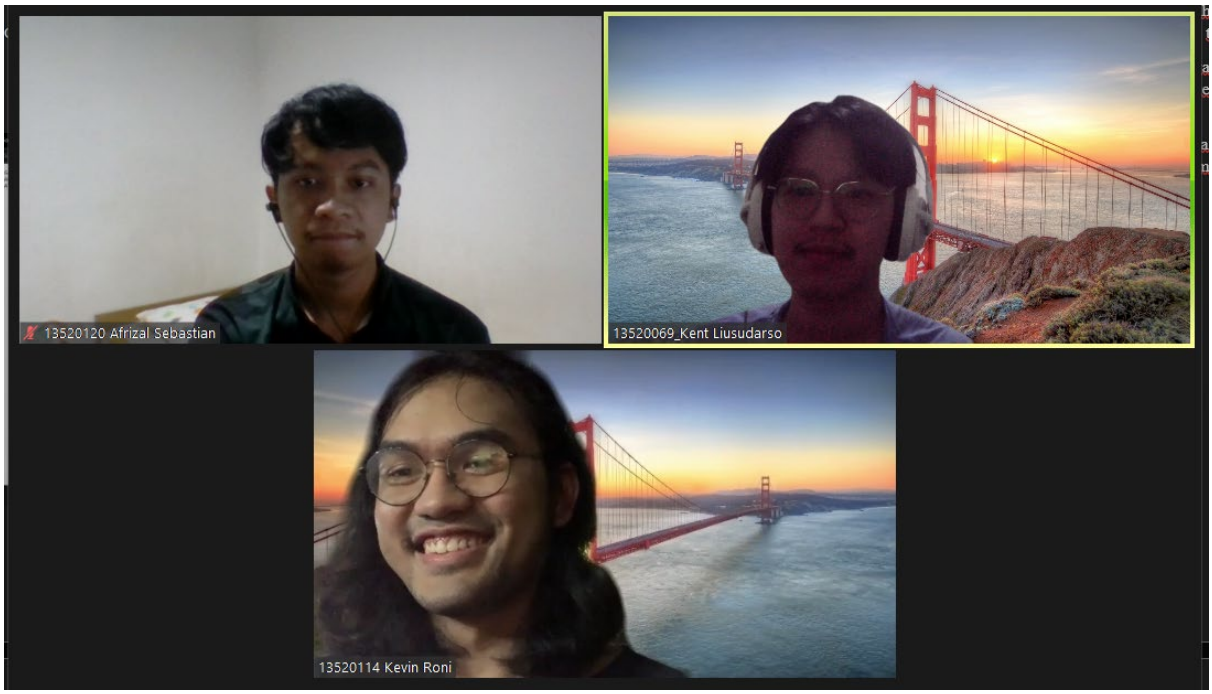
Tugas Besar 3 IF 2211 Strategi Algoritma
Semester II Tahun 2021/2022

Penerapan String Matching dan Regular Expression dalam DNA Pattern Matching



DISUSUN OLEH

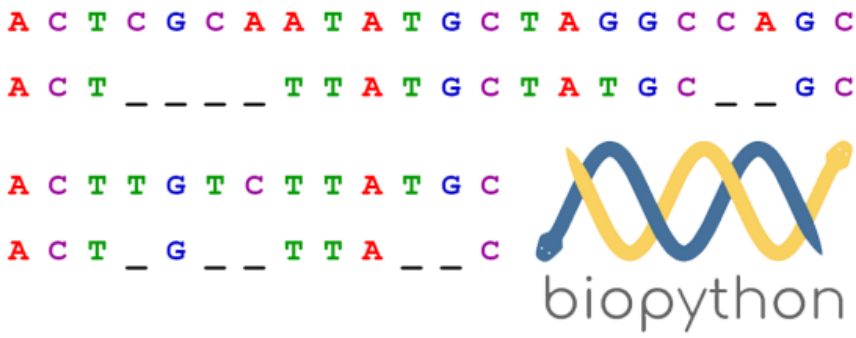
Kent Liusudarso	**13520069**
Kevin Roni	**13520114**
Afrizal Sebastian	**13520120**

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

BAB I
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.



Gambar 1. Ilustrasi Sekuens
DNA Sumber: https://towardsdatascience.com/pairwise-sequence-alignment-using-biopython-d1a9d0ba861f

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: ATTCGTAACTAGTAAGTTA. 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.

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.

1.2 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).

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.

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 pengecekkan string matching terlebih dahulu.

1.3 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

1. Knuth-Morris-Pratt (KMP) Algorithm

Pada dasarnya, algoritma ≠ mirip seperti brute-force, yaitu melakukan pencocokan string dari kiri ke kanan. Tetapi, algoritma ini akan menggeser pattern dengan lebih cerdas sehingga tidak selalu menggeser sebanyak satu. Algoritma KMP menyatakan bahwa jika ada ketidakcocokan pada text T dan pattern P sehingga T[i] ≠ P[j], penggeseran terjauh untuk mengurangi pekerjaan yang sia-sia adalah prefix dari P[0..j-1] yang merupakan suffix dari [1..j-1]. Misalkan ada sebuah string abaab dan ditemukan ketidakcocokan. Prefix terbesar yang merupakan bagian dari suffix adalah ab, maka pencarian string akan dimajukan sebesar panjang string – panjang prefix tersebut (5-2) sehingga pattern akan dimajukan sebanyak 3 kali. Ketika dilakukan penyocokan karakter pada pattern terhadap teks, algoritma KMP melakukan sebuah preprocess untuk mencari prefix. Preprocess ini biasa disebut border function b(k). Lihat contoh di bawah ini:

- P: abaaba

  j: 012345

(k = j-1)

j	0	1	2	3	4	5
P[j]	a	b	a	a	b	a

k	0	1	2	3	4
b(k)	0	0	1	1	2

b(k) is the size of the largest border.

Untuk setiap string yang berukuran n, b(k) menyatakan ukuran prefix terbesar [0..k] yang merupakan suffix [1..k]. Lihat contoh untuk k = 4. Ukuran prefix terbesar abaab yang merupakan suffix dari baab adalah ab, maka b(4) = 2.

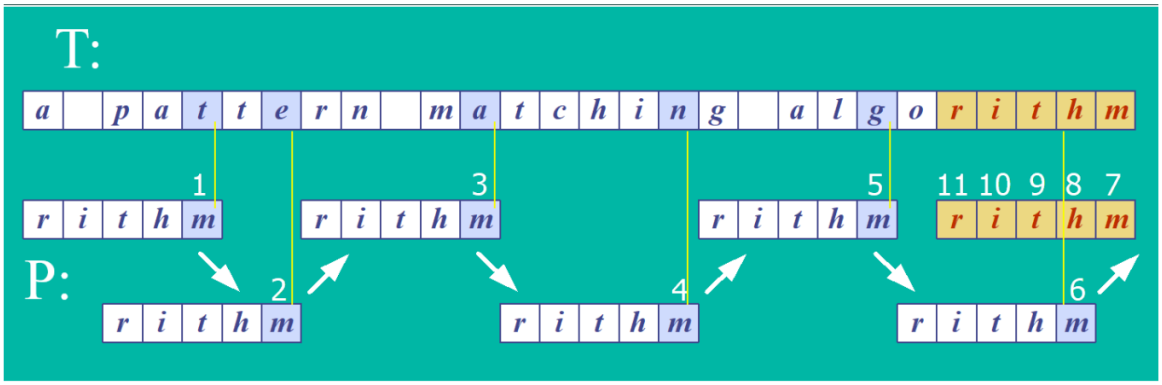
Algoritma untuk menghitung fungsi pinggiran pada KMP adalah O(m) dan untuk pencarian string adalah O(n), maka algoritma ini memiliki kompleksitas O(m+n)

2. The Boyer-Moore Algorithm

Algoritma Boyer-Moore memiliki konsep mirip seperti KMP, yaitu pencarian string yang memiliki preprocessing di dalamnya. Algoritma string matching ini didasarkan oleh dua

teknik, yaitu teknik looking-glass serta teknik character-jump. Looking-glass merupakan teknik untuk mencari Pattern pada Teks dengan mengecek secara mundur, dimulai dari akhir pattern. Sedangankan teknik character-jump dilakukan ketika ada ketidakcocokan P[j] dengan x pada T[i]. Teknik ini memiliki 3 skenario. Skenario pertama adalah jika p mengandung x, maka p digeser ke kanan sehingga sejajar dengan keberadaan terakhir x di P dengan T[i]. Skenario kedua adalah jika P mengandung x, tetapi menggeser hingga keberadaan terakhir tidak mungkin, maka P akan digeser sebanyak 1 karakter hingga sejajar T[i+1]. Skenario ketiga adalah jika P tidak mengandung x, maka P digeser hingga P[0] sejajar T[i+1]. Berikut adalah contoh skenarionya

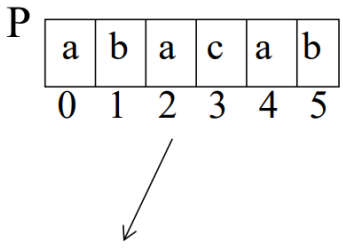
Boyer-Moore Example (1)



Fungsi preprocessing Boyer-Moore biasa disebut last occurrence function L(x). L(x) akan memetakan semua karakter di teks ke integer. L(x) dinyatakan sebagai indeks terbesar i sehingga P[i] = x atau mengembalikan -1 jika tidak ada. Contohnya sebagai berikut

L() Example

- A = {a, b, c, d}
- P: "abacab"



x	*a*	*b*	*c*	*d*
L(x)	4	5	3	-1

Algoritma Boyer-Moore memiliki kompleksitas waktu worst case O(mn) dengan m adalah panjang teks dan n adalah panjang pattern. Algoritma Boyer-Moore akan lebih efektik jika

variasi alfabet besar karena mismatching lebih mungkin terjadi dan akan sangat buruk jika variasi alfabet kecil / binary karena mismatching lebih jarang terjadi.

3. Regular Expression (Regex)

Regular expression merupakan sebuah sekuens karakter yang mespesifikkan sebuah pattern pencarian. Regular expression memiliki penggunaan yang sangat luas terutama untuk fungsi pencarian serta validasi input. Regular expression mendeskripsikan Bahasa regular dalam teori Bahasa formal dan memiliki expressive power yang sama dengan regular grammars. Regex memiliki tiga standar, yaitu BRE (basic regex), ERE (extended regex), SRE (simple regex). Hampir semua bahasa pemrograman sudah memiliki library regex yang sesuai dengan standard masing-masing.

4. Penjelasan Singkat Aplikasi

Aplikasi yang kami buat secara garis besar memiliki fitur untuk melakukan test DNA yang diimplementasikan menggunakan algoritma KMP atau Boyer-Moore dalam melakukan pengecekan stringnya. Untuk melakukan test DNA diperlukan input string berupa rantai penyusun DNA penyakit serta manusia yang ingin ditest. Input ini akan divalidasi menggunakan regular expression agar sesuai dengan rantai yang mungkin, yaitu kombinasi karakter ACGT. Selain itu, aplikasi kami dapat melakukan fitur search dengan memasukkan input tanggal dan/atau nama penyakit. Web aplikasi yang kami buat menggunakan Bahasa pemrograman golang untuk server side (backend) serta postgresql untuk database. Untuk client-side kami menggunakan framework next.js.

BAB III
ANALISIS PEMECAHAN MASALAH

1. Langkah Penyelesaian Masalah

A. Fitur menambahkan penyakit

-Fitur ini akan menerima input berupa nama penyakit serta rantai penyusunnya.

-Pertama akan dicek apakah isi file valid atau tidak menggunakan regex.

-Jika tidak maka fitur menambahkan penyakit tidak dapat dilakukan.

-Jika valid maka akan dicek apakah sudah ada di database atau belom.

-Jika sudah ada, maka rantai penyakit yang ada di dalam database akan diupdate.

-Jika tidak, maka penyakit akan ditambahkan ke dalam database.

B. Fitur test DNA

-Fitur ini akan menerima input berupa nama orang yang akan diuji, penyakit yang ingin dites, rantai DNA penyusun manusia yang ingin diuji serta metode string matching yang ingin digunakan ketika pengecekan dna.

-Ketika dilakukan submit, akan dilakukan pengecekan input rantai DNA penyusun dengan regex.

-Jika valid akan diteruskan untuk dilakukan pemrosesan sesuai dengan metode yang dipilih.

-Pengecekan akan dilakukan dengan algoritma string matching terlebih dahulu.

-Jika tidak match, maka akan dicek similaritynya menggunakan algoritma hamming distance. Hamming distance akan menghitung similarity berdasarkan jumlah karakter yang sama ketika teks dan pattern disejajarkan.

-Fungsi hamming distance akan diiterasi dengan menggeser pattern sebanyak 1 ke kanan dan mengambil nilai maksimum hamming distance dari setiap iterasi. Hamming distance pada algoritma ini sedikit dimodifikasi. Hamming distance sebenarnya adalah mencari jumlah karakter yang tidak sesuai antar string dengan panjang sama. Pada program ini dibuat untuk mencari jumlah karakter yang sama. Hasil akhir Hamming distance yang dimodifikasi ini akan dibagi dengan panjang pattern yang dicari lalu dikalikan dengan 100 agar didapat persentase kemiripannya.

-Jika pada pengecekan ditemukan kecocokan pada algoritma Boyer-Moore/KMP maka program otomatis mengassign value isSick menjadi true dan persentase kemiripan 100.
-Jika tidak, maka persentase akan mengikuti fungsi yang telah dibuat berdasarkan penjelasan di atas dan value isSick menjadi true apabila persentase lebih besar atau sama dengan 80 dan false jika sebaliknya.
-Data akan ditambahkan ke database.

C. Fitur Search

-Fitur ini akan menerima input berupa query yang ingin dicari.
-Setelah query dimasukkan query akan diperiksa menggunakan regex.
-Query akan diteruskan ke database jika memenuhi salah satu dari 3 kasus berikut. Query dalam bentuk tanggal, tanggal+nama penyakit atau nama penyakit saja.
-Jika memenuhi syarat yang divalidasi dengan regex, query akan diteruskan ke database dan menampilkan data sesuai yang diinginkan.
-Query dalam bentuk tanggal + nama penyakit akan diparsing terlebih dahulu agar didapat 2 sintaks query, yaitu tanggal dan nama penyakit.
-Query tanggal akan digunakan untuk kolom tanggal pada database dan query nama penyakit akan digunakan untuk kolom nama penyakit pada database.

2. Fitur Fungsional & Arsitektur Web Aplikasi

A. Fitur Fungsional

- Search Database berdasarkan query yang diberikan
- Test DNA dengan memberikan input nama orang, nama penyakit, sekuens DNA serta metode yang dipilih
- Add Penyakit dengan memberikan nama penyakit serta sekuens DNA

B. Arsitektur Web Aplikasi

- Client side menggunakan bahasa pemrograman javascript dengan framework next.js. Client side juga dideploy pada platform hosting Vercel
- Server side menggunakan Bahasa pemrograman golang dengan framework. Server side dideploy pada platform hosting digital Ocean
- Database menggunakan postgresql yang dideploy pada platform hosting Heroku

BAB IV

IMPLEMENTASI DAN PENGUJIAN

1. Spesifikasi Teknis Program

A. Struktur Data

```go
package models

type Sickness struct {
    Id   int    `json:"id" gorm:"primaryKey"`
    Name string `json:"name"`
    DNA  string `json:"dna"`
}
```

```go
package models

type User struct {
    Id         int    `json:"id" gorm:"primaryKey"`
    Date       string `json:"date"`
    Name       string `json:"name"`
    Prediction string `json:"prediction"`
    Percentage int    `json:"percentage"`
    IsSick     bool   `json:"isSick"`
    DNA        string `json:"dna"`
    Method     string `json:"method"`
}
```

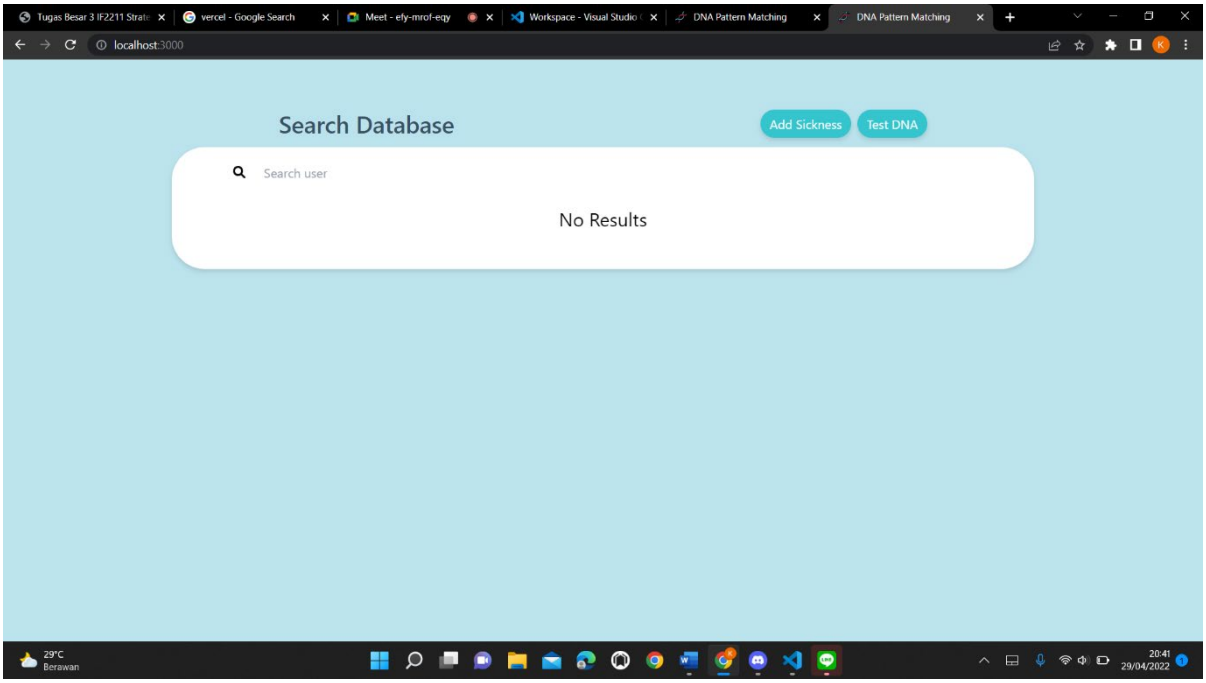
B. Fungsi dan Prosedur

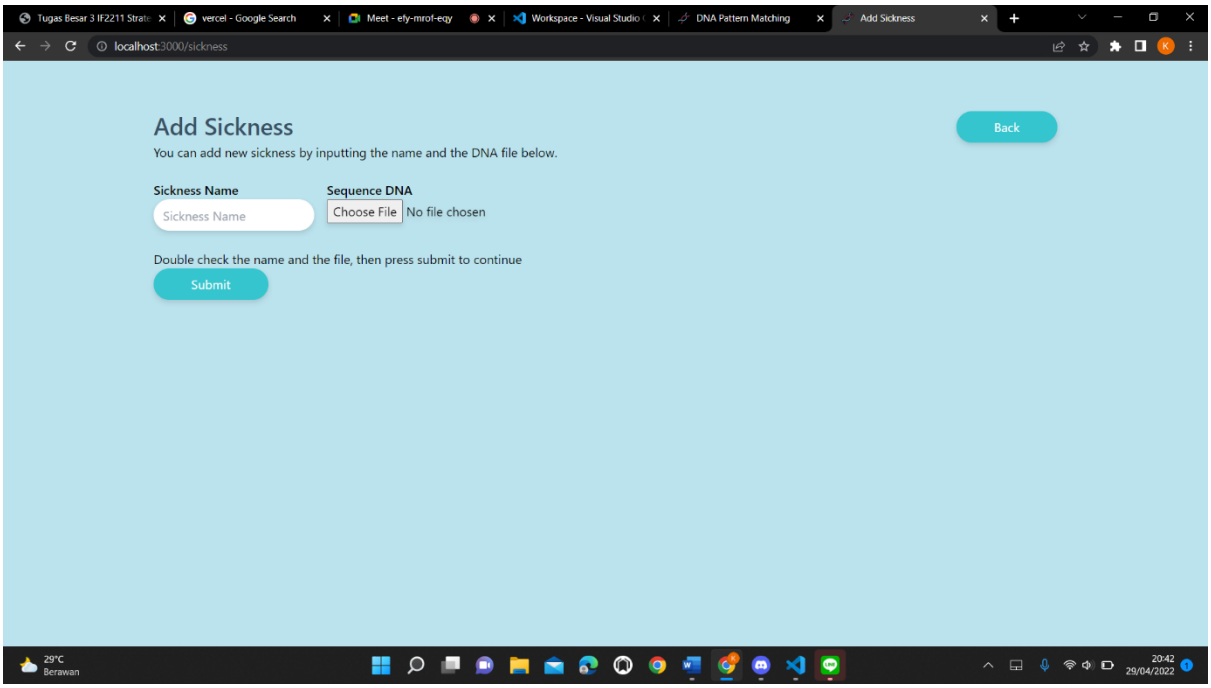
No	Fungsi dan Prosedur yang dibangun	Deskripsi
1	Getlastoccurrence(string) -> [128]int	Mengembalikan kejadian terakhir sebuah karakter untuk fungsi boyer-moore
2	Boyermoore(string, string)-> boolean	Mengembalikan true jika ada matching pattern dengan metode boyer-moore pada string, false jika kebalikan
3	Borderfunction(string, int, []int)	Mengubah [int] untuk fungsi border yang digunakan untuk fungsi KMP
4	KMP(string, string) -> Boolean	Mengembalikan true jika ada matching pattern dengan metode KMP pada string, false jika kebalikan

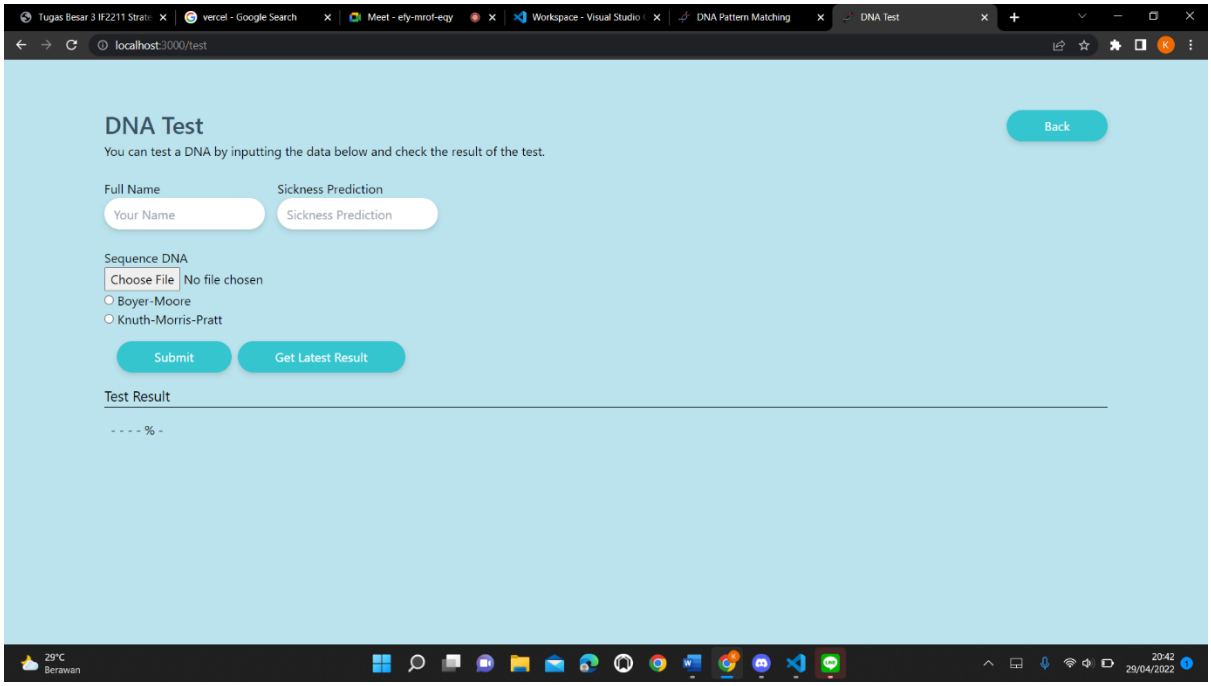
5	Hammingdistance(string, string)->int	Menghitung berapa huruf yang sama di antara 2 string yang sama panjang
6	Countsimilarity(string, string) -> int	Mengiterasi hammingdistance dan mengembalikan hammingdistance maksimum dalam bentuk persentase
7	IsValid(string) -> bool	Validasi sekuens dna dengan regex
8	IsDDMMYYYY(string) -> bool	Validasi tanggal dengan regex
9	IsDDMMYYYYandName -> bool	Validasi tanggal + nama dengan regex
10	Adduser()	Menambahkan data riwayat ke database
11	Addsickness()	Menambahkan data penyakit ke database

2. Tata Cara Penggunaan Program

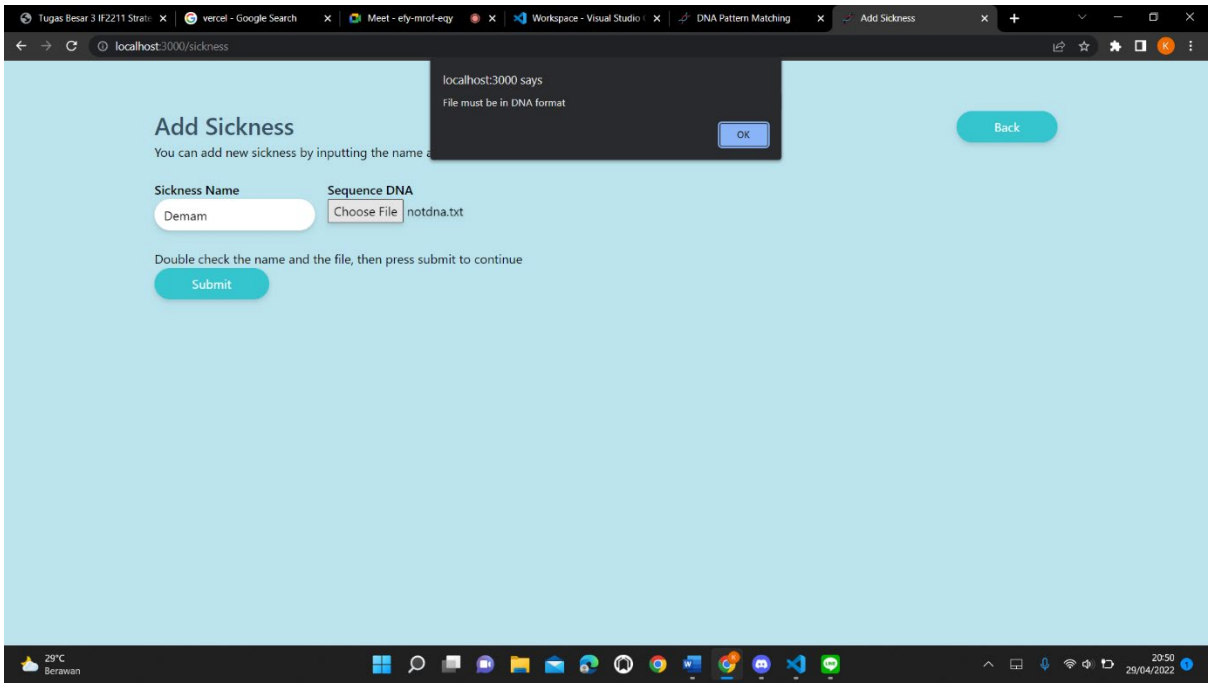
- Buka aplikasi pada url https://dna.kentang.fun/ atau deploy program manual
- Pada page awal, akan disambut dengan page search database. Masukkin query dengan sintaks “dd/mm/yyyy” dan/atau “{nama_penyakit}”. Setelah memasukkan query akan otomatis
- Buka page add sickness untuk menambahkan penyakit. Masukkan nama penyakit serta sekuens dna sesuai ketentuan
- Buka page test DNA untuk melakukan test dna. Masukkan nama orang, nama penyakit, sekuens dna, serta metode untuk test dna. Pencet tombol submit untuk mensubmit ke database, Pencet tombol get latest result untuk mendapatkan hasil terakhir

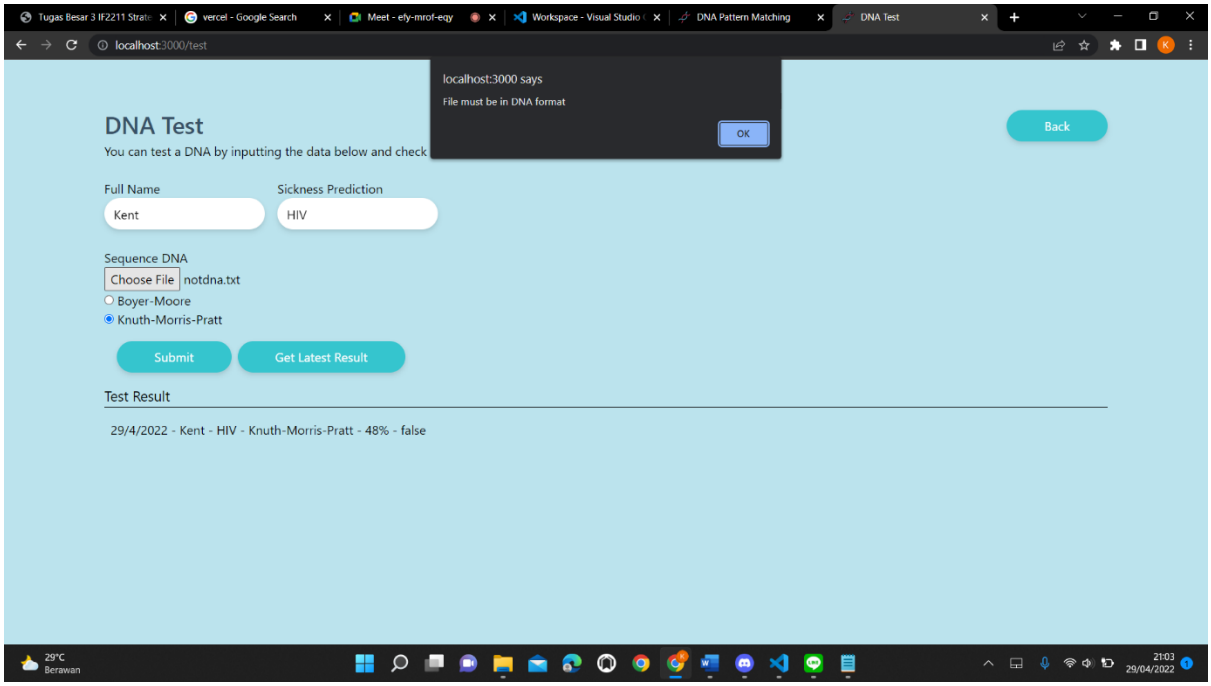

Search Database
Add Sickness
Test DNA
Search user
No Results


Add Sickness
You can add new sickness by inputting the name and the DNA file below.
Back
Sickness Name
Sickness Name
Sequence DNA
Choose File
No file chosen
Double check the name and the file, then press submit to continue
Submit

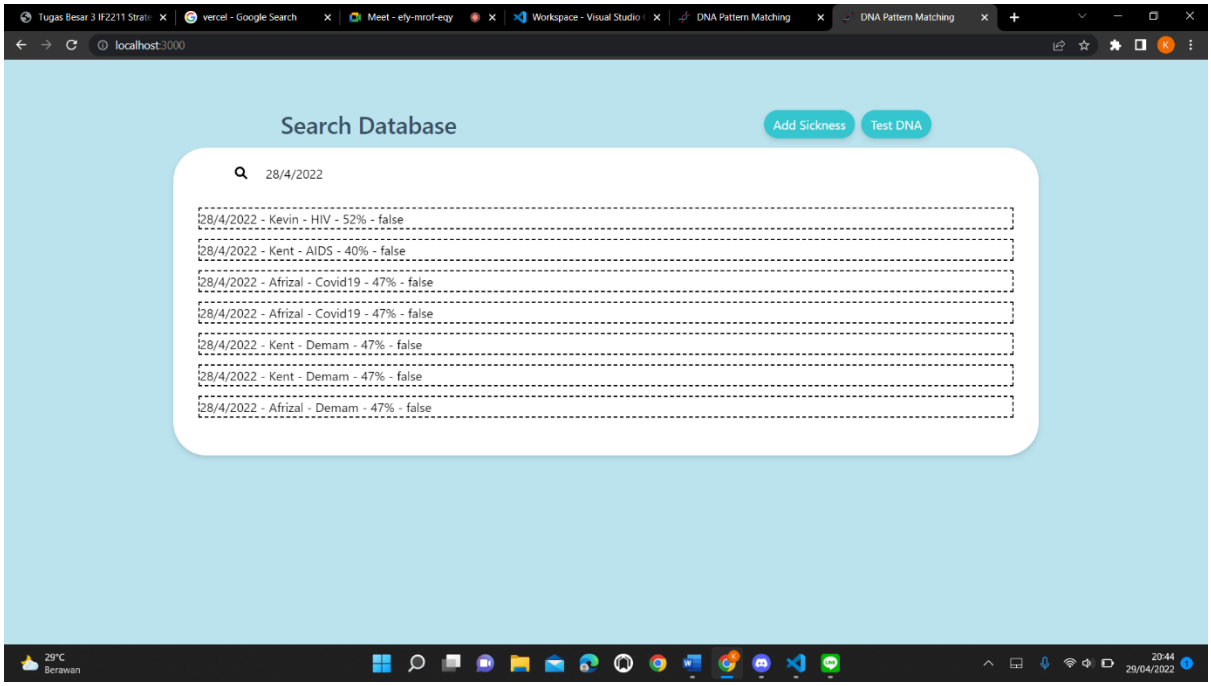


3. Hasil Pengujian

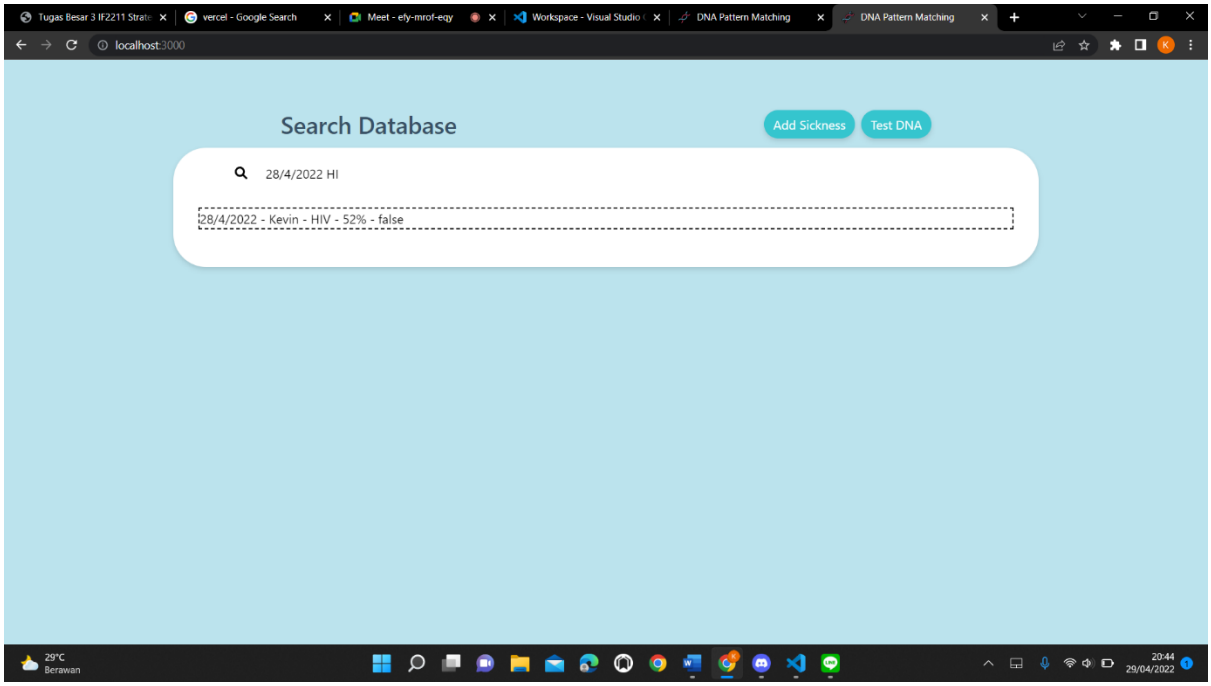




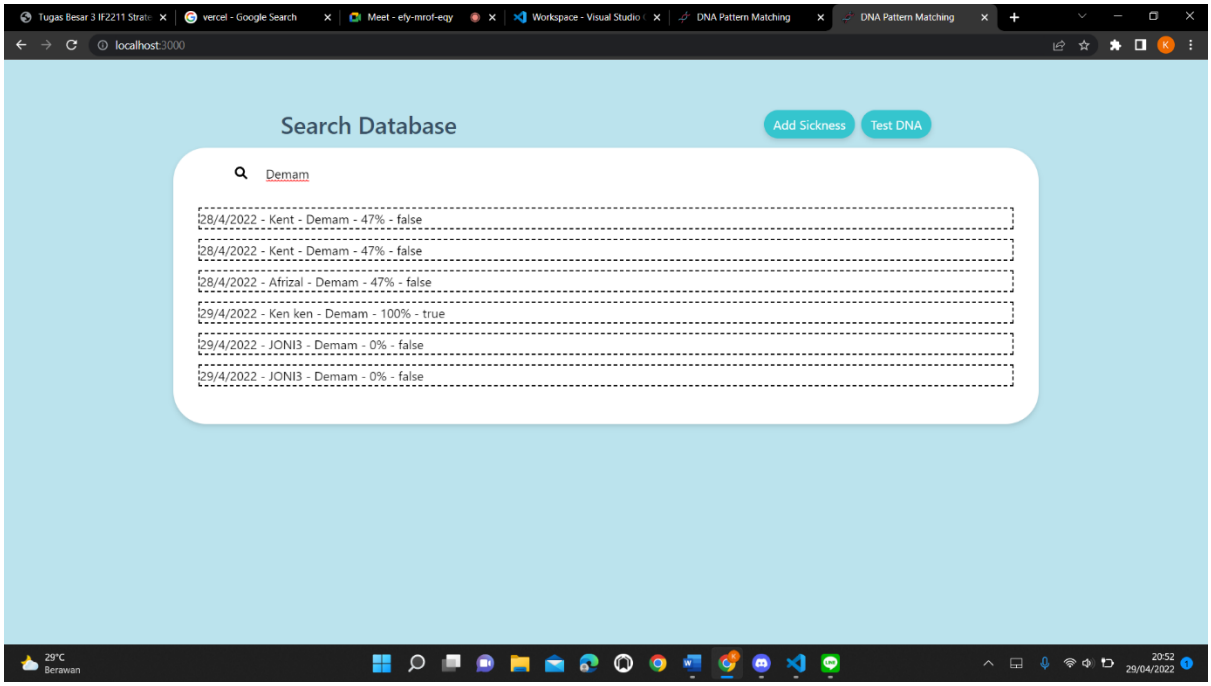
Sanitasi input



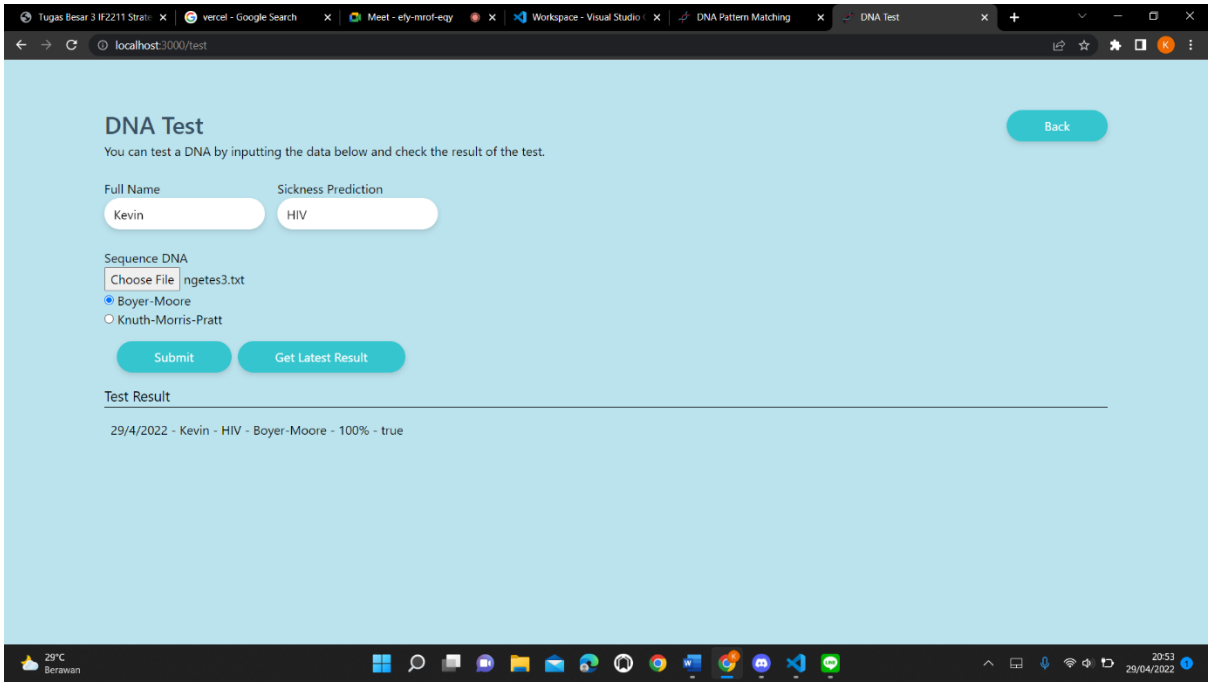
Pencarian dengan tanggal



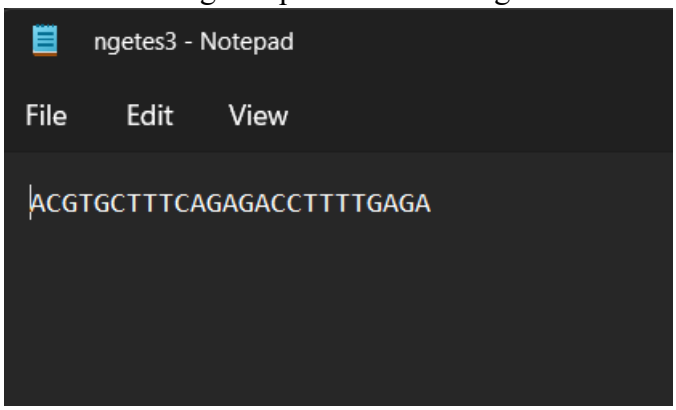
Pencarian dengan tanggal + nama



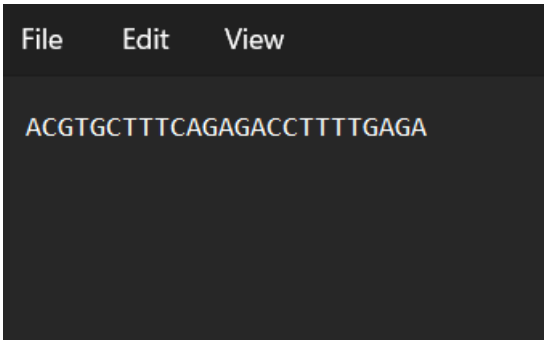
Pencarian dengan nama penyakit



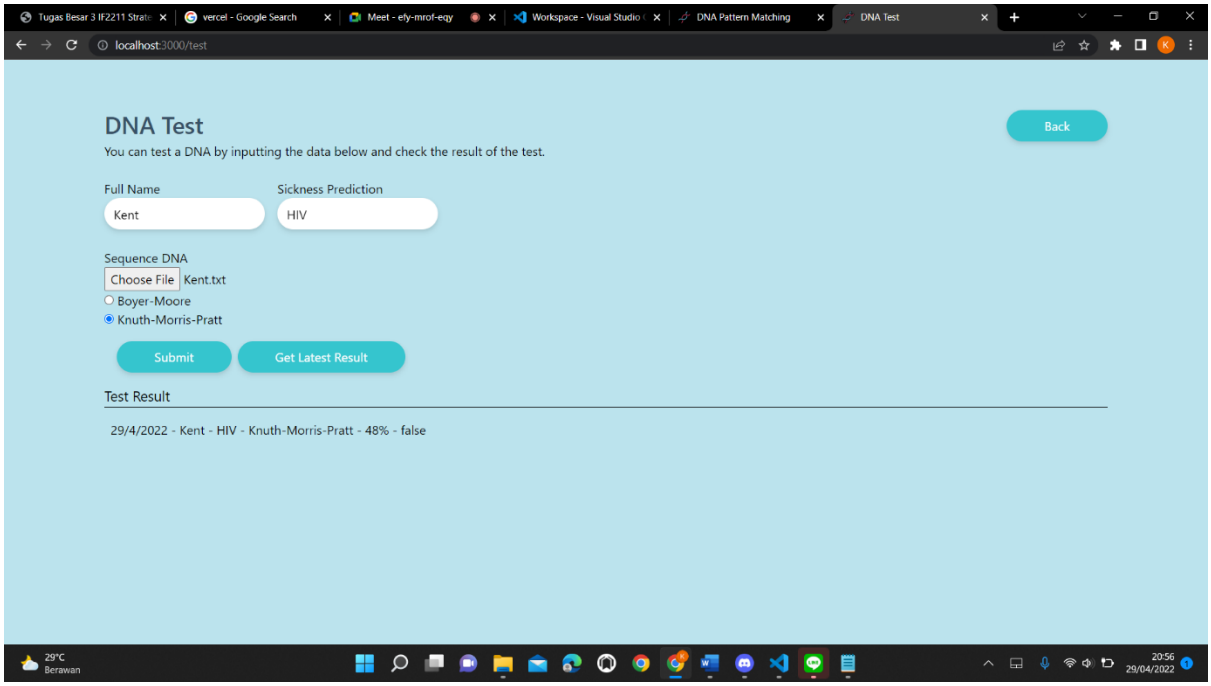
Tes dna dengan input sekuens sebagai berikut



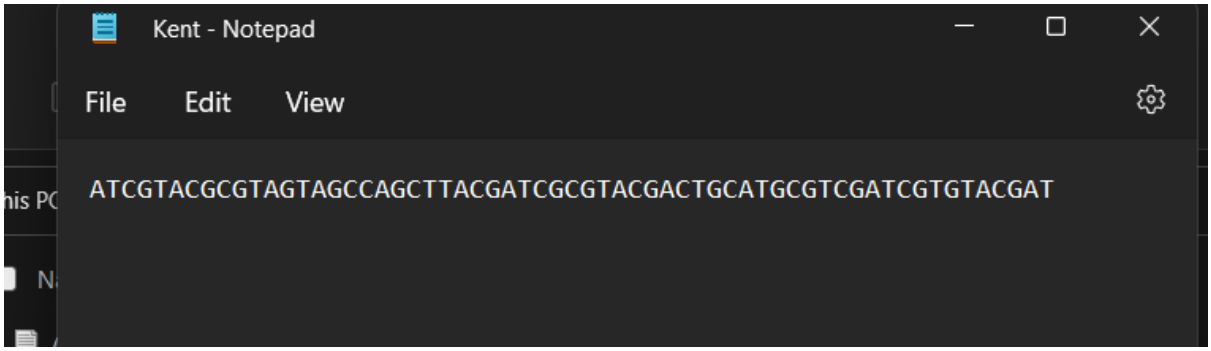
dan sekuens hiv sebagai berikut

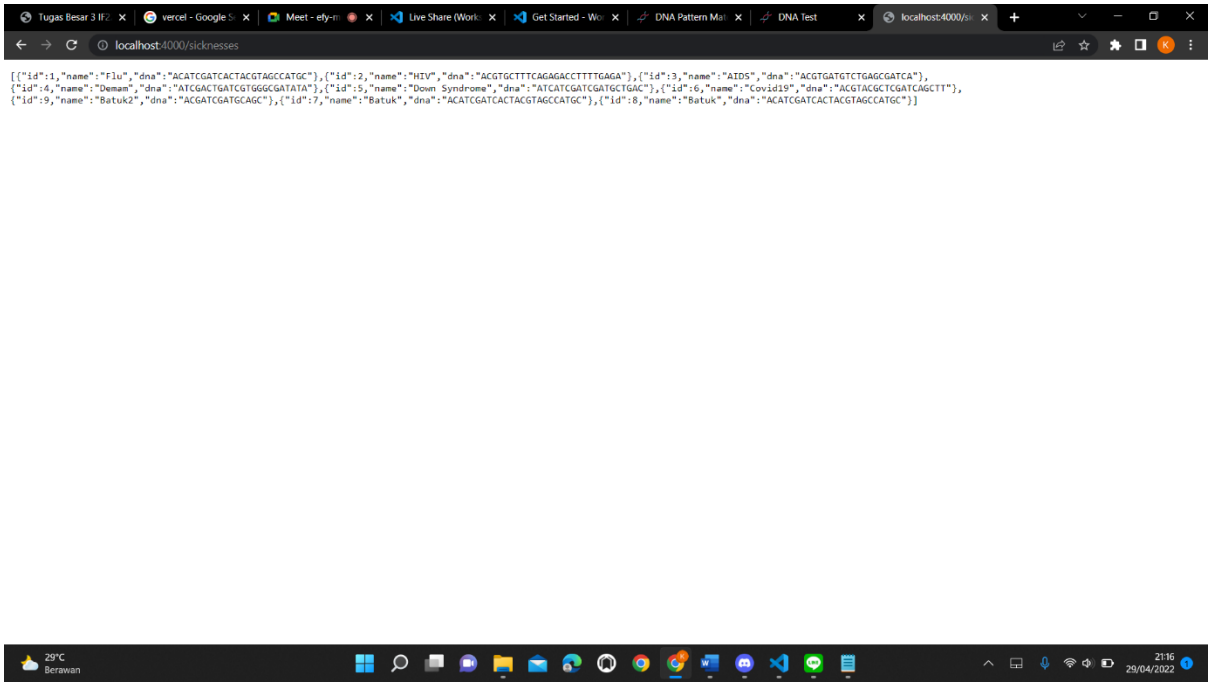


Contoh lainnya

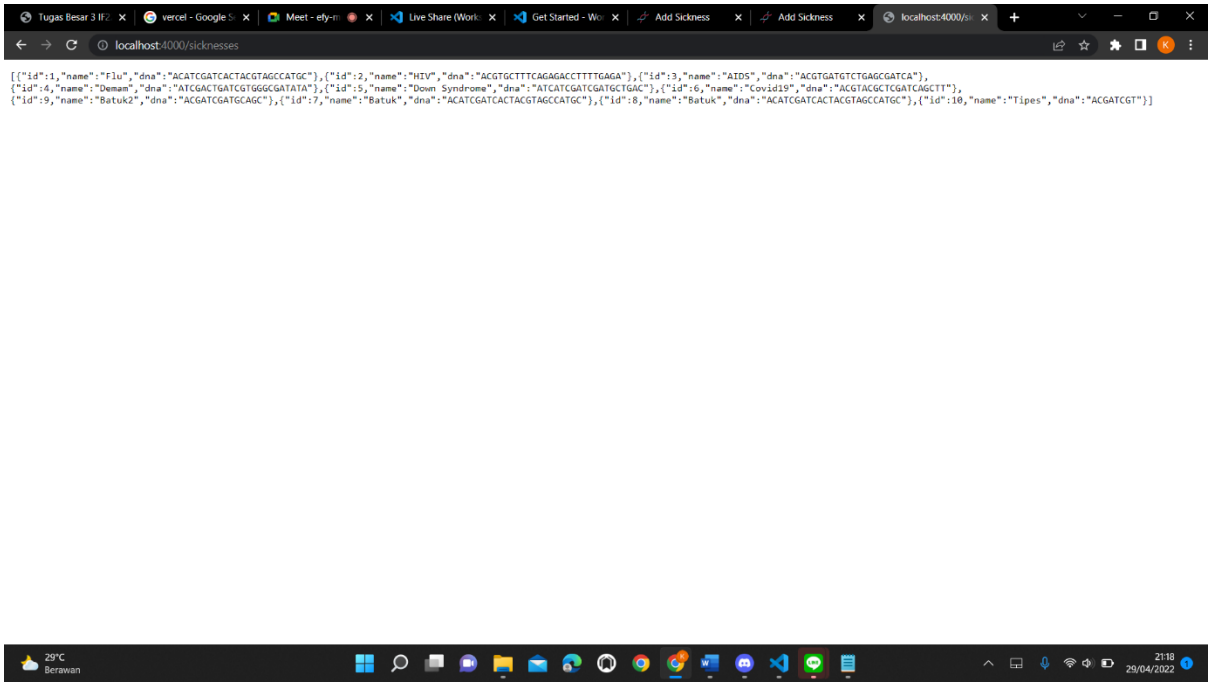


Dengan sekuens sebagai berikut





Daftar penyakit sebelum ditambahkan



Setelah ada penambahan penyakit tipes

4. Analisis Hasil Pengujian

Berdasarkan hasil pengujian, fitur searching database sudah dapat berjalan dengan baik. Query sudah dapat berjalan sesuai dengan 3 skenario yang terjadi.

Fitur add penyakit juga sudah dapat menjalankan fungsinya dengan baik. Menambahkan database serta juga dapat mensanitasi input dengan baik. Selain itu juga sudah bisa menambahkan data ke database

Fitur Test DNA sudah dapat menjalankan fungsi dengan baik. Fitur sudah dapat mensanitasi input serta dapat menerima kedua tipe algoritma pencarian dengan baik. Selain itu, sudah dapat menambahkan data ke database.

BAB V
KESIMPULAN DAN SARAN

I. Kesimpulan

Test DNA dapat dilakukan dengan cara string matching dengan algoritma KMP dan Boyer-Moore. Program kami juga sudah dapat mengimplementasikannya dalam pembuatan aplikasi website yang telah kami susun. Web aplikasi yang kami susun juga sudah bisa menghubungkan Frontend, Backend, serta Database meskipun dideploy pada platform hosting yang berbeda.

II. Saran

Tugas yang diberikan sangat baik untuk eksplorasi mandiri dalam web programming serta menambah wawasan dalam pemakaian basis data. Alangkah baik apabila diberikan sumber untuk pembelajaran yang lebih banyak lagi agar lebih mudah dalam proses eksplorasi.

Daftar Pustaka

Munir, R. 2022. Pencocokan string (String matching/pattern matching) (2022). Slide kuliah IF2211 Strategi Algoritma, Institut Teknologi Bandung.

Munir, R. 2022. Pencocokan string Pencocokan string dengan Regular Expression (Regex) (2022). Slide kuliah IF2211 Strategi Algoritma, Institut Teknologi Bandung.

Kumpulan Link

Source Code: https://github.com/kentlius/Tubes3_13520069

Deploy: https://dna.kentang.fun

Video Demo: https://youtu.be/ORJZA9FALPQ