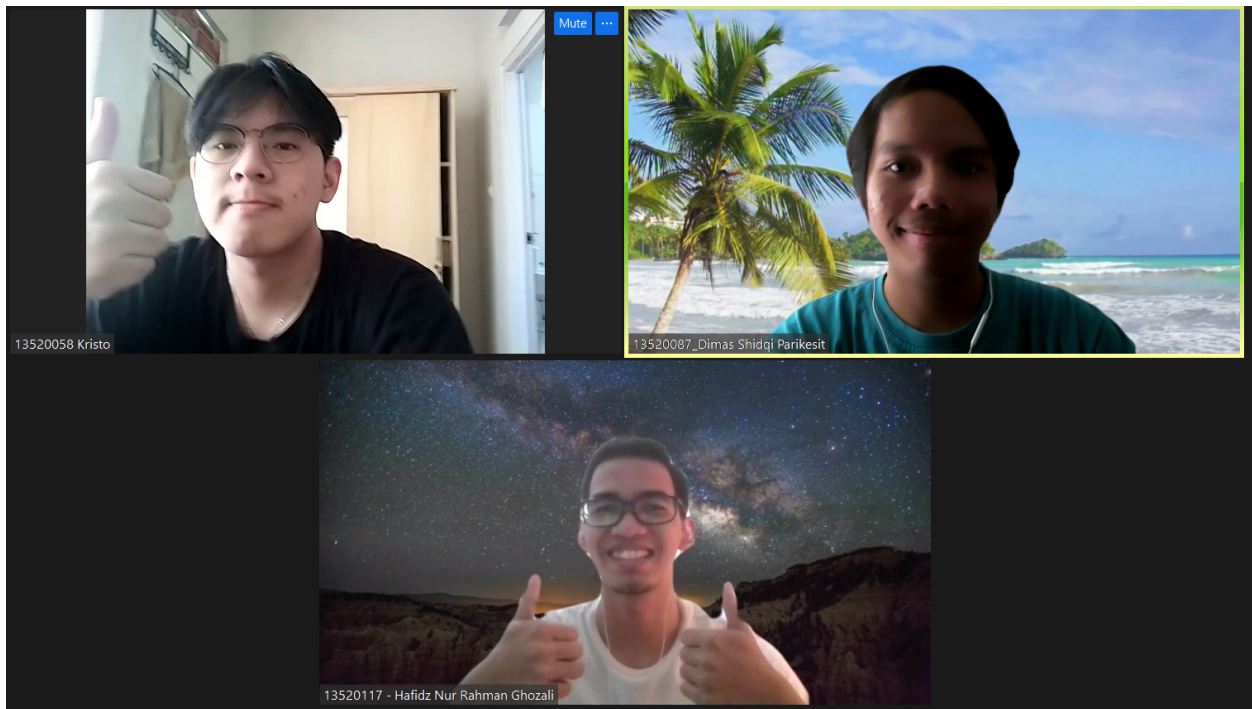


LAPORAN TUGAS BESAR III

PENERAPAN STRING MATCHING DAN REGULAR EXPRESSION DALAM DNA PATTERN MATCHING

Laporan dibuat untuk memenuhi salah satu tugas mata kuliah

IF2211 Strategi Algoritma



Disusun oleh:
Kelompok 54 Backend

Kristo Abdi Wiguna	13520058
Dimas Shidqi Parikesit	13520087
Hafidz Nur R. G.	13520117

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

2022

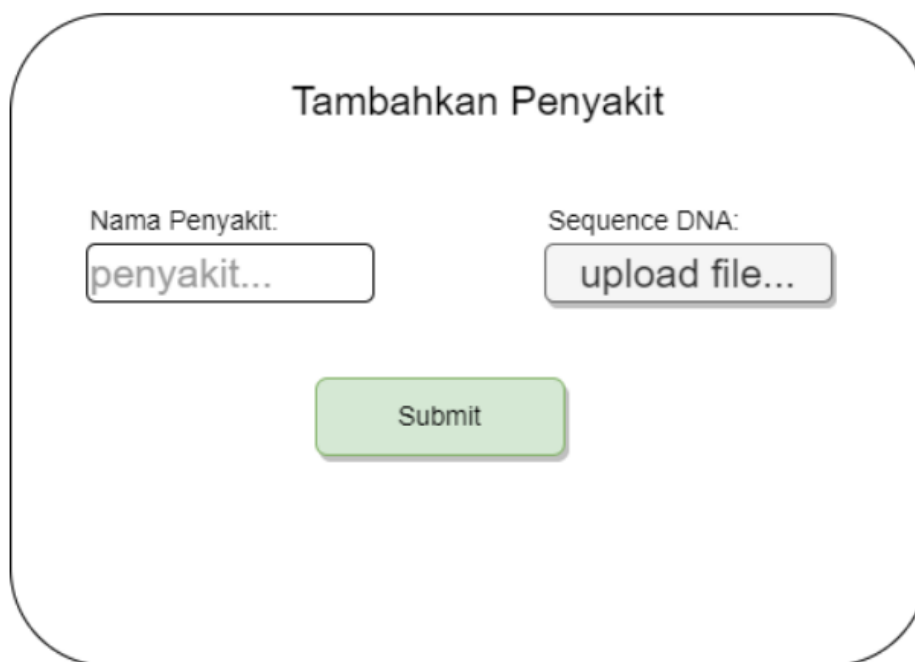
DAFTAR ISI

DAFTAR ISI	1
Bab 1 Deskripsi Tugas	2
Bab 2 Landasan Teori	7
2.1. Dasar Teori	7
2.2 Penjelasan Aplikasi Web	10
Bab 3 Analisis Pemecahan Masalah	11
3.1 Langkah Pemecahan Masalah	11
3.2 Fitur Fungsional dan Arsitektur Aplikasi Web	12
Bab 4 Implementasi dan Pengujian	14
4.1 Spesifikasi Teknis Program	14
4.2 Penjelasan Tata Cara Penggunaan Program	15
4.3 Hasil Pengujian	18
4.3.1 Pengujian Penambahan Penyakit	18
4.3.2 Pengujian Tes DNA	19
4.3.3 Pengujian Pencarian Riwayat Tes DNA	22
4.4 Analisis Hasil Pengujian	25
Bab 5 Kesimpulan dan Saran	27
Link Video	27
Link Repository	27
Link Website	27
Daftar Pustaka	28

Bab 1 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.

1. Aplikasi dapat menerima input penyakit baru berupa nama penyakit dan sequence DNA-nya (dan dimasukkan ke dalam database).
 - a. Implementasi input sequence DNA dalam bentuk file.
 - b. Dilakukan sanitasi input menggunakan regex untuk memastikan bahwa masukan merupakan sequence DNA yang valid (tidak boleh ada huruf kecil, tidak boleh ada huruf selain AGCT, dan tidak ada spasi).
 - c. Contoh input penyakit:



The image shows a web form titled "Tambahkan Penyakit" (Add Disease). It contains two input fields: "Nama Penyakit:" (Disease Name) with a placeholder text "penyakit..." and "Sequence DNA:" with a placeholder text "upload file...". Below these fields is a green "Submit" button.

Gambar 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 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: , contoh “13 April 2022 HIV”.

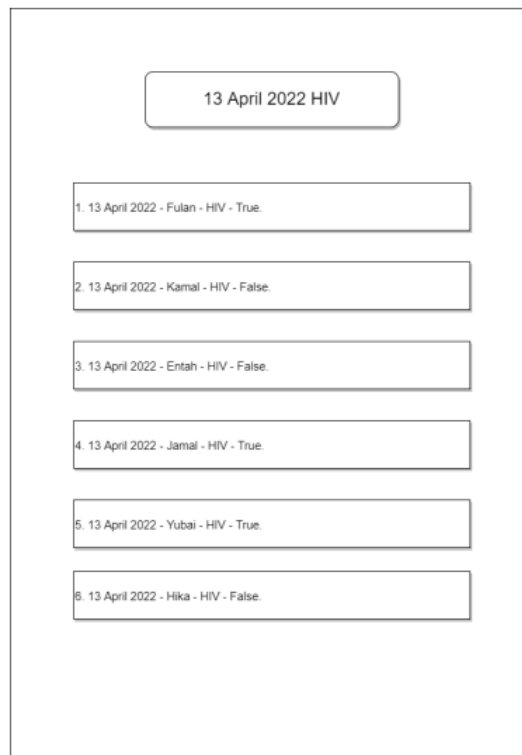
Format penanggalan dibebaskan, jika bisa menerima >1 format lebih baik.

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 screenshot shows a web application 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 search 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

The screenshot shows a web application interface. At the top, there is a rounded rectangular input field containing the text "13 April 2022". Below this, there are six rectangular boxes, each containing a numbered list item. The items are as follows:

- 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 4. Ilustrasi Interaksi 2

iii. Masukan hanya nama penyakit

The screenshot shows a web application interface. At the top, there is a rounded rectangular input field containing the text "HIV". Below this, there are six rectangular boxes, each containing a numbered list item. The items are as follows:

- 1. 13 April 2022 - Fulan - HIV - True.
- 2. 14 April 2022 - Kamal - HIV - False.
- 3. 15 April 2022 - Entah - HIV - False.
- 4. 16 April 2022 - Jamal - HIV - True.
- 5. 17 April 2022 - Yubai - HIV - True.
- 6. 18 April 2022 - Hika - HIV - False.

Gambar 5. Ilustrasi Interaksi 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 2 Landasan Teori

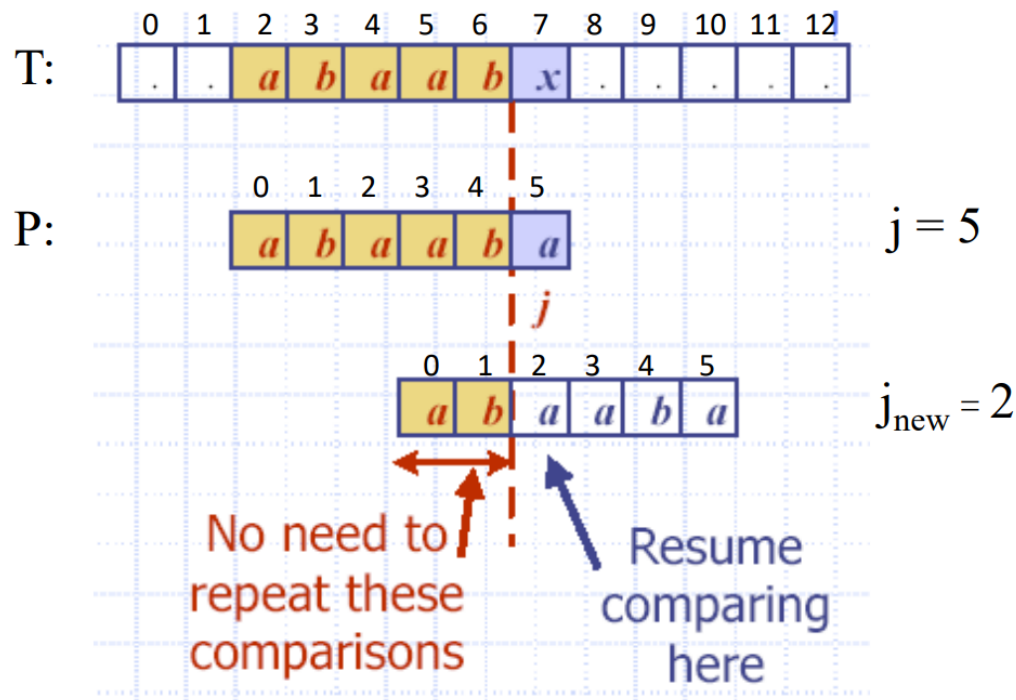
2.1. Dasar Teori

Algoritma KnuthMorrisPratt merupakan salah satu algoritma pencarian string yang dikembangkan secara terpisah oleh Donald E. Knuth pada tahun 1967 dan James H. Morris dengan Vaughan R. Pratt pada tahun 1966, tetapi keduanya menerbitkan secara bersamaan pada tahun 1977. Jika algoritma brute force dilihat lebih dalam, dengan mengingat beberapa perbandingan sebelumnya, kita dapat meningkatkan besarnya pergeseran sehingga akan menghemat perbandingan dan lebih mempercepat pencarian.

Pada algoritma KMP, informasi yang digunakan untuk melakukan pergeseran lebih jauh akan disimpan, tidak hanya satu karakter per karakter diperbandingkan seperti algoritma brute force. Algoritma ini melakukan pencocokan dari kiri ke kanan.

Kompleksitas algoritma pencocokan string dengan KMP ini dihitung dari kompleksitas untuk menghitung fungsi pinggiran dan pencarian string. Untuk menghitung fungsi pinggiran dibutuhkan waktu $O(m)$, sementara untuk pencarian string dibutuhkan $O(n)$, sehingga kompleksitas waktu algoritma KMP adalah $O(m + n)$.

Fungsi pinggiran $b(k)$ dapat dicari dengan menghitung panjang prefix yang sama dengan suffix dari string pattern. Dengan membuat *array* panjang yang sama dengan panjang pattern, tiap indeks k array tersebut akan diisi angka yang merupakan panjang terbesar prefix $P[0..k]$ dan suffix $P[1...k]$ yang sama dengan k adalah $j-1$ dimana j adalah indeks terjadinya gagal komparasi yang berbeda.



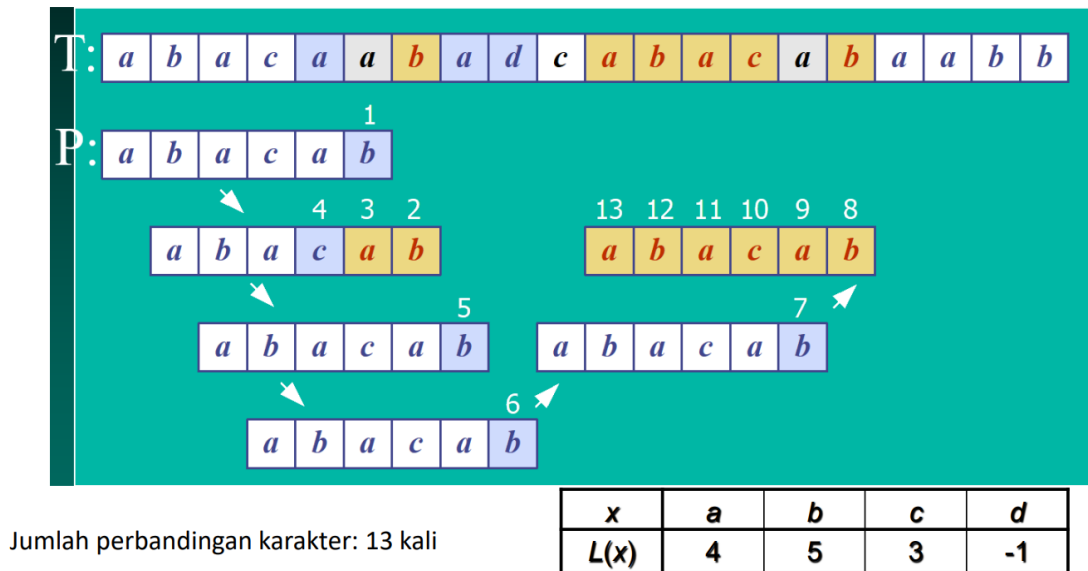
Gambar 6. Ilustrasi KMP

Algoritma Boyer-Moore adalah salah satu algoritma pencarian string, dipublikasikan oleh Robert S. Boyer, dan J. Strother Moore pada tahun 1977. Algoritma ini dianggap sebagai algoritma yang paling efisien pada aplikasi umum. Tidak seperti algoritma pencarian string yang ditemukan sebelumnya, algoritma Boyer-Moore mulai mencocokkan karakter dari sebelah kanan pattern. Ide di balik algoritme ini adalah bahwa dengan memulai pencocokan karakter dari kanan, dan bukan dari kiri, maka akan lebih banyak informasi yang didapat.

Jika algoritma KMP merupakan algoritma yang tidak memperdulikan isi dari string pattern dan text maka, algoritma Boyer-Moore adalah algoritma yang memperdulikan isi dari string pattern dan text. Pada awalnya, algoritma Boyer-Moore akan mencari fungsi kemunculan terakhir yang dimana didapatkan dengan mengiterasi teks pattern dan mencari kemunculan terakhir karakter yang ada di pattern. Misalnya jika pattern berisi “ATTACGTAGCTAG” maka fungsi kemunculan terakhir akan mencari indeks kemunculan terakhir dari tiap karakter yang ada di pattern yaitu A, T, C, dan G yang masing - masing bernilai 11, 10, 9, dan 12.

Boyer-Moore memiliki kompleksitas $O(nm + A)$ untuk worst case. Namun, Boyer-Moore sangat efisien jika teks yang diberikan cukup besar dan lambat jika teks yang diberikan pendek.

Algoritma Boyer-Moore memiliki keefisienan tinggi ketika mengiterasi teks berbahasa Inggris dan keefisienan rendah ketika mengiterasi teks biner.



Gambar 7. Ilustrasi Boyer-Moore

Regular expression merupakan notasi yang digunakan untuk mendeskripsikan himpunan karakter string. Sebagai contoh, himpunan yang terdiri dari string “Handel”, “Händel”, dan “Haendel” dapat dideskripsikan dengan “H(ä|ae?)ndel” (atau alternatif lainnya, notasi ini menunjukkan bahwa notasi tersebut sesuai dengan setiap string yang diberikan).

Regular expression didefinisikan berdasarkan aturan teori bahasa formal. Regular expression terdiri dari konstanta dan operator yang menunjukkan himpunan-himpunan string dan operasi antar himpunan string tersebut secara berurutan.

Konstanta yang telah didefinisikan adalah:

- Himpunan kosong, diberi notasi \emptyset .
- String kosong, diberi notasi ϵ .
- Karakter, diberi notasi sesuai dengan karakter bahasa yang digunakan.

Operator yang telah didefinisikan adalah:

- Konkatenasi, misal $\{ "ab", "c" \} \{ "d", "ef" \} = \{ "abd", "abef", "cd", "cef" \}$.

- Alternasi, misal $\{"ab", "c"\} \mid \{"ab", "d", "ef"\} = \{"ab", "c", "d", "ef"\}$.
- Kleene star, menunjukkan semua himpunan yang dapat dibuat dengan melakukan konkatenasi 0 atau lebih banyak string dari string yang dilakukan .

2.2 Penjelasan Aplikasi Web

Aplikasi web yang kelompok kami buat menggunakan bahasa pemrograman Golang untuk backend dan bahasa pemrograman React untuk frontend dan menggunakan PostgreSQL untuk database penyimpanan DNA sequence. Algoritma yang diimplementasikan yaitu KMP, Boyer-Moore, dan mengimplementasikan bonus yaitu Similarity Pattern Matching yang didasarkan algoritma Levenshtein Distance.

Dalam section backend, disimpan beberapa tipe data model untuk menyimpan data di database yaitu terdiri dari Disease, History, QueryMatch, dan QueryHistory. Model disease untuk menyimpan nama orang dan DNA sequence yang mereka miliki. History model menyimpan riwayat pencocokan DNA sequence seseorang terhadap DNA sequence penyakit tertentu, similaritas, hasil pemeriksaan exact matching, dan tanggal pemeriksaan. Query Match berfungsi untuk menyimpan riwayat query nama, DNA sequence seseorang, dan penyakit. Query History berfungsi untuk menyimpan nama orang dan tanggal pemeriksaan pencocokan DNA dalam bentuk satu string.

Dalam section frontend, disimpan beberapa pages yang tersedia yaitu 3, page pencarian riwayat, page penambahan DNA sequence suatu penyakit, dan page pemeriksaan DNA sequence seseorang terhadap suatu penyakit.

Bab 3 Analisis Pemecahan Masalah

3.1 Langkah Pemecahan Masalah

Terdapat 3 fitur pada program ini. Pertama adalah menambahkan penyakit pada database, kedua melakukan pencocokan antara dna masukan dengan dna yang berada di database, ketiga adalah melakukan pencarian riwayat pencocokan.

1. Menambahkan penyakit pada database

Dalam menambahkan penyakit pada database terdapat beberapa langkah yang harus diikuti.

1. Pengguna memasukkan nama penyakit serta file yang berisi sequence dna ke dalam antar muka web.
2. Melakukan pembacaan isi file secara client-side.
3. Data nama penyakit serta sequence dna tersebut dikirimkan ke server-side dalam bentuk JSON.
4. Dilakukan validasi isi sequence dna yang akan dimasukkan.
5. Apabila valid, maka kemudian dilakukan pemanggilan query pada database untuk mencari apakah penyakit yang akan dimasukkan sudah ada di database atau belum.
6. Apabila sudah ada di database, server akan mengirimkan respon berisi error.
7. Apabila belum, maka server akan melakukan proses memasukkan data pada database.
8. Apabila proses memasukkan server tersebut gagal, maka server akan mengirimkan respon berisi error.
9. Apabila berhasil, maka server akan mengirimkan respon berupa string OK.

2. Pencocokan dna masukan dengan dna yang berada di database

Tahapan yang perlu dilakukan untuk melakukan pencocokan dna adalah

1. Memasukkan nama orang, sequence dna orang tersebut, dan nama penyakit yang dicocokkan. Sequence dna dimasukkan dalam bentuk file, dimana file tersebut akan dibaca oleh client-side.
2. Mengirimkan ke server-side dalam bentuk JSON.

3. Dilakukan validasi sequence DNA tersebut.
4. Kemudian akan dicari penyakit yang sesuai pada database.
5. Apabila ditemukan, digunakan algoritma pencocokan string sesuai keinginan pengguna.
6. Apabila hasilnya tidak sama, akan digunakan algoritma similarity untuk mencari tahu seberapa mirip dna pengguna dengan penyakit pada database.
7. Hasil pencocokan dan penghitungan similarity tersebut dimasukkan pada database.
8. Server akan memberikan respon sebuah string hasil pencocokan tersebut.

3. Melakukan pencarian riwayat pencocokan

Pencarian riwayat pencocokan dilakukan dengan tahapan berikut ini

1. Validasi input query pengguna
2. Mengirim query dari client ke server
3. Pencarian riwayat pada database
4. Mengirim data dari server ke client
5. Menampilkan data pada client

3.2 Fitur Fungsional dan Arsitektur Aplikasi Web

Aplikasi web memiliki tiga fitur, yaitu

1. Penambahan data penyakit pada database
2. Pencocokan dna pengguna dengan database penyakit
3. Melihat history pencocokan.

Aplikasi web dibuat dengan menggunakan Golang pada sisi backend, React JS pada sisi frontend, dan PostgreSQL sebagai database. Folder frontend dan backend dipisahkan, sementara database langsung disimpan di cloud dengan menggunakan penyedia layanan Supabase.

Pada sisi backend, digunakan framework fiber untuk pembuatan server, dan orm Gorm untuk mempermudah komunikasi antara database dengan server.

Pada sisi frontend, digunakan library React JS dengan typescript untuk pembuatan client, react-bootstrap untuk memperindah tampilan, serta react-router untuk menangani perpindahan antar halaman web.

Bab 4 Implementasi dan Pengujian

4.1 Spesifikasi Teknis Program

a. Struktur Data

Dalam pemindahan data dari client ke server dan sebaliknya digunakan struktur data JSON atau *Javascript Object Notation* dan string. Kedua struktur data ini memungkinkan pemindahan yang fleksibel untuk digunakan pada keseluruhan aplikasi web ini.

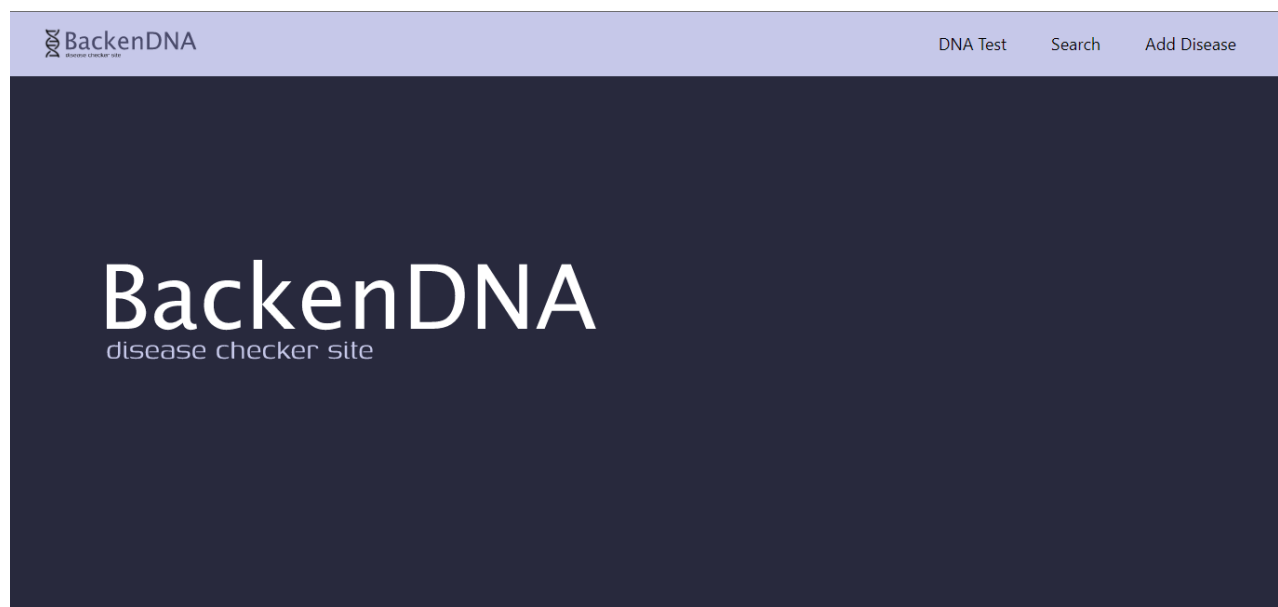
b. Fungsi dan Prosedur

No	Fungsi / Prosedur	Keterangan
1	IsValidDNA(s)	Fungsi menerima parameter string sequence DNA s dan mengeluarkan boolean hasil pengecekan terhadap RegEx DNA yang valid
2	IsValidInputSearch(s)	Fungsi menerima parameter string sequence DNA s dan mengeluarkan boolean hasil pengecekan terhadap RegEx input yang valid
3	IsValidDate(s)	Fungsi menerima parameter string sequence DNA s dan mengeluarkan boolean hasil pengecekan terhadap RegEx tanggal yang valid
4	BMMatch(dna, disease)	Fungsi menerima parameter dua buah string yaitu sequence dna orang yang dicek dan dna penyakit. Fungsi ini mengembalikan boolean apakah sama atau tidak setelah dicek menggunakan algoritma Boyer-Moore.
5	KMPMatch(dna, disease)	Fungsi menerima parameter dua buah string yaitu sequence dna orang yang dicek dan dna penyakit. Fungsi ini mengembalikan boolean apakah sama atau tidak setelah dicek menggunakan algoritma Knuth Morris Pratt.
4	CalculateLevenshtein Dist(dna, disease)	Fungsi menerima parameter dua buah string yaitu sequence dna orang yang dicek dan dna penyakit. Fungsi ini mengembalikan variabel float yang menyatakan tingkat kemiripan dna sequence orang dan dna penyakit.
6	SimilarityMatching(dna, disease)	Fungsi menerima parameter dua buah string yaitu sequence dna orang yang dicek dan dna penyakit. Fungsi ini mengembalikan 2 variabel yaitu, boolean apakah sequence dna dan penyakit mirip, dan float tingkat kemiripan tertinggi similarity dna sequence dan

		sequence penyakit.
7	ConvertTime(time)	Fungsi menerima parameter tipe data date time dan mengembalikan string pengejaan tanggal dan error
8	ConvertString(string)	Fungsi menerima parameter tipe data string dan mengembalikan tipe data date time representasi tanggal dan error
9	ConnectSupabase	Prosedur untuk membuat koneksi dengan database cloud
10	SplitText(text, delimiter)	Fungsi menerima parameter string dan delimiter kemudian mengembalikan array string dari text yang dipisahkan oleh delimiter
11	JoinArray(arr, idx)	Fungsi menerima parameter array string dan index dimulainya penggabungan kemudian mengembalikan gabungan menjadi string yang dihasilkan

4.2 Penjelasan Tata Cara Penggunaan Program

Pada saat pertama kali memasuki laman web, pengguna akan disambut oleh *homepage* seperti berikut

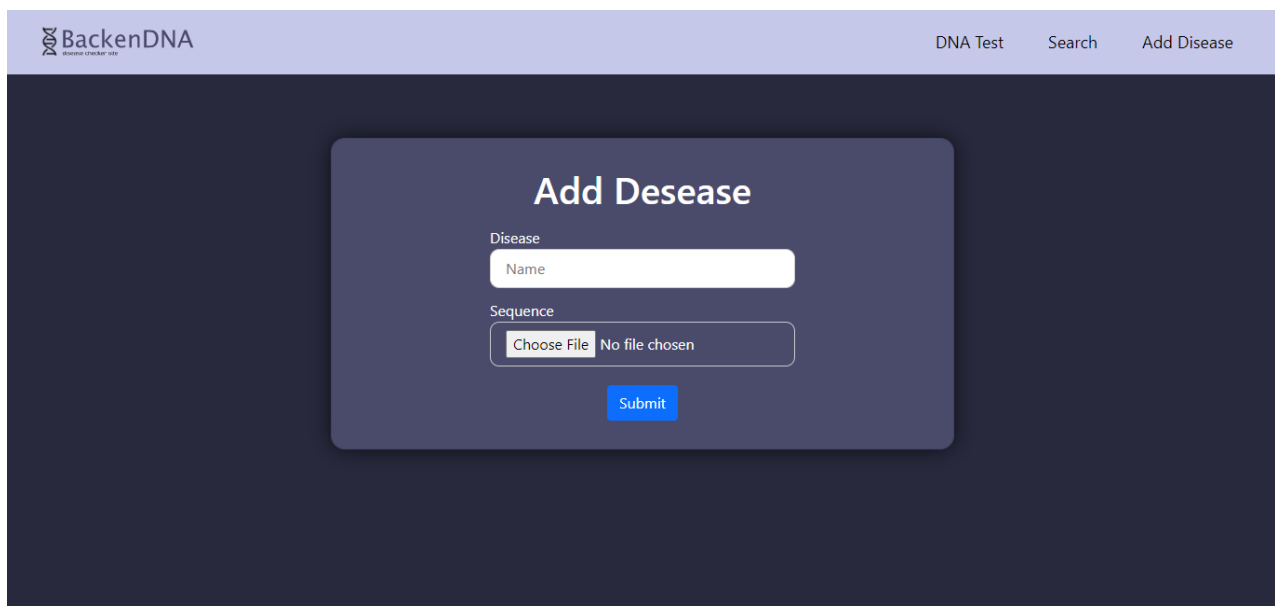


Gambar 8. Laman homepage

Aplikasi web memiliki tiga fitur utama, yaitu

1. Penambahan data penyakit pada database

- Pengguna menuliskan nama penyakit, nama penyakit hanya boleh alphanumeric, spasi, dan karakter “-”
- Pengguna mengunggah file berisi teks rantai DNA penyakit tersebut
- Pengguna menekan tombol submit
- Setelah tombol submit ditekan, akan muncul *pop up* berisi informasi error atau konfirmasi sukses dalam menambahkan penyakit



Gambar 9. Laman Tambah Penyakit

2. Pencocokan dna pengguna dengan database penyakit

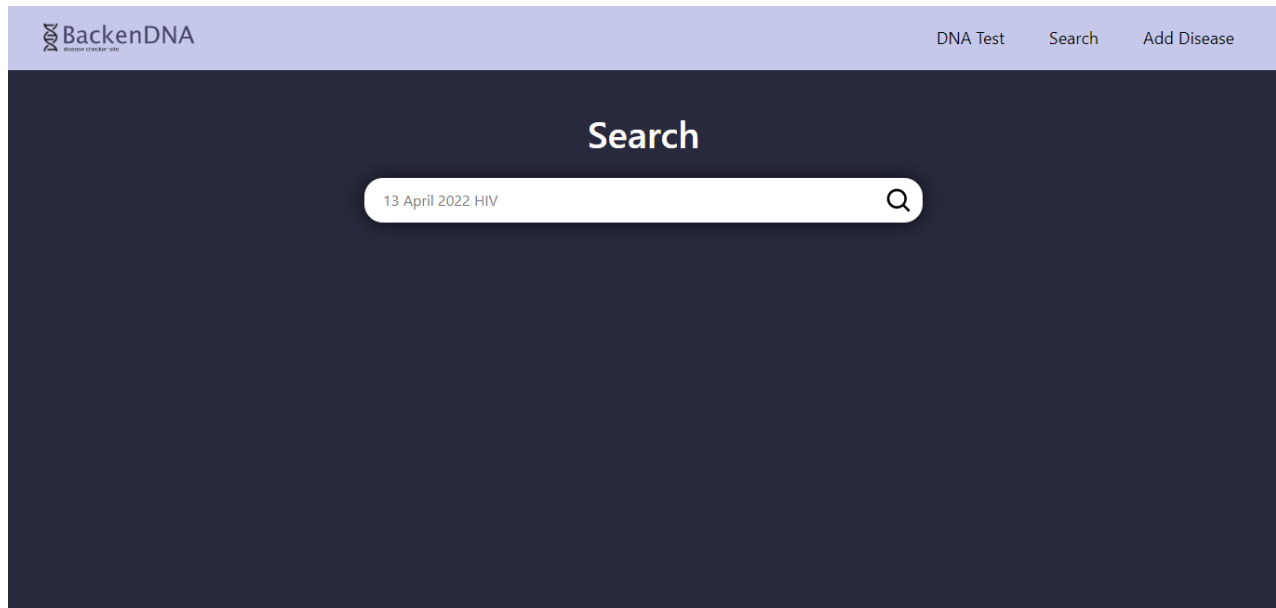
- Pengguna memasukkan nama dan nama penyakit, nama penyakit hanya boleh alphanumeric, spasi, dan karakter “-”
- Pengguna mengunggah file berisi teks rantai DNA pengguna
- Pengguna memilih algoritma yang akan digunakan dalam string matching. Secara *default* akan memilih algoritma Boyer-Moore
- Pengguna menekan tombol submit
- Setelah tombol submit ditekan, akan muncul *pop up* berisi informasi error atau konfirmasi sukses dalam menambahkan penyakit

The screenshot shows a web application interface for a DNA test. At the top, there is a header bar with the logo 'BackenDNA' on the left and navigation links 'DNA Test', 'Search', and 'Add Disease' on the right. The main content area is dark blue and features a central white form titled 'DNA Test'. The form contains the following fields and controls: a 'Username' section with a text input labeled 'Name'; a 'Sequence' section with a file upload button labeled 'Choose File' and the text 'No file chosen'; a 'Disease' section with a text input labeled 'Disease'; and an 'Algorithm' section with two radio buttons, 'Boyer-Moore' (selected) and 'Knuth-Morris-Pratt'. A blue 'Submit' button is located at the bottom right of the form.

Gambar 10. Laman Tes DNA

3. Melihat history pencocokan.

- Pengguna memasukkan input pada kolom pencarian, nama penyakit hanya boleh alphanumeric, spasi, dan karakter “-”
- Input yang diterima berupa ‘Tanggal Nama_Penyakit’ atau ‘Tanggal’ atau ‘Nama_Penyakit’, dengan format tanggal ‘DD Month YYYY’ dan format nama_penyakit berupa huruf alfanumerik, tanda hubung dan spasi
- Pengguna dapat menekan enter atau tombol cari di bagian kanan
- Apabila ditemukan history yang cocok dengan input yang dimasukkan, akan ditampilkan hasil berbagai test dalam bentuk tabel
- Apabila tidak ditemukan, akan muncul sebuah teks yang mengindikasikan error yang terjadi



Gambar 11. Laman Pencarian

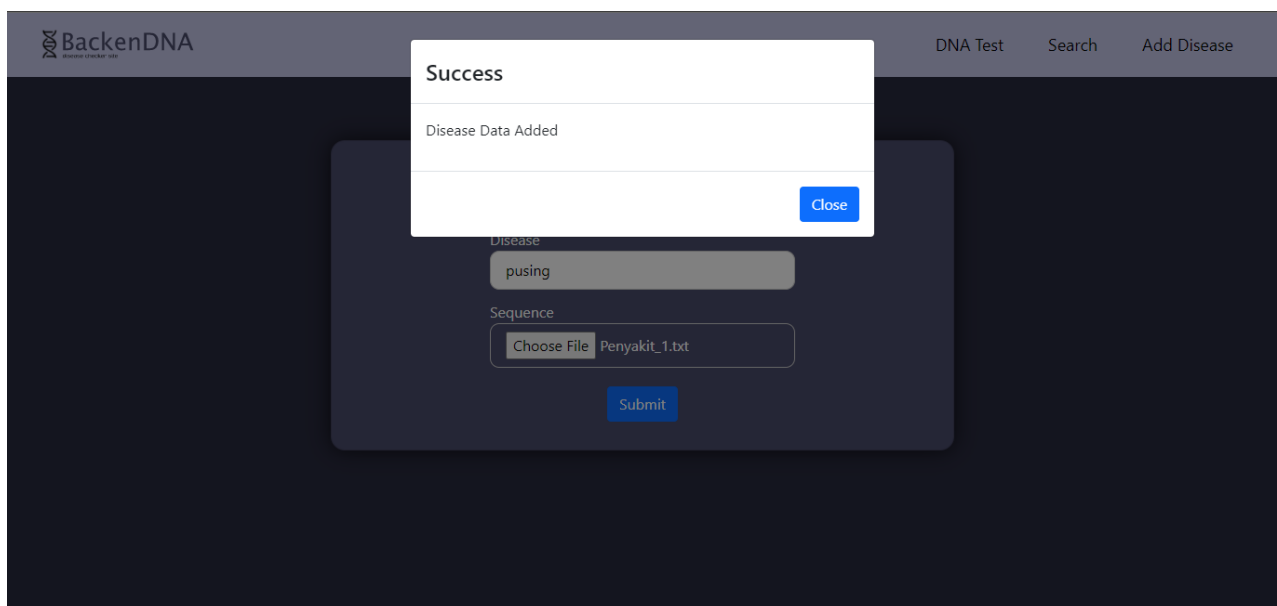
4.3 Hasil Pengujian

4.3.1 Pengujian Penambahan Penyakit

a. Berhasil

File masukan : Penyakit_1.txt

ATGGTTGCATTATCCTCCTTGCAGGACTA

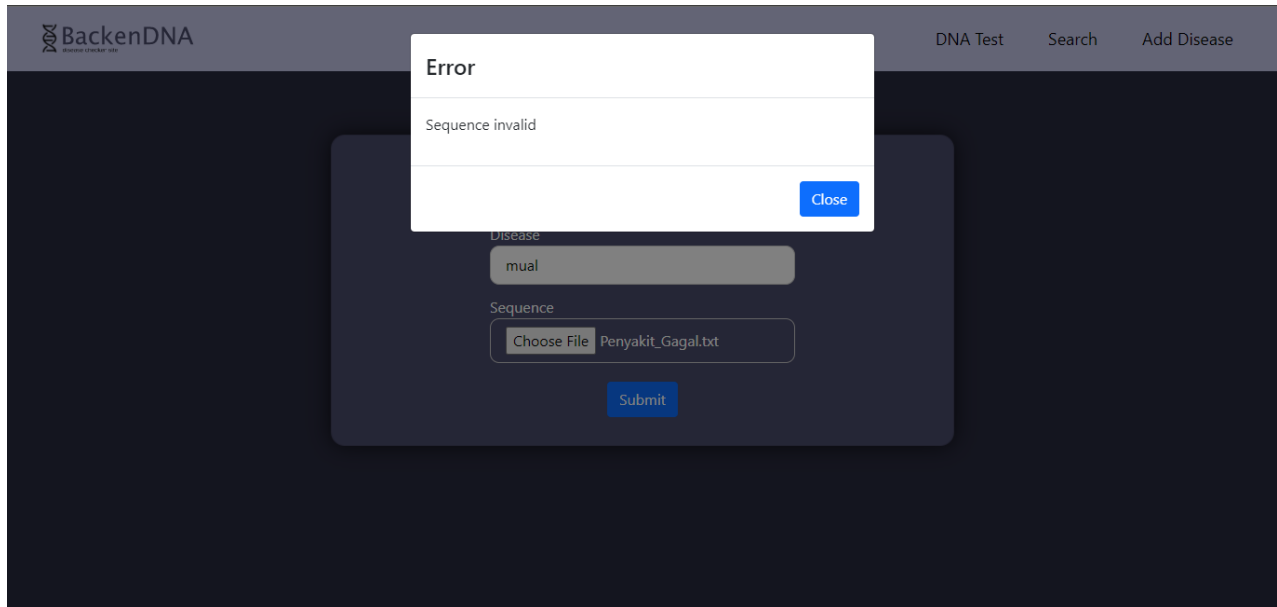


Gambar 11. Pengujian sukses menambahkan penyakit

b. Gagal

File masukan: Penyakit_Gagal.txt

UCCGACGUGTCUCGAGACAA



Gambar 12. Pengujian gagal menambahkan penyakit

4.3.2 Pengujian Tes DNA

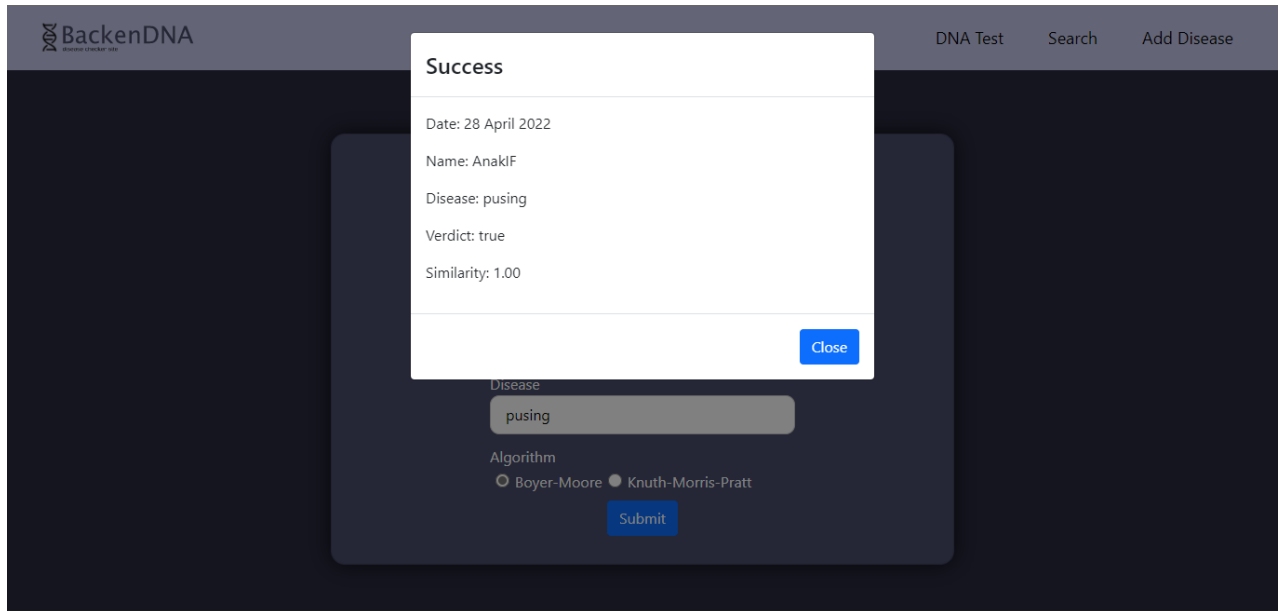
a. *Exact Matching*

File masukan DNA:

```
TCAGTTATCCACCAGCTTCGAGGATAACTAAGTCATAGTGGAACGCCGATGTTG
GGCTGAAATGGGACGGGGATTGAAGTGCTACGTGCTTAAATAGGTGCCGAGGTGG
CAAGCTCGGCATTGTAATGGACGCGGTGACCGTGCCAAGCGGGAAATTCTTTTTC
CCCCATGAGAGTCTAGACTCAGTTCCAAGTACCTACCAGGAAGCATATTGGTAGC
CGGACCTTGCCAGCTCCTTCGCCCCACGGGGCTGCTATTTATTACCTAGTAGTCCG
TTCTCCTAGAACACATGTTCCCTGTGGCGCAAGATAGGGTATATCTGTCACCCCG
TTTATACCAAGTGTAACGAATTAACGTCGGCTGTGAGTAGCTGCCAGGGCATCAA
AACCGTCTGATGATGACTATCCCCCTCTCACTGTGCGTCCCTAGTGAGTCATTCTG
AGCTTCAGCGTTAATAGTATTGATGGTTGCATTATCCTCCTTGTTACAGGGCATT
CATCCGTTCTAAGAAAAGTGTCGAACGCAAACGTGAAAAGTTATGCCAAGTGGTG
CTACTGAATAAGGCTTTGTTATTAAACTCTCTCGAAAACCACACTTACTGTGTTT
ACCTATGATCCTACTGCAATGTTAACCGGCGTGTCATGTCTAGTGGTCCACATTG
ACGGATAGTCATAGCTTTAGAGTTACCTCGCAGTGTCGGTGGGATAGTCACAGGG
CGGCACGCCGTATATTTCCATCCTCACGATCTGACTTGAGCTGTGGTGCTCCCGT
GGATGCGGTAGCAACTATGATTCTTATCATAGAGACCCGGCTCAAGACAGCAGGC
ACTTTTATTAAATCACAGCATCCCAACCATACATATAGAACCTCCACCGAATAAC
```

CATCCAGCTCTCCTCTTACTGACGCGGCTGAGTTATTTTTATCGTAAACTGCTGC
 AGTGAAGGCCTCATAGGTCCGAGTGTGGGGAAAAGCTGGAACACCCAAGAGGGAT
 ACAATTTAGT

Sequence penyakit: ATGGTTGCATTATCCTCCTTGCAGGACTA



Gambar 13. Pengujian Tes DNA dengan hasil yang exact

b. Similarity lebih dari atau sama dengan 80%

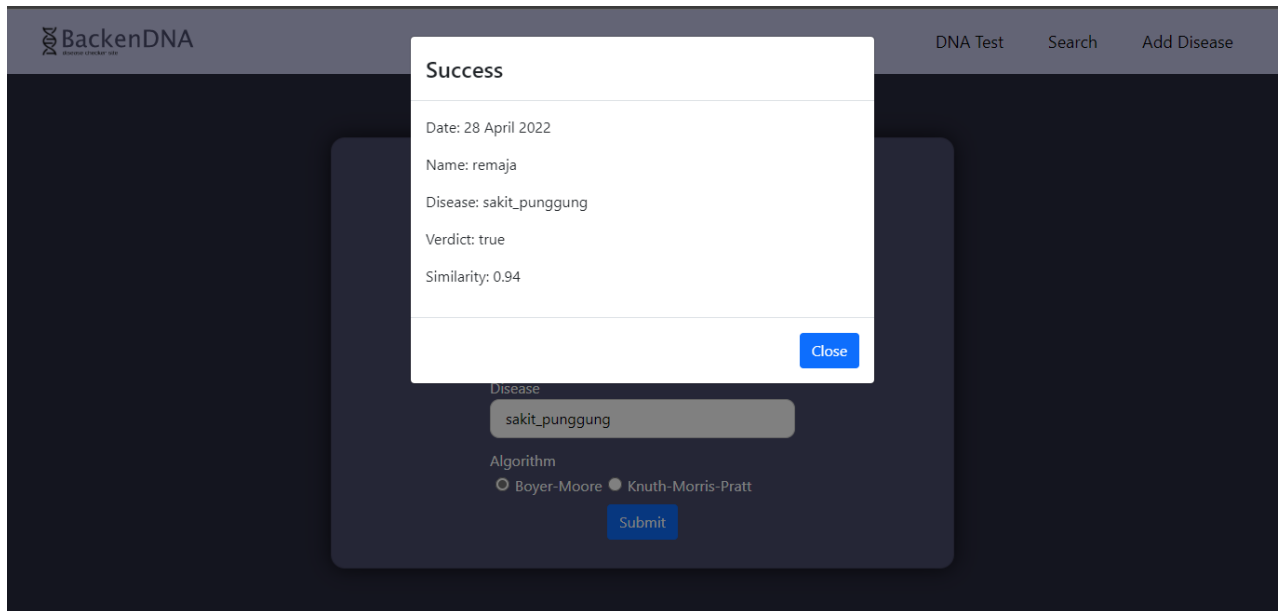
File masukan DNA:

GCAGCGGCCGGAGCGTATACGAACAGTTTTCTCCTGATGACTATTCTGCCAAGC
 CGTTTACACCATCGAGAAAAGAGGTCTGTCTGTAGCGACACCGCGAACTGTGGGG
 GGCTTAGCGGAACAGGGGACGCACCTTTGGCCGAAATACCCTTTAGGAGGCTTCA
 GGGTAATTCGCTGTAATGCAGGGATGAATATTAGTACCACTTGATTGGTAAATTG
 GTGTAGCGCCGTGGTTTCAGGAAATCACCCATCCAGCGTAGATAGAGGGAACATC
 TGTCAGTGCCCTTCTGTGCCCAAAGTACTGTCCAGCAACCCGGTTCTCCCACTTG
 GCGCCAAGTAGGATTGGTAGAGCGGATCTAAGACTAAAAGCGTATTCGTCTCGTT
 TCTTGGTCAAGTCCATTCTCCATCTATATTCGGGCACCGTCGCCTCGCTCAATTC
 TTAGCCTACCGCCAGTACATCTTAGCACTACATGCACGAAATGCCCTGTAGCGGT
 AACCGGGAAGGTCGTTGCATGCGTCAATGTCTCATGCAAGGAGGTGAGTAACCTG
 GCCCCCACGTTGAAGACAAGCCAGTATGTATGGTTAACCGTGTGCGAGTTTGGGA
 CCTTTTCCCCTGTCATTTCGAATAGACGCGCCGACTTATCCAACCGACAAGGAAGC
 GGTTGGTATTTGCTGAACTAGTCCGCGCCACCCCCGTCCGCACGACTAAATGGA
 CAGCGCAAGCCTATTCGCTACAGCTCACGTGCGTTGCACGCCGTGGCCCAACCT
 TTGGGCAGGGGTGAGGGAGGACTCAGTGAACGGTTTACAAAACACCTAAGGCACA
 AGTGTGCGCGGTATCTGTAGACAACTCCATGGAAGTCATGTACACGGGCTGTGAC

CTCGCTAAAGGTAGCGTGGGCGCTGACAATTGAGCGTTGTTCTGTCGATGCCAGG
GTCAACATTGTTAAACAATCTCACCGGCTCGGTAGCGATAACCAATTCTAGCAGA
CCATATCTACATTCTAACGCGTGTGCCGACGCC

Sequence penyakit:

TATTCCGCTACAGCTCACGTGCGTTGCACGCCGTGGCCCAACCTTTCCCA



Gambar 14. Pengujian Tes DNA dengan hasil yang tinggi

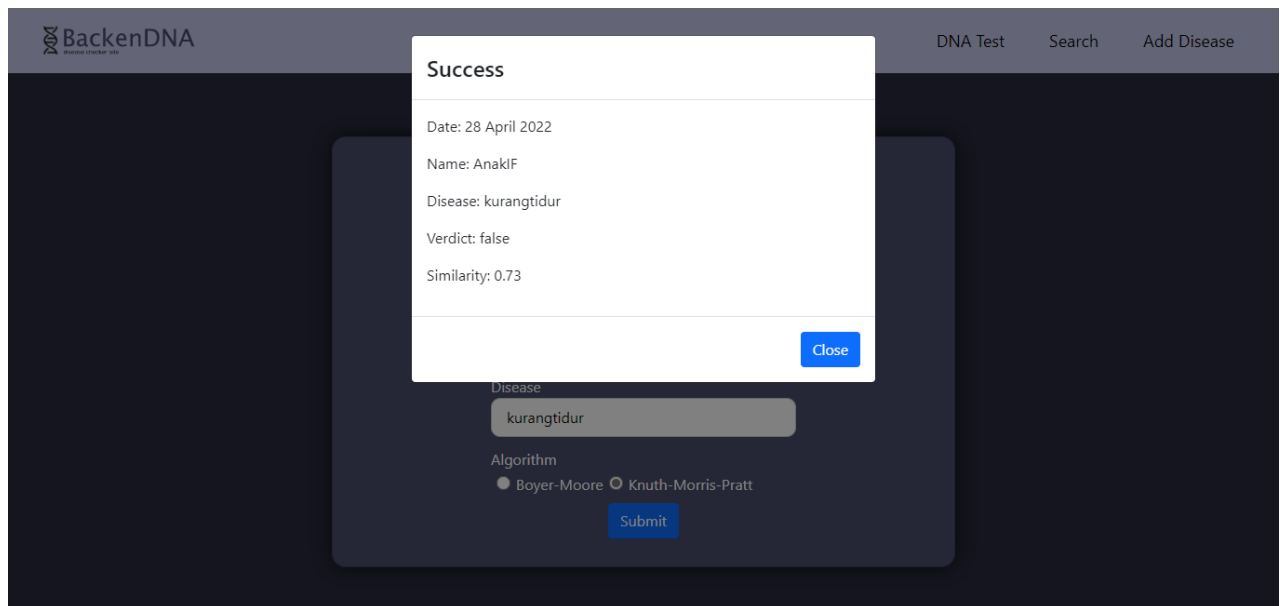
c. Similarity kurang dari 80%

File masukan DNA:

TCAGTTATCCACCAGCTTCGAGGATAACTAAGTCATAGTGGAACGCCGATGTTG
GGCTGAAATGGGACGGGGATTGAAGTGCTACGTGCTTAAATAGGTGCCGAGGTGG
CAAGCTCGGCATTGTAATGGACGCGGTGACCGTGCCAAGCGGGAAATTCTTTTTC
CCCCATGAGAGTCTAGACTCAGTTCCAAGTACCTACCAGGAAGCATATTGGTAGC
CGGACCTTGCCAGCTCCTTCGCCCACGGGGCTGCTATTTATTACCTAGTAGTCCG
TTCTCCTAGAACACATGTTCCCTGTGGCGCAAGATAGGGTATATCTGTCACCCCG
TTTATACCAAGTGTAACGAATTAACGTCGGCTGTGAGTAGCTGCCAGGGCATCAA
AACCGTCTGATGATGACTATCCCCCTCTCACTGTGCGTCCCTAGTGAGTCATTCTG
AGCTTCAGCGTTAATAGTATTGATGGTTGCATTATCCTCCTTGTTACAGGGCATT
CATCCGTTCTAAGAAAAGTGTCGAACGCAAACGTGAAAAGTTATGCCAAGTGGTG
CTACTGAATAAGGCTTTGTTATTAACTCTCTCGAAAACCACACTTACTGTGTTT
ACCTATGATCCTACTGCAATGTTAACCGGCGTGTCATGTCTAGTGGTCCACATTG
ACGGATAGTCATAGCTTTAGAGTTACCTCGCAGTGTCGGTGGGATAGTCACAGGG
CGGCACGCCGTATATTTCCATCCTCACGATCTGACTTGAGCTGTGGTGCTCCCGT
GGATGCGGTAGCAACTATGATTCTTATCATAGAGACCCGGCTCAAGACAGCAGGC
ACTTTTATTAAATCACAGCATCCCAACCATACATATAGAACCTCCACCGAATAAC
CATCCAGCTCTCCTCTTACTGACGCGGCTGAGTTATTTTTATCGTAAACTGCTGC

AGTGAAGGCCTCATAGGTCCGAGTGTGGGGAAAAGCTGGAACACCCAAGAGGGAT
ACAATTTAGT

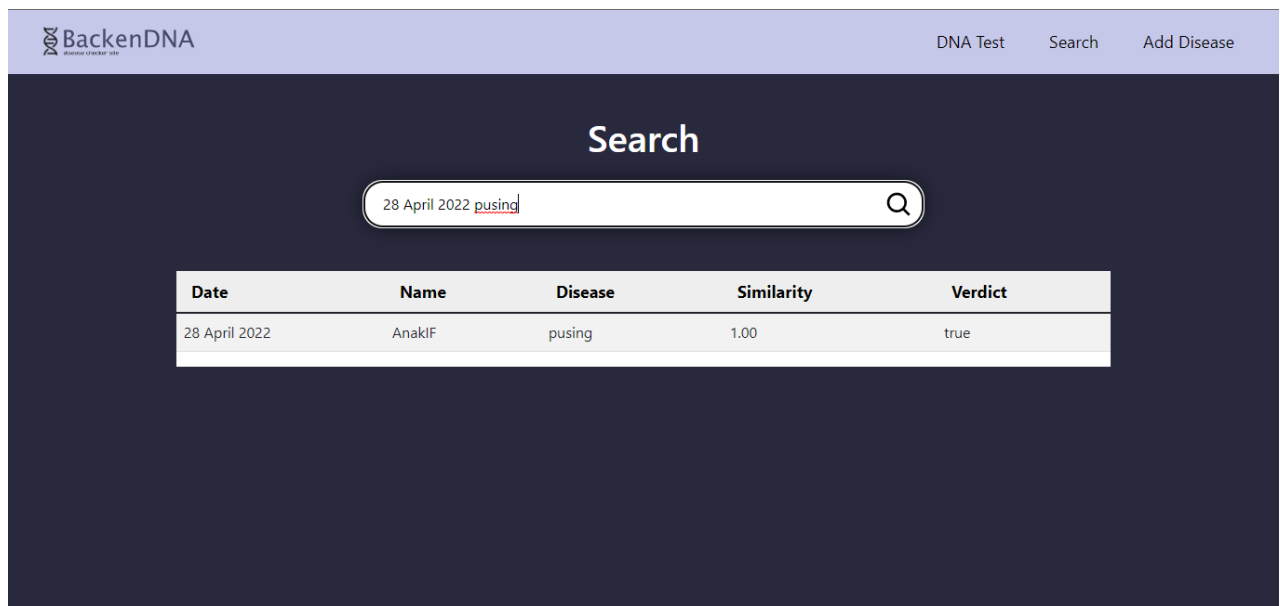
Sequence penyakit: AGGCAAATCACATTC



Gambar 15. Pengujian Tes DNA dengan hasil yang tidak cukup tinggi

4.3.3 Pengujian Pencarian Riwayat Tes DNA

a. Format masukan “Tanggal Nama_Penyakit”



Gambar 15. Pengujian riwayat tes DNA dengan format “Tanggal Nama_Penyakit”

b. Format masukan “Tanggal”

The screenshot shows the BackenDNA application interface. At the top, there is a navigation bar with the logo 'BackenDNA' and three links: 'DNA Test', 'Search', and 'Add Disease'. The main section is titled 'Search' and features a search input field containing the date '28 April 2022'. Below the input field is a table with the following data:

Date	Name	Disease	Similarity	Verdict
28 April 2022	Orang1	penyakit_2	0.74	false
28 April 2022	Orang2	penyakit_2	0.74	false
28 April 2022	Orang2	penyakit_1	0.74	false
28 April 2022	AnakIF	pusing	1.00	true
28 April 2022	AnakIF	kurangtidur	0.73	false
28 April 2022	remaja	sakit_punggung	0.94	true
28 April 2022	remaja	sakit_punggung	0.94	true

Gambar 16. Pengujian riwayat tes DNA dengan format “Tanggal”

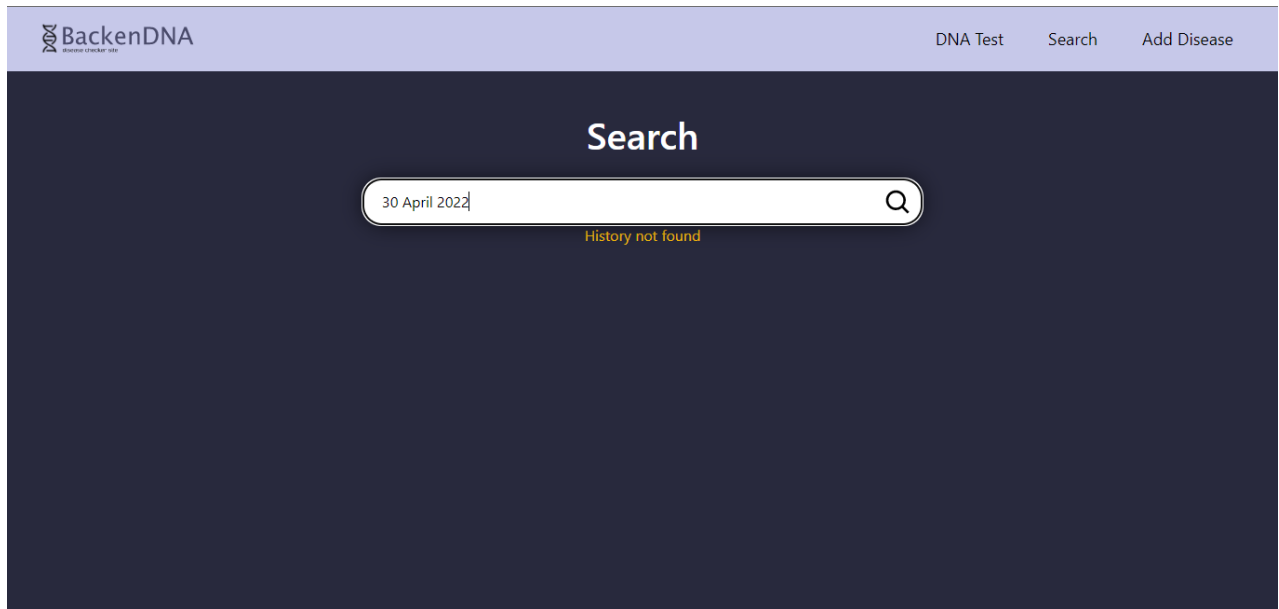
c. Format masukan “Nama_Penyakit”

The screenshot shows the BackenDNA application interface. At the top, there is a navigation bar with the logo 'BackenDNA' and three links: 'DNA Test', 'Search', and 'Add Disease'. The main section is titled 'Search' and features a search input field containing the text 'penyakit_1'. Below the input field is a table with the following data:

Date	Name	Disease	Similarity	Verdict
28 April 2022	Orang1	penyakit_1	1.00	true
28 April 2022	Orang2	penyakit_1	0.74	false

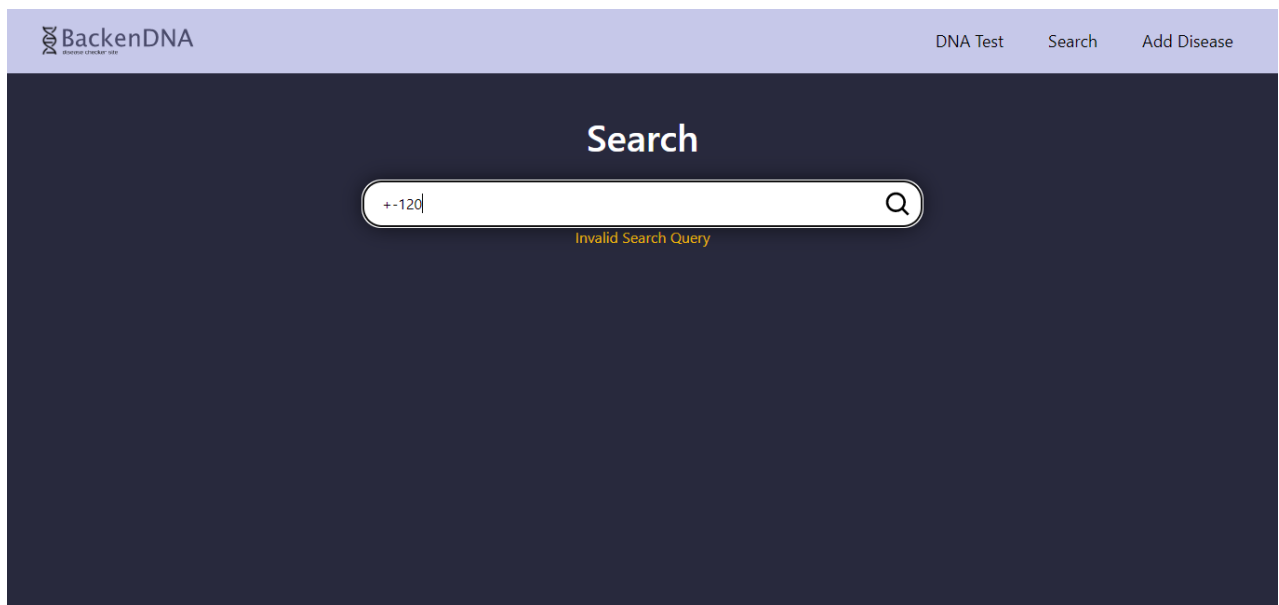
Gambar 17. Pengujian riwayat tes DNA dengan format “Nama_Penyakit”

d. History Not Found



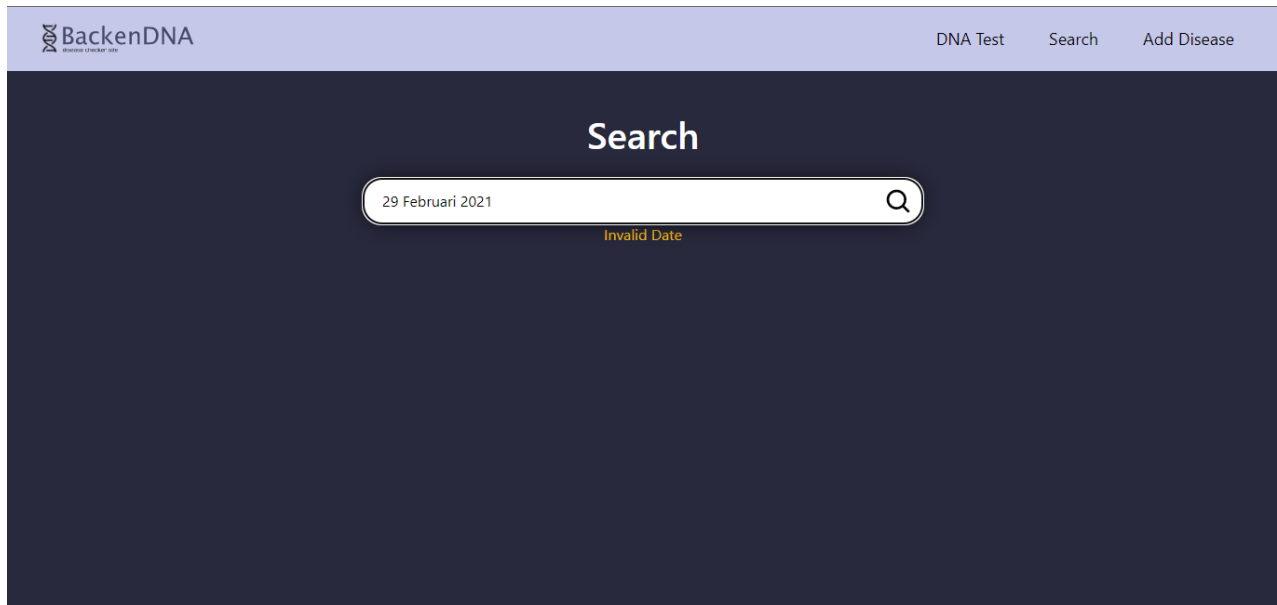
Gambar 18. Pengujian riwayat tes DNA dengan hasil yang kosong

e. Invalid Search Query



Gambar 19. Pengujian riwayat tes DNA dengan input yang tidak valid

f. Invalid Date



Gambar 20. Pengujian riwayat tes DNA dengan input tanggal yang tidak valid

4.4 Analisis Hasil Pengujian

Pengujian yang dilakukan telah mencakup seluruh skenario dari kemungkinan interaksi antara pengguna dan aplikasi berbasis web. Pada pengujian fitur penambahan penyakit, aplikasi ini dapat melakukan sanitasi input pada *sequence* penyakit yang dimasukkan. Dapat dilihat pada gambar 11 dan 12 beserta dengan lampiran *sequence*-nya yang menolak *sequence* DNA yang tidak hanya tersusun dari huruf C,G,A, atau T. Apabila *sequence* penyakit lolos tahap sanitasi, maka penyakit tersebut akan ditambahkan ke dalam basis data. Apabila tidak lolos, maka akan mengembalikan pesan error seperti yang tertampil pada gambar 12.

Pada pengujian fitur Tes DNA, aplikasi dapat berjalan sesuai dengan spesifikasi yang diberikan. Pada poin 4.3.2 a, kami menguji dengan kasus uji yang *exact matching* sehingga kemiripan yang dihasilkan adalah sebesar 100%. Pada poin 4.3.2 b, kami menguji dengan kasus uji yang lumayan mirip sehingga menghasilkan kemiripan yang tinggi. Pada poin 4.3.2 c, kami menguji dengan kasus uji yang tidak terlalu mirip dan tidak menunjukkan adanya kecocokan yang berarti.

Pada pengujian fitur Riwayat Tes DNA, kami menguji dengan berbagai kasus yang mungkin terjadi. Aplikasi ini dapat melakukan pengecekan pada input yang dimasukkan oleh pengguna apakah sesuai dengan spesifikasi yang diinginkan atau tidak. Pengecekan ini dilakukan dengan *regular expression*. Pada poin 4.3.3 a-d, kami menguji dengan masukan yang valid dan menghasilkan riwayat yang berbeda-beda. Sedangkan pada poin 4.3.3 e dan f, kami menguji dengan memasukkan input yang tidak valid. Aplikasi ini dapat menolak masukan jika masukan tersebut merupakan bukanlah tanggal yang valid. Aplikasi ini juga dapat menolak masukan jika masukan tersebut tidak cocok dengan spesifikasi yang ada.

Bab 5 Kesimpulan dan Saran

Kesimpulan

Berdasarkan yang sudah dipaparkan, dapat diambil kesimpulan bahwa implementasi algoritma KMP, Boyer-Moore, Levenshtein Distance dan Regex untuk sanitasi input dapat dijalankan dengan baik bersamaan dengan implementasi backend Golang dan frontend React. Integrasi keduanya dapat dilakukan dengan DBMS PostgreSQL sehingga implementasi aplikasi web dapat berjalan dengan baik. Algoritma KMP dan Boyer-Moore dapat mencari nilai kebenaran kecocokan suatu pattern di dalam suatu DNA sequence. Implementasi bonus Similarity Matching menggunakan algoritma Levenshtein Distance juga berjalan dengan baik dan dapat menghasilkan ratio kemiripan sehingga dapat menentukan nilai kebenaran kecocokan suatu pattern di dalam suatu DNA sequence.

Saran

Algoritma KMP, Boyer-Moore, Levenshtein Distance dan Regex dapat berjalan lancar begitu juga dengan frontend serta backend. Beberapa hal yang dapat dikembangkan berikutnya adalah, penambahan layout yang lebih estetik serta modularitas dalam algoritma dan backend.

Komentar dan Refleksi

Tugas besar 3 kali ini sangat menantang karena kami menggunakan Golang, bahasa yang belum pernah kami gunakan sebelumnya pada bagian backend. Kami juga berhasil mengimplementasi semua bonus di tengah kesibukan tugas besar yang lain. Kami merasa bangga akan karya kami meskipun ada beberapa hal yang bisa ditingkatkan lebih lagi untuk kedepannya.

Link Video

<https://youtu.be/Tg7Y0pPnVX0>

Link Repository

https://github.com/kristabdi/Tubes3__13520058

Link Website

<https://stima-backend.onrender.com>

Daftar Pustaka

- [1] Munir, Rinaldi dan Maulidevi, Nur Ulfa. (2022), Pencocokan String Matching. Diakses online dari <https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Pencocokan-string-2021.pdf> pada 15 April 2022.
- [2] Go. (2022). Go Programming Guide. Diakses online dari <https://go.dev/doc/> pada 14 April 2022.