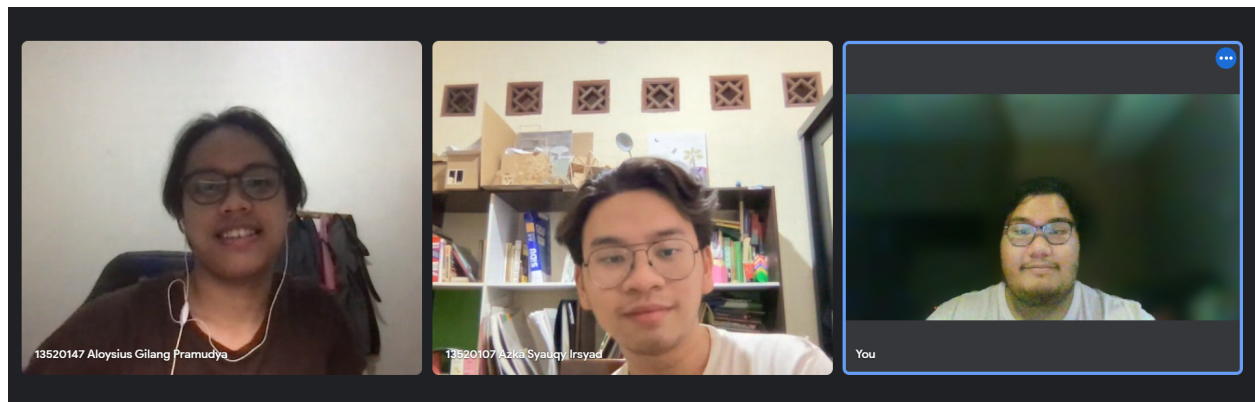


TUGAS BESAR 3 SEMESTER II TAHUN 2021/2022
IF2211 STRATEGI ALGORITMA

Penerapan String Matching dan Regular Expression
dalam DNA Pattern Matching



13520107 Azka Syauqy Irsyad
13520141 Yoseph Alexander Siregar
13520147 Aloysius Gilang Pramudya

**PROGRAM STUDI SARJANA TEKNIK INFORMATIKA
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG
2021**

Daftar Isi

Deksripsi Tugas	3
Landasan Teori	8
Deskripsi Singkat Algoritma	8
Web Development dengan ReactJS	11
Analisis Pemecahan Masalah	13
Langkah-langkah Penyelesaian Masalah	13
Fitur fungsional dan arsitektur aplikasi web yang dibangun	15
Implementasi dan Pengujian	16
Spesifikasi Teknis Program	16
Tata cara penggunaan program	18
Hasil pengujian	19
Analisis Hasil Pengujian	30
Saran dan Kesimpulan	31
Kesimpulan	31
Saran	31
Daftar Pustaka	32
Link	33

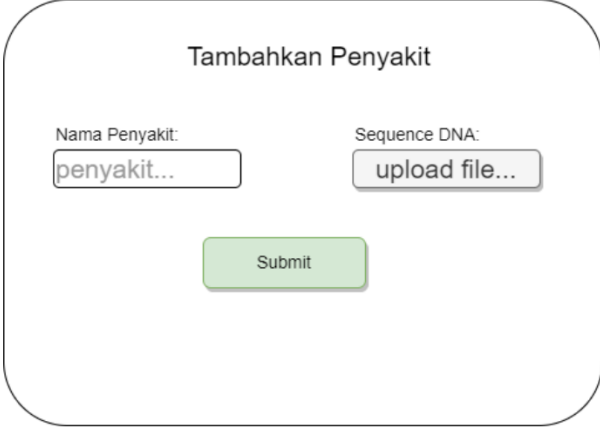
I. Deksripsi Tugas

Dalam tugas besar ini, diminta untuk membangun sebuah aplikasi DNA Pattern Matching. Dengan memanfaatkan algoritma String Matching dan Regular Expression yang telah dipelajari di kelas IF2211 Strategi Algoritma, 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.

Gambar 1.1. Ilustrasi Input Penyakit

Sumber :

<https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2021-2022/Tugas-Besar-3-IF2211-Strategi-Algoritma-2022.pdf>

2. Aplikasi dapat memprediksi seseorang menderita penyakit tertentu berdasarkan sequence DNA-nya.

- a. Tes DNA dilakukan dengan menerima input nama pengguna, sequence DNA pengguna, dan nama penyakit yang diuji. Asumsi sequence DNA pengguna > sequence DNA penyakit.
- b. Dilakukan sanitasi input menggunakan regex untuk memastikan bahwa masukan merupakan sequence DNA yang valid (tidak boleh ada huruf kecil, tidak boleh ada huruf selain AGCT, tidak ada spasi, dll).
- c. Pencocokan sequence DNA dilakukan dengan menggunakan algoritma string matching.
- d. Hasil dari tes DNA berupa tanggal tes, nama pengguna, nama penyakit yang diuji, dan status hasil tes. Contoh: 1 April 2022 - Mhs IF - HIV - False
- e. Semua komponen hasil tes ini dapat ditampilkan pada halaman web (refer ke poin 3 pada “Fitur-Fitur Aplikasi”) dan disimpan pada sebuah tabel database.
- f. Contoh tampilan web:

Tes DNA

Nama Pengguna:

Sequence DNA:

Prediksi Penyakit:

Hasil Tes

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

Gambar 1.2. Ilustrasi Output Prediksi

Sumber :

<https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2021-2022/Tugas-Besar-3-IF2211-Strategi-Algoritma-2022.pdf>

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:

1) Masukkan tanggal dan nama penyakit

The screenshot shows a web application interface. At the top, there is a rounded rectangular box containing the text "13 April 2022 HIV". Below this, there are six rectangular boxes, each containing a numbered list item. The items are: 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 1.3. Ilustrasi Interaksi 1

Sumber :

<https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2021-2022/Tugas-Besar-3-IF2211-Strategi-Algoritma-2022.pdf>

2) Masukkan hanya tanggal

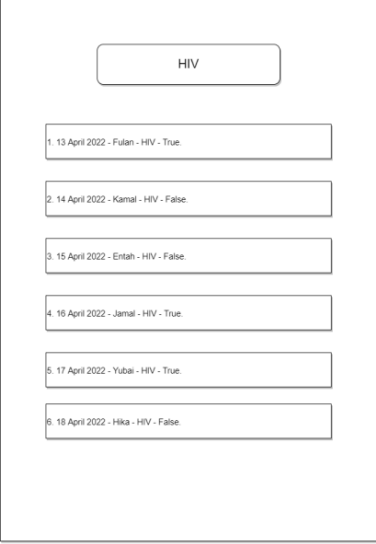
The screenshot shows a web application interface. At the top, there is a rounded rectangular box containing the text "13 April 2022". Below this, there are six rectangular boxes, each containing a numbered list item. The items are: 1. 13 April 2022 - Fulan - Diabetes - True; 2. 13 April 2022 - Kamal - Sinusitis - False; 3. 13 April 2022 - Entah - Down Syndrome - False; 4. 13 April 2022 - Jamal - Polio - True; 5. 13 April 2022 - Yubai - TBC - True; 6. 13 April 2022 - Hika - Hepatitis A - False.

Gambar 1.4. Ilustrasi Interaksi 2

Sumber :

<https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2021-2022/Tugas-Besar-3-IF2211-Strategi-Algoritma-2022.pdf>

3) Masukan hanya nama penyakit



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

Gambar 1.5. Ilustrasi Interaksi 3

Sumber :

<https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2021-2022/Tugas-Besar-3-IF2211-Strategi-Algoritma-2022.pdf>

4. (Bonus) Menghitung tingkat kemiripan DNA pengguna dengan DNA penyakit pada tes DNA

- Ketika melakukan tes DNA, terdapat persentase kemiripan DNA dalam hasil tes. Contoh hasil tes: 1 April 2022 - Mhs IF - HIV - 75% - False
- Perhitungan tingkat kemiripan dapat dilakukan dengan menggunakan Hamming distance, Levenshtein distance, LCS, atau algoritma lainnya (dapat dijelaskan dalam laporan).
- Tingkat kemiripan DNA dengan nilai lebih dari atau sama dengan 80% dikategorikan sebagai True. Perlu diperhatikan mengimplementasikan atau tidak mengimplementasikan bonus ini tetap dilakukan pengecekan string matching terlebih dahulu.
- Contoh tampilan:

Tes DNA

Nama Pengguna:

Sequence DNA:

Prediksi Penyakit:

Hasil Tes

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

Gambar 1.6. Ilustrasi Program dengan Bonus

Sumber :

<https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2021-2022/Tugas-Besar-3-IF2211-Strategi-Algoritma-2022.pdf>

Untuk tugas besar ini, implementasi Backend program menggunakan Node.js, implementasi Frontend menggunakan React, dan penyimpanan data / DBMS menggunakan MongoDB.

II. Landasan Teori

A. Deskripsi Singkat Algoritma

String Matching adalah algoritma pencocokan string yang bekerja dengan mencari lokasi pertama suatu *pattern* dalam suatu teks. Misalkan pencocokan antara string pattern ABAC pada string teks ABADABACDE. Dengan algoritma string matching akan dikeluarkan hasil 4 (indeks ke-4) atau huruf ke-5.

Dalam Tugas Besar ini, diimplementasikan dua algoritma string matching yaitu, Knuth-Morris-Pratt (KMP) dan Boyer-Moore.

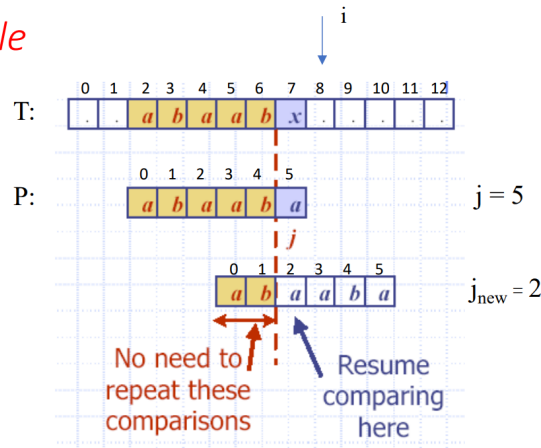
1. Knuth-Morris-Pratt (KMP)

Algoritma KMP bekerja seperti algoritma Brute Force dalam string matching tetapi penggeseran *pattern* dalam proses pengecekan lebih efektif dan efisien dibandingkan Brute Force.

Misalkan saat proses perbandingan *pattern* dengan *text*, ditemukan ketidaksetaraan pada indeks ke- j dari *pattern*. Untuk penggeseran *pattern*, sebelumnya dicek terlebih dahulu prefix dan suffix dari *pattern* tersebut, dimana prefix dilihat dari indeks 0 hingga $j-1$ sedangkan suffix pada indeks 1 hingga $j-1$.

Pada pengecekan prefix dan suffix dari *pattern* tersebut, akan dicari pasangan prefix dan suffix yang sama dan terbesar / terpanjang. Setelah didapatkan, maka *pattern* akan diberikan pergeseran sebanyak j dikurangi dengan panjang prefix-suffix tersebut. Lalu setelah digeser, pengecekan dilakukan dimulai dari indeks ke- i dari *pattern*, dengan i adalah panjang prefix-suffix tersebut.

Example



Gambar 2.1. Ilustrasi Algoritma KMP

Sumber :

<https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Pencocokan-string-2021.pdf>

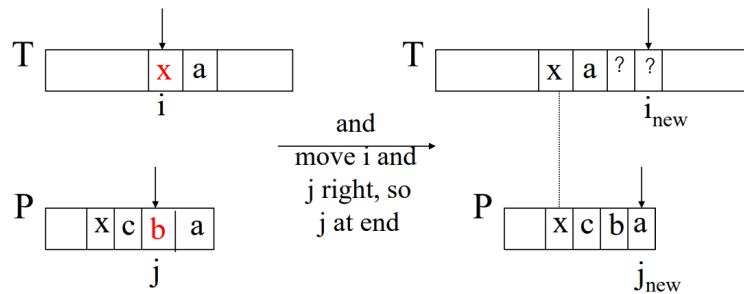
Mengapa hal ini dilakukan ? Karena saat mengambil prefix dan suffix yang sama, maka setelah pergeseran posisi prefix yang baru akan berada pada posisi suffix yang lama sehingga tidak perlu dicek lagi karena mereka adalah hal yang sama dan sudah sesuai dengan *text*.

2. Boyer-Moore(BM)

Pada algoritma BM, proses string matching didasarkan pada 2 teknik, yaitu *The looking-glass technique* dan *The character-jump technique*. Pada teknik *The looking-glass*, perbandingan pada *pattern* dilakukan dimulai dari posisi paling belakang dan maju hingga posisi awal *pattern* (apabila sudah tepat).

Pada teknik *The character-jump*, apabila terdapat ketidaksesuaian pada *pattern* dan *text* dengan karakter pada indeks ke- i pada *text* adalah x dan karakter pada indeks ke- j dari *pattern* adalah tidak sama dengan x , maka terdapat 3 kasus penggeseran *pattern* yang dilakukan satu persatu dan berurut.

Pada kasus 1, apabila pada *pattern* setelah indeks ke- j terdapat x maka *pattern* akan digeser hingga huruf x terkiri (apabila lebih dari satu) sejajar dengan x pada indeks ke- i dari *text*.

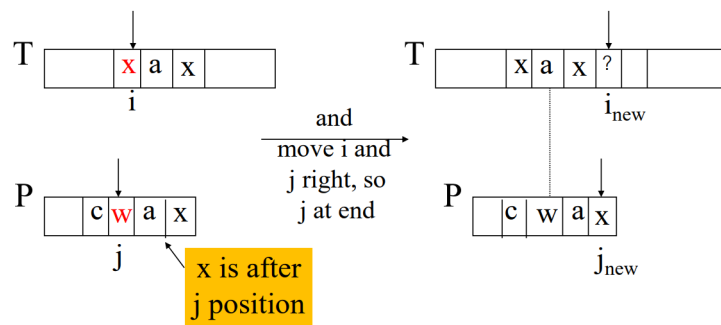


Gambar 2.2. Ilustrasi case 1

Sumber :

<https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Pencocokan-string-2021.pdf>

Pada kasus 2, apabila pada *pattern* hanya terdapat x disebelah kanan dari indeks ke-j dari *pattern*(sudah dilewati / dicek), maka dilakukan pergeseran *pattern* hingga posisi indeks ke-j dari *pattern* sejajar dengan posisi indeks ke- i+1 dari *text*.

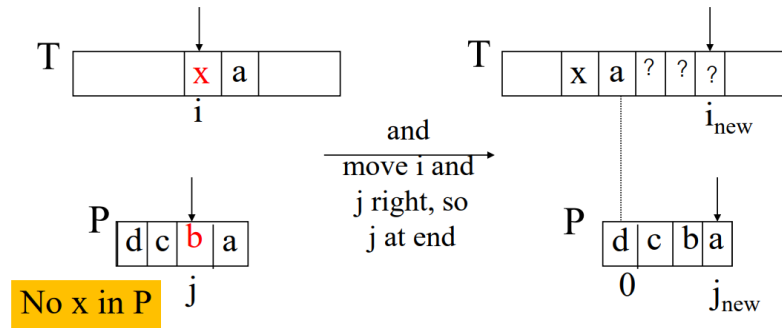


Gambar 2.3. Ilustrasi case 2

Sumber :

<https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Pencocokan-string-2021.pdf>

Pada kasus 3, apabila kasus 1 dan 2 tidak berlaku (tidak terdapat x pada *pattern*) maka geser *pattern* hingga posisi indeks ke-0 dari *pattern* sejajar dengan posisi indeks ke-i+1 dari *text*.



Gambar 2.4. Ilustrasi case 3

Sumber :

<https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Pencocokan-string-2021.pdf>

3. Regex

Pada Tugas Besar kali ini juga dimanfaatkan Regex (regular expression) untuk beberapa bagian. Regex dimanfaatkan dalam bagian input *sequence* DNA dan juga kolom pencarian riwayat.

Pada bagian input *sequence* DNA, regex dimanfaatkan untuk melakukan sanitasi / pengecekan pada *sequence* dimana *sequence* DNA hanya bisa berisi karakter AGCT, dalam huruf besar dan tanpa spasi.

Pada bagian kolom pencarian riwayat, regex digunakan untuk menampilkan hasil pencarian riwayat berdasarkan 3 input yang bisa diterima, yaitu hanya tanggal, hanya nama penyakit dan keduanya.

B. Web Development dengan ReactJS

ReactJs adalah suatu pustaka atau *library* yang merupakan produk besutan Facebook. Teknologi ini dibuat khusus untuk membuat *user interface* pada website, bahkan dapat juga untuk *mobile*, dengan menggunakan React Native. Kebanyakan pengembang menyebutnya sebagai suatu kerangka, padahal aslinya React adalah *library* yang digunakan dalam pembuatan UI.

React menyajikan Virtual DOM yang diketahui cukup cepat. Para pengembang web saat ini banyak menggunakan React karena dapat membuat serta mendesain tampilan yang simple bagi tiap tingkatan pada web yang sedang dikembangkan. Ada beberapa fitur-fitur unggulan dari *library* ini, seperti komponen dapat dibuat secara

encapsulated, sehingga dapat membuat rangkaian UI yang lain dengan dasar kemampuan komponen tersebut. Selain itu, fitur baru yang nantinya ingin ditambahkan juga dapat dengan mudah diaplikasikan tanpa harus mengganti kode sebelumnya, dimana React dapat bekerja menggunakan *library* NodeJS ataupun *mobile apps* yang menggunakan React Native. Dengan dasar bahasanya yang menggunakan JavaScript, React dapat dipadukan dengan kerangka JavaScript lainnya, seperti AngularJS dan MVC.

Untuk membuat projek React, langkah-langkah yang diperlukan cukup mudah. Pastikan terlebih dahulu bahwa node dan npm sudah terinstall pada perangkat yang digunakan. Lalu, kita hanya tinggal membuka terminal dan mengetikkan `npm create-react-app <nama-project>` pada directory yang diinginkan. Package akan langsung dinstall setelah itu bersama dengan dependencies yang diperlukan. Bahkan, pembuatan project tersebut sudah langsung membuat repository git. Kita dapat melihat module-module yang telah dibuat secara default dan dapat kita rubah sesuai kemauan website yang dikehendaki. Untuk melihat tampilan UI dari projek kita, ketikkan command `npm start` pada terminal pada directory tempat kita menginisiasi projek.

III. Analisis Pemecahan Masalah

A. Langkah-langkah Penyelesaian Masalah

1. Menerima input penyakit baru berupa nama penyakit dan sequence DNA-nya (dan dimasukkan ke dalam database)
 - Menerima *input sequence* DNA dalam bentuk file.
 - Melakukan sanitasi untuk *input* menggunakan regex (melakukan pengecekan apakah *input* sudah benar, yaitu tidak ada huruf kecil, tidak ada huruf selain AGCT dan tidak ada spasi).
 - Apabila hasil sanitasi tidak sesuai dengan yang diinginkan, maka menampilkan pemberitahuan bahwa input ditolak.
 - Apabila lolos sanitasi, maka *input sequence* akan dimasukkan ke dalam *database* dengan nama penyakit sesuai dengan yang dimasukkan.
2. Memprediksi seseorang menderita penyakit tertentu berdasarkan sequence DNA-nya
 - Menerima *input* dari pengguna berupa nama pengguna, nama penyakit yang akan diuji, *input sequence* DNA yang akan diuji dalam bentuk file dan algoritma pengecekan yang diinginkan.
 - Dilakukan sanitasi untuk *input* menggunakan regex (sebagaimana halnya seperti pada nomor 1 - input penyakit baru)
 - Apabila hasil sanitasi tidak sesuai dengan yang diinginkan, maka menampilkan pemberitahuan bahwa input ditolak.
 - Apabila lolos sanitasi, maka *input sequence* akan diuji menggunakan algoritma *string matching* yang dipilih terhadap *sequence* DNA penyakit pada database yang sudah ditentukan pengguna untuk diuji. Algoritma *string matching* yang dapat dipilih adalah algoritma Knuth-Morris-Pratt (KMP) dan Boyer-Moore (BM).
 - Setelah dilakukan pencocokan, maka aplikasi akan mengeluarkan *output* kepada pengguna berupa rangkaian kalimat yang terdiri atas ‘tanggal tes - nama pengguna - penyakit yang diuji - hasil tes’, hasil tes akan berupa *True / False* berdasarkan hasil dari pencocokan.

- *Output* dari aplikasi juga memberikan tingkat kemiripan DNA pengguna dengan DNA penyakit yang juga bisa memengaruhi *output* pada bagian hasil tes (**Bonus dikerjakan**), lebih lanjut dapat dilihat pada **nomor 4** di bawah.
 - *Output* yang dikeluarkan juga akan disimpan pada *database*.
3. Memiliki halaman yang menampilkan urutan hasil prediksi dengan kolom pencarian di dalamnya. Kolom pencarian bekerja sebagai filter dalam menampilkan hasil.
- Mengecek *input* dari pengguna menggunakan regex untuk mencari dan menampilkan hasil yang sesuai dengan pencarian pada *database*.
 - Aplikasi dapat menerima *input* berupa nama penyakit dan tanggal tes, hanya nama penyakit, dan hanya tanggal tes.
 - Apabila hasil pencocokan dengan regex sesuai dengan salah satu dari 3 pola *input* di atas, maka aplikasi akan melakukan pencarian pada *database* berdasarkan input yang diterima dan menampilkannya.
 - Apabila *input* yang dimasukkan tidak terdapat pada *database* atau input tidak sesuai dengan pencocokan regex, maka aplikasi tidak akan menampilkan apa-apa.
4. Menghitung tingkat kemiripan DNA pengguna dengan DNA penyakit pada tes DNA
- Fitur ini adalah fitur bonus yang diimplementasikan dalam aplikasi ini
 - Diimplementasikan algoritma penghitung kemiripan menggunakan algoritma Hamming Distance sebagai dasarnya
 - Melalui hasil perhitungan ini, *output* aplikasi untuk pengguna pada nomor 2 akan berubah menjadi ‘tanggal tes - nama pengguna - penyakit yang diuji - persentase kemiripan - hasil tes’
 - Dengan diimplementasikannya fitur ini, maka penentuan hasil tes mendapatkan ketentuan baru, yaitu apabila hasil pencocokan dari algoritma *string matching* memberikan informasi bahwa *sequence DNA* tidak cocok maka akan dilakukan pengecekan pada persentase kemiripan dimana apabila persentase kemiripan memiliki nilai lebih atau sama dengan 80 % maka hasil tes akan berupa *True*, selain kedua ketentuan itu akan berupa *False*.

B. Fitur fungsional dan arsitektur aplikasi web yang dibangun

1. Fitur Fungsional dari aplikasi web yang dibangun

- Menerima *input* penyakit baru dan memasukkannya ke dalam *database*
- Melakukan pengecekan *input sequence* DNA (sanitasi)
- Melakukan prediksi penyakit seseorang berdasarkan *input sequence* DNA-nya dengan algoritma *string matching* dengan pertimbangan persentase kemiripan (bonus)
- Dalam tes prediksi dapat memilih algoritma *string matching* yang ingin digunakan
- Menyimpan seluruh hasil tes ke dalam *database*
- Menampilkan hasil pencarian dari hasil tes yang disimpan pada *database* berdasarkan *input* dari pengguna (3 jenis : nama penyakit saja, tanggal saja, atau keduanya)

2. Arsitektur aplikasi web yang dibangun

- **Frontend**

Frontend dari aplikasi menggunakan library ReactJS. Situs dibangun dengan UI utama berada pada App.js, yang komponen lainnya berada pada di folder component. Komponen-komponen yang dipakai berasal dari node package manager yang telah diinstall dan juga digunakan react bootstrap untuk styling umum.

- **Backend**

Backend dari aplikasi dibangun dengan Node.js, untuk keperluan DBMS (*Database Management System*) digunakan cloud Atlas dari MongoDB.

IV. Implementasi dan Pengujian

A. Spesifikasi Teknis Program

- Struktur data

Struktur data yang digunakan pada pengerjaan tugas besar ini ada beberapa yaitu bentuk *collection* (relasi) pada DBMS MongoDB seperti berikut :

1. Pada *collection* dna_penyakit

```
{
  "_id" : ObjectId{"..."},
  "nama" : "Talasemia",
  "dna" : "CAGTATCG"
}
```

2. Pada *collection* log

```
{
  "_id" : ObjectId{"..."},
  "tanggal" : "2022-04-24T12:36:34:2032",
  "nama_pengguna" : "Nama",
  "nama_penyakit" : "Talasemia",
  "dna" : "AGTCAGTATCGGAT",
  "hasil" : true,
  "kemiripan" : 100
}
```

- Fungsi dan Program yang dibangun

No	Fungsi dan Prosedur	Keterangan
1.	bmStringMatching(text, pattern)	Algoritma <i>string matching</i> Boyer-Moore yang akan mencari dan mengembalikan posisi patern pada text, apabila tidak ada maka mengembalikan -1

2.	charLastOcc(pattern)	Menyimpan posisi kemunculan terakhir tiap karakter pada pattern pada sebuah array berukuran 256 (standar ASCII) dan mengembalikannya
3.	bm(diseaseList, dnaTest, nama_penyakit)	Fungsi yang akan memanggil algoritma bmStringMatching dan similarity untuk mengecek dnaTest terhadap sequence dna pada diseaseList yang memiliki nama sama dengan nama_penyakit
4.	kmpStringMatching(text, pattern)	Algoritma <i>string matching</i> Knuth-Morris-Pratt yang akan mencari dan mengembalikan posisi patren pada text, apabila tidak ada maka mengembalikan -1
5.	kmpBorderFunction(pattern)	Mengolah pattern untuk menyimpan urutan prefix terbesar yang juga merupakan suffixnya terhadap pattern itu sendiri untuk tiap posisi kemungkinan ketidaksamaan. Disimpan pada sebuah array
6.	kmp(diseaseList, dnaTest, nama_penyakit)	Fungsi yang akan memanggil algoritma kmpStringMatching dan similarity untuk mengecek dnaTest terhadap sequence dna pada diseaseList yang memiliki nama sama dengan nama_penyakit
7.	similarity(text, pattern)	Menghitung persentase kemiripan pattern pada text dengan menggunakan basis algoritma Hamming Distance
8.	AddDisease()	Prosedur untuk menambahkan penyakit ke dalam database berdasarkan input dari pengguna apabila seluruh input sudah benar (<i>sequence</i> DNA lolos sanitasi)
9.	getRecords() pada Check.js	Prosedur untuk mengambil data <i>sequence</i> DNA dari database

10.	getRecords() pada History.js	Prosedur untuk mengambil data riwayat tes prediksi penyakit dari database
11.	searchFromDatabase(loglist)	Mencari riwayat prediksi dari loglist yang sesuai dengan input yang dicari pengguna
12.	search(pattern)	Memilah input pencarian riwayat dari pengguna menjadi 3 kategori yang ada dan menyimpan data input tersebut

B. Tata cara penggunaan program

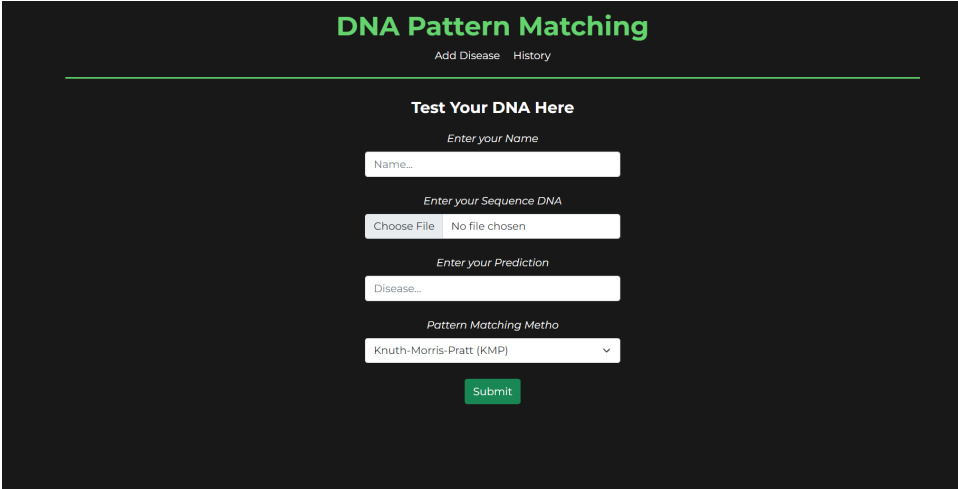
- Program **dideploy** pada <https://dnapattern.vercel.app> dan dapat diakses melalui link tersebut.
- Page default yang pertama kali ditampilkan kepada pengguna adalah page Home dimana pengguna dapat melakukan tes prediksi penyakit.
- Pada page ini, pengguna perlu mengisi form yang ada untuk nama pengguna, *input file sequence* DNA, nama penyakit yang ingin diprediksi dan pilihan algoritma pengecekan yang diinginkan.
- Apabila *sequence* DNA yang diberikan oleh pengguna sudah tepat format isinya (lolos hasil regex) dan menekan tombol submit, maka program akan menampilkan hasil prediksi dari *sequence DNA* yang diinput beserta tingkat kemiripannya.
- Apabila *sequence* DNA yang diberikan oleh pengguna tidak lulus regex maka program akan mengeluarkan peringatan bahwa *input* ditolak.
- Page lain yang dapat diakses oleh pengguna adalah page “Add Disease” dimana pengguna dapat memasukkan nama dan *sequence* DNA dari sebuah penyakit yang akan dimasukkan ke dalam *database*.
- Input *sequence* DNA pada page ini juga akan melalui proses yang sama seperti pada page Home.
- Page berikutnya adalah page “History” dimana pengguna dapat mencari riwayat hasil tes prediksi yang sudah disimpan pada *database*, untuk pencarian pengguna dapat mencari dengan tiga jenis *keyword* yaitu tanggal saja, nama penyakit saja

dan tanggal beserta nama penyakit. Program hanya akan menampilkan riwayat berdasarkan ketiga jenis *keyword* tersebut.

C. Hasil pengujian

Screenshot antarmuka dan skenario yang memperlihatkan berbagai kasus yang mencakup seluruh fitur pada aplikasi DNA Sequence Matching

1. Home Page



The screenshot shows the home page of a web application titled "DNA Pattern Matching" in green text. Below the title are two links: "Add Disease" and "History". A horizontal green line separates the header from the main content area. The main content area is titled "Test Your DNA Here" and contains several input fields and a submit button. The fields are labeled "Enter your Name", "Enter your Sequence DNA", "Enter your Prediction", and "Pattern Matching Metho". The "Enter your Sequence DNA" field has a "Choose File" button and a "No file chosen" text. The "Pattern Matching Metho" field is a dropdown menu with "Knuth-Morris-Pratt (KMP)" selected. A green "Submit" button is at the bottom.

DNA Pattern Matching

[Add Disease](#) [History](#)

Test Your DNA Here

Enter your Name

Name...

Enter your Sequence DNA

Choose File No file chosen

Enter your Prediction

Disease...

Pattern Matching Metho

Knuth-Morris-Pratt (KMP) ▾

Submit

Gambar 4.1. Home Page

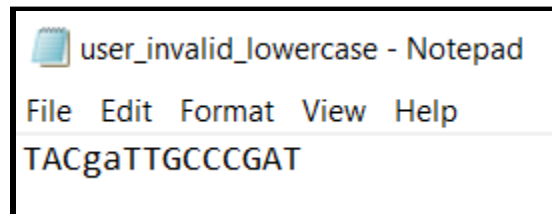
2. Add Disease Page

Gambar 4.2. Add Disease Page

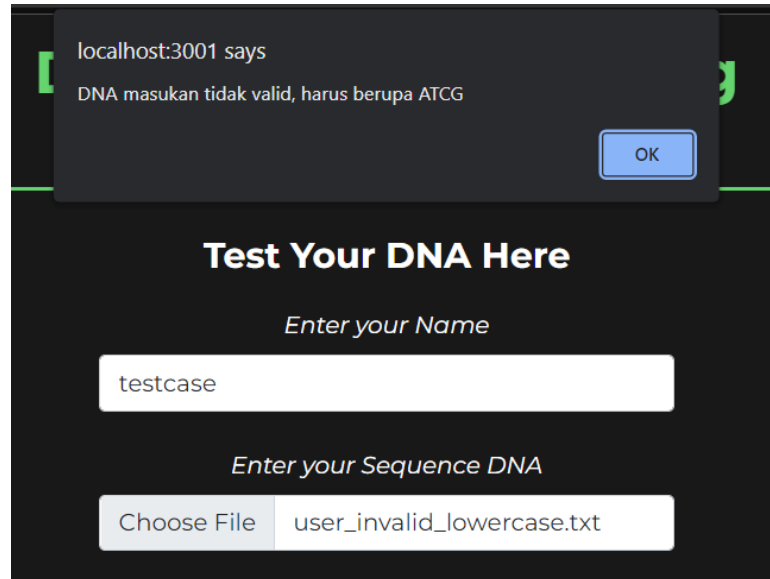
3. History Page

Gambar 4.3. History Page

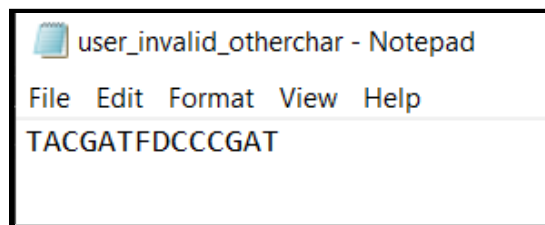
4. Sanitasi Input *Sequence DNA*



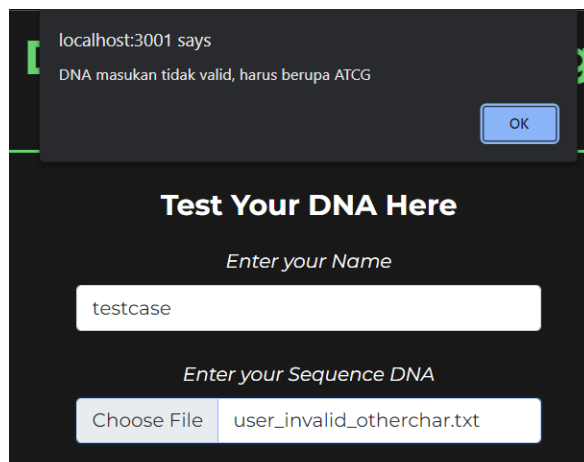
Gambar 4.4. *Sequence* Tidak Valid terdapat huruf kecil



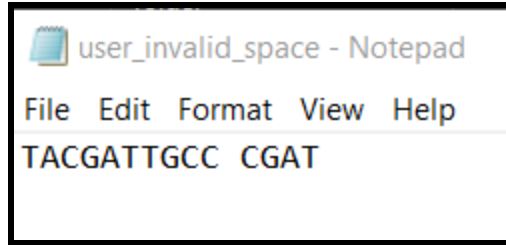
Gambar 4.5. *Output Program*



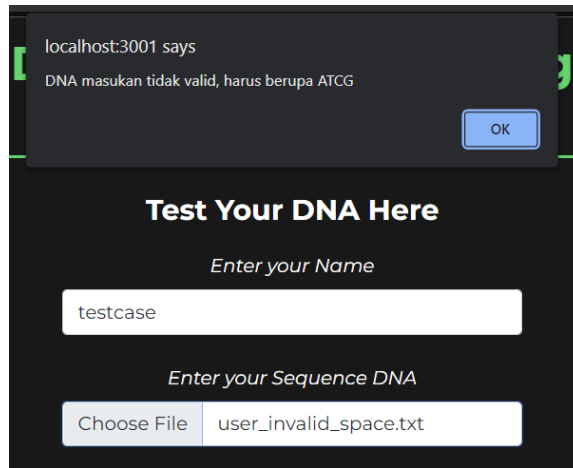
Gambar 4.6. *Sequence Tidak Valid* terdapat huruf selain ATCG



Gambar 4.7. *Output Program*



Gambar 4.6. *Sequence* Tidak Valid terdapat spasi

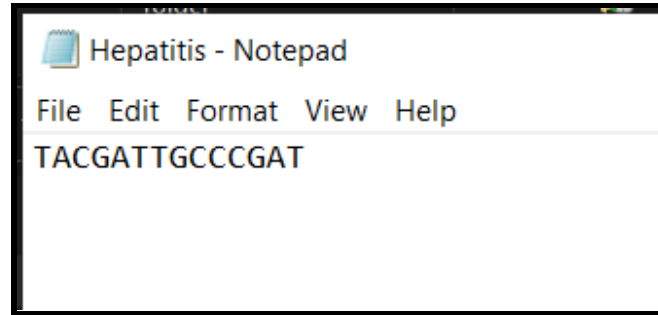


Gambar 4.8. *Output* Program

5. Fitur Menambahkan Penyakit ke *Database*



Gambar 4.9. Menambahkan Penyakit Hepatitis

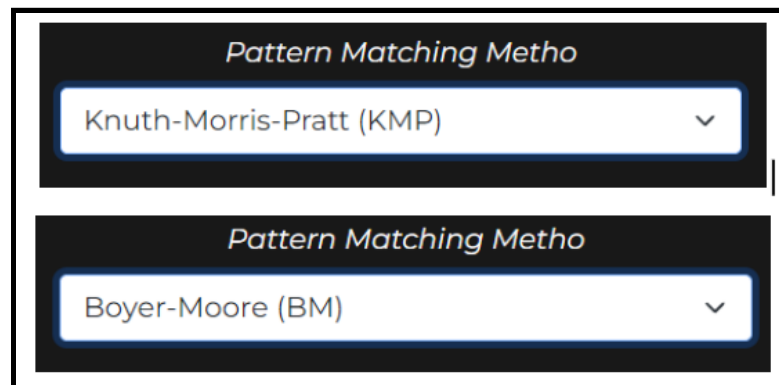


Gambar 4.10. *Sequence* DNA Hepatitis



Gambar 4.11 Hepatitis masuk ke *database*

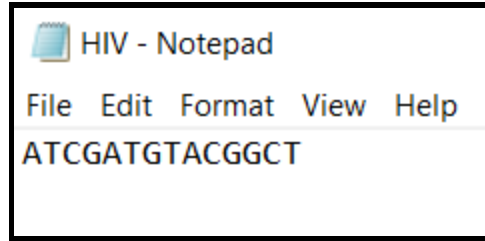
6. Fitur Memilih Algoritma *String Matching*



Gambar 4.12. Algoritma *String Matching* dapat dipilih

7. Tes Prediksi

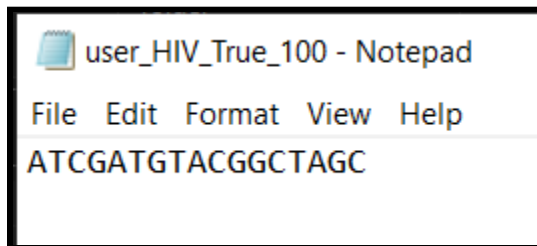
Pada tes prediksi ini digunakan *sequence* DNA HIV yang pada database bernama Fever



Gambar 4.13. *Sequence* DNA yang ada pada database

a. Tes 1 - Kemiripan 100%

Pada tes ini, karena algoritma *string matching* memberikan informasi bahwa ditemukan *sequence* pada database di *sequence* yang diuji maka hasilnya adalah True dan kemiripan 100% .



Gambar 4.14. *Sequence* DNA yang diuji

Test Your DNA Here

Enter your Name

testcase

Enter your Sequence DNA

Choose File

user_HIV_True_100.txt

Enter your Prediction

Fever

Pattern Matching Metho

Knuth-Morris-Pratt (KMP) ▾

Submit

Your Result

Name : testcase

Date : 2022-4-29

Disease : Fever

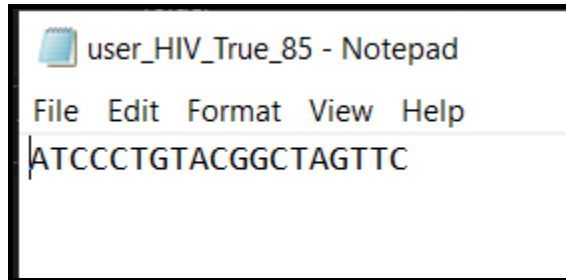
Result : true

Kemiripan : 100.00

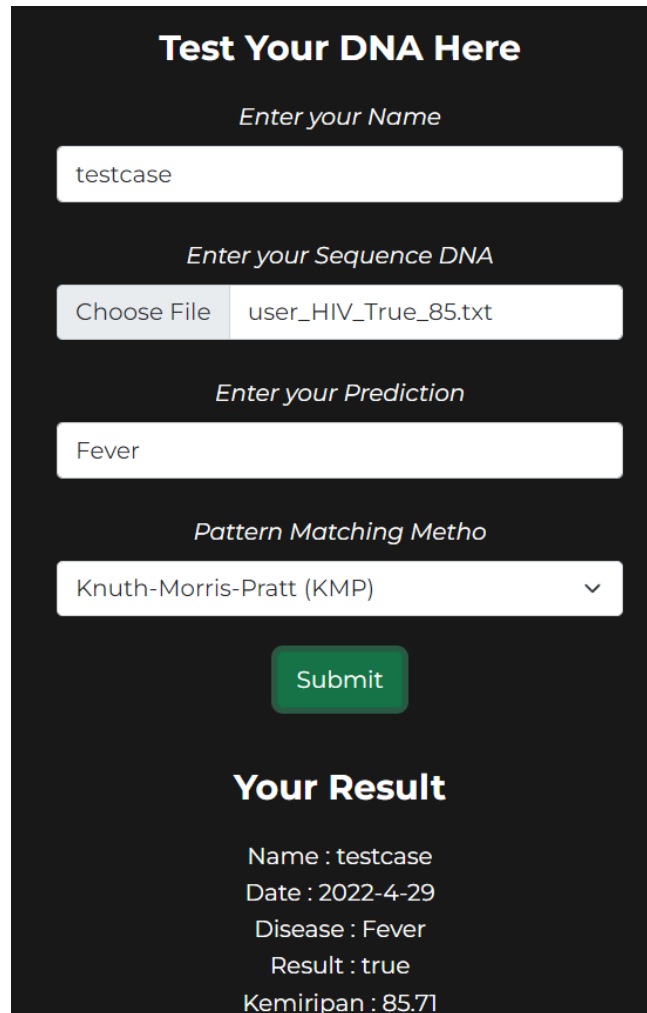
Gambar 4.15. Hasil Tes

b. Tes 2 - Kemiripan $\geq 80\%$

Pada tes ini, karena algoritma *string matching* memberikan informasi bahwa tidak ditemukan *sequence* pada database di *sequence* yang diuji maka selanjutnya dipertimbangkan persentase kemiripan yang dimana bernilai $\geq 80\%$, maka hasil menjadi True.



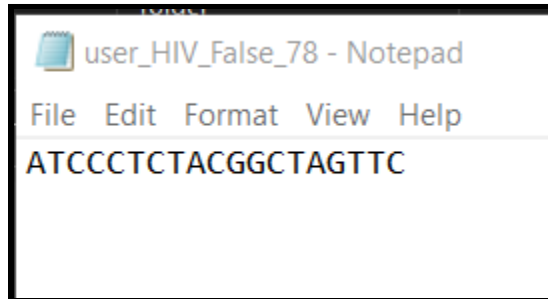
Gambar 4.16. *Sequence* DNA yang diuji

A screenshot of a web application interface with a dark background. At the top, it says "Test Your DNA Here". Below this are four input sections: "Enter your Name" with a text box containing "testcase"; "Enter your Sequence DNA" with a "Choose File" button and a text box containing "user_HIV_True_85.txt"; "Enter your Prediction" with a text box containing "Fever"; and "Pattern Matching Metho" with a dropdown menu showing "Knuth-Morris-Pratt (KMP)". A green "Submit" button is below these sections. At the bottom, under the heading "Your Result", the following information is displayed: "Name : testcase", "Date : 2022-4-29", "Disease : Fever", "Result : true", and "Kemiripan : 85.71".

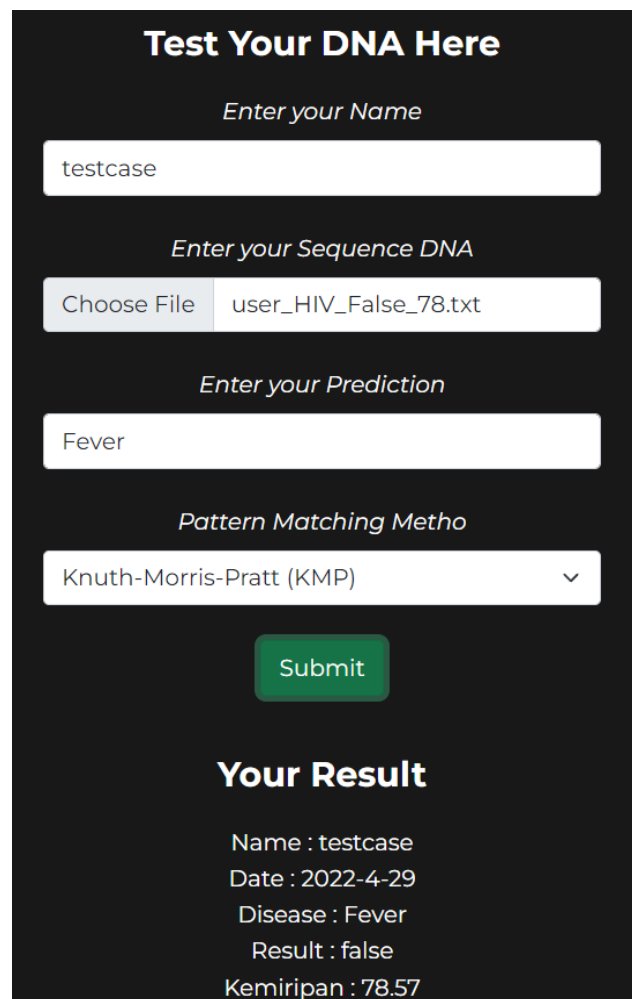
Gambar 4.17. Hasil Tes

c. Tes 3 - Kemiripan < 80%

Pada tes ini, karena algoritma *string matching* memberikan informasi bahwa tidak ditemukan *sequence* pada database di *sequence* yang diuji maka selanjutnya dipertimbangkan persentase kemiripan yang dimana bernilai < 80% , maka hasil tetap False.



Gambar 4.18. *Sequence* DNA yang diuji

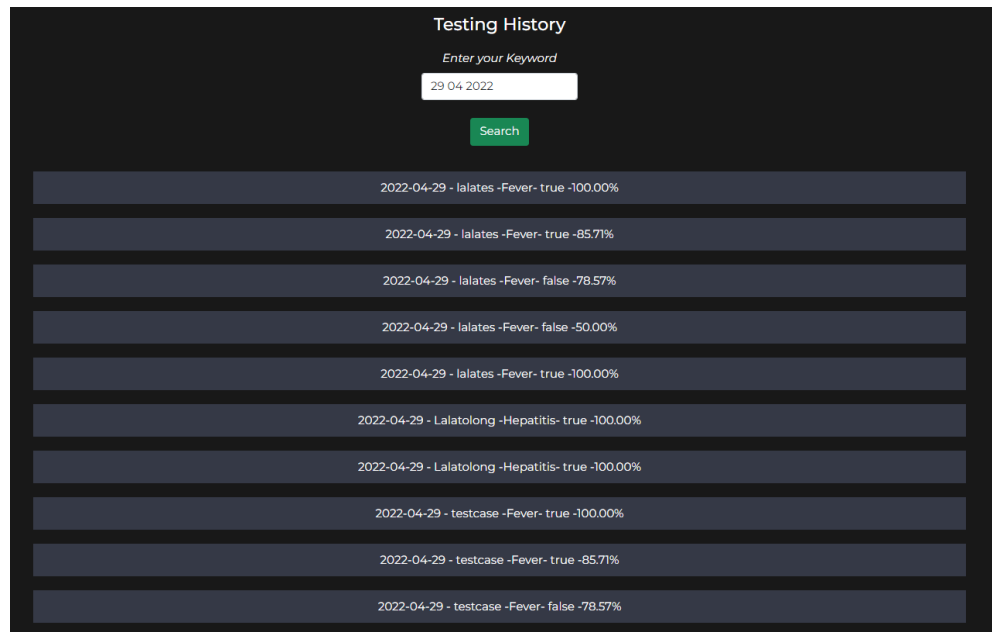
A screenshot of a web application interface for testing DNA sequences. The title is "Test Your DNA Here". It features several input fields: "Enter your Name" with the value "testcase", "Enter your Sequence DNA" with a file selection button "Choose File" and the filename "user_HIV_False_78.txt", "Enter your Prediction" with the value "Fever", and "Pattern Matching Metho" with a dropdown menu showing "Knuth-Morris-Pratt (KMP)". A green "Submit" button is located below the inputs. At the bottom, under the heading "Your Result", the following information is displayed: "Name : testcase", "Date : 2022-4-29", "Disease : Fever", "Result : false", and "Kemiripan : 78.57".

Gambar 4.19. Hasil Tes

8. Fitur Pencarian Riwayat

a. Tanggal saja

Format Tanggal : dd mm yyyy



Testing History

Enter your Keyword

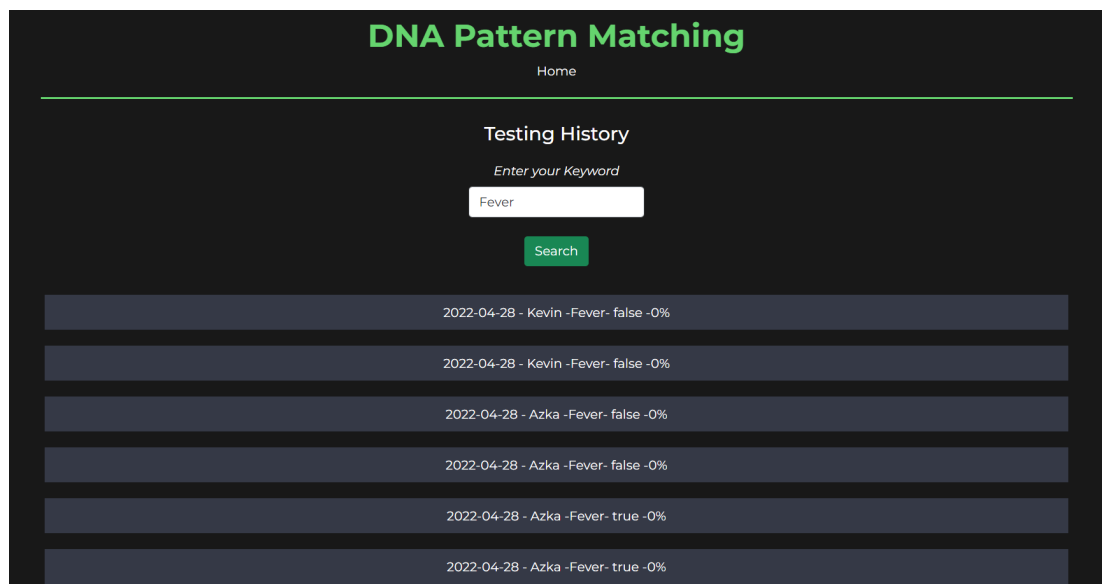
29 04 2022

Search

2022-04-29 - lalates -Fever- true -100.00%
2022-04-29 - lalates -Fever- true -85.71%
2022-04-29 - lalates -Fever- false -78.57%
2022-04-29 - lalates -Fever- false -50.00%
2022-04-29 - lalates -Fever- true -100.00%
2022-04-29 - Lalatolong -Hepatitis- true -100.00%
2022-04-29 - Lalatolong -Hepatitis- true -100.00%
2022-04-29 - testcase -Fever- true -100.00%
2022-04-29 - testcase -Fever- true -85.71%
2022-04-29 - testcase -Fever- false -78.57%

Gambar 4.20. Riwayat Tanggal

b. Nama Penyakit saja



DNA Pattern Matching

Home

Testing History

Enter your Keyword

Fever

Search

2022-04-28 - Kevin -Fever- false -0%
2022-04-28 - Kevin -Fever- false -0%
2022-04-28 - Azka -Fever- false -0%
2022-04-28 - Azka -Fever- false -0%
2022-04-28 - Azka -Fever- true -0%
2022-04-28 - Azka -Fever- true -0%

Gambar 4.21 Riwayat Nama Penyakit

c. Tanggal dan Nama Penyakit

Format : dd mm yyy <nama_penyakit>

Testing History	
Enter your Keyword	
29 04 2022 Fever	
Search	
2022-04-29	- lalates -Fever- true -100.00%
2022-04-29	- lalates -Fever- true -85.71%
2022-04-29	- lalates -Fever- false -78.57%
2022-04-29	- lalates -Fever- false -50.00%
2022-04-29	- lalates -Fever- true -100.00%
2022-04-29	- testcase -Fever- true -100.00%
2022-04-29	- testcase -Fever- true -85.71%
2022-04-29	- testcase -Fever- false -78.57%

Gambar 4.22. Riwayat Tanggal dan Nama Penyakit

D. Analisis Hasil Pengujian

Berdasarkan hasil pengujian yang didapatkan, aplikasi sudah menyelesaikan seluruh permasalahan yang ada dan mengimplementasikan seluruh fitur yang diinginkan.

Aplikasi dapat menerima input nama dan *sequence* DNA penyakit baru dan dimasukkan ke dalam database.

Aplikasi juga dapat melakukan sanitasi dalam tiap *input sequence* DNA untuk memastikan bahwa file yang di-input oleh pengguna sudah tepat.

Aplikasi dapat melakukan tes prediksi penyakit yang diinginkan pengguna terhadap *sequence* DNA yang diberikan oleh pengguna dengan *sequence* DNA yang ada pada *database*.

Dalam penanganan permasalahan tes prediksi, aplikasi juga mengimplementasikan dan menggunakan algoritma *string matching* yang dapat dipilih oleh pengguna, yaitu Knuth-Morris-Pratt (KMP) dan Boyer-Moore(BM).

Aplikasi juga dapat memperhitungkan tingkat kemiripan dari *sequence* DNA input-an pengguna dan *sequence* DNA penyakit pada *database*.

Aplikasi juga merekam seluruh tes prediksi penyakit pada *database* dan dapat ditampilkan pada page riwayat kepada pengguna berdasarkan *input* pencarian yang diinginkan pengguna.

V. Saran dan Kesimpulan

A. Kesimpulan

Dalam pembuatan aplikasi web ini diimplementasikan algoritma *string matching* Knuth-Morris-Pratt (KMP) dan Boyer-Moore dan juga algoritma penghitung tingkat kemiripan (bonus) dengan basis algoritma Hamming Distance. Diimplementasikan juga algoritma regex untuk keperluan sanitasi input dan pencarian riwayat. Pembuatan aplikasi dibuat dengan library ReactJS untuk frontend, dibangun dengan Node.js pada backend dan menggunakan DBMS MongoDB Atlas.

B. Saran

Saran yang kami bisa berikan adalah untuk tidak takut melakukan eksplorasi terhadap berbagai hal yang sekiranya belum pernah dicoba untuk memperluas wawasan dan pengalaman yang ada.

Komentar dan Refleksi

Meskipun waktu yang diberikan cukup panjang tapi karena berbarengan dengan beberapa tugas besar lain sehingga cukup membuat kelabakan dan menjadi pelajaran untuk pintar me-manage waktu lebih baik

VI. Daftar Pustaka

Pencocokan string (String matching/pattern matching), Rinaldi Munir

<https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Pencocokan-string-2021.pdf>

Pencocokan string dengan Regular Expression (Regex), Rinaldi Munir

<https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2018-2019/String-Matching-dengan-Regex-2019.pdf>

What is Hamming Distance, tutorialspoint

<https://www.tutorialspoint.com/what-is-hamming-distance>

Hamming Distance between two strings, geeksforgeeks

<https://www.geeksforgeeks.org/hamming-distance-two-strings/>

VII. Link

Video :

<https://youtu.be/1wnq1liqbT4>

Github :

https://github.com/irsyadazka/Tubes3_13520107

Deploy :

<https://dnapattern.vercel.app>

<https://backend-website-dnapattern.herokuapp.com> (backend)