LAPORAN TUGAS BESAR 3

IF2211/Strategi Algoritma

Semester II Tahun 2021/2022

Penerapan String Matching dan Regular Expression dalam

DNA Pattern Matching

Dipersiapkan oleh:

Kelompok 55

Rozan Fadhil Al Hafidz	13520039
Amar Fadil	13520103
Malik Akbar Hashemi Rafsanjani	13520105

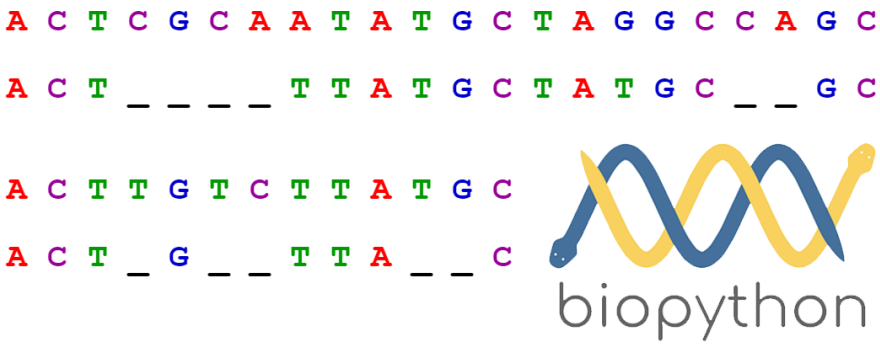
Sekolah Teknik Elektro dan Informatika - Institut Teknologi Bandung

Jl. Ganesha 10, Bandung 40132

Daftar Isi

1 Deskripsi Tugas

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*.



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.

2 Landasan Teori

Misalkan diberikan sebuah *string* dan sebuah *pattern* yang memiliki panjang lebih pendek dari *string*. String matching adalah proses pencarian lokasi pertama *pattern* pada *string*. Algoritma string matching digunakan pada pencarian di dalam teks editor, *web search engine*, analisis citra, dan *bioinformatics*. Ada beberapa algoritma *string matching*, contohnya dari Knuth–Morris–Pratt (KMP) dan Boyer-Moore (BM)

2.1 Algoritma Knuth–Morris–Pratt (KMP)

Algoritma Knuth-Morris-Pratt (KMP) melakukan pencocokan *string* dari kiri ke kanan (seperti algoritma *brute force*). Tetapi, algoritma ini lekaukan pergeseran lebih cerdas dibandingkan dengan algoritma *brute force*. Hal tersebut dilakukan untuk mengurangi perbandingan yang sia-sia.

Misalkan *pattern* disimpan di dalam variabel P. Langkah pertama yang dilakukan algoritma ini adalah dengan membuat larik yang berisikan nilai *border function* b(k) untuk setiap kemungkinan posisi ketidakcocokan yang terjadi di P[].

Dengan :

j = ketidakcocokan yang terjadi di P[]
k = posisi sebelum ketidakcocokan (k = j – 1)

Border function b(k) didefinisikan sebagai *prefix* terpanjang dari P[0..k] yang juga merupakan *suffix* dari P[1..k]. Berikut ini adalah contoh dari *border function* pada pattern “abaaba”.

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

Dengan menggunakan *border function* tersebut, algoritma ini memodifikasi pergeseran pada algoritma *brute force*. Misalnya jika ketidakcocokan karakter terjadi pada indeks ke-j, maka ubah nilai j menjadi b(k) dengan k = j – 1.

2.2 Algoritma Boyer-Moore (BM)

Misalkan *pattern* P akan dicocokkan dengan teks T. Algoritma Boyer-Moore melakukan pencocokan string menggunakan dua teknik, yaitu :

1. Teknik *looking-glass*
   Mencari P di T dengan bergerak ke belakang melewati P, dimulai dari huruf terakhir P

2. Teknik *character-jump*
   Dilakukan ketika karakter yang dicocokkan tidak sama. Misalkan huruf yang tidak sama pada T adalah x. Terdapat 3 kasus
   2.1. Kasus 1 :
   Jika P mengandung x, cobalah geser P ke kanan untuk menyesuaikan posisi x di P dengan T[i]
   2.2. Kasus 2 :
   Jika P mengandung x, namun penggeseran ke kemunculan terakhir tidak mungkin, maka geser P sebesar 1 karakter ke T[i+1]
   2.3. Kasus 3 :
   Jika kasus 1 dan kasus 2 tidak memenuhi, geser P sehingga menyesuaikan P[0] dengan T[i+1]

   Pada kasus 1 dan 2, algoritma BM membutuhkan informasi mengenai kemunculan terakhir sebuah karakter pada T dalam pola P. Oleh karena itu, sebelum menjalankan pengecekan string, mula-mula algoritma ini melakukan pembuatan larik yang menyimpan kemunculan terakhir elemen pada P dengan menggunakan fungsi *Last Occurrence* L(x).

   Misalkan P sebagai berikut

a	b	a	c	a	b
0	1	2	3	4	5

   Dengan A = {a, b, c, d}

   Maka larik yang berisi hasil dari fungsi *Last Occurrence* sebagai berikut

x	a	b	c	d	b
L(x)	1	2	3	4	5

   Larik tersebut digunakan untuk menentukan lokasi pergeseran.

2.3 Regular Expression (Regex)

Regular expression adalah sekumpulan notasi dan karakter yang digunakan untuk mendeskripsikan suatu pola pada pencarian berbasis karakter. Biasanya regex digunakan pada operasi “find” atau “find and replace” pada string.

Regex juga bisa digunakan untuk melakukan validasi input. Berikut ini adalah ekspresi regex untuk melakukan validasi email

```
^([a-zA-Z0-9_\-\.]+)@([a-zA-Z0-9_\-\.]+)\.([a-zA-Z]{2,5})$
```

Contoh string yang diterima ekspresi regex tersebut adalah “abc@yyy.zzz” dan contoh string yang akan ditolak ekspresi tersebut adalah “www.com”

2.4 Aplikasi Web yang Dibangun

Aplikasi web yang dibangun adalah aplikasi yang memiliki beberapa fitur. Aplikasi ini dapat melakukan penambahan penyakit berdasarkan nama dan file sequence DNA penyakit tersebut. Penyakit-penyakit yang telah ditambahkan dapat dites terhadap sequence DNA dari pengguna dengan cara menggunggah nama pengguna, file sequence DNA, memilih penyakit dan memilih metode pengetesan. Hasilnya akan langsung ditampilkan apakah DNA pengguna memiliki kemiripan dengan DNA penyakit. Hasil-hasil pengetesan yang telah dilakukan dapat dilakukan pencarian berdasarkan query pengguna menggunakan regular expression.

Aplikasi ini memiliki berbagai manfaat. Dengan memiliki sequence DNA dari penyakit dan pengguna, kita dapat melakukan pengetesan seberapa beresiko pengguna mendapatkan penyakit yang dites. Hal ini akan membantu para pengguna untuk mendapatkan informasi lebih awal sebelum mendapatkan penyakit tersebut, ataupun penyakit menjadi parah. Informasi tersebut dapat digunakan untuk melakukan langkah-langkah medis preventif dari awal.

3 Analisis Pemecahan Masalah

3.1 Langkah Pemecahan Masalah Setiap Fitur

3.1.1 Menerima Input DNA Penyakit Dalam Bentuk File

Aplikasi yang kami buat dapat menerima input nama penyakit pada *input field* beserta *sequence* DNA penyakit tersebut berupa string yang disimpan dalam bentuk file. File tersebut akan dibaca dan kemudian akan melalui tahap sanitasi sebelum disimpan ke database

3.1.2 Menerima Input DNA Seseorang Dalam Bentuk File

Aplikasi yang kami buat juga dapat menerima input nama pengguna pada *input field*, *sequence* DNA pengguna tersebut berupa string yang disimpan dalam bentuk file, serta *sequence* DNA penguna tersebut berupa string yang disimpan dalam bentuk file. File tersebut kemudian akan dibaca dan kemudian akan melalui tahap sanitasi sebelum dilakukan pengecekan penyakit. Selain itu, aplikasi yang kami buat juga memberikan pilihan algoritma pencocokan string untuk melakukan penyecekan penyakit, yaitu menggunakan algoritma KMP ataupun BM.

3.1.3 Sanitasi Input DNA menggunakan Regex

Sebelum disimpan ke database, input tersebut harus divalidasi terlebih dahulu menggunakan regex. DNA yang valid adalah string yang hanya terdiri dari huruf A, G, C, atau T, case sensitive dan tanpa spasi. Oleh karena itu, input yang diberikan harus memenuhi ekspresi regex /^[ACGT]+$/.

3.1.4 Melakukan Pengecekan Penyakit Pengguna

Aplikasi yang kami buat dapat melakukan pengecekan penyakit pengguna menggunakan algoritma pencocokan string Knuth–Morris–Pratt (KMP) ataupun Booyer-Moore (BM). Selain mencari pola string yang sama, algoritma yang kami implementasikan juga bisa menghitung persentasi kemiripan DNA penyakit dengan DNA pengguna yang dicek.

3.1.5 Menampilkan Hasil Pengecekan Penyakit Pengguna

Aplikasi yang kami buat dapat langsung menampilkan hasil pengecekan penyakit pengguna setelah data di-*submit* ke aplikasi. Pengguna dikategorikan terinfeksi penyakit yang dicek jika tingkat kemiripan DNA pengguna dan tingkat kemiripan DNA penyakit lebih dari atau sama dengan 80%.

3.1.6 Melihat Riwayat Pengecekan Penyakit Pengguna

Aplikasi yang kami buat dapat menyimpan dan menampilkan riwayat pengecekan yang pernah dilakukan oleh pengguna. Selain itu, terdapat juga kolom pencarian yang menerima input riwayat yang ingin dilihat dan berfungsi untuk memfilter riwayat pencarian sesuai input yang diberikan. Riwayat pencarian tersebut difilter menggunakan regex.

3.2 Fitur Fungsional Aplikasi

3.2.1 Menambahkan DNA Penyakit Baru

Fitur menambahkan DNA penyakit baru digunakan untuk menambahkan penyakit baru berupa nama penyakit dan sequence DNA-nya yang disimpan ke dalam *database*. Aplikasi menerima file input sequence DNA dan melakukan sanitasi input dengan regex sehingga hanya menerima sequence DNA yang valid. DNA yang valid terdiri dari huruf A, G, C, dan T saja tanpa spasi, baris baru, dan karakter lainnya. Penyimpanan sequence DNA dalam bentuk file pada backend, dimana semua file DNA akan di-rename menjadi <timestamp>.dna dan disimpan pada folder files/penyakit. Nama file akan disimpan pada database dan sewaktu-waktu akan dilakukan pembacaan file penyakit sesuai dengan nama file yang tersimpan dalam *database*.

3.2.2 Memprediksi Penderita Penyakit Berdasarkan Sequence DNA

Fitur memprediksi penderita penyakit berdasarkan sequence DNA digunakan untuk memprediksi seseorang apakah terjangkit suatu penyakit berdasarkan sequence DNA seseorang tersebut. Sequence DNA orang tersebut memiliki asumsi lebih besar atau sama dengan sequence DNA penyakit yang ingin diprediksi. Prediksi dilakukan dengan mencocokkan sequence DNA seseorang dan sequence DNA penyakit yang telah disimpan pada *database*. Pencocokan sequence DNA menggunakan algoritma string matching Boyer-Moore atau Knuth-Morris-Pratt (KMP). Hasil prediksi akan ditampilkan dengan tingkat kemiripan menggunakan metode Longest Common Subsequence (LCS) yang didapatkan sesuai alur pencarian algoritma terpilih, sehingga mungkin saja tingkat kemiripan akan berbeda untuk kedua algoritma tersebut. Prediksi akan menampilkan tanggal prediksi, nama pengguna, nama penyakit yang diprediksi, hasil prediksi, dan tingkat kemiripan dalam persen.

3.2.3 Menampilkan Riwayat Prediksi Penderita Penyakit

Fitur menampilkan riwayat prediksi penderita penyakit digunakan untuk menampilkan daftar riwayat prediksi yang tercatat selama pengguna menggunakan fitur prediksi penyakit. Daftar ini dapat dicari berdasarkan nama penyakit, nama pengguna, tanggal prediksi, atau kombinasi dari ketiganya. Jika diinginkan, riwayat juga dapat dihapus dari basis data. Daftar ini menampilkan tanggal prediksi, nama pengguna, nama penyakit yang diprediksi, hasil prediksi, dan tingkat kemiripan dalam persen untuk setiap riwayat prediksi.

3.3 Arsitektur Aplikasi

Backend aplikasi ini menggunakan bahasa pemograman Go dengan framework Echo (Web Framework) dan GORM (Object-Relational Mapping). Pendekatan yang digunakan adalah REST-ful API dengan arsitektur Model-View-Controller (View tidak diperlukan karena pendekatan REST API). Terdapat 3 controller utama, CRUD untuk penyakit (/penyakit), CRUD untuk check (/check), dan *endpoint* untuk testing algoritma (/testing/kmp dan /testing/bm). Masing-masing controller (kecuali testing) memiliki *responsibility* satu model yang terkait, yakni DNAPenyakit untuk controller Penyakit dan DNACheck untuk controller Check. Migrasi basis data dan pemodelan dilakukan dengan bantuan *auto migrate* dari GORM dan fungsionalitasnya sebagai ORM. Selain itu, basis data yang digunakan adalah MariaDB (dengan driver MySQL bawaan GORM) dengan koneksi ke *localhost* pada user *stima* dengan basis data *tubes3stima*.

Frontend aplikasi ini menggunakan bahasa pemrograman Typescript dengan framework Next JS. Sementara itu, untuk membantu *styling* dan penggunaan komponen *reusable*, digunakan library bootstrap dan react-bootstrap. Sementara itu, untuk melakukan *hit endpoint* API digunakan library bantuan dari axios untuk mempermudah. Library axios tersebut di-*wrap* pada kelas ApiSrv yang bertindak sebagai kelas yang dapat dipanggil untuk membantu pengambilan data dari backend.

4 Implementasi dan Pengujian

4.1 Spesifikasi Teknis Program

4.1.1 Library

Spesifikasi teknis program untuk library terdiri dari fungsi-fungsi berikut:

1. Max(a, b : int)
   Mengembalikan nilai terbesar di antara a dan b
2. Min(a, b : int)
   Mengembalikan nilai terkecil di antara a dan b
3. GetDnaCode(c : byte)
   Mengembalikan code DNA dari input karakter DNA yang diberikan. Digunakan pada algoritma Boyer-Moore.
4. buildLast(pattern : string)
   Melakukan pencarian kemunculan terakhir tiap karakter pada pattern. Digunakan pada algoritma Boyer-Moore
5. BMMatch(text, pattern: string)
   Melakukan pencocokan string dengan mengecek apakah terdapat pattern di dalam teks dengan menggunakan algoritma Boyer-Moore. Mengembalikan nilai index kemunculan pattern dan jumlah karakter maksimum yang cocok.
6. computeFail(pattern : string)
   Melakukan perhitungan prefix yang sama dengan suffix untuk setiap kemungkinan terjadi kesalahan pencocokan. Digunakan pada algoritma Knuth-Morris-Pratt
7. KMPMatch(text, pattern : string)
   Melakukan pencocokan string dengan mengecek apakah terdapat pattern di dalam teks dengan menggunakan algoritma Knuth-Morris-Pratt. Mengembalikan nilai index kemunculan pattern dan jumlah karakter maksimum yang cocok.

4.1.2 Backend

Adapun spesifikasi teknis program untuk backend terdiri dari fungsi-fungsi berikut:

1. getAllCheck(c: echo.Context)
   Mengambil semua data riwayat prediksi yang terdaftar di basis data. *Endpoint* ada pada GET /check/
2. getCheck(c: echo.Context)
   Mengambil satu data riwayat prediksi yang teridentifikasi unik dengan id pada parameternya. *Endpoint* ada pada GET /check/:id
3. createCheck(c: echo.Context)
   Membuat prediksi penyakit baru yang menerima multipart/form-data berupa nama penyakit (“penyakit”), nama pengguna (“pengguna”), algoritma yang digunakan (“method”), dan file sekuen DNA pengguna (“dna”). *Endpoint* ada pada POST /check/
4. updateCheck(c: echo.Context)
   Memperbarui data prediksi berupa data nama pengguna (“pengguna”). *Endpoint* ada pada PUT /check/:id
5. deleteCheck(c: echo.Context)

Menghapus data prediksi yang tersimpan melalui id unik. *Endpoint* ada pada DELETE /check/:id

6. getAllPenyakit(c: echo.Context)
   Mengambil semua data penyakit yang terdaftar di basis data. *Endpoint* ada pada GET /penyakit/
7. getPenyakit(c: echo.Context)
   Mengambil satu data penyakit yang teridentifikasi unik dengan nama penyakit (“id”) pada parameternya. *Endpoint* ada pada GET /penyakit/:id
8. createPenyakit(c: echo.Context)
   Membuat penyakit baru yang menerima multipart/form-data berupa nama penyakit (“penyakit”) dan file sekuen DNA penyakit (“dna”). *Endpoint* ada pada POST /penyakit/
9. updatePenyakit(c: echo.Context)
   Memperbarui data penyakit berupa data nama penyakit (“penyakit”) dan file sekuen DNA penyakit (“dna”). *Endpoint* ada pada PUT /penyakit/:id
10. deletePenyakit(c: echo.Context)
    Menghapus data penyakit yang tersimpan dengan nama penyakit. *Endpoint* ada pada DELETE /penyakit/:id
11. bmRoute(c: echo.Context)
    Melakukan testing string terhadap pattern dengan algoritma Boyer-Moore. *Endpoint* ada pada GET /testing/bm/:string/:pattern
12. kmpRoute (c: echo.Context)
    Melakukan testing string terhadap pattern dengan algoritma Knuth-Morris-Pratt. *Endpoint* ada pada GET /testing/kmp/:string/:pattern

4.1.3 Frontend

Terakhir, spesifikasi teknis program untuk frontend terdiri dari fungsi dan struktur berikut:

1. Fungsi updateRes(newQuery: string)
   Melakukan filtering ke database test berdasarkan input query dari user
2. Fungsi convertToString(data: ISearchRes)
   Melakukan konversi dari format data dari database untuk menjadi string yang nantinya akan dites terhadap query search dari pengguna menggunakan regular expression
3. Interface ISearchRes
   Interface ini digunakan sebagai interface dari hasil pencarian dari database yang terdiri atas:
   CreatedAt: string;
   DeletedAt: string | null;
   ID: number;
   Match: number;
   Pengguna: string;
   Penyakit: string;
   Result: boolean;
   UpdatedAt: string;
4. Interface ITestDNARes
   Interface ini digunakan sebagai interface dari hasil pengetesan DNA yang terdiri atas:
   CreatedAt: string;

DeletedAt: string | null;
Penyakit: string;
DNA: string;
UpdatedAt: string;

4.2 Tata Cara Penggunaan Program

4.2.1. Setup Program
Pastikan di komputer Anda telah terinstall bahasa Go, Node JS, dan MySQL

4.2.2. Cara Menjalankan Program
- Setup database
  - Jalankan command berikut pada MySQL server anda untuk membuat database, user dan memberikan priviledge
    ```sql
    CREATE DATABASE tubes3stima;
    CREATE USER 'stima'@'127.0.0.1' IDENTIFIED BY 'fFDzwk4Z!FpU_QU';
    GRANT ALL PRIVILEGES ON tubes3stima.* TO 'stima'@'127.0.0.1' WITH GRANT OPTION;
    ```
- Cara menjalankan Backend program:
  - Pindah ke directory backend
    ```
    cd src/backend
    ```
  - Jalankan program
    ```
    go run server.go
    ```
- Cara menjalankan Frontend program:
  - Pindah ke directory frontend
    ```
    cd src/frontend
    ```
  - Install semua dependencies:
    ```sh
    npm install
    yarn install    # bila anda memiliki yarn
    ```
  - Jalankan program
    ```sh
    npm run dev
    yarn dev        # bila anda memiliki yarn
    ```

4.2.3. Penggunaan Program

a. Menambahkan penyakit
- Buka halaman utama (http://localhost:3000/), akan muncul halaman add penyakit
- Masukkan nama penyakit pada input field nama penyakit
- Unggah file sequence DNA dari penyakit pada input field file sequence DNA
- Klik submit
- Ketika sukses, maka anda akan menerima pemberitahuan sukses menambahkan penyakit, dan jika error dalam penambahan penyakit ini, anda akan mendapatkan pemberitahuan error beserta pesan error.

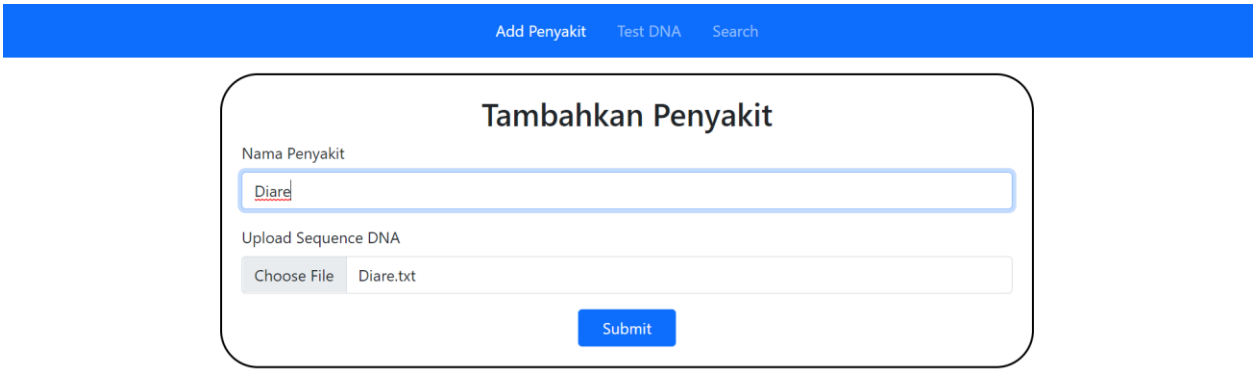
b. Test DNA
- Buka halaman test DNA (http://localhost:3000/test-dna), akan muncul halaman test DNA
- Masukkan nama pengguna yang akan dites penyakitnya
- Masukkan file sequence DNA dari pengguna pada input field file sequence DNA
- Pilih penyakit yang akan dite spada dropdown pilih penyakit
- Pilih metode pengetesan
- Klik submit
- Ketika sukses, maka anda akan menerima pemberitahuan sukses menambahkan penyakit, dan jika error dalam penambahan penyakit ini, anda akan mendapatkan pemberitahuan error beserta pesan error.

c. Search Database
- Buka halaman search (http://localhost:3000/search), akan muncul halaman search
- Masukkan query yang ingin dicari pada input field search
- Query akan tampil setelah satu detik pengguna berhenti mengetikkan query
- Query akan diproses menggunakan regular expression dan dapat berdasarkan nama pengguna, nama penyakit, tanggal, bulan dalam bahasa inggris, bulan dalam bahasa indonesia, tahun, format tanggal yyyy-mm-dd, format tanggal dd-mm-yyyy, hasil test (True / False)

4.3 Hasil Pengujian

4.3.1 Memasukkan DNA Penyakit ke Database

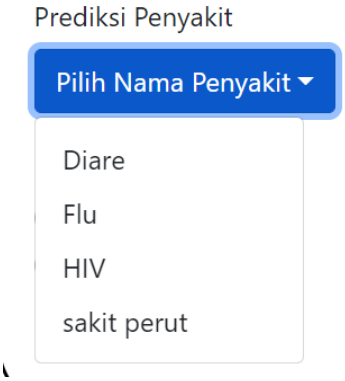
Misalnya saya ingin menambahkan penyakit diare ke dalam database. Berikut ini adalah tangkapan layarnya.



Berikut ini adalah isi file yang di-*upload*, yaitu diare.txt

Diare.txt
GGGGCGCGGGAGAAGCCCCTGGGCCTCCGGAGATGGGGGACACCCCACGCCAGTTCGGAGGCGCGAGGCC
GCGCTCGGGAGGCGCGCTCCGGGGGTGCCGCTCTCGGGGCGGGGGCAACCGGCGGGGTCTTTGTCTGAGC
CGGGCTCTTGCCAATGGGGATCGCAGGGTGGGCGCGGCGGAGCCCCCGCCAGGCCCGGTGGGGGCTGGGG
CGCCATTGCGCGTGCGCGCTGGTCCTTTGGGCGCTAACTGCGTGCGCGCTGGGAATTGGCGCTAATTGCG
CGTGCGCGCTGGGACTCAAGGCGCTAACTGCGCGTGCGTTCTGGGGCCCGGGGTGCCGCGGCCTGGGCTG
GGGCGAAGGCGGGCTCGGCCGGAAGGGGTGGGGTCGCCGCGGCTCCCGGGCGCTTGCGCGCACTTCCTGC
CCGAGCCGCTGGCCGCCCGAGGGTGTGGCCGCTGCGTGCGCGCGCGCCGACCCGGCGCTGTTTGAACCGG

Maka pada kolom pemilihan nama penyakit, kata “Diare” akan muncul dan dapat dipilih untuk melakukan pengecekan terhadap penyakit diare



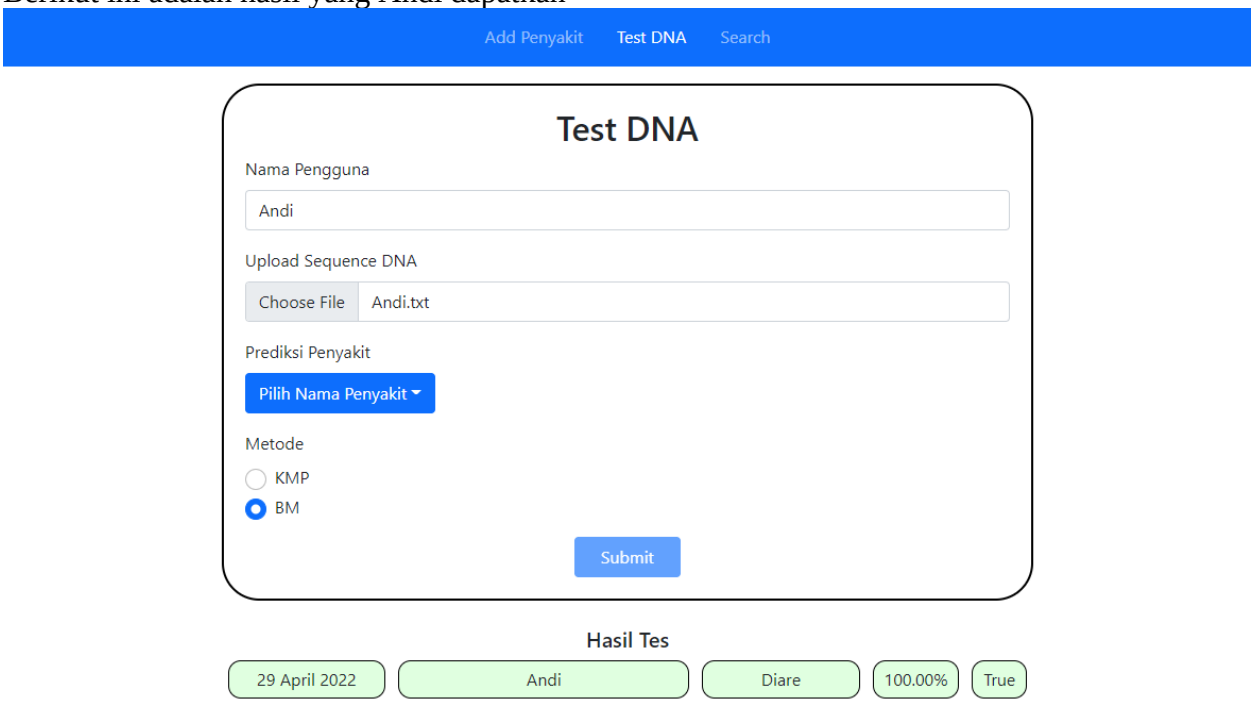
4.3.2 Melakukan Pengecekan Penyakit - Kasus True (Andi)

Misalkan Andi adalah orang yang terserang penyakit diare. Andi ingin melakukan pengecekan penyakit diare menggunakan aplikasi yang kami buat. Andi memasukkan namanya dan *sequence* DNA-nya di dalam halaman Test DNA aplikasi kami. Berikut ini adalah *sequence* DNA Andi.

Andi.txt

```
ACCGCCGAGACCGCGTCCGCCCCGCGAGCACAGAGCCTCGCCTTTGCCGATCCGCCGCCCGTCCACACCCGCCGCCAGGTAAGCCCGGC
AGCCGACCGGGGCAGGCGGCTCACGGCCCGGCCGCAGGCGGCCGCGGCCCCTTCGCCCGTGCAGAGCCGCCGTCTGGGCCGCAGCGGGG
GGCGCATGGGGGGGGAACCGGACCGCCGTGGGGGGCGCGGGAGAAGCCCCTGGGCCTCCGGAGATGGGGGACACCCCACGCCAGTTCGG
AGGCGCGAGGCCGCGCTCGGGAGGCGCGCTCCGGGGGTGCCGCTCTCGGGGCGGGGGCAACCGGCGGGGTCTTTGTCTGAGCCGGGCTC
TTGCCAATGGGGATCGCAGGGTGGGCGCGGCGGAGCCCCCGCCAGGCCCGGTGGGGGCTGGGGCGCCATTGCGCGTGCGCGCTGGTCCT
TTGGGCGCTAACTGCGTGCGCGCTGGGAATTGGCGCTAATTGCGCGTGCGCGCTGGGACTCAAGGCGCTAACTGCGCGTGCGTTCTGGG
GCCCGGGGTGCCGCGGCCTGGGCTGGGGCGAAGGCGGGCTCGGCCGGAAGGGGTGGGGTCGCCGCGGCTCCCGGGCGCTTGCGCGCACT
TCCTGCCCGAGCCGCTGGCCGCCCGAGGGTGTGGCCGCTGCGTGCGCGCGCGCCGACCCGGCGCTGTTTGAACCGGGCGGAGGCGGGGC
TGGCGCCCGGTTGGGAGGGGGTTGGGGCCTGGCTTCCTGCCGCGCGCCGCGGGGACGCCTCCGACCAGTGTTTGCCTTTTATGGTAATA
ACGCGGCCGGCCCGGCTTCCTTTGTCCCCAATCTGGGCGCGCGCCGGCGCCCCCTGGCGGCCTAAGGACTCGGCGCGCCGGAAGTGGCC
AGGGCGGGGGCGACCTCGGCTCACAGCGCGCCCGGCTATTCTCGCAGCTCACCATGGATGATGATATCGCCGCGCTCGTCGTCGACAAC
GGCTCCGGCATGTGCAAGGCCGGCTTCGCGGGCGACGATGCCCCCCGGGCCGTCTTCCCCTCCATCGTGGGGCGCCCCAGGCACCAGGT
AGGGGAGCTGGCTGGGTGGGGCAGCCCCGGGAGCGGGCGGGAGGCAAGGGCGCTTTCTCTGCACAGGAGCCTCCCGGTTTCCGGGGTGG
GGGCTGCGCCCGTGCTCAGGGCTTCTTGTCCTTTCCTTCCCAGGGCGTGATGGTGGGCATGGGTCAGAAGGATTCCTATGTGGGCGACG
AGGCCCAGAGCAAGAGAGGCATCCTCACCCTGAAGTACCCCATCGAGCACGGCATCGTCACCAACTGGGACGACATGGAGAAAATCTGG
CACCACACCTTCTACAATGAGCTGCGTGTGGCTCCCGAGGAGCACCCCGTGCTGCTGACCGAGG
```

Berikut ini adalah hasil yang Andi dapatkan



Menurut aplikasi kami, Andi positif terkena penyakit Diare.

4.3.3 Melakukan Pengecekan Penyakit - Kasus Parsial (Budi)

Misalkan Budi juga merupakan orang yang terserang penyakit diare. Budi juga ingin melakukan pengecekan penyakit diare menggunakan aplikasi yang kami buat. Budi memasukkan namanya

dan *sequence* DNA-nya di dalam halaman Test DNA aplikasi kami. Berikut ini adalah *sequence* DNA Budi.

Budi.txt

```
GTCTGTCTGTCTCTCAGAAGACTGGGTCTTGGGAAGCCCAGTACTGGCTTTCTATCCAAGAAGCTTATGTTTTCATTCCAGGTCTAG
ATTAGATGGAAAGTCCCACTGGAACTCCAGGCTTCTGAAGTCTCCTTTCTAGATTCAGTAAATGTTTATTGCTTGTCTAAGATGTGG
CAGACATTATGAGTTTAAAGGGGGATCTTAATGATTATATTATTGCTTCCAAACATATGTAGTTCTTGAAACCACTAGTTAAACATT
GGTAATCTGCTTTTCAAATATGTAAAACAGACTAGTACTCCGACTGACTTTCTTACGGTGCACATTCCCAGTGGTAGGACATGATGT
TATCTGAAGTCAGGTGGTGACACCAATAGACTCAACAGCTTCCGTTGGCTTTATTTACTTAGCCTCATAAGTCATAACTTTTCTAAA
AGGAAAAAAAAGGTGCTGGATTTGATATCTTAATCTCTGAAGGCTCTATCACTACCAAAATGTCTGGTCCTTTTGACGGGCTATCTGT
GAAACGTGTATGCTAAGAATGGCTTTCAATCTTAGCAGTTCAAGCAGCTTGCTTTAAAAGCTATATGGGAATTATCACAGGGGGCCC
TTCTTTCATGTTTACACTTCTCCCCACGCCAGTTCGGAGGCGCGAGGCCGCGCTCGGGAGGCGCGCTCCGGGGGTGCCGCTCTCGGG
GCGGGGGCAACCGGCGGGGTCTTTGTCTGAGCCGGGCTCTTGCCAATGGGGATCGCAGGGTGGGCGCGGCGGAGCCCCCGCCAGGCC
CGGTGGGGGCTGGGGCGCCATTGCGCGTGCGCGCTGGTCCTTTGGGCGCTAACTGCGTGCGCGCTGGGAATTGGCGCTAATTGCGCG
TGCGCGCTGGGACTCAAGGCGCTAACTGCGCGTGCGTTCTGGGGCCCGGGGTGCCGCGGCCTGGGCTGGGGCGAAGGCGGGCTCGGC
CGGAAGGGGTGGGGTCGCCGCGGCTCCCGGGCGCTTGCGCGCACTTCCTGCCCGAGCCGCTGGCCGCCCGAGGGTGTGGCCGCTGCG
TGCGCGCGCGCCGACCCGGCGCTGTTTGAACCGGAAGAAAATCCCCAAATGGAAAAATCATTTGGGAAAAAGAGACTACTGGTTACA
GAAGTGGCATCGAGAGTCAAGATCTGGAAAGCACACTAATGCCCTACTAAGGTCAAGTCAATAGGAAGAGAGAGAATGTTCTATACA
GTTCCAGATGAACTCCATTAATCACAGCACAATTTTGGATCTTCTACATGAAAAGACTTATTGGAGAACTTAAAGAACTGCCAGGAA
GTGGCTGCTGGTTACTATCTCCAGGAAACCTGAAGCCTGGATTGTTGGGTATCAGTGGAAATAGGGAGATGGATGACCGTTACAGGA
```

Berikut ini hasil yang Budi dapatkan

Menurut aplikasi kami, Budi positif terkena penyakit Diare.

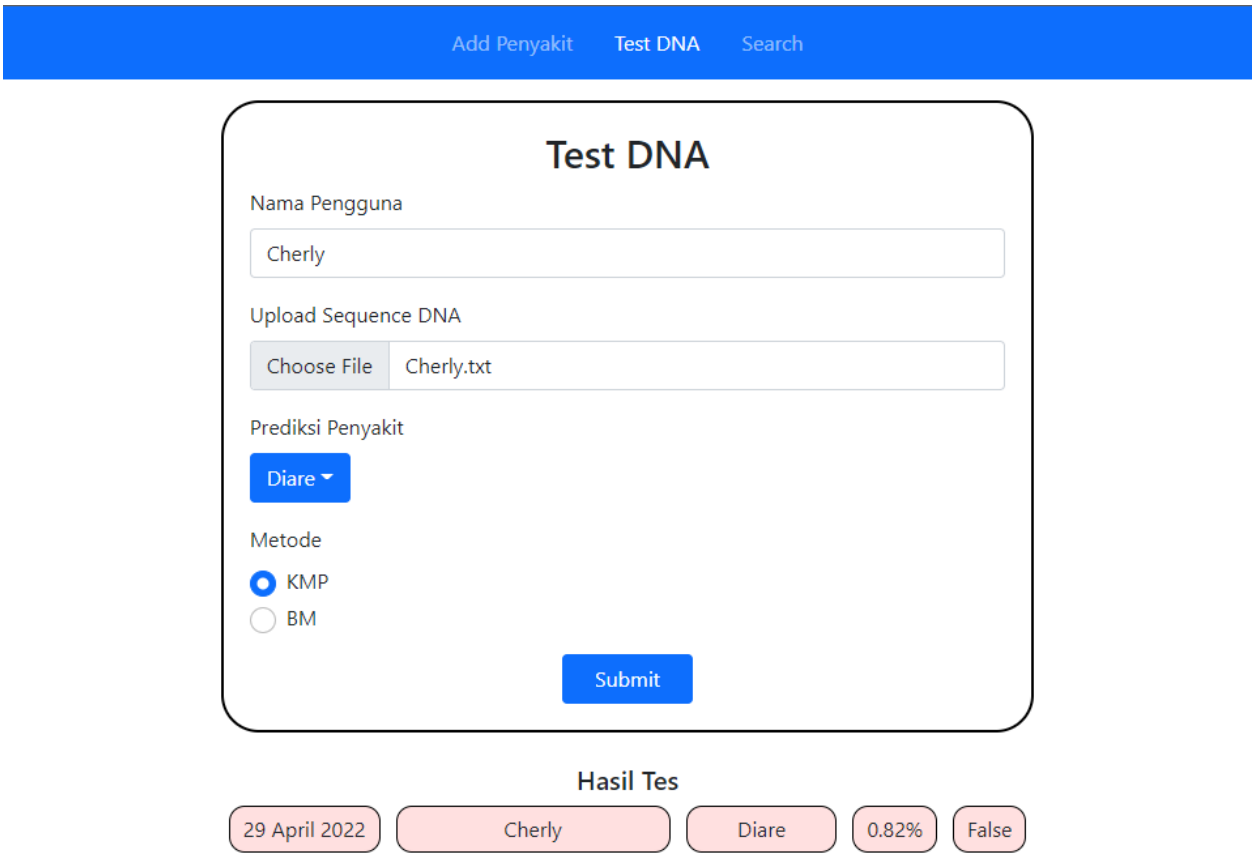
4.3.4 Melakukan Pengecekan Penyakit - Kasus False (Cherly)

Misalkan Cherly bukan merupakan orang yang terserang penyakit diare. Cherly juga ingin melakukan pengecekan penyakit diare menggunakan aplikasi yang kami buat. Cherly memasukkan namanya dan *sequence* DNA-nya di dalam halaman Test DNA aplikasi kami. Berikut ini adalah *sequence* DNA Cherly.

Cherly.txt

```
TGCCCCTGCTCTTTCTTTAAGGATAAAGATGTACAAGACCTCTTTAACTCCTTGCATTGTTACTGTAATTTACACTTAGCTCATTTCAA
ATTACACACAGCTAGAACACGAAGTAGCAGCAAAATAACAGCCTGTGTGTAAATTCCACCACCGGAAGCTTTGTATTTTCACAGACACA
CAGCAACTACAGAAACTCTGCAAAGCAAGTATCCTCTTGTAAGGACCCACAGCTCCACTCCTTGCCAGTTCCCATCCAACCTCAGCCCC
CATATGCAGAGAAACCATATACAGATAGATATGTGGATATGAGCTTGAACATCAGCACTGTGAGGCAAGAAAGCACACAAGTCACAGAG
TAAAGCCACAGCACACTGTCGATCCTGGGTAGCTTCCTGAGCCTCTGTGCCAGCCCAGCCTCCTAAGCCCTGATTCCCACACCCATTGC
AGCGTGCAACCATACAGTATGTTGCTTTACTCGCAGGGATACTTTACAGTAATAAATTTTGGAAGAACCTTTTATGTACATTTCATTTT
TCAGGATAGTTGACTGAACTAGTATGCATCAAATTCTTCCTGCATACTGCTATACTTTTAACTATGGATTGCATAAGAAGTATAATCTC
TGTTCTTAAATACCCTAAAATCCAGACTTCATTCATTACATTCCATTTCTGAAAGTGAAACTTAAATCCATTGAGGTTTGAGAATGCCT
CGGCACAGAGGATAGGGAGAGCTGTAAAAGGCCTGGCTGCATTTGCGTAATGTGTGCACAAATGTTGCCCATACCTCCGCTCTGTCTGC
GCTCTACAGAGCAGCTCAGAGTTCTCACACAACACGGCCTGTGCCACACTTTGTCTTATTTGAATAATCATGAGAGCACAGCCTTATTT
CCCTCTGTGATTGCTTGATAAAGGCAACACAGTAGGCAAGCTTCAAGTTTCTATGTTTGGATAGAACCAAATCAGTGCGAAAAATCTGT
GTGCTCTGTTGACATCAACATTGTTTACTGTAAGCTTAATAGTTGCAAGGTCTACAATTTCAATTACAGTTGAGTGTCCTTTATCTGAA
TTGCTAAGAAGGAAAGCATTTTGATTTTGGAGCTTTTCAGAGGTGAGATTTTTTTTTTCTTTTTTTCTGAGACTTTACCAGTCGAACAA
GTGGAAAACAAAAACAAAAACAAAACAAAAGTCCCAATTCTAAAATACCTTGAAATCTAAAACCTTAGGAACAGCATGTTAGAACTCAA
AACTTTTTGGATCATTTTCAACTTGGGATGGATATTCAAGCATCTTATTGTTACTATAATTTACATTTACAGTAAAGCTGGTCTTTATT
GTCAAAGCAGCTCATCTCAGATTTTTATCACAGCAGGGGAAAAGAGAAAATAGAGAGAAGAAAACCTGTGTGGTCTTTCCTGGGAGGTT
```

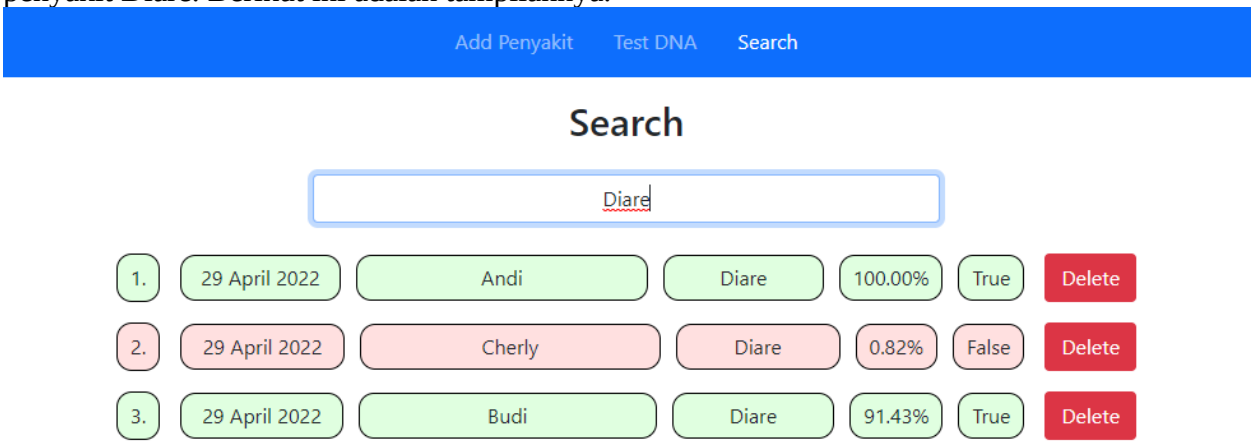
Berikut ini adalah hasil yang Cherly dapatkan



Menurut aplikasi kami, Andi negatif terkena penyakit Diare.

4.3.5 Melakukan Pencarian Riwayat Pengecekan Penyakit

Misalnya seseorang ingin melihat riwayat orang-orang yang melakukan pengecekan terhadap penyakit Diare. Berikut ini adalah tampilannya.



4.4 Analisis Hasil Pengujian

Semua pengujian berhasil dijalankan dengan baik. Berikut ini adalah analisis tiap kasus pengujian penyakit.

4.4.1 Analisis Kasus True (Andi)

Hasil yang didapat aplikasi kami sudah benar karena terdapat substring DNA penyakit diare pada DNA Andi sebagai berikut.

Andi.txt

```
ACCGCCGAGACCGCGTCCGCCCCGCGAGCACAGAGCCTCGCCTTTGCCGATCCGCCGCCCGTCCACACCCGCCGCCAGGTAAGCCCGGC
AGCCGACCGGGGCAGGCGGCTCACGGCCCGGCCGCAGGCGGCCGCGGCCCCTTCGCCCGTGCAGAGCCGCCGTCTGGGCCGCAGCGGGG
GGCGCATGGGGGGGGAACCGGACCGCCGTGGGGGGCGCGGGAGAAGCCCCTGGGCCTCCGGAGATGGGGGACACCCCACGCCAGTTCGG
AGGCGCGAGGCCGCGCTCGGGAGGCGCGCTCCGGGGGTGCCGCTCTCGGGGCGGGGGCAACCGGCGGGGTCTTTGTCTGAGCCGGGCTC
TTGCCAATGGGGATCGCAGGGTGGGCGCGGCGGAGCCCCCGCCAGGCCCGGTGGGGGCTGGGGCGCCATTGCGCGTGCGCGCTGGTCCT
TTGGGCGCTAACTGCGTGCGCGCTGGGAATTGGCGCTAATTGCGCGTGCGCGCTGGGACTCAAGGCGCTAACTGCGCGTGCGTTCTGGG
GCCCGGGGTGCCGCGGCCTGGGCTGGGGCGAAGGCGGGCTCGGCCGGAAGGGGTGGGGTCGCCGCGGCTCCCGGGCGCTTGCGCGCACT
TCCTGCCCGAGCCGCTGGCCGCCCGAGGGTGTGGCCGCTGCGTGCGCGCGCGCCGACCCGGCGCTGTTTGAACCGGGCGGAGGCGGGGC
TGGCGCCCGGTTGGGAGGGGGTTGGGGCCTGGCTTCCTGCCGCGCGCCGCGGGGACGCCTCCGACCAGTGTTTGCCTTTTATGGTAATA
ACGCGGCCGGCCCGGCTTCCTTTGTCCCCAATCTGGGCGCGCGCCGGCGCCCCCTGGCGGCCTAAGGACTCGGCGCGCCGGAAGTGGCC
AGGGCGGGGGCGACCTCGGCTCACAGCGCGCCCGGCTATTCTCGCAGCTCACCATGGATGATGATATCGCCGCGCTCGTCGTCGACAAC
GGCTCCGGCATGTGCAAGGCCGGCTTCGCGGGCGACGATGCCCCCCGGGCCGTCTTCCCCTCCATCGTGGGGCGCCCCAGGCACCAGGT
AGGGGAGCTGGCTGGGTGGGGCAGCCCCGGGAGCGGGCGGGAGGCAAGGGCGCTTTCTCTGCACAGGAGCCTCCCGGTTTCCGGGGTGG
GGGCTGCGCCCGTGCTCAGGGCTTCTTGTCCTTTCCTTCCCAGGGCGTGATGGTGGGCATGGGTCAGAAGGATTCCTATGTGGGCGACG
AGGCCCAGAGCAAGAGAGGCATCCTCACCCTGAAGTACCCCATCGAGCACGGCATCGTCACCAACTGGGACGACATGGAGAAAATCTGG
CACCACACCTTCTACAATGAGCTGCGTGTGGCTCCCGAGGAGCACCCCGTGCTGCTGACCGAGG
```

Bagian yang di-*highlight* warna kuning adalah *sequence* DNA penyakit Diare. Oleh karena itu, hasil yang dikeluarkan oleh aplikasi merupakan hasil yang benar.

4.4.2 Analisis Kasus Parsial (Budi)

Hasil yang didapat aplikasi kami sudah benar karena terdapat substring parsial DNA penyakit diare pada DNA Budi sebagai berikut.

Budi.txt

```
GTCTGTCTGTCTCTCAGAAGACTGGGTCTTGGGAAGCCCAGTACTGGCTTTCTATCCAAGAAGCTTATGTTTTCATTCCAGGTCTAG
ATTAGATGGAAAGTCCCACTGGAACTCCAGGCTTCTGAAGTCTCCTTTCTAGATTCAGTAAATGTTTATTGCTTGTCTAAGATGTGG
CAGACATTATGAGTTTAAAGGGGGATCTTAATGATTATATTATTGCTTCCAAACATATGTAGTTCTTGAAACCACTAGTTAAACATT
GGTAATCTGCTTTTCAAATATGTAAAACAGACTAGTACTCCGACTGACTTTCTTACGGTGCACATTCCCAGTGGTAGGACATGATGT
TATCTGAAGTCAGGTGGTGACACCAATAGACTCAACAGCTTCCGTTGGCTTTATTTACTTAGCCTCATAAGTCATAACTTTTCTAAA
AGGAAAAAAAAGGTGCTGGATTTGATATCTTAATCTCTGAAGGCTCTATCACTACCAAAATGTCTGGTCCTTTTGACGGGCTATCTGT
GAAACGTGTATGCTAAGAATGGCTTTCAATCTTAGCAGTTCAAGCAGCTTGCTTTAAAAGCTATATGGGAATTATCACAGGGGGCCC
TTCTTTCATGTTTACACTTCTCCCCACGCCAGTTCGGAGGCGCGAGGCCGCGCTCGGGAGGCGCGCTCCGGGGGTGCCGCTCTCGGG
GCGGGGGCAACCGGCGGGGTCTTTGTCTGAGCCGGGCTCTTGCCAATGGGGATCGCAGGGTGGGCGCGGCGGAGCCCCCGCCAGGCC
CGGTGGGGGCTGGGGCGCCATTGCGCGTGCGCGCTGGTCCTTTGGGCGCTAACTGCGTGCGCGCTGGGAATTGGCGCTAATTGCGCG
TGCGCGCTGGGACTCAAGGCGCTAACTGCGCGTGCGTTCTGGGGCCCGGGGTGCCGCGGCCTGGGCTGGGGCGAAGGCGGGCTCGGC
```

```
CGGAAGGGGTGGGGTCGCCGCGGCTCCCGGGCGCTTGCGCGCACTTCCTGCCCGAGCCGCTGGCCGCCCGAGGGTGTGGCCGCTGCG
TGCGCGCGCGCCGACCCGGCGCTGTTTGAACCGGAAGAAAATCCCCAAATGGAAAAATCATTTGGGAAAAAGAGACTACTGGTTACA
GAAGTGGCATCGAGAGTCAAGATCTGGAAAGCACACTAATGCCCTACTAAGGTCAAGTCAATAGGAAGAGAGAGAATGTTCTATACA
GTTCCAGATGAACTCCATTAATCACAGCACAATTTTGGATCTTCTACATGAAAAGACTTATTGGAGAACTTAAAGAACTGCCAGGAA
GTGGCTGCTGGTTACTATCTCCAGGAAACCTGAAGCCTGGATTGTTGGGTATCAGTGGAAATAGGGAGATGGATGACCGTTACAGGA
```

Bagian yang di-*highlight* warna kuning adalah bagian dari *sequence* DNA penyakit Diare. Oleh karena itu, hasil yang dikeluarkan oleh aplikasi merupakan hasil yang benar.

4.4.3 Analisis Kasus False (Cherly)

Hasil yang didapat aplikasi kami sudah benar karena tidak terdapat substring DNA penyakit diare pada DNA Cherly.

Cherly.txt

```
TGCCCCTGCTCTTTCTTTAAGGATAAAGATGTACAAGACCTCTTTAACTCCTTGCATTGTTACTGTAATTTACACTTAGCTCATTTCAA
ATTACACACAGCTAGAACACGAAGTAGCAGCAAAATAACAGCCTGTGTGTAAATTCCACCACCGGAAGCTTTGTATTTTCACAGACACA
CAGCAACTACAGAAACTCTGCAAAGCAAGTATCCTCTTGTAAGGACCCACAGCTCCACTCCTTGCCAGTTCCCATCCAACCTCAGCCCC
CATATGCAGAGAAACCATATACAGATAGATATGTGGATATGAGCTTGAACATCAGCACTGTGAGGCAAGAAAGCACACAAGTCACAGAG
TAAAGCCACAGCACACTGTCGATCCTGGGTAGCTTCCTGAGCCTCTGTGCCAGCCCAGCCTCCTAAGCCCTGATTCCCACACCCATTGC
AGCGTGCAACCATACAGTATGTTGCTTTACTCGCAGGGATACTTTACAGTAATAAATTTTGGAAGAACCTTTTATGTACATTTCATTTT
TCAGGATAGTTGACTGAACTAGTATGCATCAAATTCTTCCTGCATACTGCTATACTTTTAACTATGGATTGCATAAGAAGTATAATCTC
TGTTCTTAAATACCCTAAAATCCAGACTTCATTCATTACATTCCATTTCTGAAAGTGAAACTTAAATCCATTGAGGTTTGAGAATGCCT
CGGCACAGAGGATAGGGAGAGCTGTAAAAGGCCTGGCTGCATTTGCGTAATGTGTGCACAAATGTTGCCCATACCTCCGCTCTGTCTGC
GCTCTACAGAGCAGCTCAGAGTTCTCACACAACACGGCCTGTGCCACACTTTGTCTTATTTGAATAATCATGAGAGCACAGCCTTATTT
CCCTCTGTGATTGCTTGATAAAGGCAACACAGTAGGCAAGCTTCAAGTTTCTATGTTTGGATAGAACCAAATCAGTGCGAAAAATCTGT
GTGCTCTGTTGACATCAACATTGTTTACTGTAAGCTTAATAGTTGCAAGGTCTACAATTTCAATTACAGTTGAGTGTCCTTTATCTGAA
TTGCTAAGAAGGAAAGCATTTTGATTTTGGAGCTTTTCAGAGGTGAGATTTTTTTTTTCTTTTTTTCTGAGACTTTACCAGTCGAACAA
GTGGAAAACAAAAACAAAAACAAAACAAAAGTCCCAATTCTAAAATACCTTGAAATCTAAAACCTTAGGAACAGCATGTTAGAACTCAA
AACTTTTTGGATCATTTTCAACTTGGGATGGATATTCAAGCATCTTATTGTTACTATAATTTACATTTACAGTAAAGCTGGTCTTTATT
GTCAAAGCAGCTCATCTCAGATTTTTATCACAGCAGGGGAAAAGAGAAAATAGAGAGAAGAAAACCTGTGTGGTCTTTCCTGGGAGGTT
```

Oleh karena itu, hasil yang dikeluarkan oleh aplikasi merupakan hasil yang benar.

5 Kesimpulan dan Saran

5.1 Kesimpulan

Algoritma pencocokan string memiliki banyak variasi, contohnya algoritma Knuth-Morris-Pratt (KMP) dan Booyer-Moore (BM). Kedua algoritma tersebut bisa digunakan untuk melakukan pencocokan dua string. Selain itu, regular expression juga bisa digunakan untuk melakukan sanitasi input. Kami berhasil membuat aplikasi berbasis web yang dapat melakukan pencocokan DNA manusia dengan DNA penyakit menggunakan algoritma Knuth-Morris-Pratt (KMP) dan Booyer-Moore (BM). Sebelum dilakukan pencocokan, kami menggunakan regular expression untuk melakukan sanitasi input yang diberikan. Aplikasi yang kami buat dapat langsung menampilkan hasil yang diberikan serta menyimpan hasil tersebut di dalam database sebagai riwayat. Kami juga menggunakan regular expression untuk melakukan filter terhadap riwayat pengecekan penyakit.

5.2 Saran

Aplikasi berbasis web yang kami buat sudah berjalan dengan baik, namun masih bisa ditingkatkan lagi di bagian tampilan antarmuka (*user interface*) memiliki tampilan yang lebih menarik. Selain itu,

5.3 Komentar

Tugas besar ini membantu kami dalam memahami pembuatan aplikasi web yang terdiri dari library, frontend, dan backend. Namun, waktu pengerjaan tugas besar ini berbarangan dengan tugas besar lain sehingga waktu pengerjaan menjadi lebih sibuk.

6 Link Penting

Link github : https://github.com/marfgold1/Tubes3_13520103
Link youtube : https://youtu.be/Z-b9dklL0Do
Link deployment : tesdna.ddns.net