Tugas Besar 3 IF2211 Strategi Algoritma

Semester II Tahun 2020/2021

Penerapan String Matching dan Regular Expression dalam Pembangunan Deadline Reminder Assistant

Kelompok 31 (EsokKanMasihAda)

Aulia Adila (NIM 13919100)

Giovani Anggasta (NIM 13519155)

Gayuh Tri Rahutami (NIM 13519192)

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

JL. GANESHA 10, BANDUNG 40132

BAB I
Deskripsi Tugas

Dalam tugas besar ini, Anda diminta untuk membangun sebuah chatbot sederhana yang berfungsi untuk membantu mengingat berbagai deadline, tanggal penting, dan task-task tertentu kepada user yang menggunakannya. Dengan memanfaatkan algoritma String Matching dan Regular Expression, Anda dapat membangun sebuah chatbot interaktif sederhana layaknya Google Assistant yang akan menjawab segala pertanyaan Anda terkait informasi deadline tugas-tugas yang ada.

Fitur-fitur aplikasi:

Deadline Reminder Assistant dibangun dengan sistem Question and Answer dimana pengembang sudah menyediakan kumpulan formula tertentu untuk melakukan pendeteksian setiap perbedaan command atau perintah pada aplikasi Chatbot. Berikut ini adalah runtutan fitur yang dimiliki oleh Deadline Reminder Assistant tersebut:

1. Menambahkan task baru
    a. Suatu kalimat diklasifikasikan sebagai suatu task apabila mengandung semua komponen berikut ini:
        i. Tanggal
        ii. Kode Mata Kuliah / Nama Mata Kuliah
        iii. Jenis Tugas (berdasarkan daftar kata penting yang sudah disediakan)
        iv. Topik Tugas
    b. Point i sampai dengan iv diklasifikasikan menggunakan regular expression sehingga masukkan kalimat benar-benar layaknya kalimat sehari-hari
    c. Jika pesanan berhasil dikenali oleh assistant, maka assistant akan mengirim pesan balasan yang berisi ID (sesuai urutan task diinput), tanggal, kode mata kuliah, jenis tugas, dan topik tugas.
2. Melihat daftar task yang harus dikerjakan
    a. Seluruh task yang sudah tercatat oleh assistant
    b. Berdasarkan periode waktu
        i. Pada periode tertentu (DATE_1 sampai DATE_2)

ii. N minggu ke depan
iii. N hari ke depan
iv. Hari ini

c. Berdasarkan jenis task (kata penting)
i. Sesuai dengan daftar task yang didefinisikan
ii. User dapat melihat daftar task dengan jenis tertentu

3. Menampilkan deadline dari suatu task tertentu
   a. Hanya berlaku untuk task yang bersifat tugas atau memiliki tenggat waktu

4. Memperbaharui task tertentu
   a. Memperbarui tanggal dari suatu task
   b. Perintah yang dimasukkan meliputi 1 keyword untuk memperbarui suatu task dan nomor task tertentu
   c. Apabila task berhasil diperbaharui, Chatbot akan menampilkan pesan sukses memperbaharui suatu task. Sebaliknya, Chatbot akan menampilkan pesan error apabila task yang dimaksud tidak dikenali oleh Chatbot (belum masuk ke dalam Daftar Task)
5. Menandai bahwa suatu task sudah selesai dikerjakan
   a. Apabila user sudah menyelesaikan suatu task, maka task tersebut bisa ditandai bahwa task tersebut sudah selesai dan tidak perlu lagi ditampilkan pada Daftar Task selanjutnya.
   b. Apabila perintah yang dimasukkan user bisa dieksekusi, Chatbot akan menampilkan pesan sukses. Sebaliknya, Chatbot akan menampilkan pesan error apabila task yang dimaksud tidak dikenali oleh Chatbot (belum masuk ke dalam Daftar Task)
6. Menampilkan opsi help yang difasilitasi oleh assistant
   a. Berisikan command-command yang dapat digunakan oleh user
   b. Bot akan memberikan hasil berupa daftar kata-kata yang bisa digunakan untuk menambahkan dan melihat daftar task (setiap kelompok bebas membentuknya seperti apa)
7. Mendefinisikan list kata penting terkait apakah itu merupakan suatu task atau tidak
   a. Minimal terdapat 5 kata penting berbeda, contohnya adalah: [“Kuis”, “Ujian”, “Tucil”, “Tubes”, “Praktikum”]

b. Kata penting akan digunakan pada penentuan jenis tugas dari suatu task
c. Daftar kata penting tidak perlu dibuat dinamis, cukup static saja atau hardcoded.

8. Menampilkan pesan error jika assistant tidak dapat mengenali masukan user
   a. Masukan yang tidak termasuk ke dalam jenis pesan di poin 1 sampai 4 dapat dikategorikan sebagai masukan tak dikenali
   b. Error message dibebaskan sesuai kreativitas mahasiswa

BAB II
Landasan Teori

1.1. Knuth Morris Pratt Algorithm

Algoritma Knuth Morris Pratt merupakan salah satu algoritma String Matching dengan arah pergerakan dari kiri ke kanan (gerakan menyerupai algoritma brute force). Perbedaannya terdapat pada jumlah langkah shifting yang dilakukan, guna mengurangi jumlah perbandingan pola yang tak berguna (wasteful).

Langkah pertama yang dilakukan adalah mencari pola prefix (dengan panjang karakter terbesar) yang sama dengan pola suffix. Misalkan, pola yang ingin dicocokkan adalah ‘abaab’. Prefix dengan panjang terbesar, yang sama dengan suffix adalah : ab. Langkah-langkah untuk mendapatkan prefix terbesar adalah dengan menjabarkan seluruh kemungkinan prefix dari pola. Untuk ‘abaab’, terdapat beberapa kemungkinan pasangan prefix dan suffix sebagai berikut.

- Untuk ‘a’, tidak terdapat prefix ataupun suffix yang memadai.
- Untuk ‘ab’, terdapat prefix : ‘a’ dan suffix : ‘b’. Tidak ada *string* yang sama
- Untuk ‘aba’, terdapat prefix : ‘a’, ‘ab’ dan suffix : ‘ba’, ‘a’. Terdapat *string* yang sama yaitu ‘a’ dengan panjang 1.
- Untuk ‘abaa’, terdapat prefix : ‘a’, ‘ab’, ‘aba’ dan suffix : ‘a’, ‘aa’, dan ‘baa’
- Untuk ‘abaab’, terdapat prefix : ‘a’,’ab’,’aba’,’abaa’ dan suffix : ‘baab’, ‘aab’, ‘ab’, ‘b’. Terdapat *string* yang sama yaitu ‘ab’ dengan panjang 2.

  Berdasarkan uraian tersebut, dapat disimpulkan bahwa *string* dengan panjang terbesar adalah 2 yaitu *string* ‘ab’. 2 menjadi indeks terbaru yang menjadi indeks dimulainya perbandingan. Dari panjang *string* tersebut, dapat ditentukan jumlah pergeseran.

Knuth Morris Pratt (KMP) memiliki border function b(k) yang didefinisikan sebagai ukuran terbesar dari prefix yang juga merupakan suffix. Contohnya, pada pola ‘abaaba’ tadi, jika dibuat tabel yang merepresentasikan b(k), maka akan terbentuk seperti berikut.

Tabel 1.1.1 Border Function

j	0	1	2	3	4	5
P[j]	a	b	a	a	b	a
k = j - 1	-	0	1	2	3	4
b(k)	-	0	0	1	1	2

Algoritma ini cocok untuk diaplikasikan dalam ukuran file yang besar, namun tidak cocok digunakan untuk ukuran karakter yang dinamis. Kompleksitas waktu KMP terbangun atas dua perhitungan, yaitu menghitung *border function* b(k) dengan kompleksitas O(m), serta pencarian *string* dengan kompleksitas O(n). Sehingga, dapat disimpulkan kompleksitas waktu algoritma KMP adalah O(m+n).

1.2. Boyer Moore Algorithm

Algoritma Boyer Moore merupakan salah satu algoritma String Matching yang banyak diaplikasikan, karena memiliki kompleksitas waktu yang jauh lebih baik daripada algoritma Brute Force. Terdapat dua mekanisme umum dari algoritma ini. Yang pertama yaitu menggunakan teknik *the looking-glass* saat iterasi untuk pemeriksaan P. Pada teknik ini, kecocokan antara pola P dengan teks T dimulai dari indeks terbesar/terakhir, namun pemeriksaan terhadap T tetap dimulai dari indeks terkecil, dengan indeks i = (panjang pola P) - 1.

Teknik kedua adalah the character-jump, yang diterapkan saat terjadi mismatch (akan terjadi lompatan). Terdapat 3 jenis kasus yang mungkin terjadi pada kondisi ini.

1. Kasus 1 : terjadi mismatch pada T[i] dan P[j], dimana T[i] terdapat dalam P, namun dengan indeks yang lebih kecil daripada j (i<j). Yang kita periksa kecocokannya adalah teks T, maka yang akan kita geser adalah posisi indeks i yang dimiliki T.
   i baru = i saat mismatch + ((panjang pola P) - 1) - (kemunculan terakhir T[i] pada P)
2. Kasus 2 : terjadi mismatch pada T[i] dan P[j], dimana T[i] terdapat dalam P, namun dengan indeks yang lebih besar daripada j (i≥j). Dalam hal ini, lakukan pergeseran P 1 karakter ke kanan (karena kita tahu pemeriksaan terhadap T[i] sudah terlewat), hingga indeks terakhir P (j baru) sejajar dengan (indeks akhir dari T (i lama)) + 1.

i baru = i saat mismatch + panjang pola P - j saat mismatch

3. Kasus 3 : terjadi mismatch pada T[i] dan P[j], dimana T[i] tidak terdapat dalam P. Dalam hal ini, pola P bergerak ke kanan hingga indeks pertama P menyentuh (i saat mismatch) + 1, karena kita tahu T[i] tidak akan ada di P. Maka, dapat disimpulkan bahwa pergeseran indeks i pada T menjadi i baru adalah:
   i baru = i saat mismatch + panjang pola P

Untuk menyelesaikan kasus 1 dan kasus 2, kita membutuhkan informasi mengenai indeks kemunculan terakhir sebuah karakter pada T dalam pola P. Maka dari itu, dibutuhkan sebuah fungsi Last Occurrence L(x) yang memeriksa indeks kemunculan terakhir seluruh karakter pada T dalam pola P. Jika karakter pada T tidak pernah muncul dalam P, nilai L(x) akan bernilai -1. Misal terdapat variasi karakter pada T: A = {a,b,c,d} yang akan dicocokkan dengan pola P : 'abacab'. Maka dapat dibuat tabel x dan L(x) sebagai berikut.

Tabel 1.2.1 Last Occurrence Function

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

Alur pemeriksaan kecocokan *string* T terhadap pola P adalah sebagai berikut. Pertama, hitung nilai L(x) untuk seluruh variasi karakter pada T. Representasi dalam bentuk tabel sangat cocok dalam hal ini. Kedua, periksa kecocokan *string* T terhadap pola P dimulai dari indeks terakhir P (teknik *the looking-glass*). Selama mismatch belum terjadi, pemeriksaan terus dilakukan dari indeks terbesar P hingga ke indeks terkecil (indeks awal) P. Jika terjadi mismatch, periksa jenis kasus yang terjadi (kasus 1/ kasus 2/ kasus 3). Lalu, geser indeks i (sebagai awal pemeriksaan yang baru) sesuai dengan penanganan masing-masing kasus.

Algoritma ini sangat baik untuk diaplikasikan pada pencocokan string pada variasi karakter yang sangat beragam, seperti karakter pada bahasa Inggris, dengan kompleksitas waktu maksimum adalah O(mn + A). A yang dimaksud adalah karakter pada T yang terdapat dalam P.

1.3. Regular Expression

Regular Expression adalah salah satu cara yang digunakan untuk mencari teks yang bersesuaian dengan pattern/pola tertentu. Teks, atau yang dinotasikan T, adalah string yang memiliki panjang n karakter. Sedangkan pattern, atau yang dinotasikan P, adalah string dengan panjang m karakter (m <<< n). Terdapat 2 kasus utama dalam pencocokan menggunakan regex, yaitu *exact matching* dan *regex matching*.

Exact matching, sesuai dengan namanya, akan mengeluarkan string pada T yang memiliki kesamaan 100% dengan pattern P. Sedangkan *regex matching* memiliki aturan pencocokan yang lebih 'leluasa', disesuaikan dengan notasi regex yang diberikan. Terdapat beberapa notasi umum dalam regex.

Tabel 1.3.1 Notasi Regular Expression

Notasi	Keterangan
.	Semua karakter kecuali newline
\.	Period (akhir suatu kata/kalimat)
^	Awalan dari string
$	Akhiran dari string
[abc]	Karakter a, b, atau c
[a-z]	Karakter a hingga z
[^abc]	Karakter kecuali a, b, atau c
aa | bb	aa atau bb
?	Nol atau satu dari elemen yang dikenakan (?)
*	Nol atau lebih dari elemen yang dikenakan (*)
+	Satu atau lebih dari elemen yang dikenakan (+)

\d, \w, \s	Sebuah digit, kata yang terdiri atas karakter, atau spasi
\D, \W, \S	Apapun kecuali sebuah digit, kata yang terdiri atas karakter, atau spasi

Contoh regex adalah sebagai berikut.

[A-Z][a-z]* → Regex untuk kata yang diawali huruf kapital. Dalam hal ini, huruf setelah kapital boleh tidak ada atau boleh ada (dan boleh >1)

BAB III
Analisis Pemecahan Masalah

3.1. Langkah Penyelesaian Masalah

a. Menambahkan Task Baru

Pada masukan pengguna, apabila seluruh keyword yang telah ditentukan tidak ditemukan, maka program akan mencari apakah terdapat empat kata kunci, yaitu: kata penting, kata “matkul”, kata “topik”, dan tanggal. Apabila salah satu dari keempat kata tersebut tidak ditemukan, maka ditampilkan pesan “Perintah tidak dapat dikenali.” Apabila keempat kata ditemukan, maka akan dicari nama mata kuliah dengan mengambil kata-kata yang berada di antara kata “matkul” dan kata kunci selanjutnya serta nama topik dengan mengambil kata-kata yang berada di antara kata “topik” dan kata kunci selanjutnya. Kemudian id, kata penting, nama matkul, nama topik, dan tanggal akan disimpan di suatu dictionary dan menambahkannya ke daftar task. Program kemudian akan menampilkan pesan keberhasilan.

b. Melihat Daftar Task yang Harus Dikerjakan

Pada masukan pengguna, apabila terdapat kata “apa saja” / ”apa aja” / “tampil”, maka program akan mencari batas tanggal dari task yang ingin ditampilkan. Kemudian, program akan memfilter task berdasarkan tanggalnya. Kemudian, akan dicari apakah terdapat kata penting di dalam masukan pengguna. Jika ada, daftar task akan difilter menjadi task-task yang memiliki kata penting yang sama saja. Kemudian akan dicari apakah terdapat task yang nama mata kuliahnya terdapat di masukan pengguna. Jika ada, task-task tersebut akan diambil dan task-task yang lain akan dibuang. Jika tidak, semua task akan tetap disimpan. Kemudian akan dicari apakah terdapat task yang nama topiknya terdapat di masukan pengguna. Jika ada, task-task tersebut akan diambil dan task-task yang lain akan dibuang. Jika tidak, semua task akan tetap disimpan. Task-task yang tersisa akan ditampilkan.

c. Menampilkan Deadline dari Suatu Task Tertentu

Pada masukan pengguna, apabila terdapat kata “kapan,” maka program akan menjalankan fungsi checkDeadline. Pada fungsi checkDeadline, akan dicari apakah terdapat kata penting di dalam masukan pengguna. Jika ada, daftar task akan difilter menjadi task-task yang memiliki

kata penting yang sama saja. Kemudian akan dicari apakah terdapat task yang nama mata kuliahnya terdapat di masukan pengguna. Jika ada, task-task tersebut akan diambil dan task-task yang lain akan dibuang. Jika tidak, semua task akan tetap disimpan. Kemudian akan dicari apakah terdapat task yang nama topiknya terdapat di masukan pengguna. Jika ada, task-task tersebut akan diambil dan task-task yang lain akan dibuang. Jika tidak, semua task akan tetap disimpan. Deadline dari task-task yang tersisa akan ditampilkan.

d. Memperbarui Task Tertentu

Pada masukan pengguna atau user, apabila terdapat kata “ubah” maka akan menjalankan fungsi ubahTask. Pada fungsi ubahTask akan mencari masukkan id dari command pengguna menggunakan regular expression, selain itu fungsi ubahTask juga mencari masukkan tanggal dari command pengguna dengan menggunakan fungsi searchDate dimana fungsi searchDate juga menggunakan regular expression. Jika pada list task terdapat id yang sama dengan id dari command pengguna maka task tersebut akan diubah tanggalnya sesuai dengan masukkan command pengguna. Jika tidak ditemukan id yang sesuai dengan masukkan pengguna maka akan mengembalikan pesan bahwa task tidak ditemukan

e. Menandai Bahwa Suatu Task Sudah Selesai Dikerjakan

Pada masukkan pengguna atau user, apabila terdapat kata “selesai” maka akan menjalankan fungsi hapusTask. Pada fungsi hapusTask akan mencari masukkan id dari command pengguna menggunakan regular expression. Jika pada list task terdapat id yang sama dengan id dari command pengguna maka task tersebut akan dihapus dari list task dan mengembalikan pesan bahwa task berhasil diselesaikan. Jika tidak ditemukan id yang sesuai dengan masukkan pengguna maka akan mengembalikan pesan bahwa task tidak ditemukan.

f. Menampilkan Opsi Help yang Difasilitasi Oleh Assistant

Pada masukkan pengguna atau user, apabila terdapat kata “bantuan” maka akan menjalankan fungsi showHelp. Pada fungsi showHelp terdapat array fitur yang berisi list fitur apa saja yang terdapat pada chatbot berupa string, array kata_penting yang berisi list kata penting yang ada pada chatbot berupa string, dan array command yang berisi list command apa saja yang dapat dijalankan pada chatbot berupa string. Fungsi showHelp mengembalikan array

fitur apa saja, kata penting apa saja, serta command apa saja yang terdpat pada chatbot tersebut dengan rapih.

g. Menampilkan Pesan Error Jika Assistant Tidak Dapat Mengenali Masukkan User

Masukkan pengguna akan diterima oleh chatbot jika pada command pengguna terdapat kata “apa saja”, “tampil”, “apa aja”, “kapan”, “ubah”, “selesai”, “semua task”, “berdasarkan, “bantuan”, kata penting, “matkul”, “topik”, dan tanggal dengan format DD/MM/YYYY, DD/MM/YY, atau DD [nama bulan] YYYY. Jika masukkan pengguna tidak sesuai dengan kata-kata tersebut maka akan menampilkan pesan error yaitu “Perintah tidak dapat dikenali.”

3.2. Fitur Fungsional

Fitur-fitur yang ada pada chatbot kami:

1. Menambahkan Task Baru
2. Melihat Daftar Task yang Harus Dikerjakan
3. Menampilkan Deadline dari Suatu Task Tertentu
4. Memperbarui Task Tertentu
5. Menandai Bahwa Suatu Task Sudah Selesai Dikerjakan
6. Menampilkan Opsi Help yang Difasilitasi Oleh Assistant
7. Menampilkan Pesan Error Jika Assistant Tidak Dapat Mengenali Masukkan User

3.3. Arsitektur Chatbot

Frontend: HTML/CSS/Javascript

Backend: Python

Web Framework: Flask

BAB IV
Implementasi dan Pengujian

4.1. Spesifikasi Teknis Program

Berikut ini merupakan spesifikasi teknis dari program yang kami kembangkan:

1. makeBorderFunction(string str1)
   Membuat border function untuk KMP.
2. searchKMP(string line, string[] words)
   Mencari apakah salah satu kata yang berada di array words terdapat di string line dengan menggunakan algoritma KMP.
3. searchDate(string line)
   Mencari masukkan tanggal dari command pengguna.
4. searchKataPenting(string line)
   Mencari masukkan kata penting dari command pengguna.
5. searchKeywords(string line, string[] keywords)
   Mencari apakah salah satu kata yang terdapat di array keywords terdapat di string line.
6. extractTaskFromLine(string line, integer id)
   Membuat dictionary task dari suatu masukan pengguna. Mengembalikan -1 apabila masukan pengguna tidak valid.
7. extractDateStartDateEnd(string line)
   Mencari tanggal awal dan tanggal akhir untuk fitur menampilkan daftar task.
8. tambahTask(string line, integer id, array of dictionary taskList)
   Menambahkan task baru ke dalam taskList sesuai dengan masukkan pengguna
9. daftarTask(string line, array of dictionary taskList)
   Menampilkan daftar tugas yang deadline atau waktu pengumpulannya belum lewat.
10. fileBasedOnKataPenting(string line, array of dictionary taskList)
    Memfilter line atau masukkan pengguna berdasarkan kata penting.
11. fileBasedOnMatkul(string line, array of dictionary taskList)
    Memfilter line atau masukkan pengguna berdasarkan mata kuliah.
12. fileBasedOnTopik(string line, array of dictionary taskList)
    Memfilter line atau masukkan pengguna berdasarkan topik

13. checkDeadline(string line, array of dictionary taskList)
    Menampilkan tanggal atau daftar task dari deadline tugas tertentu
14. sendMeme()
    Menampilkan gambar meme secara random.
15. checkFitur(string line, integer availID, array of dictionary taskList)
    Mengecek kata kunci yang ada pada command pengguna dan mengembalikan fitur atau fungsi yang sesuai dengan kata kunci tersebut
16. ubahTask(string line, array of dictionary taskList)
    Melakukan perubahan tanggal pada task tertentu, dan akan menampilkan pesan keberhasilan jika task ditemukan atau pesan kegagalan jika task tidak ditemukan.
17. hapusTask(string line, array of dictionary taskList)
    Menandai task yang sudah selesai dengan menghapusnya dari tasklist. Akan mengembalikan pesan keberhasilan jika task ditemukan dan akan menampilkan pesan kegagalan jika task tidak ditemukan.
18. allTask(string line, array of dictionary taskList)
    Menampilkan seluruh task yang ada tanpa terkecuali
19. kataPentingTask(string line, array of dictionary taskList)
    Menampilkan seluruh task berdasarkan kata pentingnya.
20. showHelp()
    Menampilkan daftar kata penting, fitur, dan command apa saja yang terdapat pada chatbot
21. removeWords(string line, string[] words)
    Menghapus kata-kata yang berada di array words dari line.
22. convertTaskToMessage(dictionary task)
    Mengubah task dalam bentuk dictionary menjadi string untuk ditampilkan ke pengguna
23. convertMonth(string month)
    Mengubah masukkan bulan dari pengguna menjadi sesuai dengan taskList
24. convertDate(string date)
    Mengubah masukkan tanggal dari pengguna menjadi sesuai dengan taskList
25. convertDateToDays(string date)
    Mengkonversi tanggal menjadi hari.

26. isBefore(string date1, string date2)

    Menentukan apakah date1 merupakan tanggal sebelum date2

27. isQualified(dictionary task, string katapenting, string startDate, string endDate)

    Menentukan apakah task memiliki kata penting katapenting

4.2. Tata Cara Penggunaan Program

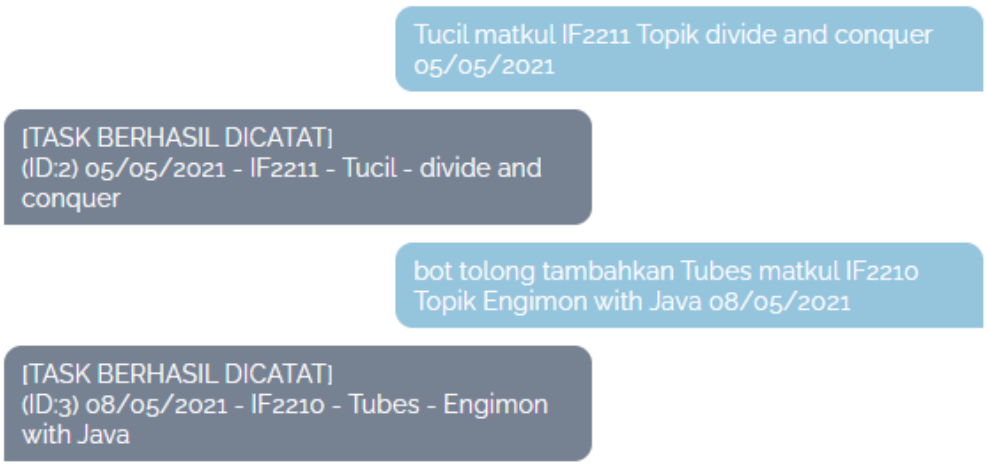
Aplikasi dapat dibuka melalui website https://esok-kan-masih-ada.herokuapp.com/. Apabila ingin menjalankan program di *local host*, berikut langkah-langkahnya:

1. Buka terminal dan buka folder src dari program yang telah diunduh.
2. Jalankan `pip install virtualenv`
3. Jalankan `python -m venv env`
4. Jalankan `source env/bin/activate` apabila menggunakan sistem operasi berbasis unix.

   Apabila menggunakan sistem operasi windows Jalankan `.\env\Scripts\activate`
5. Jalankan `pip install -r requirements.txt`
6. Jalankan `python run.py`
7. Buka link yang tertera di terminal.

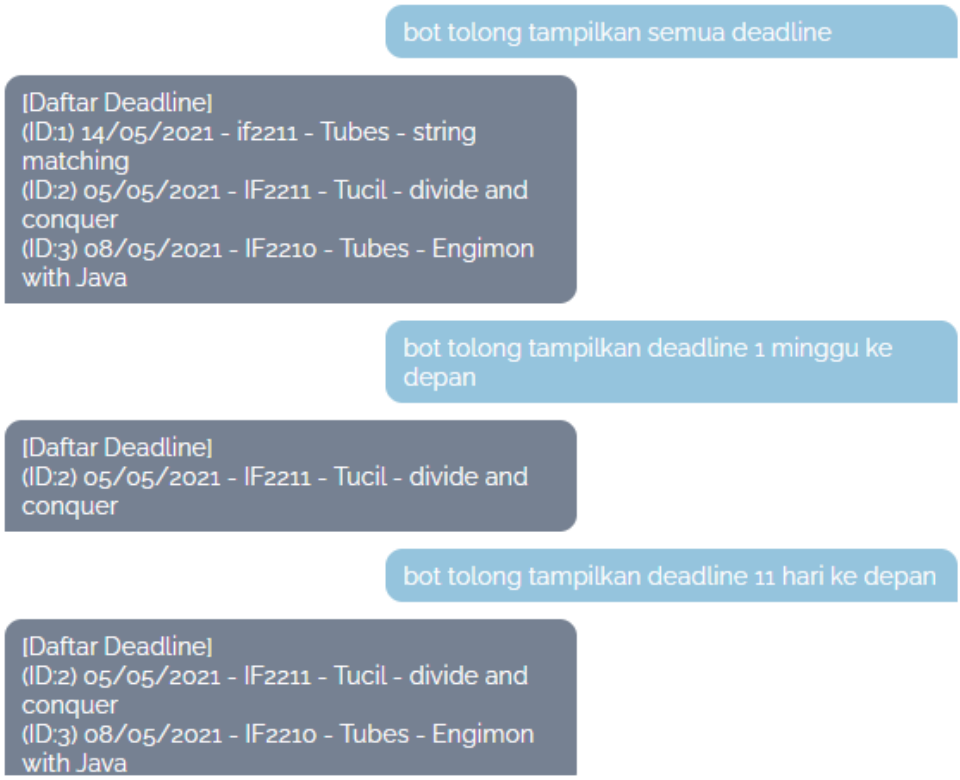
4.3. Hasil Pengujian

1. Fitur Menambahkan Task

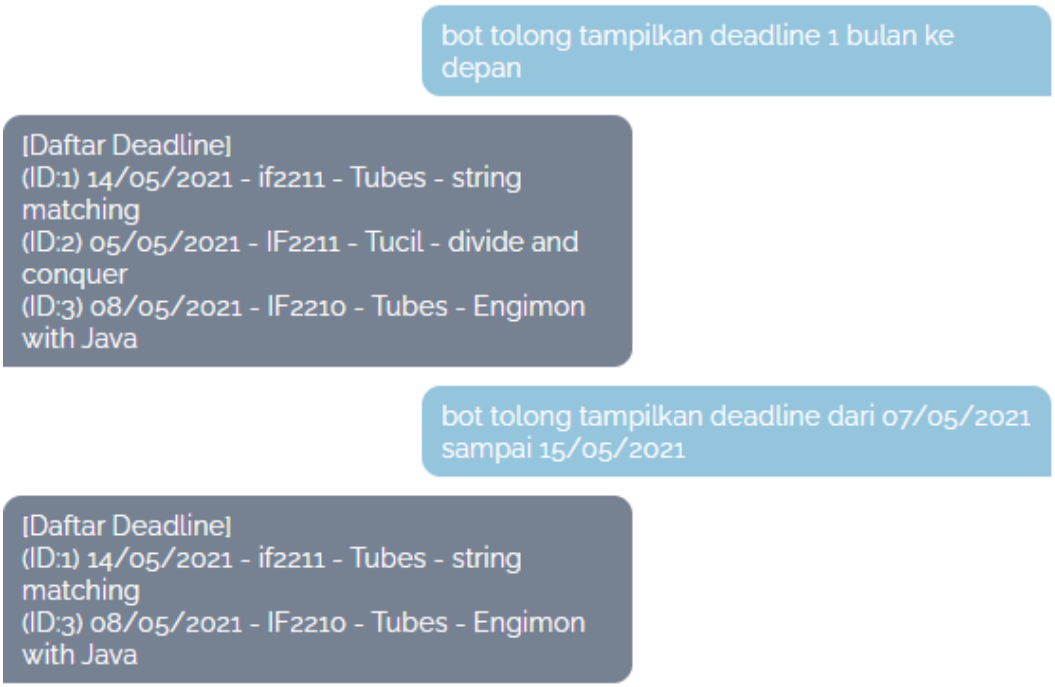


Gambar 4.3.1 Hasil Pengujian Fitur Menambahkan Task

2. Fitur Melihat Daftar Task yang Harus Dikerjakan

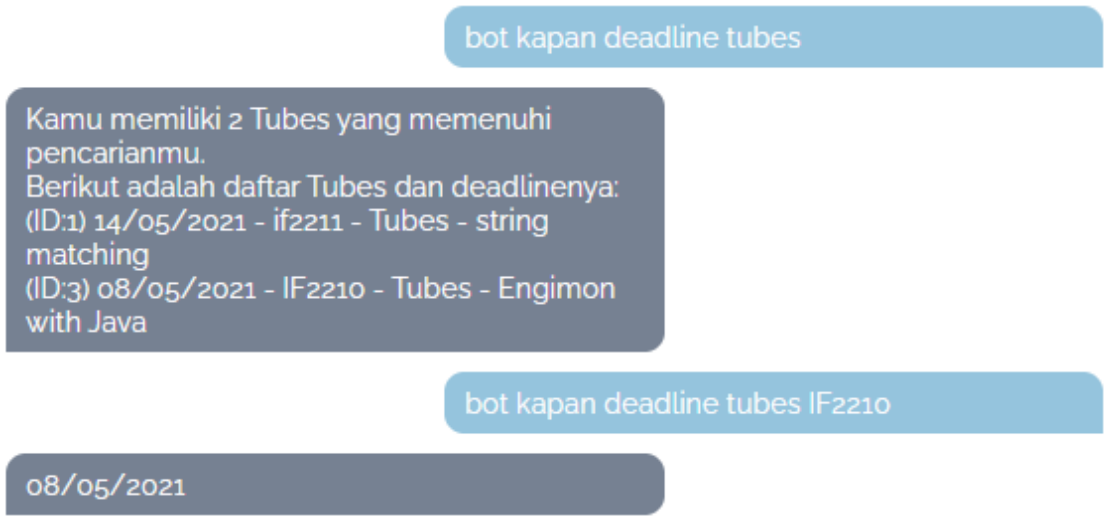


Gambar 4.3.2 Hasil Pengujian Fitur Melihat Daftar Task yang Harus Dikerjakan (1)



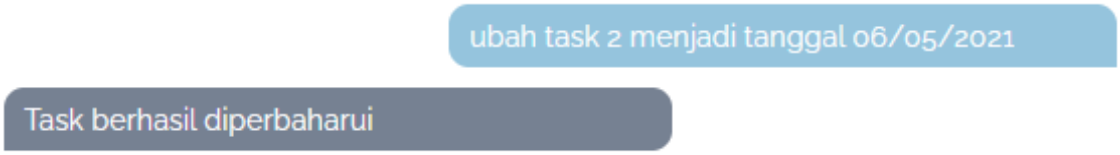
Gambar 4.3.3 Hasil Pengujian Fitur Melihat Daftar Task yang Harus Dikerjakan (2)

3. Fitur Menampilkan Deadline dari Suatu Task Tertentu



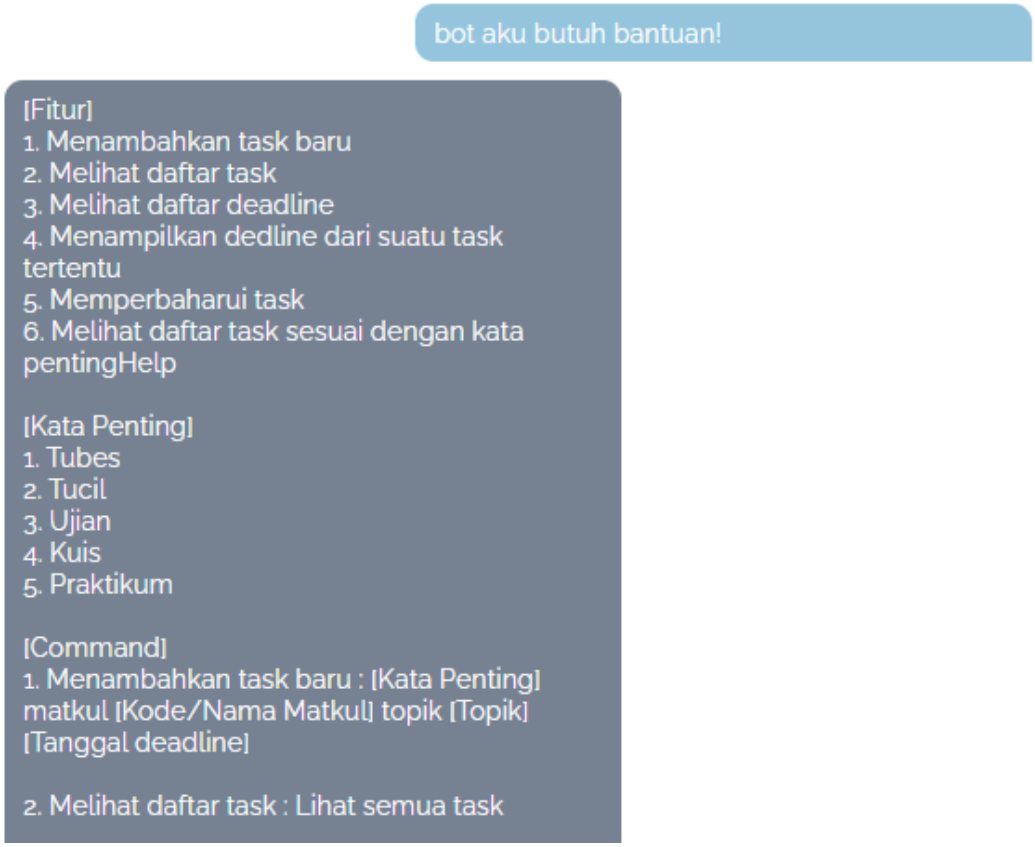
Gambar 4.3.4 Hasil Pengujian Fitur Menampilkan Deadline dari Suatu Task Tertentu

4. Fitur Memperbarui Task Tertentu



Gambar 4.3.5 Hasil Pengujian Fitur Memperbarui Task Tertentu

5. Fitur Meminta Bantuan

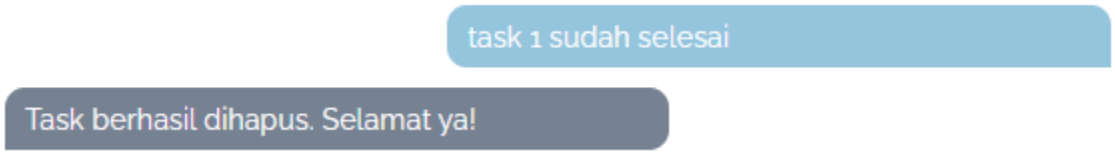


Gambar 4.3.6 Hasil Pengujian Fitur Meminta Bantuan

6. Fitur Menampilkan Pesan Error

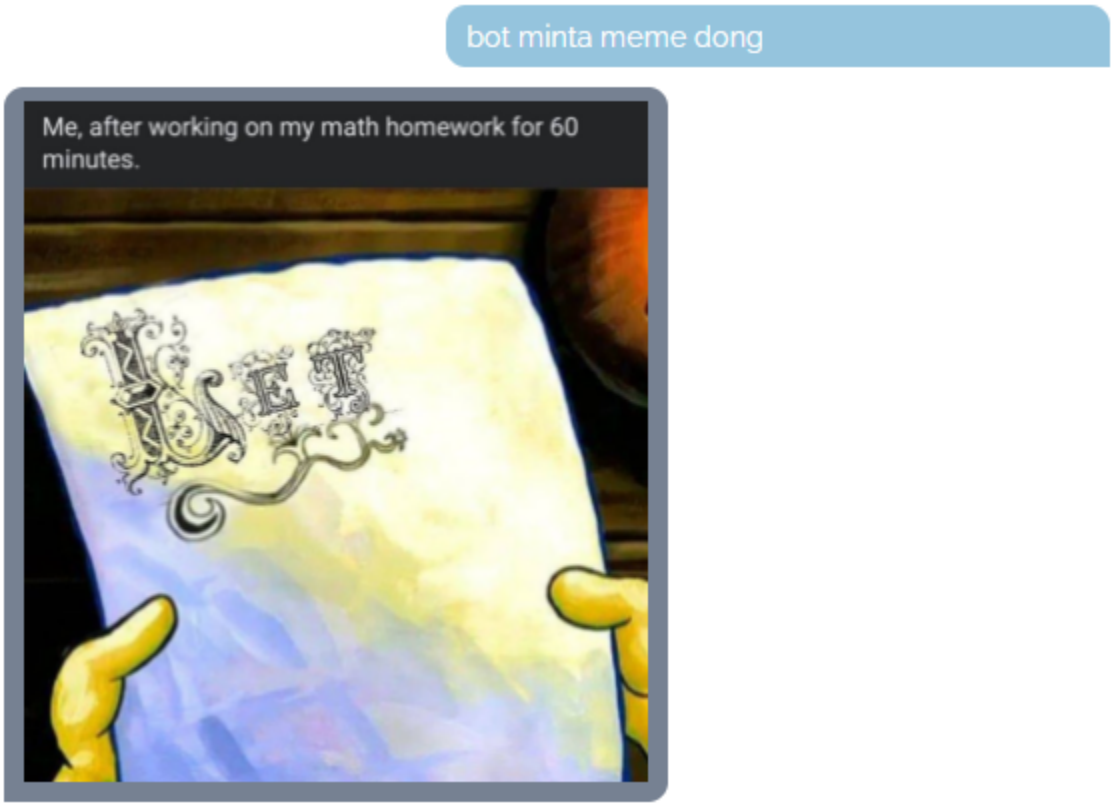
Gambar 4.3.7 Hasil Pengujian Fitur Menampilkan Pesan Error

7. Fitur Menghapus Task



Gambar 4.3.8 Hasil Pengujian Fitur Menghapus Task

8. Fitur Meme (Unsur Kreativitas)



Gambar 4.3.9 Hasil Pengujian Fitur Meme

4.4. Analisis

Seluruh fitur pada chatbot EsokKanMasihAda dapat berjalan dengan sesuai jika masukkan atau command pengguna sesuai. Apabila masukkan dari pengguna tidak sesuai, chatbot akan memberikan pesan bahwa masukkan pengguna tidak sesuai. Keberjalanan fitur-fitur dari chatbot ini memanfaatkan algoritma Knuth Morris Pratt dan juga regular expression.

BAB V
Kesimpulan, Saran, dan Refleksi

5.1. Kesimpulan

Algoritma Knuth Morris Pratt, algoritma Boyer Moore, dan juga regular expression dapat dimanfaatkan untuk membuat sebuah program chatbot karena program tersebut menerapkan *string matching* dalam pencocokan command pengguna dengan aksi yang akan dilakukan oleh chatbot tersebut. Pada tugas ini, telah dibuat sebuah aplikasi chatbot bernama "EsokKanMasihAda Bot" dimana chatbot ini memiliki beberapa fitur yang diterapkan dengan menggunakan algoritma Knuth Morris Pratt dan juga regular expression.

5.2. Saran

Aplikasi chatbot "EsokKanMasihAda Bot" ini dapat lebih dikembangkan dan ditingkatkan dengan memperhatikan beberapa hal berikut:

a. Membuat chatbot pada official account line agar dapat diakses oleh banyak pengguna dengan lebih mudah.
b. Memberikan rekomendasi kata yang mungkin apabila terjadi kesalahan pengetikan kata kunci pada command.

5.3. Refleksi

Program yang kami baut masih jauh dari kata sempurna, namun sudah dapat melakukan fitur-fitur yang diharapkan dari program ini. Tugas ini membantu kami memahami pemanfaatan algoritma Knuth Morris Pratt dan juga regular expression dalam penggunaan string matching pada pembuatan aplikasi yang dapat digunakan sehari-hari. Selain itu, tugas ini juga membantu kami mempelajari dasar dari pengembangan aplikasi berbasis web.

Daftar Pustaka

1. https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Pencocokan-string-2021.pdf, diakses pada 28 April 2021.
2. https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2018-2019/String-Matching-dengan-Regex-2019.pdf, diakses pada 28 April 2021.
3. https://www.geeksforgeeks.org/kmp-algorithm-for-pattern-searching/, diakses pada 28 April 2021.
4. https://www.geeksforgeeks.org/boyer-moore-algorithm-for-pattern-searching/, diakses pada 28 April 2021.
5. https://docs.python.org/3/library/re.html, diakses pada 28 April 2021.