

**LAPORAN TUGAS BESAR 3 IF2211 STRATEGI ALGORITMA
SEMESTER II TAHUN 2020/2021**

**PENERAPAN STRING MATCHING DAN REGULAR EXPRESSION
DALAM PEMBANGUNAN DEADLINE REMINDER ASSISTANT**



Anggota:

1. Dzaki Muhammad - 13519049 - K1
2. Ramadhana Bhanuharya Wishnumurti - 13519203 - K4
3. Rafidika Samekto - 13519207 - K4

**SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG
BANDUNG
2021**

BAB 1

DESKRIPSI TUGAS

Dalam tugas besar ini, Anda akan 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. akan dibangun dengan sistem **Question and Answer** dimana pengembang diharapkan 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 (format dibebaskan)
 - ii. Kode Mata Kuliah / Nama Mata Kuliah (dibebaskan)
 - iii. Jenis Tugas (berdasarkan daftar kata penting yang sudah disediakan)
 - iv. Topik Tugas (tidak ada batasan)
 - b. Point i sampai dengan iv diklasifikasikan menggunakan regular expression sehingga masukan kalimat benar-benar layaknya kalimat sehari-hari
 - c. Jika pesan 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. Contoh pesan balasan dari bot sebagai berikut.

[TASK BERHASIL DICATAT]

(ID: 1) 14/04/2021 - IF2211 - Tubes - String matching

d. Contoh interaksi :



Gambar 1.1. Contoh interaksi menambahkan tugas baru

2. Melihat daftar task yang harus dikerjakan

a. Seluruh task yang sudah tercatat oleh assistant

Contoh perintah yang dapat digunakan: “Apa saja deadline yang dimiliki sejauh ini?”

b. Berdasarkan periode waktu

i. Pada periode tertentu (DATE_1 until DATE_2)

Contoh perintah yang dapat digunakan: “Apa saja deadline antara DATE_1 sampai DATE_2?”

ii. N minggu ke depan

Contoh perintah yang dapat digunakan: “Deadline N minggu ke depan apa saja?”

iii. N hari ke depan

Contoh perintah yang dapat digunakan: “Deadline N hari ke depan apa saja?”

iv. Hari ini

Contoh perintah yang dapat digunakan: “Apa saja deadline hari ini?”

c. Berdasarkan jenis task (kata penting)

i. Sesuai dengan daftar task yang didefinisikan

ii. User dapat melihat daftar task dengan jenis task tertentu

- iii. Misalnya: “3 minggu ke depan ada kuis apa saja?”, maka Chatbot akan menampilkan daftar kuis selama 3 minggu kedepan

Catatan: Eksekusi perintah pengguna bisa mencakup ketiga poin sekaligus sehingga formula pengenalan command sebaiknya dibuat sebagai satu kesatuan utuh.

Contoh interaksi :

Apa saja deadline yang dimiliki sejauh ini?

[Daftar Deadline]

1. (ID: 1) 14/04/2021 - IF2211 - Tubes - String Matching
2. (ID: 2) 22/04/2021 - IF3110 - Kuis - Bab 2 sampai 3

Apa saja deadline antara 03/04/2021 sampai 15/04/2021?

[Daftar Deadline]

1. (ID: 2) 14/04/2021 - IF3110 - Kuis - Bab 2 sampai 3

Deadline 3 minggu ke depan apa saja?

[Daftar Deadline]

1. (ID: 1) 14/04/2021 - IF2211 - Tubes - String Matching
2. (ID: 2) 22/04/2021 - IF3110 - Kuis - Bab 2 sampai 3

Deadline 1 hari ke depan apa saja?

Tidak ada

3 minggu ke depan ada kuis apa saja?

[Daftar Deadline]

1. (ID: 2) 22/04/2021 - IF3110 - Kuis - Bab 2 sampai 3

Gambar 1.2. Contoh interaksi melihat daftar task

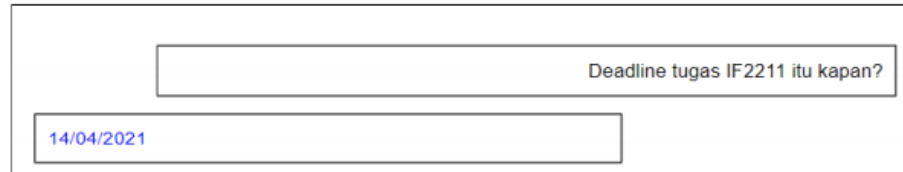
Keterangan penting:

- Perintah yang digunakan pengguna bisa tidak selalu sama, asalkan mengandung kata kunci yang ditentukan (kata kunci tiap perintah bisa ditentukan sendiri). Misal kedua contoh di bawah ini memberikan output yang sama
 - Apa saja deadline antara 03/04/2021 sampai 15/04/2021?

- Antara 03/04/2021 dan 15/04/2021 ada deadline apa saja ya?

3. Menampilkan deadline dari suatu task tertentu

- a. Hanya berlaku untuk task yang bersifat Tugas atau memiliki tenggat waktu
- b. Misalnya: “Deadline tugas IF2211 itu kapan?”
- c. Contoh interaksi :



Gambar 1.3. Contoh interaksi menampilkan deadline dari suatu task

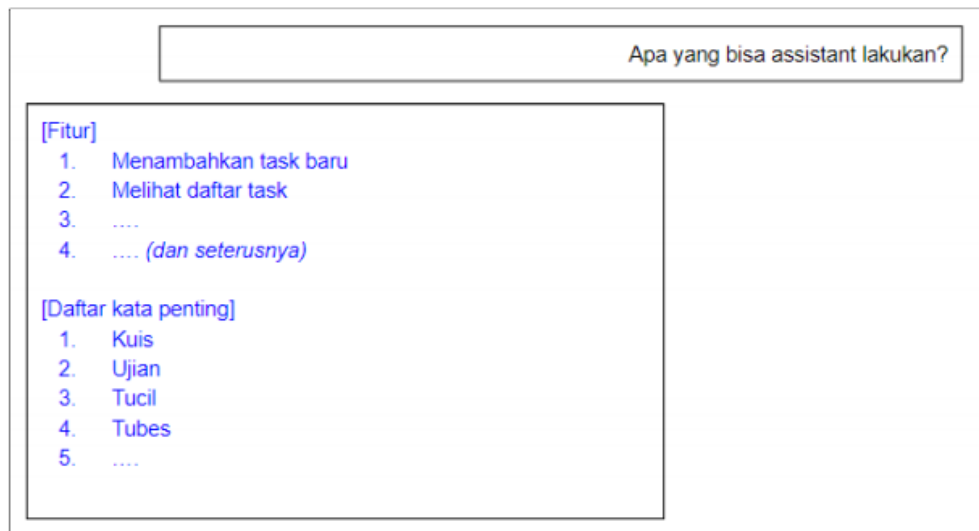
4. Memperbaharui task tertentu

- a. Memperbarui tanggal dari suatu task (dalam kehidupan nyata, tentu ada kejadian dimana deadline dari suatu task diundur)
- b. Perintah yang dimasukkan meliputi 1 keyword untuk memperbaharui suatu task dan nomor task tertentu.
- c. Misalnya:
 - “Deadline task X diundur menjadi 28/04/2021” dimana X merupakan nomor ID dari suatu task.
- d. 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. Misalnya:
 - “Saya sudah selesai mengerjakan task X” dimana X merupakan nomor ID dari suatu task.

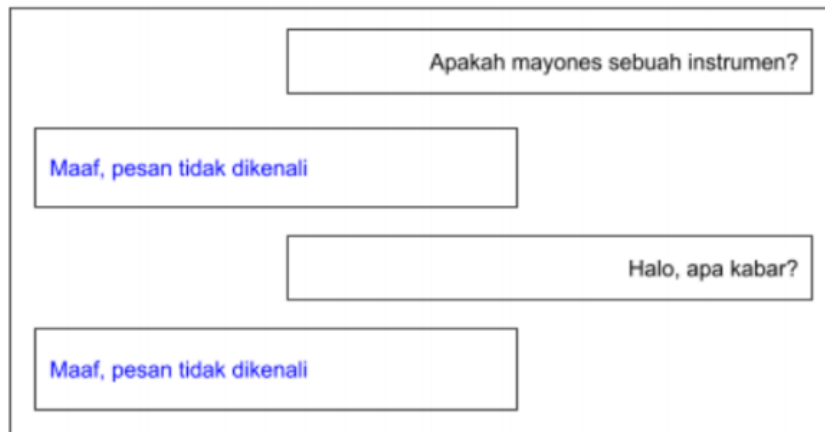
- c. Apabila perintah yang dimasukkan user bisa dieksekusi, Chatbot akan menampilkan pesan sukses. Sebaliknya, Chatbot akan menampilkan pesan error apabila task yang dimaksud tidak dikenali oleh Chatbot (belum masuk ke dalam Daftar Task)
- 6. Menampilkan opsi help yang difasilitasi oleh assistant
 - a. Berisikan command-command yang dapat digunakan oleh user
 - b. Misalnya: “Apa yang bisa assistant lakukan?”
 - c. Bot akan memberikan hasil berupa daftar kata-kata yang bisa digunakan untuk menambahkan dan melihat daftar task (setiap kelompok bebas membentuknya seperti apa)
 - d. Contoh interaksi :



Gambar 1.4. Contoh interaksi menampilkan opsi help

- 7. Mendefinisikan list kata penting terkait apakah itu merupakan suatu task atau tidak
 - a. Minimal terdapat 5 kata penting berbeda, contohnya adalah: [“Kuis”, “Ujian”, “Tupil”, “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
- c. Contoh interaksi



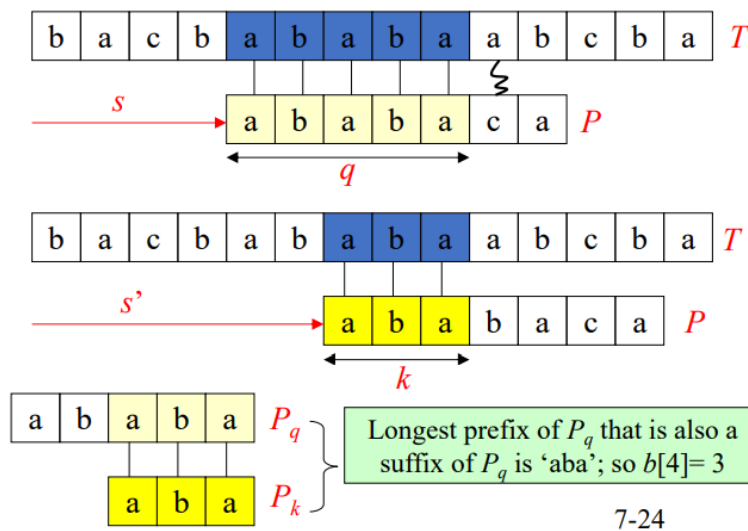
Gambar 1.5. Contoh interaksi menampilkan pesan error

BAB 2

LANDASAN TEORI

2.1 Algoritma Knuth-Morris-Pratt

Algoritma Knuth-Morris-Pratt (KMP) mencari pola dalam sebuah teks dengan urutan kiri ke kanan seperti algoritma *brute force* tetapi dengan pergeseran yang lebih cerdas dibanding algoritma *brute force*. Jika terjadi ketidakcocokan antara teks dengan pola P misalkan pada indeks j , maka pergeseran pola dilakukan dengan membandingkan prefiks terbesar dari $P[0..j-1]$ yang merupakan sufiks dari $P[1..j-1]$.



Gambar 2.1.1 Contoh pergeseran pola pada algoritma Knuth-Morris-Pratt

KMP melakukan *preprocessing* pola terlebih dahulu untuk menemukan kecocokan prefiks-prefiks dari pola dengan pola itu sendiri. Fungsi yang melakukan *preprocessing* ini disebut dengan fungsi pinggiran (*border function*). Fungsi pinggiran $b(k)$ didefinisikan sebagai ukuran dari prefiks terbesar dari $P[0..k]$ yang juga merupakan sufiks dari $P[1..k]$. Fungsi ini juga sering disebut dengan *failure function* karena menangani permasalahan saat terjadi ketidakcocokan.

➤ P: abaaba
j: 012345

($k = j-1$)

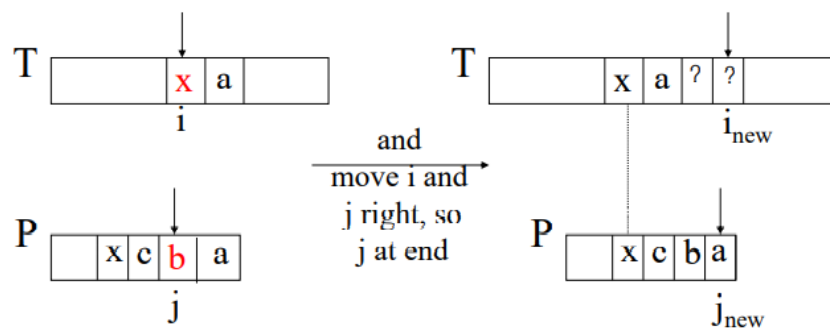
j	0	1	2	3	4	5
$P[j]$	a	b	a	a	b	a
k	-	0	1	2	3	4
$b(k)$	-	0	0	1	1	2

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

Gambar 2.1.2 Contoh visualisasi fungsi pinggiran KMP

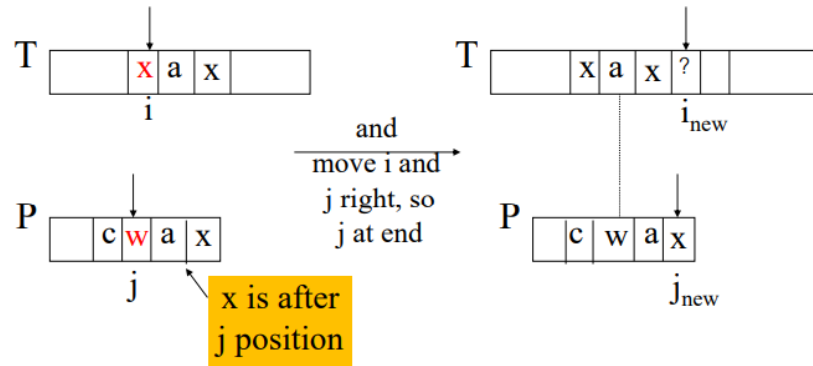
2.2 Algoritma Boyer-Moore

Algoritma *pattern matching* Boyer-Moore (BM) didasari dua teknik yakni, teknik *looking-glass* dan teknik *character-jump*. Teknik *looking-glass* adalah teknik mencari pola P dalam teks T dengan bergerak mundur melalui P mulai dari akhir P. Teknik *character-jump* adalah teknik pergeseran dengan mencari x dalam pola P kemudian mencocokkan posisi x sebelum mencocokkan kembali teks dengan pola ketika terjadi ketidakcocokan pada $T[i] \neq x$ dan karakter dalam pola $P[j]$ juga tidak sama dengan $T[i]$. Terdapat 3 kasus pergeseran yang mungkin terjadi, kasus pertama, jika P mengandung x, maka geser P ke kanan untuk menyelaraskan x terakhir dalam P dengan $T[i]$.



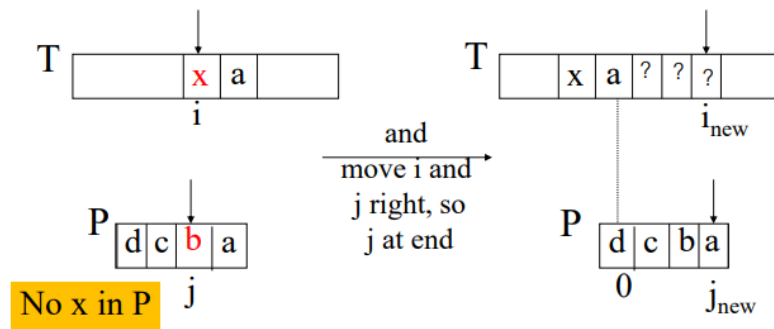
Gambar 2.2.1 Kasus pertama pergeseran pola P dalam algoritma Boyer-Moore

Kasus kedua, jika P mengandung x tetapi pergeseran P ke kanan tidak menghasilkan keselarasan x dalam P dengan $T[i]$ maka geser P ke kanan sebanyak satu karakter.

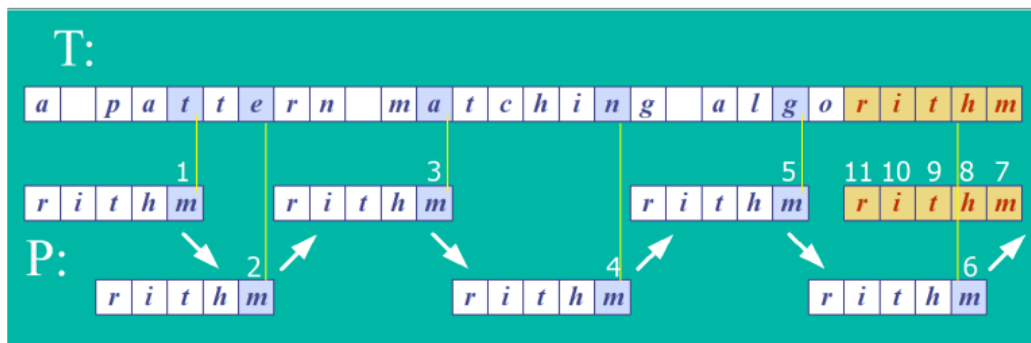


Gambar 2.2.2 Kasus kedua pergeseran pola P dalam algoritma Boyer-Moore

Kasus ketiga, yaitu ketika kasus pertama dan kasus kedua tidak dapat diaplikasikan, P tidak mengandung x, maka geser P untuk menyelaraskan $P[0]$ dengan $T[i+1]$.



Gambar 2.2.3 Kasus ketiga pergeseran pola P dalam algoritma Boyer-Moore



Gambar 2.2.4 Contoh pergeseran pola pada algoritma Boyer-Moore

Algoritma BM melakukan *preprocessing* pola terlebih dahulu untuk membangun fungsi kemunculan terakhir (*last occurrence function*) $L()$. $L(x)$ didefinisikan sebagai indeks i terbesar dimana $P[i] == x$ atau -1 jika tidak ada indeks yang memenuhi.

L() Example

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

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





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

$L()$ stores indexes into $P[]$

Gambar 2.2.5 Contoh fungsi kemunculan terakhir algoritma Boyer-Moore

2.3 Algoritma *Regular Expression*

Selain dengan menggunakan string yang didefinisikan, *string matching* atau *pattern matching* juga dapat dilakukan dengan menggunakan *regular expression*. *Regular expression* atau yang biasa disebut juga dengan regex adalah objek atau hal yang menggambarkan pola dari suatu string. Dengan regex, pola string dapat dibentuk dengan menggunakan metakarakter, yaitu karakter seperti '\', '*', '+', '^', dsb. Hal ini membuat penggunaan regex sangat kuat untuk menerapkan pencarian pola yang tidak bergantung pada string yang didefinisikan.

Regex book	Version History	Feedback	Blog
Options		Quick Reference	
.	Any character except newline.		
\.	A period (and so on for *, \{, \\\, etc.)		
^	The start of the string.		
\$	The end of the string.		
\d,\w,\s	A digit, word character [A-Za-z0-9_], or whitespace.		
\D,\W,\S	Anything except a digit, word character, or whitespace.		
[abc]	Character a, b, or c.		
[a-z]	a through z.		
[^abc]	Any character except a, b, or c.		
aa bb	Either aa or bb.		
?	Zero or one of the preceding element.		
*	Zero or more of the preceding element.		
+	One or more of the preceding element.		
{n}	Exactly n of the preceding element.		
{n,}	n or more of the preceding element.		
{m,n}	Between m and n of the preceding element.		
??,*?,+?,{n}?, etc.	Same as above, but as few as possible.		
(expr)	Capture expr for use with \1, etc.		
(?:expr)	Non-capturing group.		
(?=expr)	Followed by expr.		
(?!expr)	Not followed by expr.		
Near-complete reference			

Gambar 2.3.1 Notasi Umum Regex

/udi/g

Test String

udi Budi rudi yudi udi

/\.\.\./g

Test String

Kalimat ini panjang ... dipotong dan selesai

/s/g

Test String

kata yang dipisahkan spasi

/rekans?/g

Test String

halo rek rekan dan rekans

/[a-z]+\d+/g

Test String

024 24b saya123

Gambar 2.3.2 Contoh-contoh *string matching* dengan regex

2.4 Chatbot

Pada dasarnya, chatbot adalah program komputer yang mensimulasikan dan memproses percakapan manusia (baik tertulis maupun lisan), memungkinkan manusia berinteraksi dengan

mengirim pesan dalam perangkat digital seolah-olah mereka sedang berkomunikasi dengan manusia. Chatbot bisa sesederhana program dasar yang menjawab pertanyaan sederhana dengan respons satu baris, atau secanggih asisten digital yang belajar dan berevolusi untuk memberikan tingkat personalisasi yang semakin meningkat saat mereka mengumpulkan dan memproses informasi.

Ada dua tipe utama chatbot, chatbot berorientasi tugas dan chatbot *data-driven*. Chatbot berorientasi tugas (deklaratif) adalah program tujuan tunggal yang berfokus pada melakukan satu fungsi. Menggunakan aturan, NLP, dan ML yang sangat sedikit, mereka menghasilkan respons otomatis namun cakap untuk pertanyaan pengguna. Interaksi dengan chatbot ini sangat spesifik dan terstruktur dan paling dapat diterapkan untuk fungsi dukungan dan layanan, contohnya FAQ interaktif yang andal. Sedangkan chatbot *data-driven* (percakapan) sering disebut sebagai asisten virtual atau asisten digital, mereka jauh lebih canggih, interaktif, dan dipersonalisasi daripada chatbot yang berorientasi tugas. Mereka menerapkan kecerdasan prediktif dan analitik untuk mengaktifkan personalisasi berdasarkan profil pengguna dan perilaku pengguna sebelumnya.

Chatbot sering digunakan untuk meningkatkan pengalaman manajemen layanan teknologi, yang mempelajari proses swalayan dan otomatisasi yang ditawarkan. Di sisi bisnis, chatbot paling sering digunakan di pusat kontak pelanggan untuk mengelola komunikasi yang masuk dan mengarahkan pelanggan ke sumber daya yang sesuai. Di sisi konsumen, chatbot dapat melakukan berbagai layanan pelanggan, mulai dari memesan tiket acara hingga memesan dan check-in hotel hingga membandingkan produk dan layanan.

BAB 3

ANALISIS PEMECAHAN MASALAH

3.1 Langkah-langkah Pemecahan Masalah

3.1.1 Fitur Menambahkan Task Baru

1. Melihat apakah perintah pengguna mengandung kata kunci “PR”, “Praktikum”, “Kuis”, “UAS”, “UTS”, “Tucil”, dan “Tubes”.
2. Melakukan pengecekan setiap kata kunci yang didapatkan dari masukan user apakah ada dalam command atau tidak. Kata kunci ini berupa tanggal deadline, kode mata kuliah, dan topik task yang akan dimasukkan.
3. Melakukan pengecekan validitas kata kunci yang dimasukkan ke dalam fungsi. Semua kata kunci yang disebut dalam poin 2 akan dilakukan validasi apakah sesuai dengan format input atau tidak. Jika ada yang tidak sesuai format, fungsi akan mengembalikan pesan error.
4. Jika command user lolos seleksi poin 3, seluruh kata kunci yang ada di command user akan dimasukkan ke dalam database dan sistem akan menampilkan pesan berhasil.

3.1.2 Fitur Melihat Daftar Task

1. Melihat apakah perintah pengguna memuat kata kunci “deadline” dan “apa saja”
2. Jika iya, ekstrak string perintah untuk mendapatkan jenis tugas
3. Melihat apakah perintah pengguna memuat kata kunci waktu seperti “ke depan”, “minggu”, “hari”, “antara”, “hari ini”, “sejauh ini”
4. Membagi kasus untuk setiap kata kunci waktu:
 - a. Memuat “minggu” dan “ke depan” : menampilkan semua daftar task yang harus dikerjakan dalam n minggu ke depan
 - b. Memuat “hari” dan “ke depan” : menampilkan semua daftar task yang harus dikerjakan dalam n hari ke depan
 - c. Memuat “antara” : menampilkan semua daftar task yang harus dikerjakan antara dua tanggal
 - d. Memuat “hari ini” : menampilkan semua daftar task yang harus dikerjakan pada hari ini

- e. Memuat “sejauh ini” : menampilkan semua daftar task yang belum diselesaikan
5. Jika perintah memuat kata kunci jenis tugas, tambahkan batasan jenis tugas dalam menampilkan daftar task.

3.1.3 Fitur Menampilkan Deadline Suatu Task

1. Melihat apakah perintah pengguna memuat kata kunci “deadline” dan “kapan”
2. Jika iya, ekstrak string perintah untuk mendapatkan kode mata kuliah, jenis tugas, dan topik tugas.
3. Cari data tanggal deadline pada database yang cocok dengan kode mata kuliah, jenis tugas, dan topik tugas yang didapatkan.
4. Jika tidak ditemukan data yang diinginkan, tampilkan pesan bahwa tidak ada data. Jika data ditemukan, tampilkan semua data tanggal deadline yang diperoleh.

3.1.4 Fitur Memperbaharui Task Tertentu

1. Melakukan pengecekan apakah kata kunci yang diperlukan ada dalam command atau tidak. Jika tidak, sistem akan mengembalikan pesan error. Kata kunci tersebut adalah salah satu keyword untuk melakukan update, ID task yang akan diperbaharui, dan tanggal deadline baru task tersebut.
2. Dilakukan validasi kesamaan format kata kunci yang dimasukkan. Jika gagal, sistem akan mengirimkan pesan error.
3. Proses selanjutnya adalah mengubah tanggal deadline task yang diminta menjadi tanggal baru yang dimasukkan pengguna. Jika berhasil, sistem akan mengembalikan pesan berhasil.

3.1.5 Fitur Menandai Suatu Task Sudah Dikerjakan

1. Melakukan pengecekan apakah kata kunci yang diperlukan ada dalam command atau tidak. Jika tidak, sistem akan mengembalikan pesan error. Kata kunci tersebut adalah salah satu keyword untuk menyelesaikan task dan ID task yang akan diperbaharui.
2. Dilakukan validasi kesamaan format kata kunci yang dimasukkan. Jika gagal, sistem akan mengirimkan pesan error.
3. Pengecekan selanjutnya dilakukan dengan melihat kesamaan value dalam database, yaitu value atribut yang mencatat apakah task sudah selesai atau belum. Jika task sudah selesai, maka akan dikembalikan pesan error yang menyatakan task sudah selesai dikerjakan dan

tidak perlu ditandai sudah dikerjakan kembali. Jika belum, sistem akan melakukan update ke database dan mengirimkan pesan berhasil.

3.1.6 Fitur Menampilkan Opsi “Help” yang Difasilitasi Chatbot

1. Melihat apakah perintah pengguna memuat kata kunci “bisa” dan “lakukan”
2. Jika iya, chatbot akan menampilkan daftar fitur dan kata penting yang disediakan oleh chatbot

3.1.7 Fitur Menampilkan Pesan Error Jika Tidak Mengenali Masukan User

1. Tampilkan pesan error perintah sama sekali tidak memuat kata kunci atau kata kunci yang dimuat tidak cukup untuk menggunakan fitur-fitur yang disediakan chatbot

3.2 Fitur Fungsional Chatbot

Chatbot yang dibangun memiliki fitur fungsional untuk menambahkan task baru, melihat daftar task, menampilkan deadline suatu task, memperbaharui task tertentu, menandai suatu task sudah dikerjakan, menampilkan opsi bantuan, dan menampilkan pesan error jika tidak mengenal masukan pengguna. Cara kerja setiap fitur fungsional dari chatbot ini dapat dilihat pada bab 3.1, perintah masukan pengguna diproses oleh chatbot untuk menentukan fungsionalitas yang akan digunakan kemudian mengembalikan pesan berdasarkan fungsionalitas tersebut.

3.3 Arsitektur Chatbot

Chatbot dibangun menggunakan bahasa Python dalam framework Flask. Chatbot yang dibangun menggunakan SQL sebagai basis data kebutuhan chatbot penyimpanan data yang diberikan pengguna. Untuk kebutuhan frontend, aplikasi ini menggunakan HTML, CSS, dan JavaScript untuk tampilan web.

BAB 4

IMPLEMENTASI DAN PENGUJIAN

4.1 Spesifikasi Teknis Program

4.1.1 File KnuthMorrisPratt.py

1. Fungsi computeFail

Tipe luaran : List of integer

Parameter : String pattern.

Prekondisi : Masukan terdefinisi.

Kegunaan : Fungsi untuk membuat tabel pinggiran KMP dalam bentuk list of integer

Implementasi :

```
def computeFail(pattern):  
    fail = [0 for i in range (len(pattern))]  
  
    fail[0] = 0  
  
    m = len(pattern)  
    j = 0  
    i = 1  
  
    while i<m :  
        if pattern[j] == pattern[i] :  
            fail[i] = j+1  
            i+=1  
            j+=1  
  
        elif j>0 :  
            j = fail[j-1]  
        else:  
            fail[i] = 0  
            i+=1  
  
    return fail
```

2. Fungsi kmpMatch

Tipe luaran : integer

Parameter : String text dan string pattern.

Prekondisi : Masukan terdefinisi.

Kegunaan : Fungsi untuk mengetahui apakah terdapat substring pattern dalam string text. Mengembalikan indeks posisi pattern jika ditemukan, -1 jika tidak ditemukan.

Implementasi :

```
def kmpMatch(text, pattern):  
    n = len(text)  
    m = len(pattern)  
  
    fail = computeFail(pattern)  
  
    i = 0  
    j = 0  
  
    while i < n :  
        if pattern[j] == text[i]:  
            if j == m-1 :  
                return i-m+1  
            i+=1  
            j+=1  
        elif j > 0 :  
            j = fail[j-1]  
        else:  
            i+=1  
  
    return -1 # tidak ada kesamaan
```

4.1.2 File BoyerMoore.py

1. Fungsi buildLast

Tipe keluaran : List of integer

Parameter : String pattern.

Prekondisi : Masukan terdefinisi.

Kegunaan : Fungsi untuk membuat tabel kemunculan terakhir karakter dari pattern dalam bentuk list of integer

Implementasi :

```
def buildLast(pattern):  
    last = [-1 for i in range(256)]
```

```

for i in range(len(pattern)):
    last[ord(pattern[i])] = i

return last

```

2. Fungsi bmMatch

Tipe luaran : integer

Parameter : String text dan string pattern.

Prekondisi : Masukan terdefinisi.

Kegunaan : Fungsi untuk mengetahui apakah terdapat substring pattern dalam string text. Mengembalikan indeks posisi pattern jika ditemukan, -1 jika tidak ditemukan.

Implementasi :

```

def bmMatch(text, pattern):
    last = buildLast(pattern)
    n = len(text)
    m = len(pattern)
    i = m-1

    if (i > n-1):
        return -1 # tidak match kalau pattern lebih panjang dari
text

    j = m-1
    while (i<n-1):
        if pattern[j]==text[i]:
            if j==0 :
                return i
            else:
                i-=1
                j-=1
        else:
            lo = last[ord(text[i])]
            i = i+m - min(j,1+lo)
            j = m-1

    return -1 # Tidak match

```

4.1.3 File handleNewTask.py

1. Fungsi isValidDate

Tipe keluaran : boolean

Parameter : String date.

Prekondisi : Masukan terdefinisi.

Kegunaan : Fungsi untuk mengetahui apakah string date merupakan tanggal yang Valid.

Implementasi :

```
def isValidDate(date): # bertipe string
    listTampungan = []
    if (date[2] == '-'):
        listTampungan = date.split('-')
    elif (date[2] == '/'):
        listTampungan = date.split('/')
    day = int(listTampungan[0])
    month = int(listTampungan[1])
    year = int(listTampungan[2])
    if (day >= 1 and day <= 28):
        return True
    elif (day == 29 or day == 30):
        if ((month >= 3 and month <= 12) or month == 1):
            return True
        else: # If month == 2 -> Februari
            if (day == 29): # Tanggal 29 Februari
                if (year % 4 == 0): # Tahun kabisat
                    return True
                else:
                    return False
            else: # Tanggal 30 Februari
                return False
    elif (day == 31):
        if (month == 2 or month == 4 or month == 6 or month == 9
or month == 11):
            return False
        else:
            return True
    else:
        return False
```

2. Fungsi getDateWithRegex

Tipe luaran : String

Parameter : String command.

Prekondisi : Masukan terdefinisi.

Kegunaan : Mengembalikan string dengan format tanggal dari string command menggunakan regex matching. Mengembalikan None jika tidak ditemukan

Implementasi :

```
def getDateWithRegex(command):
    matchObject = None
    # Format tanggal valid -> DD-MM-YYYY atau DD/MM/YYYY
    allowedDateFormat =
['\d\d-\d\d-\d\d\d\d', '\d\d\/\d\d\/\d\d\d\d']
    for format in allowedDateFormat: # Pasti date yang ada sesuai
sama salah satu format atau ga sama sekali
        if (re.search(format, command)):
            matchObject = re.search(format, command)
    if (matchObject):
        dateFormat = matchObject.group()
        if (isDateValid(dateFormat)):
            return dateFormat # bertipe string
        else: # Tanggal yang dimasukkan ga valid
            return None
    else: # Ada kesalahan penulisan tanggal (ga sesuai format)
        return None
```

3. Fungsi reverseDate

Tipe luaran : string

Parameter : String date.

Prekondisi : Masukan terdefinisi.

Kegunaan : Mengubah string tanggal dengan format DD-MM-YYYY atau DD/MM/YYYY menjadi YYYY-MM-DD

Implementasi :

```
def reverseDate(date): # bertipe string. Masukan berupa input
string dari pengguna dan digunakan untuk SQL
    listTampungan = []
    newDate = ''
    if (date[2] == '-'):

```

```

        listTampungan = date.split('-')
    elif (date[2] == '/'):
        listTampungan = date.split('/')
        newDate = listTampungan[2] + "-" + listTampungan[1] + "-" +
listTampungan[0]
    return newDate

```

4. Fungsi getKodeMatkul

Tipe keluaran : string

Parameter : String command.

Prekondisi : Masukan terdefinisi.

Kegunaan : Mengembalikan string dengan format kode mata kuliah. Mengembalikan None jika tidak ditemukan

Implementasi :

```

def getKodeMatkul(command):
    matchObject = None
    allowedKodeFormat = '[A-Z][A-Z]\d\d\d\d'
    if (re.search(allowedKodeFormat, command)):
        matchObject = re.search(allowedKodeFormat, command)
    if (matchObject):
        kodeFormat = matchObject.group()
        return kodeFormat
    else: # Ada kesalahan penulisan format kode matkul
        return None

```

5. Fungsi getTaskTopic

Tipe keluaran : string

Parameter : String command.

Prekondisi : Masukan terdefinisi.

Kegunaan : Mengembalikan string dengan format topik task. Mengembalikan None jika tidak ditemukan

Implementasi :

```

def getTaskTopic(command): # Topik task dikutip sama petik dua
atau petik satu
    matchObject = None
    # allowedTopikFormat = ["\"(\w|\s)*\"", "\"'(\w|\s)*'"]

```

```

# for format in allowedTopikFormat:
    if (re.search("\"(\w|\s)*\"|'(\w|\s)*'",command)):
        matchObject =
re.search("\"(\w|\s)*\"|'(\w|\s)*'",command)
        if (matchObject):
            topikFormat = matchObject.group()
            if (len(topikFormat) <= 255):
                return topikFormat
            else: # Panjang string melebihi 255 karakter
                return None
        else: # Ada kesalahan penulisan format topik task
            return None

```

6. Fungsi getStringFromResult

Tipe keluaran : string
 Parameter :
 Prekondisi : Masukan terdefinisi.
 Kegunaan :
 Implementasi :

```

def getStringFromResult(result):
    temp = str(result)
    newString = ""
    for i in temp:
        if (ord(i) >= 48 and ord(i) <= 57):
            newString = newString + i
    return newString

```

7. Fungsi handleNewTask

Tipe keluaran : string
 Parameter : String command dan string jenisTask.
 Prekondisi : Masukan terdefinisi.
 Kegunaan : Menambahkan task baru jika command yang diberikan sesuai dengan kriteria yang dibutuhkan lalu mengembalikan string pesan keberhasilan. Jika tidak memenuhi, akan mengembalikan string pesan kegagalan
 Implementasi :

```

def handleNewTask(command, jenisTask):
    # Define error message

```

```

    isTanggalError = False
    isKodeError = False
    isTopikError = False
    errorMessageTanggal = 'Terdapat kesalahan penulisan format
tanggal atau tanggal yang dimasukkan tidak valid!\n'
    errorMessageKode = 'Terdapat kesalahan penulisan format kode
mata kuliah!\n'
    errorMessageTopik = 'Terdapat kesalahan penulisan format topik
task atau panjang karakter topik melebihi 255 karakter!\n'
    errorMessage = ''

    # Check Tanggal error
    tanggal_deadline = getDateWithRegex(command)
    if (tanggal_deadline == None):
        isTanggalError = True
    else:
        tanggal_deadline = reverseDate(getDateWithRegex(command))
# Disiapin buat dimasukin ke database

    # Check Kode error
    kode_matkul = getKodeMatkul(command)
    if (kode_matkul == None):
        isKodeError = True

    # Check Topik Error
    topik_task = getTaskTopic(command)
    if (topik_task == None):
        isTopikError = True

    # Generate output
    if (isTanggalError or isKodeError or isTopikError): # Return
error message
        if (isTanggalError):
            errorMessage = errorMessage + errorMessageTanggal
        if (isKodeError):
            errorMessage = errorMessage + errorMessageKode
        if (isTopikError):
            errorMessage = errorMessage + errorMessageTopik
    return errorMessage

```



```

else: # Put into database and return success message
    # Establish connection to DB
    mydb = mysql.connector.connect(
        # host="localhost",
        # user="root",
        # password="",
        # database="task"
        host="localhost",
        user="hariya",
        password="31213121",
        database="task"
    )
    mycursor = mydb.cursor()

    # Generate insert statement
    insertQuery = "INSERT INTO taskList
(tanggal_deadline,kode_matkul,jenis_task,topik_task,isDone) VALUES
(\'"+tanggal_deadline+"\',\'"+kode_matkul+"\',\'"+jenisTask+"\',\'
"+topik_task+"\',\'Belum\');"
    mycursor.execute(insertQuery)
    mydb.commit()

    # Selecting and returning the previously added task
    selectQuery = "SELECT id_task FROM taskList WHERE
tanggal_deadline=\'"+tanggal_deadline+"\' and
kode_matkul=\'"+kode_matkul+"\' and jenis_task=\'"+jenisTask+"\'
and topik_task=\'"+topik_task+"\';"
    mycursor.execute(selectQuery)
    result = mycursor.fetchall()
    newID = getStringFromResult(result[0])
    successMessage = "Task berhasil ditambahkan!\n"
    newTask = "(ID: "+newID+") "+tanggal_deadline+" -
"+kode_matkul+" - "+jenisTask+" - "+topik_task+"\n"
    return successMessage + newTask

```

4.1.4 File handleUpdateTask.py

1. Fungsi getID

Tipe luaran : string

Parameter : String command.
Prekondisi : Masukan terdefinisi.
Kegunaan : Mengembalikan string dengan format ID. Mengembalikan None jika tidak ditemukan

Implementasi :

```
def getID(command):  
    matchObject = None  
    allowedIDFormat = '\s[0-9]*\s'  
    if (re.search(allowedIDFormat,command)):  
        matchObject = re.search(allowedIDFormat,command)  
    if (matchObject):  
        IDFormat = matchObject.group()  
        newIDFormat = getStringFromResult(IDFormat)  
        return newIDFormat  
    else: # Input tidak sesuai format  
        return None
```

2. Fungsi handleUpdateTask

Tipe luaran : string
Parameter : String command.
Prekondisi : Masukan terdefinisi.
Kegunaan : Memperbaharui task tertentu sesuai command jika command yang diberikan sesuai dengan kriteria yang dibutuhkan lalu mengembalikan string pesan keberhasilan. Jika tidak memenuhi, akan mengembalikan string pesan kegagalan

Implementasi :

```
def handleUpdateTask(command):  
    # Define error message  
    isIDError = False  
    isTanggalError = False  
    errorMessageID = 'Terdapat kesalahan penulisan format ID!\n'  
    errorMessageTanggal = 'Terdapat kesalahan penulisan format  
tanggal atau tanggal yang dimasukkan tidak valid!\n'  
    errorMessage = ''  
  
    # Check ID error  
    ID = getID(command)  
    if (ID == None):
```

```

        isIDError = True

    # Check Tanggal error
    tanggal_baru = getDateWithRegex(command)
    if (tanggal_baru == None):
        isTanggalError = True
    else:
        tanggal_baru = reverseDate(getDateWithRegex(command))

    # Generate Output
    if (isIDError or isTanggalError):
        if (isIDError):
            errorMessage = errorMessage + errorMessageID
        if (isTanggalError):
            errorMessage = errorMessage + errorMessageTanggal
        return errorMessage
    else: # Cari ID di database
        # Establish connection to DB
        mydb = mysql.connector.connect(
            # host="localhost",
            # user="root",
            # password="placeholder",
            # database="task"
            host="localhost",
            user="hariya",
            password="31213121",
            database="task"
        )
        mycursor = mydb.cursor()

        # Generate select statement
        searchQuery = 'SELECT id_task FROM taskList WHERE
id_task='+ID+';"
        mycursor.execute(searchQuery)
        result = mycursor.fetchall()
        if (len(result) == 0): # List kosong, artinya tidak ada
        hasil yang diinginkan
            return "ID Task yang dimasukkan tidak terdapat dalam
database!\n"

```

```

        else: # List tidak kosong, ada hasilnya
            # Generate update statement
            updateQuery = "UPDATE taskList SET
tanggal_deadline=\'"+tanggal_baru+"\' WHERE id_task="+ID+";"
            mycursor.execute(updateQuery)
            mydb.commit()

            # Selecting previously updated task and generate
            success message
            # selectQuery = 'SELECT tanggal_deadline FROM tasklist
            WHERE id_task='+ID+';'
            # mycursor.execute(selectQuery)
            # result = mycursor.fetchall()
            newTanggal = toDate(tanggal_baru)
            successMessage = "Task berhasil di-update!\n"
            updatedTask = "Deadline task dengan ID = "+ID+"
            berhasil di-update menjadi tanggal "+newTanggal+"!\n"
            return successMessage + updatedTask

```

4.1.5 File handleMarkDoneTask.py

1. Fungsi getStringFromResult2

Tipe luaran : string
 Parameter : String result.
 Prekondisi : Masukan terdefinisi.
 Kegunaan :
 Implementasi :

```

def getStringFromResult2(result): # version: input = string
    temp = str(result)
    newString = ""
    for i in temp:
        if ((ord(i) >= 65 and ord(i) <= 90) or (ord(i) >= 97 and
ord(i) <= 122)):
            newString = newString + i
    return newString

```

2. Fungsi handleMarkDoneTask

Tipe luaran : string
 Parameter : String command.

Prekondisi : Masukan terdefinisi.

Kegunaan : Menandai task tertentu selesai sesuai command jika command yang diberikan sesuai dengan kriteria yang dibutuhkan lalu mengembalikan string pesan keberhasilan. Jika tidak memenuhi, akan mengembalikan string pesan kegagalan

Implementasi :

```
def handleMarkDoneTask(command):
    # Setup error message
    isIDError = False
    errorMessageID = 'Terdapat kesalahan penulisan format ID!\n'
    errorMessage = ''

    # Check ID Error
    ID = getID(command)
    if (ID == None):
        isIDError = True

    # Generate output
    if (isIDError):
        errorMessage = errorMessage + errorMessageID
        return errorMessage
    else:
        # Establish connection to DB
        mydb = mysql.connector.connect(
            # host="localhost",
            # user="root",
            # password="placeholder",
            # database="task"
            host="localhost",
            user="hariya",
            password="31213121",
            database="task"
        )
        mycursor = mydb.cursor()

        # Check if task already marked done
        # Creating select statement
        selectQuery = "SELECT isDone FROM taskList WHERE
id_task="+ID+";"
```

```

mycursor.execute(selectQuery)
result = mycursor.fetchall()
newResult = getStringFromResult2(result[0])
if (newResult == 'Sudah'):
    return "Task sudah selesai dikerjakan!\n"
else:
    # Creating update statement
    updateQuery = "UPDATE taskList SET isDone='\Sudah\'
WHERE id_task="+ID+";"
    mycursor.execute(updateQuery)
    mydb.commit()

    successMessage = "Task sudah ditandai selesai
dikerjakan!\n"
    doneTask = "Task dengan ID = "+ID+" sudah selesai
dikerjakan!\n"
    return successMessage + doneTask

```

4.1.6 File showTask.py

1. Fungsi getAngka

Tipe keluaran : integer

Parameter : String command

Prekondisi : Masukan terdefinisi.

Kegunaan : Mengembalikan bilangan n dalam format perintah “n minggu”
atau “n hari”.

Implementasi :

```

def getAngka(command):
    x = re.search("\d+ minggu|\d+ hari", command)
    a = x.group()
    y = re.search("\d+", a)
    angka = y.group()
    return int(angka)

```

2. Fungsi getDates

Tipe keluaran : Tuple of string

Parameter : String command.

Prekondisi : Masukan terdefinisi.

Kegunaan : Mendapatkan string date1, date2 dalam format perintah “antara date1 dan

date2”

Implementasi :

```
def getDates(command):  
    x = re.findall("\d\d\/\d\d\/\d\d\d\d", command)  
    date1 = reverseDate(x[0])  
    date2 = reverseDate(x[1])  
    return date1, date2
```

3. Fungsi getHariIni

Tipe luaran : String

Parameter : -

Prekondisi : -

Kegunaan : Mengembalikan string tanggal hari ini dalam format YYYY-MM-DD

Implementasi :

```
def getHariIni():  
    fdate = date.today().strftime('%Y-%m-%d')  
    return fdate
```

4. Fungsi getNHarikeDepan

Tipe luaran : String

Parameter : integer N

Prekondisi : Masukan terdefinisi.

Kegunaan : Mengembalikan string tanggal N hari ke depan dalam format
YYYY-MM-DD

Implementasi :

```
def getNHarikeDepan(N):  
    fdate = date.today() + timedelta(days=N)  
    fdate = fdate.strftime('%Y-%m-%d')  
    return fdate
```

5. Fungsi getJenisTugas

Tipe luaran : String

Parameter : String command.

Prekondisi : Masukan terdefinisi.

Kegunaan : Mengembalikan string jenis tugas jika di dalam string command terdapat kata kunci jenis tugas yang telah ditentukan

Implementasi :

```
def getJenisTugas(command):
    jenisTugas = ["PR", "Praktikum", "Tubes", "Tucil", "Kuis",
"UTS", "UAS"]
    for jenis in jenisTugas:
        if(kmpMatch(command.lower(), jenis.lower())!=-1):
            return jenis
    return None
```

6. Fungsi toDate

Tipe luaran : string

Parameter : String tanggal.

Prekondisi : Masukan terdefinisi.

Kegunaan : Mengubah format string tanggal dari format YYYY-MM-DD menjadi format DD/MM/YYYY lalu mengembalikan string tanggal dengan format baru.

Implementasi :

```
def toDate(tanggal):
    list = tanggal.split("-")
    newDate = list[2]+"-"+list[1]+"-"+list[0]
    return newDate
```

7. Fungsi showTask

Tipe luaran : String

Parameter : String command.

Prekondisi : Masukan terdefinisi.

Kegunaan : Mengembalikan string berisi daftar task yang diperoleh berdasarkan command yang diberikan pengguna

Implementasi :

```
def showTask(command):
    # Connect database
    mydb = mysql.connector.connect(
        host="localhost",
        user="root",
        password="",
        database="task"
    )
    mycursor = mydb.cursor()
```



```

#getJenisTugas
jenis_tugas = getJenisTugas(command)
selectQuery = ''
if(kmpMatch(command, "ke depan")!=-1):
    if(kmpMatch(command, "minggu")!=-1):
        #getangka
        angka = getAngka(command)
        #gethariini
        today = getHariIni()
        updatedDay = getNHarikeDepan(angka*7)
        #select
        if(jenis_tugas==None):
            selectQuery = "SELECT * FROM taskList WHERE
tanggal_deadline<=\""+updatedDay+"\" and
tanggal_deadline>=\""+today+"\";"
        else:
            selectQuery = "SELECT * FROM taskList WHERE
jenis_task = '\""+jenis_tugas+"\" and
tanggal_deadline<=\""+updatedDay+"\" and
tanggal_deadline>=\""+today+"\";"

    if(kmpMatch(command, "hari")!=-1):
        #getangka
        angka = getAngka(command)
        #gethariini
        today = getHariIni()
        updatedDay = getNHarikeDepan(angka)
        #select task from task where date >= today and date <=
lastday
        if(jenis_tugas==None):
            selectQuery = "SELECT * FROM taskList WHERE
tanggal_deadline<=\""+updatedDay+"\" and
tanggal_deadline>=\""+today+"\";"
        else:
            selectQuery = "SELECT * FROM taskList WHERE
jenis_task = '\""+jenis_tugas+"\" and
tanggal_deadline<=\""+updatedDay+"\" and
tanggal_deadline>=\""+today+"\";"

```

```

else:
    if (kmpMatch(command, "antara") != -1):
        #get date1, date2
        date1, date2 = getDates(command)
        #select task from task where date >=date1 and date
        <=date2

        if(jenis_tugas==None):
            selectQuery = "SELECT * FROM taskList WHERE
tanggal_deadline<=\""+date2+"\" and
tanggal_deadline>=\""+date1+"\"";
        else:
            selectQuery = "SELECT * FROM taskList WHERE
jenis_task = '\"'+jenis_tugas+'\" and
tanggal_deadline<=\""+date2+"\" and
tanggal_deadline>=\""+date1+"\"";

    else:
        if (kmpMatch(command, "hari ini") != -1):
            #gethariini
            today = getHariIni()
            #select task from task where date = today
            if(jenis_tugas==None):
                selectQuery = "SELECT * FROM taskList WHERE
tanggal_deadline= '\""+today+"\"";
            else:
                selectQuery = "SELECT * FROM taskList WHERE
jenis_task = '\"'+jenis_tugas+'\" and
tanggal_deadline= '\""+today+"\"";
            else:
                #select all
                if jenis_tugas == None:
                    # selectQuery = "SELECT * FROM taskList
WHERE tanggal_deadline >= "+getHariIni()+"";
                    selectQuery = "SELECT * FROM taskList
WHERE isDone = '\"Belum\"' and tanggal_deadline >=
 '\""+getHariIni()+"\"";
                else:
                    selectQuery = "SELECT * FROM taskList

```

```

WHERE isDone = \'Belum\' and jenis_task = \'"+jenis_tugas+"\' and
tanggal_deadline >= \'"+getHariIni()+"\'";

print(selectQuery)
mycursor.execute(selectQuery)
result = mycursor.fetchall()

if len(result)==0:
    return "Tidak ada deadline :D"
else:
    message = "[Daftar Deadline]<br>"
    n = 1
    for tup in result:
        idTask = tup[0]
        deadline = toDate(str(tup[1]))
        matkul = tup[2]
        jenis = tup[3]
        topik = tup[4]
        task = str(n) + ". (ID: " + str(idTask) + " )
"+str(deadline)+" - " + matkul + " - " + jenis + " - " + topik +
"<br>"
        message = message + task
        n+=1
    return message

```

4.1.7 File showDeadline.py

1. Fungsi showDeadline

Tipe keluaran : String

Parameter : String command.

Prekondisi : Masukan terdefinisi.

Kegunaan : Mengembalikan string berisi daftar tanggal deadline yang diperoleh berdasarkan command yang diberikan pengguna

Implementasi :

```

def showDeadline(command):
    # Connect database
    mydb = mysql.connector.connect(
        host="localhost",
        user="root",

```

```

        password="",
        database="task"
    )
    mycursor = mydb.cursor()

    # getKodeKuliah
    kode_matkul = getKodeMatkul(command)

    # getJenisTugas
    jenis_tugas = getJenisTugas(command)

    # getTopik
    topik = getTaskTopic(command)

    # Search tanggal based on kodekuliah, jenistugas, topik
    #select
    selectQuery = ""
    if(jenis_tugas==None):
        selectQuery = "SELECT tanggal_deadline FROM taskList WHERE
isDone = \'Belum\' and kode_matkul=\'"+kode_matkul+"\';"
    else:
        if(topik == None):
            selectQuery = "SELECT tanggal_deadline FROM taskList
WHERE isDone = \'Belum\' and jenis_task = \'"+jenis_tugas+"\' and
kode_matkul=\'"+kode_matkul+"\';"
        else:
            selectQuery = "SELECT tanggal_deadline FROM taskList
WHERE jenis_task = \'"+jenis_tugas+"\' and
kode_matkul=\'"+kode_matkul+"\' and topik_task=\'"+topik+"\';"
    print(selectQuery)
    mycursor.execute(selectQuery)
    result = mycursor.fetchall()

    if len(result)==0:
        return "Tidak ada deadline :D"
    else:
        message = ""
        for tuple in result:
            tanggal = toDate(str(tuple[0]))

```

```

        tanggal = str(tanggal)+"<br>"
        message = message + tanggal
    return message

```

4.1.8 File helpBot.py

1. Fungsi helpBot

Tipe keluaran : String

Parameter : -

Prekondisi : -

Kegunaan : Mengembalikan string berisi daftar fitur dan daftar kata penting yang disediakan oleh chatbot

Implementasi :

```

def helpBot():
    fitur = ("Menambah task baru", "Melihat daftar task",
"Menampilkan deadline task tertentu", "Memperbaharui task",
"Menandai task sudah selesai dikerjakan", "Menampilkan opsi
bantuan yang difasilitasi oleh assistant")
    kataPenting = ("Kuis", "Ujian", "Tucil", "Tubes", "Tugas",
"Praktikum", "UTS", "UAS")
    message = "Fitur : <br>"
    for tuple in fitur:
        fiturList = tuple
        fiturList = "["+str(fitur.index(tuple)+1)+"]
"+str(fiturList)+"\n"+"<br>"
        message = message + fiturList

    message = message + "<br> Kata Penting : " + "<br>"
    for tuple in kataPenting:
        kataList = tuple
        kataList = "["+str(kataPenting.index(tuple)+1)+"]
"+str(kataList)+"\n"+"<br>"
        message = message + kataList
    return message

```

4.1.9 File processInput.py

1. Fungsi removeNewLine

Tipe keluaran : string

Parameter : String bernama string
Prekondisi : Masukan valid
Kegunaan : Menghilangkan newline pada suatu string lalu mengembalikan string yang sudah tidak terdapat newline tersebut
Implementasi :

```
def removeNewLine(string):  
    newString = ''  
    for char in string:  
        if (char != '\n'):  
            newString = newString + char  
    return newString
```

2. Fungsi checkIfTanggalExist

Tipe keluaran : boolean
Parameter : String command
Prekondisi : Masukan valid
Kegunaan : Memeriksa apakah string command mengandung tanggal
Implementasi :

```
def checkIfTanggalExist(command):  
    matchObject = None  
    allowedDateFormat =  
    ['\d\d-\d\d-\d\d\d\d', '\d\d\d\d/\d\d\d\d/\d\d\d\d']  
    for format in allowedDateFormat: # Pasti date yang ada sesuai  
    sama salah satu format atau ga sama sekali  
        if (re.search(format, command)):  
            matchObject = re.search(format, command)  
    if (matchObject): # Ada tanggal dalam command  
        return True  
    else: # Tidak ada tanggal dalam command  
        return False
```

3. Fungsi checkIfKodeExist

Tipe keluaran : boolean
Parameter : String command
Prekondisi : Masukan valid
Kegunaan : Memeriksa apakah string command mengandung kode mata kuliah
Implementasi :

```
def checkIfKodeExist(command):
    matchObject = None
    allowedKodeFormat = '[A-Z][A-Z]\d\d\d\d'
    if (re.search(allowedKodeFormat,command)):
        matchObject = re.search(allowedKodeFormat,command)
    if (matchObject): # Ada kode matkul
        return True
    else: # Tidak ada kode matkul
        return False
```

4. Fungsi checkIfTopikExist

Tipe keluaran : boolean

Parameter : String command

Prekondisi : Masukan valid

Kegunaan : Memeriksa apakah string command mengandung topik task

Implementasi :

```
def checkIfTopikExist(command):
    matchObject = None
    if (re.search("\"(\w|\s)*\"|'(\w|\s)*'",command)):
        matchObject =
re.search("\"(\w|\s)*\"|'(\w|\s)*'",command)
    if (matchObject):
        return True
    else:
        return False
```

5. Fungsi checkIfIDExist

Tipe keluaran : boolean

Parameter : String command

Prekondisi : Masukan valid

Kegunaan : Memeriksa apakah string command mengandung ID

Implementasi :

```
def checkIfIDExist(command):
    matchObject = None
    allowedIDFormat = '\s[0-9]*\s'
    if (re.search(allowedIDFormat,command)):
        matchObject = re.search(allowedIDFormat,command)
```

```

if (matchObject):
    return True
else:
    return False

```

6. Fungsi processInput

Tipe keluaran : string
 Parameter : String command
 Prekondisi : Masukan valid
 Kegunaan : Membaca command lalu mengklasifikasikan command tersebut untuk mengakses fitur tertentu yang disediakan chatbot kemudian mengembalikan string jawaban berdasarkan fitur yang digunakan tersebut.

Implementasi :

```

def processInput(command):
    # Command List
    listTask = []
    listUpdateCommand = []
    listDoneCommand = []

    # Load keyword
    addToListTask = False
    addToListUpdateCommand = False
    addToListDoneCommand = False
    with open("../test/KeywordList.txt", 'r') as keywordFile:
        for line in keywordFile.readlines():
            # print(removeNewLine(line))
            if (bmMatch(removeNewLine(line), "Task:") != -1):
                addToListTask = True
                continue
            elif (bmMatch(removeNewLine(line), "Task Update:") !=
-1):
                addToListUpdateCommand = True
                addToListTask = False
                continue
            elif (bmMatch(removeNewLine(line), "Task Done:") !=
-1):
                addToListDoneCommand = True

```



```

        addToListUpdateCommand = False
        continue
    if (addToListTask):
        listTask.append(removeNewLine(line))
    elif (addToListUpdateCommand):
        listUpdateCommand.append(removeNewLine(line))
    elif (addToListDoneCommand):
        listDoneCommand.append(removeNewLine(line))

# Process command
for update in listUpdateCommand:
    if ((kmpMatch(command,update) != -1) and
(checkIfTanggalExist(command)) and (checkIfIDExist(command))):
        # Mengandung salah satu keyword untuk update, tanggal
baru, dan ID Task
        return handleUpdateTask(command)

for done in listDoneCommand:
    if ((kmpMatch(command,done) != -1) and
(checkIfIDExist(command))):
        # Mengandung salah satu keyword untuk done dan ID Task
        return handleMarkDoneTask(command)

for task in listTask: # Kalo mau pake command add task harus
ada:
    if ((kmpMatch(command,task) != -1) and
(checkIfTanggalExist(command)) and (checkIfKodeExist(command)) and
(checkIfTopikExist(command))):
        # Mengandung salah satu keyword task
(PR,Kuis,Praktikun,UTS,UAS,Tucil,Tubes), tanggal, kode matkul, dan
topik task dalam command
        return handleNewTask(command,task)

    if(kmpMatch(command.lower(), "apa saja") != -1 and
kmpMatch(command.lower(), "deadline") != -1):
        return showTask(command)

    if(kmpMatch(command.lower(), "kapan") != -1 and
kmpMatch(command.lower(), "deadline") != -1):

```

```

        return showDeadline(command)

    if(kmpMatch(command.lower(), "bisa") != -1 and
kmpMatch(command.lower(), "lakukan") != -1):
        return helpBot()

    if(kmpMatch(command.lower(), "hi") != -1 or
kmpMatch(command.lower(), "halo") != -1):
        return "Hi!, Aku Harobot ^_^"

    if(kmpMatch(command.lower(), "exit") != -1 or
kmpMatch(command.lower(), "keluar") != -1):
        return exit()

    return "Command tidak dikenali!" # Masuk kesini kalo di for
loop task ga diketahuin task apa yg mau ditambahin
# Artinya bisa langsung
dikasih tau kalo commmand ga dikenalin

```

7. Fungsi exit

Tipe keluaran : string

Parameter : String command

Prekondisi : Masukan valid

Kegunaan : Menghapus data-data pada tabel tasklist lalu mengembalikan pesan terima kasih

Implementasi :

```

def exit():
    mydb = mysql.connector.connect(
        host="localhost",
        user="root",
        password="",
        database="task"
    )

    mycursor = mydb.cursor()
    searchQuery = "DELETE FROM taskList;"
    mycursor.execute(searchQuery)
    mydb.commit()

```

```
return "Terima kasih sudah menggunakan Harobot :)"
```

4.1.10 File app.py

1. Fungsi index

Tipe keluaran : Render HTML

Parameter : -

Prekondisi : -

Kegunaan : Menampilkan tampilan file index.html yang berada pada folder templates

Implementasi :

```
@app.route('/')
def index():
    return render_template('index.html')
```

2. Fungsi start

Tipe keluaran : Render HTML

Parameter : -

Prekondisi : -

Kegunaan : Menampilkan tampilan file start.html yang berada pada folder templates

Implementasi :

```
@app.route('/start/')
def start():
    return render_template('start.html')
```

3. Fungsi get_bot_response

Tipe keluaran : String

Parameter : -

Prekondisi : -

Kegunaan : Mengembalikan string balasan (*answer*) terhadap masukan perintah pengguna (*question*) yang dimasukkan pada box yang disediakan pada tampilan web

Implementasi :

```
@app.route("/get")
def get_bot_response():
    userText = request.args.get('msg')
    answer = str(processInput(userText))
```

```

# if(userText == "halo"):
#     # return str(english_bot.get_response(userText))
#     return str('Haro, Warudo!')
# if(userText == "hi"):
#     # return str(english_bot.get_response(userText))
return answer

```

4.1.11 Folder templates

1. File index.html

index.html adalah template html tampilan awal web yang digunakan dalam aplikasi chatbot ini.

Implementasi :

```

<!DOCTYPE html>
<html>
  <head>
    <meta name="viewport" content="width=device-width,
initial-scale=1">
    <link rel="stylesheet" href="../static/styles.css">
    <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min
.js"></script>
    <title>
      Harobot | Assistant yang Membantu Hidupmu dengan Fitur
Reminder Pekerjaan!
    </title>
    <style>
      .logo {
        margin-top: 5rem !important;
      }
    </style>
  </head>
  <body>
    <div class="logo">
      
    </div>

```

```

<h1>Harobot</h1>
<h3>Penerapan String Matching dan Regular Expression dalam
    Pembangunan Deadline Reminder Assistant</h3>
<div class="container-logo">
    <a href="start" class="button">Mulai!</a>

</div>

</body>
<div class="footer">
    <p>
        @ Dzaki, Rafidika, Wishnu, 2021
    </p>
</div>

</html>

```

2. File start.html

start.html adalah template html tampilan aplikasi chatbot yang digunakan.

Implementasi :

```

<!DOCTYPE html>
<html>
    <head>
        <meta name="viewport" content="width=device-width,
initial-scale=1">
        <link rel="stylesheet" href="style/styles.css">
        <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min
.js"></script>
        <title>
            Harobot | Assistant yang Membantu Hidupmu dengan Fitur
Reminder Pekerjaan!
        </title>
        <style>
            body {
                font-family: Garamond;

```

```

}

h1 {
    color: rgb(0 193 202);
    margin-bottom: 0;
    margin-top: 2rem;
    text-align: center;
    font-size: 40px;
}

h3 {
    color: black;
    font-size: 20px;
    margin-top: 3px;
    text-align: center;
}

#chatbox {
    margin-left: auto;
    margin-right: auto;
    width: 40%;
    margin-top: 60px;
}

#userInput {
    margin-left: auto;
    margin-right: auto;
    width: 40%;
    margin-top: 60px;
    margin-bottom: 8rem;
}

#textInput {
    width: 87%;
    border: none;
    border-bottom: 3px solid rgb(0 193 202);
    font-family: monospace;
    font-size: 17px;
    margin-top: 1.5rem

```

```

}

#buttonInput {
    padding: 3px;
    font-family: monospace;
    font-size: 17px;
}

.userText {
    color: white;
    font-family: monospace;
    font-size: 17px;
    text-align: right;
    line-height: 30px;
}

.userText span {
    background-color: rgb(0 193 202);
    padding: 10px;
    border-radius: 2px;
}

.botText {
    color: rgb(255, 255, 255);
    font-family: monospace;
    font-size: 17px;
    text-align: left;
    line-height: 30px;
}

.botText span {
    background-color: rgb(0 193 202);
    padding: 10px;
    border-radius: 2px;
}

#tidbit {
    position: absolute;
    bottom: 0;

```

```

    right:0;
    width: 300px;
}

.footer {
    position: fixed;
    left: 0;
    bottom: 0;
    width: 100%;
    background-color: rgb(0 193 202);
    color: white;
    text-align: center;
}

.chat {
    list-style-type: none;
    width: 20em;
}

.chat__bubble {
    margin-bottom: 3px;
    padding: 5px 10px;
    clear: both;
    border-radius: 10px 10px 2px 2px ;
}

.chat__bubble--rcvd {
    background: #f2f2f2;
    color: black;
    float: left;
    border-bottom-right-radius: 10px;
}

.chat__bubble--rcvd span{
    padding: 1rem;
}

.chat__bubble--sent {
    background: rgb(0 193 202);

```



```

    color: white;
    float: right;
    border-bottom-left-radius: 10px;
}

.chat__bubble--sent span{
    padding: 1rem;
}

.chat__bubble--sent + .chat__bubble--sent {
    border-top-right-radius: 2px;
}

.chat__bubble--rcvd + .chat__bubble--rcvd {
    border-top-left-radius: 2px;
}

.chat__bubble--stop {
    border-bottom-right-radius: 10px;
    border-bottom-left-radius: 10px;
}

a:-webkit-any-link {
    color: rgb(0 193 202);
    cursor: pointer;
    text-decoration: none;
}

@media (max-width:768px) {
    #chatbox {

        width: 95%;

    }

    #userInput{
        width: 95%;
    }
}

```

```

    #textInput{

    width: 80%;
    }

}

</style>
</head>
<body>
    <a href="/"><h1>Harobot</h1></a>

    <h3>Penerapan String Matching dan Regular Expression dalam
    Pembangunan Deadline Reminder Assistant</h3>
    <div>
        <div id="chatbox">

        </div>
        <div id="userInput">
            <input id="textInput" type="text" name="msg"
placeholder="Message">
            <input id="buttonInput" type="submit" value="Send"
href="textInput">
        </div>
        <script>
            function getBotResponse() {
                var rawText = $("#textInput").val();
                var userHtml = '<p class="chat__bubble
chat__bubble--sent"><span>' + rawText + '</span></p>';
                $("#textInput").val("");
                $("#chatbox").append(userHtml);

document.getElementById('textInput').scrollIntoView({block:
'start', behavior: 'smooth'});
                $.get("/get", { msg: rawText }).done(function(data) {
                    var botHtml = '<p class="chat__bubble
chat__bubble--rcvd chat__bubble--stop"><span>' + data +
'</span></p>';

```

```

        $("#chatbox").append(botHtml);

document.getElementById('textInput').scrollIntoView({block:
'start', behavior: 'smooth'});
    });
}
    $("#textInput").keypress(function(e) {
        if ((e.which == 13) &&
document.getElementById("textInput").value != "" ) {
            getBotResponse();
        }
    });
    $("#buttonInput").click(function() {
        if (document.getElementById("textInput").value != "")
{
            getBotResponse();
        }
    })
</script>
</div>

</body>
<div class="footer">
    <p>
        By Dzaki, Rafidika, Wishnu
    </p>
</div>

</html>

```

4.1.12 File task.sql

File task.sql adalah basis data yang digunakan dalam program. Dalam basis data ini hanya terdapat satu tabel yaitu tabel “tasklist”. Tabel tasklist memiliki atribut id_task bertipe int(11) yang merupakan primary key dan menandakan ID suatu task, tanggal_deadline bertipe date yang menandakan deadline dari suatu task, kode_matkul bertipe varchar(10) yang menandakan kode mata kuliah dari suatu task, jenis_task bertipe varchar(255) yang menandakan jenis dari suatu task, topik_task bertipe varchar(255) yang menandakan topik dari suatu task, dan

isDone bertipe enum('Sudah', 'Belum') yang menandakan apakah suatu tugas telah selesai dikerjakan atau belum.

4.2 Tata Cara Penggunaan Program

4.2.1 Antarmuka Program



Gambar 4.2.1.1 Tampilan awal antarmuka program

Harobot

Penerapan String Matching dan Regular Expression dalam Pembangunan Deadline Reminder Assistant

The screenshot displays a chat interface for 'Harobot'. At the top right, a teal button says 'Halo'. Below it, a teal bubble contains the text 'Apa saja yang kamu bisa lakukan?'. On the left, a grey bubble says 'Hi!, Aku Harobot ^_^'. In the center, a grey box lists features and keywords. At the bottom, there is a text input field labeled 'Message' and a 'Send' button.

Halo

Hi!, Aku Harobot ^_^

Apa saja yang kamu bisa lakukan?

Fitur :

- [1] Menambah task baru
- [2] Melihat daftar task
- [3] Menampilkan deadline task tertentu
- [4] Memperbaharui task
- [5] Menandai task sudah selesai dikerjakan
- [6] Menampilkan opsi bantuan yang difasilitasi oleh assistant

Kata Penting :

- [1] Kuis
- [2] Ujian
- [3] Tugil
- [4] Tubes
- [5] Tugas
- [6] Praktikum
- [7] UTS
- [8] UAS

Message Send

Gambar 4.2.1.2 Tampilan kedua antarmuka program

4.2.2 Penggunaan Fitur

1. Fitur menambahkan task baru

Untuk menggunakan fitur menambahkan task baru, masukkan perintah yang mengandung tanggal, jenis tugas, kode mata kuliah, dan topik tugas. Tanggal yang dimasukkan harus dalam format DD-MM-YYYY atau DD/MM/YYYY. Ada 7 jenis tugas yang dapat dimasukkan, yaitu “Kuis”, “Tubes”, “Tugil”, “Praktikum”, “PR”, “UAS”, dan “UTS”. Kode mata kuliah yang dimasukkan harus dalam format: 6 karakter dengan 2 karakter pertama berupa huruf dan 4 karakter terakhir berupa angka. Terakhir, topik tugas dikenali oleh sistem menggunakan petik di antara topik. Contoh perintah : “Tubes IF2211 ‘Regex’ pada 30/04/2021”

2. Fitur melihat daftar task

Untuk menggunakan fitur melihat daftar task, masukkan perintah yang mengandung kata “apa saja” dan “deadline”. Perintah masukan juga dapat mengandung batasan lain seperti jenis tugas dan periode waktu. Contoh perintah : “Apa saja deadline tubes yang ada 1 minggu ke depan?”

3. Fitur menampilkan deadline suatu task

Untuk menggunakan fitur menampilkan deadline suatu task, masukkan perintah yang mengandung kata “kapan” dan “deadline”. Perintah masukan harus mengandung kode mata kuliah dan dapat mengandung batasan jenis tugas. Contoh perintah : “Deadline tugas IF2211 itu kapan?”

4. Fitur memperbaharui task tertentu

Untuk menggunakan fitur menandai suatu task sudah dikerjakan, masukkan perintah yang mengandung nomor ID task, kata kunci “diundur” atau “Diundur” atau “update” atau “Update”, dan tanggal deadline baru. Format ID yang diperbolehkan haruslah angka dan diakhir serta diawali dengan sebuah spasi. Format tanggal sama dengan yang dijelaskan pada nomor 1. Contoh perintah : “Deadline task 2 diundur menjadi 08/05/2021”.

5. Fitur menandai suatu task sudah dikerjakan

Untuk menggunakan fitur menandai suatu task sudah dikerjakan, masukkan perintah yang mengandung nomor ID task dan kata kunci “selesai” atau “Selesai” atau “done” atau “Done”. Format ID sama dengan yang dijelaskan pada nomor 4. Contoh perintah : “Saya sudah selesai mengerjakan task 2” .

6. Fitur menampilkan opsi bantuan

Untuk menggunakan fitur menampilkan opsi bantuan, masukkan perintah yang mengandung kata “bisa” dan “lakukan”. Contoh perintah : “Apa yang bisa bot lakukan?”

4.3 Hasil Pengujian dan Analisis

4.3.1 Kasus Uji 1 Fitur Menambahkan Task Baru

Input
Tubes IF2242 "Milestone 1" 05/05/2021
Hasil:
Task berhasil ditambahkan! (ID: 2) 2021-05-05 - IF2242 - Tubes - "Milestone 1"
Analisis:
Program mendeteksi keyword Tubes, IF2242, string "Milestone 1", dan 05/05/2021. Lalu keyword tersebut dimasukan ke database dan bot akan mengirimkan pesan bahwa task sudah ditambahkan

4.3.2 Kasus Uji 2 Fitur Menambahkan Task Baru

Input
tolong ingetin kalau ada Kuis IF3110 "Bab 2 sampai Bab3" pada 01/05/2021
Hasil:
Task berhasil ditambahkan! (ID: 3) 2021-05-01 - IF3110 - Kuis - "Bab 2 sampai Bab3"
Analisis:
Program mendeteksi keyword , IF3110, string "Bab 2 sampai Bab3", dan 01/05/2021. Lalu keyword tersebut dimasukan ke database dan bot akan mengirimkan pesan bahwa task sudah ditambahkan

4.3.3 Kasus Uji 3 Fitur Menambahkan Task Baru

Input
olong ingetin kalau ada kuis IF3110 "Bab 2 sampai Bab3" pada 01/05/2021
Hasil:
Command tidak dikenali!
Analisis:
Program mendeteksi keyword kuis, IF3110, string "Bab 2 sampai Bab3", dan 01/05/2021. Command tidak dikenali karena seharusnya kata penting kuis dalam kapital (Kuis)

4.3.4 Kasus Uji 1 Fitur Melihat Daftar Task

Input
Apa saja deadline yang ada sejauh ini?
Hasil:
[Daftar Deadline] 1. (ID: 1) 28/04/2021 - IF2211 - Tubes - Regex 2. (ID: 2) 05/05/2021 - IF2242 - Tubes - "Milestone 1" 3. (ID: 3) 01/05/2021 - IF3110 - Kuis - "Bab 2 sampai Bab3"
Analisis:
Program mendeteksi keyword deadline, apa, saja. Harobot akan mengeluarkan pesan yang berisi daftar task yang ada.

4.3.5 Kasus Uji 2 Fitur Melihat Daftar Task

Input
Apa saja deadline yang ada dalam 3 hari ke depan?
Hasil:
[Daftar Deadline] 1. (ID: 1) 28/04/2021 - IF2211 - Tubes - Regex 2. (ID: 3) 01/05/2021 - IF3110 - Kuis - "Bab 2 sampai Bab3"
Analisis:
Program mendeteksi keyword deadline, apa, saja, dan 3 hari. Harobot akan mengeluarkan pesan yang berisi daftar task yang ada selama 3 hari kedepan.

4.3.6 Kasus Uji 3 Fitur Melihat Daftar Task

Input
Deadline yang ada ?
Hasil:
Command tidak dikenali!
Analisis:

Program mendeteksi keyword “deadline” saja sehingga tidak cukup untuk menunjukkan daftar task yang berakibat Harobot mengeluarkan pesan command tidak dikenali.

4.3.7 Kasus Uji 1 Fitur Menampilkan deadline task tertentu

Input
Deadline Tubes IF2242 itu kapan?
Hasil:
05/05/2021
Analisis:
Program mendeteksi keyword “deadline” “Tubes” “IF2242” “kapan” sehingga Harobot akan mengeluarkan deadline task tersebut.

4.3.8 Kasus Uji 1 Fitur Memperbaharui Task

Input
Deadline task 1 diundur menjadi 29/04/2021
Hasil:
Task berhasil di-update! Deadline task dengan ID = 1 berhasil di-update menjadi tanggal 29/04/2021!
Analisis:
Program mendeteksi keyword “deadline” “task 1” “diundur” 29/04/2021 sehingga deadline task 1 akan diubah di database sesuai dengan command dari user. Harobot akan mengirimkan pesan berhasil update dan deadline yang baru untuk task tersebut.

4.3.9 Kasus Uji 1 Fitur Menandai task sudah selesai dikerjakan

Input
task 1 sudah selesai
Hasil:
Task sudah ditandai selesai dikerjakan! Task dengan ID = 1 sudah selesai dikerjakan!

Analisis:
Program mendeteksi keyword “task 1” “selesai”, lalu pada database task yang memiliki id = 1 akan dihilangkan. Lalu Harobot mengirimkan pesan bahwa task tersebut sudah di-remove dari database.

4.3.10 Kasus Uji 2 Fitur Menandai task sudah selesai dikerjakan

Input
Sudah selesai task 2
Hasil:
Command tidak dikenali!
Analisis:
Program mendeteksi keyword “selesai” “task 2” dimana seharusnya id dari task harus berada ditengah kata “task” dan “selesai” supaya fitur mendai task yang sudah dikerjakan berjalan.

4.3.11 Kasus Uji 1 Fitur Menampilkan opsi bantuan yang difasilitasi oleh assistant

Input
Apa yang harobot bisa lakukan?
Hasil:
<p>Fitur :</p> <ul style="list-style-type: none"> [1] Menambah task baru [2] Melihat daftar task [3] Menampilkan deadline task tertentu [4] Memperbaharui task [5] Menandai task sudah selesai dikerjakan [6] Menampilkan opsi bantuan yang difasilitasi oleh assistant <p>Kata Penting :</p> <ul style="list-style-type: none"> [1] Kuis [2] Ujian [3] Tugil [4] Tugas [5] Tugas [6] Praktikum

[7] UTS [8] UAS
Analisis:
Program mendeteksi keyword “bisa” “lakukan” dan akan mengirim pesan yang berisi fitur dan kata penting yang terdapat pada program Harobot.

4.3.12 Kasus Uji 1 Keluar dari Program

Input
exit
Hasil:
Terima kasih sudah menggunakan Harobot :)
Analisis:
Program mendeteksi keyword “exit” dan akan melakukan reset dengan perintah delete from pada database user.

BAB 5

KESIMPULAN, SARAN, DAN REFLEKSI

5.1 Kesimpulan

Dari tugas besar IF2211 Strategi Algoritma semester 2 2020/2021 berjudul “Penerapan String Matching dan Regular Expression dalam Pembangunan Deadline Reminder Assistant”, kami berhasil membuat aplikasi chatbot berbasis web dengan sistem *question and answers* yang dapat melakukan fitur menambahkan task baru, melihat daftar task, menampilkan deadline suatu task, memperbaharui task tertentu, menandai suatu task sudah dikerjakan, menampilkan opsi bantuan, dan menampilkan pesan error jika tidak mengenal masukan pengguna. Aplikasi chatbot yang dibangun memanfaatkan *string matching* menggunakan algoritma *Knuth-Morris-Pratt*, *Boyer-Moore*, dan *Regex Matching* untuk memproses perintah yang dimasukkan pengguna sehingga fitur yang digunakan dan keluaran chatbot sesuai dengan keinginan pengguna.

Dari pengerjaan tugas besar ini, kami mendapatkan beberapa kesimpulan, yaitu:

1. Semua fitur chatbot dapat diimplementasikan dengan baik menggunakan ketiga algoritma *string matching* dan menghasilkan keluaran yang diharapkan.
2. Pemrosesan string yang memuat angka atau tanggal lebih efektif dengan menggunakan *regex matching*.
3. Algoritma Knuth-Morris-Pratt dan Boyer-Moore efektif digunakan dalam pencocokan string yang memuat kata-kata penting yang telah didefinisikan terlebih dahulu.

5.2 Saran

Saran-saran yang dapat kami berikan untuk tugas besar ini adalah:

1. Mengaplikasikan fitur rekomendasi perintah, yaitu fitur yang dapat merekomendasikan perintah kepada pengguna ketika perintah yang dimasukkan pengguna tidak dikenali sehingga pengguna lebih terbantu dalam menggunakan chatbot.
2. Membuat perintah yang dapat memberi tahu instruksi untuk menggunakan fitur tertentu sehingga membantu pengguna dalam menggunakan fitur-fitur yang tersedia dalam chatbot tanpa harus melihat dokumentasi.

5.3 Refleksi

Setelah menyelesaikan tugas besar ini, kami dapat merefleksikan beberapa hal, yaitu:

1. Perencanaan dan pembagian tugas sangat penting dalam mengerjakan tugas besar secara berkelompok.
2. Lebih merapikan *source code* agar program lebih mudah dipahami.
3. Pentingnya *comments* dalam bagian-bagian kode program sehingga rekan-rekan tim yang lain dapat memahami bagian kode yang kita tulis.
4. Dalam menyelesaikan sebuah masalah dengan pendekatan informatika, penting bagi kita untuk menyusun strategi algoritma yang sesuai sehingga penyelesaian masalah bisa dilakukan dengan efisien.

DAFTAR PUSTAKA

- Anonim. (2021), What Is a Chatbot?. Diakses online dari <https://www.oracle.com/chatbots/what-is-a-chatbot/> pada 27 April 2021.
- Khodra, Masayu Leylia. (2019), String Matching dengan Regular Expression. Diakses online dari <https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2018-2019/String-Matching-dengan-Regex-2019.pdf> pada 27 April 2021.
- Munir, Rinaldi dan Maulidevi, Nur Ulfa. (2021), Pencocokan String (String/Pattern Matching). Diakses online dari <https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Pencocokan-string-2021.pdf> pada 25 April 2021.