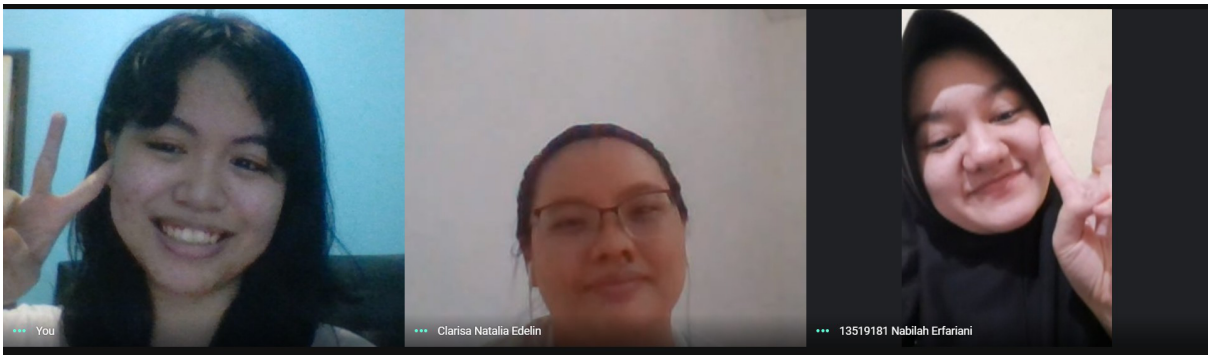
Tugas Besar III IF2211 Strategi Algoritma

Penerapan String Matching dan Regular Expression dalam Pembangunan Deadline Reminder Assistant

Semester II Tahun 2020/2021



Kelompok 47

lasteffort

Rhea Elka Pandumpi (13519047)

Nabilah Erfariani (13519181)

Clarisa Natalia Edelin (13519213)

PROGRAM STUDI TEKNIK INFORMATIKA

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

2021

BAB I
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
    1. Suatu kalimat *diklasifikasikan* sebagai suatu *task* apabila mengandung semua komponen berikut ini:
        1. Tanggal (format dibebaskan)
        2. Kode Mata Kuliah / Nama Mata Kuliah (dibebaskan)
        3. Jenis Tugas (berdasarkan daftar kata penting yang sudah disediakan)
        4. Topik Tugas (tidak ada batasan)
    2. Point i sampai dengan iv diklasifikasikan menggunakan regular expression sehingga masukan kalimat benar-benar layaknya kalimat sehari-hari
    3. 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

4. Contoh interaksi

2. Melihat daftar *task* yang harus dikerjakan
    1. Seluruh *task* yang sudah tercatat oleh assistant
        - Contoh perintah yang dapat digunakan: “Apa saja deadline yang dimiliki sejauh ini?”
    2. Berdasarkan periode waktu
        1. Pada periode tertentu (DATE_1 until DATE_2)

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

        2. N minggu ke depan

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

        3. N hari ke depan

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

        4. Hari ini

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

    3. Berdasarkan jenis task (kata penting)
        1. Sesuai dengan daftar task yang didefinisikan

2. User dapat melihat daftar task dengan jenis task tertentu
3. Misalnya: “3 minggu ke depan ada kuis apa saja?”, maka Chatbot akan menampilkan daftar kuis selama 3 minggu kedepa

Catatan:

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

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
    1. Hanya berlaku untuk task yang bersifat Tugas atau memiliki tenggat waktu
    2. Misalnya: “Deadline tugas IF2211 itu kapan?”
4. Memperbaharui task tertentu
    1. Memperbarui tanggal dari suatu task (dalam kehidupan nyata, tentu ada kejadian dimana deadline dari suatu task diundur)
    2. Perintah yang dimasukkan meliputi 1 keyword untuk memperbaharui suatu task dan nomor task tertentu.
    3. Misalnya:

- “Deadline task X diundur menjadi 28/04/2021” dimana X merupakan nomor ID dari suatu task.

1. 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
    1. 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.
    2. Misalnya:

        - “Saya sudah selesai mengerjakan task X” dimana X merupakan nomor ID dari suatu task.

    1. 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
    1. Berisikan command-command yang dapat digunakan oleh user
    2. Misalnya: “Apa yang bisa assistant lakukan?”
    3. Bot akan memberikan hasil berupa daftar kata-kata yang bisa digunakan untuk menambahkan dan melihat daftar task (setiap kelompok bebas membentuknya seperti apa)
    4. Contoh interaksi
7. Mendefinisikan list kata penting terkait apakah itu merupakan suatu *task* atau tidak
    1. Minimal terdapat 5 kata penting berbeda, contohnya adalah:

        [“Kuis”, “Ujian”, “Tucil”, “Tubes”, “Praktikum”]

    2. Kata penting akan digunakan pada penentuan jenis tugas dari suatu *task*.
    3. Daftar kata penting tidak perlu dibuat dinamis, cukup static saja atau hardcoded.
8. Menampilkan pesan error jika assistant tidak dapat mengenali masukan user.
    1. Masukan yang tidak termasuk ke dalam jenis pesan di poin 1 sampai 4 dapat dikategorikan sebagai masukan tak dikenali.
    2. Error message dibebaskan sesuai kreativitas mahasiswa
9. (Bonus) Chatbot dapat memberikan rekomendasi kata jika terdapat kesalahan kata (*typo*) pada perintah yang ditulis pengguna
    1. Berikan rekomendasi kata jika perintah masukan pengguna *mismatch* dengan daftar kata yang diterima chatbot, namun masih memiliki tingkat kemiripan di atas 75%.
    2. Contoh interaksi
    3. Ada berbagai metriks yang dapat dimanfaatkan untuk mencari kemiripan kata, salah satunya adalah Levenshtein distance yang diukur melalui pendekatan *dynamic programming*. Anda dapat mempelajari Levenshtein distance melalui pranala ini.

BAB II
LANDASAN TEORI

2.1. Algoritma Knuth Morris Pratt(KMP)

KMP adalah algoritma yang mencari *pattern* dari kiri ke kanan seperti *brute force*, tetapi secara lebih 'cerdas'. KMP melakukan *preprocess* terhadap *pattern* untuk menemukan *match* dari *prefix* dari *pattern* dengan *pattern* itu.

j = posisi *mismatch* pada P[]

k = posisi sebelum *mismatch* (k=j-1)

Border function b(k) didefinisikan sebagai ukuran dari *prefix* terbesar dari P[0..k] yang juga adalah *suffix* dari P[1..k]

2.2. Algoritma Boyer-Moore(BM)

Algoritma ini didasarkan pada dua teknik.

1. Teknik *looking-glass*
    - Temukan P dalam T dengan bergerak *backwards* melalui P, dimulai dari ujung,
2. Teknik *character-jump*
    - Ketika *mismatch* muncul pada T[i]==x
    - Karakter dalam *pattern* P[j] tidak sama dengan T[i]

Ada 3 kasus:

1. Jika P memiliki x di suatu tempat, geser P ke kanan untuk menyesuaikan kemunculan terakhir dari x dalam P dengan T[i]
2. Jika P memiliki x di suatu tempat, tapi suatu pergeseran ke kanan ke kemunculan terakhir tidak bisa dilakukan, geser P ke kanan 1 karakter ke T[i+1].
3. Jika tidak memenuhi kasus 1 dan kasus 2, maka geser P untuk menyesuaikan P[0] dengan T[i+1].

2.3. Algoritma Regex

Regular expression (regex) merupakan sekumpulan notasi dan karakter yang digunakan untuk mendeskripsikan suatu pola pada pencarian berbasis

huruf. Dengan regex dimungkinkan untuk mengenali suatu string yang mempunya karakteristik dan pola tertentu, seperti email, tanggal, nomor kartu kredit, dll. Misal dengan notasi regex p.+ingmaka akan menyaring semua kata-kata kecuali kata yang diawali huruf "p"dan mempunyai akhiran, seperti playing, praying, pulling, dll.

Notasi regex dapat dikombinasikan sedemikian kompleksnya sehingga dapat menemukan kata yang mempunya pola-pola cukup rumit. Adapun contohnya:

^[_a-z0-9-]+(\.[_a-z0-9-]+)*@[a-z0-9-]+(\.[a-z0-9-]+)*(\.[a-z]{2,4})$

akan menampilkan error atau warning ketika string yang dimasukkan tidak memiliki format seperti alamat email, misal john@yahoo.com

2.4. Chatbot

Chatbot adalah sebuah program komputer berbasis AI (Artificial Intelligence), alias robot virtual yang dapat mensimulasikan percakapan layaknya manusia. Teknologi ini juga dikenal sebagai asisten digital yang dapat memahami serta memproses permintaan pengguna, dan memberikan jawaban yang relevan dengan cepat.

Secara sederhana, cara kerja Chatbot adalah dengan mengandalkan keyword alias kata kunci yang sudah tertanam pada sistem. Maka, setiap kali Chatbot memperoleh pertanyaan dari pengguna, secara otomatis ia akan menyesuaikan jawaban mana yang sesuai dengan keyword pertanyaan yang diajukan.

Selain dirancang dengan kemampuan analisa dan identifikasi yang begitu responsif, ada tiga macam metode sistem operasional yang dianut oleh Chatbot yaitu pattern matching, decision tree-based, contextual.

BAB III
ANALISIS PEMECAHAN MASALAH

3.1. Langkah penyelesaian masalah setiap fitur

1. Menambah task baru

- Membuat fungsi addTask dengan parameter tanggal, kode kuliah, jenis tugas, dan topik tugas
- Mengambil hari, bulan, dan tahun pada tanggal dengan melakukan split
- Memasukkan semua info deadline berupa hari, bulan, tahun, kode kuliah, jenis tugas, dan topik tugas ke satu variabel bernama data_deadline
- Terdapat array of array bernama listOfDeadlinesComponent yang berisi komponen-komponen info dari suatu deadline. Misal terdapat deadline : 27 04 2020 IF2210 Tubes String Matching, maka komponen-komponen infonya berisi : “27”, “04”, “2020”, “IF2210”, “Tubes”, “String”, “ Matching
- listOfDeadlinesComponent diperoleh dengan mengiterasi listOfDeadlines, lalu melakukan append ke listOfDeadlinesComponent. listOfDeadlines diperoleh dari pembacaan file “deadline.txt”
- Memasukkan data_deadline ke listOfDeadlinesComponent

2. Melihat daftar task yang harus dikerjakan

- Seluruh *task*
    - Membuat fungsi tampilkanSeluruhDeadline tanpa parameter.
    - Membuat variabel string bernama deadline
    - Melakukan iterasi pada listOfDeadlinesComponent
    - Memasukkan tiap iterasi i ke dalam variabel deadline yang diakhiri dengan new line
    - Fungsi lalu mengembalikan deadline
- Berdasarkan periode waktu
    - Membuat fungsi tampilkanDeadline dengan parameter input dari user
    - Melakukan split pada input lalu memasukkannya ke dalam variabel arraystring
    - Melakukan iterasi variabel i pada arraystring
    - Melakukan pengecekan apakah dalam string i terdapat kata “minggu” atau “hari”. Jika ada, maka akan dicari banyak minggu atau banyak hari yang diminta dengan menggunakan regex

- Terdapat fungsi readAdaBerapaTanggal yang menghasilkan banyak tanggal pada suatu string. Satu tanggal terdiri dari DD/MM/YYYY
- Jika fungsi readAdaBerapaTanggal menghasilkan angka 2, yang berarti ada 2 tanggal yang dimasukkan user, maka akan dipanggil fungsi deadlinesBetween yang berparameter listOfDeadlines, tanggal awal, dan tanggal akhir. Fungsi deadlinesBetween akan mengembalikan list deadline yang ada di dalam range tanggal yang diberikan
- Jika fungsi readAdaBerapaTanggal menghasilkan angka 1, yang berarti ada 1 tanggal yang dimasukkan user. Maka akan dipanggil fungsi deadlineFromNow yang berparameter banyak hari dari hari ini. Untuk input yang menggunakan minggu, maka di dalam parameter akan diubah dahulu menjadi hari dengan mengalikan banyakminggu dengan 7. Fungsi deadlineFromNow akan menghasilkan list deadline sesuai dengan range tanggal.

- Berdasarkan jenis *task*
  - Membuat fungsi deadlineKataPenting dengan parameter input user.
  - Melakukan split pada input user dan memasukkannya ke variabel arraystring
  - Melakukan iterasi variabel i pada listOfDeadlinesComponent. Jika indeks ke 4 dari i sama dengan input user, maka i akan dimasukkan ke array arrayOfDeadline
  - Mengembalikan arrayOfDeadline yang berisi list deadline berdasarkan jenis tugas.

3. Menampilkan *deadline* dari suatu *task* tertentu

- Membuat fungsi tanyakanDeadline dengan parameter tugas(input user).
- Melakukan iterasi variabel i pada listOfDeadlinesComponent, jika indeks ke-3 dari i(yang berarti kodekuliah) sama dengan parameter input fungsi, maka akan ditampilkan tanggal deadline dari seluruh deadline untuk matakuliah tertentu.

4. Memperbaharui *task* tertentu

- Membuat fungsi perbaharuiTask dengan parameter ID task yang dicari serta tanggal baru deadline tugas dari input user.
- Melakukan split pada tanggalbaru(input user)
- Melakukan iterasi variabel i pada listOfDeadlinesComponent. Jika indeks ke-0 dari i sama dengan ID task yang dicari, maka akan di-assign indeks ke 1,2,3 dari i ke dalam tanggalbaru
- Fungsi akan mengembalikan output berupa pesan List berhasil diperbaharui jika task yang dicari ada di listOfDeadlinesComponent. Jika tidak, maka fungsi akan mengembalikan pesan error.

5. Menandai bahwa suatu *task* sudah selesai

- Membuat fungsi deleteTask dengan parameter ID Task yang dicari dari input user
- Melakukan iterasi variabel i pada listOfDeadlinesComponent. Jika indeks ke-0 dari i sama dengan ID Task yang dicari, maka akan dipanggil fungsi remove pada listOfDeadlinesComponent ke-i.
- Fungsi akan mengembalikan pesan List berhasil diperbaharui jika task yang diinginkan berhasil dihapus. Jika task yang diinginkan tidak ada di dalam listOfDeadlinesComponent, maka akan diberikan pesan error.

6. Menampilkan opsi *help*

- Membuat fungsi help tanpa parameter
- Membuat variabel bernama listfitur yang berisi string daftar fitur-fitur yang disediakan chatbot
- Melakukan iterasi variabel i pada katapenting. Lalu me-assign i ke variabel katapentingstr yang diakhiri dengan newline.
- Membuat variabel output yang berisi gabungan listfitur dan katapentingstr
- Fungsi mengembalikan variabel output.

7. Mendefinisikan *list* kata penting

- Membuat fungsi deteksiKataPenting dengan parameter input user
- Membuat variabel katapenting yang berisi list kata penting pada file “katapenting.txt”
- Melakukan split pada input user lalu me-assign ke variabel arraystring.

- Melakukan iterasi variabel j pada arraystring, lalu melakukan pengecekan apakah variabel j terdapat pada katapenting. Jika ada, maka akan j akan di assign ke variabel jenis. Jika tidak, maka tidak dilakukan apa-apa
- Fungsi akan mengembalikan variabel jenis yang berisi katapenting yang terdeteksi pada suatu kalimat

8. **Menampilkan pesan *error* jika *assistant* tidak dapat mengenali masukan *user***
    - Membuat fungsi deteksiPerintah dengan parameter input user
    - Melakukan split pada input user lalu me-assign ke variabel arraystring
    - Membuat variabel keyword_perintah yang berisi list keyword perintah yang ada pada file "keyword_perintah.txt"
    - Melakukan iterasi variabel j pada arraystring. Jika j terdapat pada keyword_perintah maka akan dihasilkan variabel boolean valid yang bernilai true. Jika tidak, akan variabel valid akan bernilai false.
    - Membuat fungsi deteksiKodeKuliah, deteksiKeywords, dan deteksiKataPenting
    - Jika fungsi deteksiPerintah bernilai false, fungsi deteksiKodeKuliah bernilai none, fungsi deteksiKataPenting bernilai none, dan fungsi deteksKeywords bernilai none yan artinya perintah user tidak dikenali oleh chatbot, maka akan diberikan pesan error : "Maaf, pesan tidak dikenali"

3.2. Fitur fungsional dan arsitektur Chatbot yang dibangun

Fitur Fungsional

1. **Menambah task baru**

   → menambahkan suatu tugas ke list of Deadlines, dan memasukkannya ke file "deadline.txt"

2. **Melihat daftar task yang harus dikerjakan**
    - Seluruh *task*

      → menampilkan seluruh tugas yang ada pada file "deadline.txt"
    - Berdasarkan periode waktu

      → menampilkan seluruh tugas pada periode waktu tertentu
    - Berdasarkan jenis *task*

→ menampilkan seluruh tugas sesuai dengan jenis tugas yang diminta, keyword jenis tugas disesuaikan dengan kata yang ada pada file "katapenting.txt"

3. **Menampilkan *deadline* dari suatu *task* tertentu**

   → menampilkan semua tanggal deadline dari tugas mata kuliah tertentu

4. **Memperbarui *task* tertentu**

   → memperbarui tanggal deadline suatu tugas

5. **Menandai bahwa suatu *task* sudah selesai**

   → Menghapus suatu tugas yang telah diselesaikan dari list deadline

6. **Menampilkan opsi *help***

   → menampilkan list fitur yang tersedia

7. **Mendefinisikan *list* kata penting**

   → menyimpan list kata penting yang digunakan oleh bot

8. **Menampilkan pesan *error* jika *assistant* tidak dapat mengenali masukan *user***

   → menampilkan pesan error jika input tidak sesuai dengan pattern matching bot

Arsitektur Chatbot yang dibangun:

1. Input Box

   → menerima masukan user

2. Button Submit

   → tombol untuk melakukan aksi sesuai perintah user

3. Menampilkan chat user dengan bot.
   - Chat dari user berada di sisi kanan bot
   - Chat dari bot berada di sisi kiri bot

BAB IV

IMPLEMENTASI DAN PENGUJIAN

4.1. Spesifikasi teknis program (struktur data, fungsi, prosedur)

1. Struktur Data

1. listOfDeadlines

   → merupakan array of string yang membaca file “deadline.txt” line per line.

2. listOfDeadlinesComponent

   → merupakan array of array yang berisi kata per kata dari listOfDeadlines. Setelah selesai melakukan suatu fitur, akan ditulis ke “deadline.txt” sebagai bentuk pembaharuan.

3. katapenting

   → merupakan array of string yang membaca file “katapenting.txt” line per line

4. keyword_perintah

   → merupakan array of string yang membaca file “keyword_perintah.txt” line per line

5. 4. keywordss

   → merupakan array of string yang membaca file “keywords.txt” line per line

2. Fungsi dan prosedur

Fungsi membuat array dari file, dipisahkan per kalimat.

```
def readFile(fileName):
    fileObj = open(fileName, "r")
    words = fileObj.read().splitlines()
    fileObj.close()
    return words
```

Fungsi untuk mengembalikan tanggal menjadi format sesuai sistem, yaitu “DD MM YYYY”.

```
def convert(today_date):
  # ubah jadi sistem
  tahun = ""
  bulan = ""
```

```
    tanggal = ""
    i = 0
    while(i<=3):
        tahun = tahun + str(today_date)[i]
        i=i+1
    i=i+1
    while(i<=6):
        bulan = bulan + str(today_date)[i]
        i=i+1
    i=i+1
    while(i<=9):
        tanggal = tanggal + str(today_date)[i]
        i=i+1
    today_dateConverted= tanggal + " " + bulan + " " + tahun
    return today_dateConverted
```

Fungsi yang mengembalikan *deadline* beberapa hari dari sekarang.

```
def deadlineFromNow(N):
    Enddate = date.today() + timedelta(days=N)
    tanggalYangDicari = convert(Enddate)
    deadline=""
    for i in listOfDeadlinesComponent:
        if(i[1] + " " + i[2] + " " + i[3]==tanggalYangDicari):
            return deadline+str(i)
```

Fungsi yang mengembalikan *deadline* antara dua hari.

```
def deadlinesBetween(listOfDeadlines,date1,date2):
        date1=readDate(date1)
        date2=readDate(date2)
        start_date =
date(int(cariKomponenTahun(date1)),int(cariKomponenBulan(date1)),
int(cariKomponenHari(date1)))
        end_date = date(int(cariKomponenTahun(date2)),
int(cariKomponenBulan(date2)), int(cariKomponenHari(date2)))
        delta = end_date-start_date
        the_days = []
        for i in range(delta.days + 1):
                day_range = start_date + timedelta(days=i)
                the_days.append(convert(str(day_range)))
        deadlinesSemuanya=""
        for i in listOfDeadlinesComponent:
            for j in the_days:
```

```
            if(i[1] + " " + i[2] + " " + i[3]==j):
                deadlinesSemuanya=deadlinesSemuanya + str(i) + "\n"
    return deadlinesSemuanya
```

Fungsi yang mencari ada berapa tanggal dalam input.

```
def readAdaBerapaTanggal(input):
    pattern =
re.compile("[0-9]+(\s|/|-)+(((januari|februari|maret|april|mei|juni|juli|agustus|s
eptember|november|desember))|[0-9])+(\s|/|-)+[0-9]+[0-9]",re.IGNORECAS
E)
    zTotal = re.finditer(pattern,input)
    tanggal=[]
    for x in re.finditer(pattern,input):
        tanggal.append((x.group()))
    return tanggal
```

Fungsi KMP.

```
def computefail(pattern):
    panjangpattern = len(pattern)
    fail = [0 for i in range (panjangpattern)]
    j = 0
    i = 0

    while(i<panjangpattern):
        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

def kmpstringmatching(contohstring,contohtext):

    panjangstring = len(contohstring)
    panjangtext = len(contohtext)

    fail = computefail(contohtext)
```

```
    i = 0
    j = 0

    match = False

    while(match == False and i<panjangstring):
        #while(i<panjangstring):
            if(contohtext[j] == contohstring[i]):
                if(j == panjangtext - 1):
                    #print(i - panjangtext + 1)
                    match = True
                    #print("match found")
                i+=1
                j+=1
            elif(j >0):
                j = fail[j-1]
                #print("disini1")
            else:
                i+=1
                #print("disini2")

    #if(match == True):
    #    print("ketemu :3")
    #else:
    #    print("nothing here")
    return(match)

    #return True
```

Fungsi untuk mengubah bulan dari string ke angka.

```
def month_string_to_number(string):
    #print(string)
    m = {
        'jan': '01',
        'feb': '02',
        'mar': '03',
        'apr':'04',
         'mei':'05',
         'jun':'06',
         'jul':'07',
         'agu':'08',
         'sep':'09',
         'okt':'10',
         'nov':'11',
```

```
        'des':'12'
        }
    s = string.strip()[:3].lower()
    try:
        out = m[s]
        return out
    except:
        raise ValueError('Not a month')
```

Fungsi untuk memproses tanggal menjadi tanggal sistem.

```
def prosesTanggal(tanggal):
    tanggalDisimpan = tanggal[0]
    if(re.match(re.compile("\d"),tanggal[1])):
        tanggalDisimpan = tanggalDisimpan + tanggal[1]
    if(len(tanggalDisimpan)==1):
        tanggalDisimpan = '0' + tanggalDisimpan
        tanggal = '0' + tanggal
    #print(tanggalDisimpan)
    # TANGGAL = tanggalDisimpan
    # print(tanggalDisimpan)
    # buat yang dalam huruf
    if(re.match(re.compile("[A-Z]",re.IGNORECASE),tanggal[2]) or
(re.match(re.compile("[A-Z]",re.IGNORECASE),tanggal[3]))):
        if(re.match(re.compile("[A-Z]",re.IGNORECASE),tanggal[2])):
            i = 2
        else:
            if (re.match(re.compile("[A-Z]",re.IGNORECASE),tanggal[3])):
                i = 3
        #print(tanggal[i]+tanggal[i+1]+tanggal[i+2])
        bulan =
(month_string_to_number(tanggal[i]+tanggal[i+1]+(tanggal[i+2]))) #
DAPAT BULANNYA
        postString = tanggal.split(" ",3)[2] # Asumsi kalau menggunakan format
bulan dengan huruf, hanya pake " "
        tahun = ""
        for x in re.finditer(re.compile("\d"),postString):
            tahun=tahun+((x.group())) # DAPAT TAHUNNYA
        #print(tahun,"Ini tahun")
    else:
        postString = tanggal.split(tanggalDisimpan,2)[1]
        bulanTahun = postString[1:]
        #print(bulanTahun)
        bulan = ""
        i=0
        while(re.match(re.compile("\d"),bulanTahun[i])):
```

```
            bulan = bulan+bulanTahun[i]
            i=i+1
        if(len(bulan)==1):
            bulan = "0"+bulan
            bulanTahun = "0"+bulanTahun
        #print(bulan)
        tahun=bulanTahun.split(bulan,2)[1][1:]
        #print(tahun)
    if(len(tahun)==2): # tidak ada awalan 20
        tahun = "20" + tahun
    return str(tanggalDisimpan) + " " + str(bulan) + " " + str(tahun)
```

Fungsi untuk mencari kecocokan tanggal.

```
def readDate(input):
    ## ada 3 pilihan input
    ## 15/04/2021 atau 15-04-2021
    ## 14 April 2021
    ## 22/04/21 atau 22-04-2021
    #print(input)
    pattern =
re.compile("[0-9]+(\s|/|-)+(((januari|februari|maret|april|mei|juni|juli|agustus|s
eptember|november|desember))|[0-9])+(\s|/|-)+[0-9]+[0-9]",re.IGNORECAS
E)
    zTotal = re.finditer(pattern,input)
    tanggal=[]
    for x in re.finditer(pattern,input):
        tanggal.append((x.group()))
    if(len(tanggal)>0):
        return prosesTanggal(tanggal[0])
    else:
        return None
```

Fungsi untuk mendeteksi kode kuliah.

```
def deteksiKodeKuliah(input):
    z =
re.findall(re.compile("[a-z|A-Z]+[a-z|A-Z]+[0-9]+[0-9]+[0-9]+[0-9]"),input)
    if(len(z)>0):
        return z[0]
    else:
        return None
```

Fungsi untuk menambahkan tugas.

```
def addTask(ID, tanggal,kode,jenis,topik):
    tanggal = tanggal.split(" ")
```

```
    data_deadline = []
    data_deadline.append((str(ID) + " "+ str(tanggal[0]) + " " + str(tanggal[1])
+ " " + str(tanggal[2]) + " " + str(kode) + " " + str(jenis) + " " + str(topik)))
    listOfDeadlinesComponent.append(data_deadline)
    return "Berhasil menambahkan task"
```

Fungsi untuk menampilkan seluruh *deadline*.

```
def tampilkanSeluruhDeadline():
    deadline = ""
    for i in listOfDeadlinesComponent:
        #print(i)
        deadline=deadline + str(i) + "\n"
    return deadline
```

Fungsi untuk mendeteksi seluruh *deadline* yang memiliki suatu kata penting.

```
def deadlineKataPenting(input):
    katapenting = deteksiKataPenting(input)
    arrayOfDeadline=[]
    for i in listOfDeadlinesComponent:
        if(i[5].lower()==katapenting):
            arrayOfDeadline.append(i)
    return arrayOfDeadline
```

Fungsi untuk menampilkan *deadline*, disortir apakah mencari beberapa minggu dari sekarang, beberapa hari dari sekarang, atau menampilkan seluruh *deadline*, atau *deadline* suatu kata penting, atau *deadline* antara dua tanggal.

```
def tampilkanDeadlines(input):
        arraystring = input.split(' ')
        minggu = False
        hari = False
        for i in arraystring:
            if(i=="minggu" or i=="hari"):
                if(kmpstringmatching(i,"minggu")):
                    z=re.findall(re.compile("[0-9]+\s+minggu"),input)
                    if(len(z)>0):
                        banyakminggu=re.match(re.compile("[0-9]"),z[0]).group() #
cari angka numerik
                        minggu = True
                        break
                elif(kmpstringmatching(i,"hari")):
                    z=re.findall(re.compile("[0-9]+\s+hari"),input)
                    if(len(z)>0):
```

```
                banyakhari=re.match(re.compile("[0-9]"),z[0]).group() # cari
angka numerik
                hari = True
                break
    if (readDate(input)==None and minggu==False and hari==False): #
kalo misalnya ga ada tanggal
        if(deteksiKataPenting(input)==None):
            return tampilkanSeluruhDeadline()
        else:
            return deadlineKataPenting(input)
    else:
        if(len(readAdaBerapaTanggal(input))==2):
            x=readAdaBerapaTanggal(input)
            return deadlinesBetween(listOfDeadlines,x[0],x[1])
        else:
            if (minggu):
              if(banyakminggu!=None):
                 return deadlineFromNow(int(banyakminggu)*7)
            elif (hari):
              if(banyakhari!=None):
                 return deadlineFromNow(int(banyakhari))
            else:
              output=""
              for i in listOfDeadlinesComponent:
                 y=readDate(input)
                 x=readAdaBerapaTanggal(y)
                 if(str(i[1])+" "+str(i[2])+" "+str(i[3]) in x):
                    return output + str(i)
    return "Pesan gagal diproses"
```

Fungsi menanyakan *deadline* suatu kode mata kuliah.

```
def tanyakanDeadline(kode):
    semua=[]
    for i in listOfDeadlinesComponent:
        if(i[4].lower()==kode):
            semua.append((i[1]+" "+i[2]+" "+i[3]))
    return semua
```

Fungsi memperbaharui tugas.

```
def perbaharuiTask(IDdicari,tanggalBaru):
    #print(tanggalBaru)
    tanggalBaru=tanggalBaru.split(" ")
    for i in listOfDeadlinesComponent:
        if(i[0].lower()==IDdicari):
            #print("ada")
```

```
            print("X")
            i[1]=tanggalBaru[0]
            i[2]=tanggalBaru[1]
            i[3]=tanggalBaru[2]
            return "List berhasil diperbaharui"
    return "Gagal, tidak ada ID"
```

Fungsi untuk menghapus tugas.

```
 def deleteTask(IDdicari):
    for i in listOfDeadlinesComponent:
        if(i[0].lower()==IDdicari):
            listOfDeadlinesComponent.remove(i) # ? wkwk tergantung datanya
            return "List berhasil diperbaharui"
    return "Pesan gagal"
```

Fungsi untuk *help*.

```
def help():
    output=""
    katapentingstr = ""
    listfitur = " 1. Menambahkan task baru [#katakunci] \r\n 2. Melihat daftar
task yang harus dikerjakan [#katakunci] \r\n 3. Menampilkan deadline dari
suatu task tertentu [#katakunci]\r\n 4. Memperbaharui task tertentu
[#katakunci]\r\n 5. Menandai suatu task sudah selesai [#katakunci] \r\n 6.
Menampilkan opsi help [katakunci]\r\n "
    for i in katapenting:
        katapentingstr = katapentingstr + str(i) + "\r\n"
    output="[Fitur] " + "\r\n" + listfitur + "\r\n" + "[Daftar kata penting] " +
"\r\n" + katapentingstr
    return output
```

Fungsi untuk mendeteksi kata penting.

```
def deteksiKataPenting(input):
    #katapenting=readFile('katapenting.txt')
    arraystring = input.split(' ')
    jenis=None
    for j in arraystring:
        if j in katapenting:
            jenis=j
    return jenis
```

Fungsi untuk mendeteksi apakah ada keyword perintah.

```
def deteksiPerintah(input):
    arraystring = input.split(' ')
    valid=False
```

```
    for j in arraystring:
        if j in keyword_perintah:
            valid=True
            return valid
    return valid
```

Fungsi untuk mendeteksi apakah ada keyword topik.

```
def deteksiKeywords(input):
    keywordss=readFile('keywords.txt')
    arraystringg = input.split(' ')
    keyword=None
    for j in arraystringg:
        if j in keywordss:
            keyword=j
    return keyword
```

Fungsi apakah ada yang memenuhi KMP.

```
def carikmpgak(input,sesuatu):
    #katapenting=readFile('katapenting.txt')
    #contohstring = "aku mau mau mau mau banget tucil"
    arraystring = input.split(' ')
    #print(arraystring)

    #print(arraystring)
    #contohtext = "tucil"
    jenis=False

    for j in arraystring:
        #print(j)
        #print(sesuatu)
        if(kmpstringmatching(j,sesuatu)):
            jenis=True
            break
    return jenis
```

Fungsi untuk mengeksekusi semua pilihan *input* pengguna.

```
def pilihanInput(input): # Masuk ke fitur sesuai masukan pengguna
    input = Stopword(input)
    input = input.lower()
    arraystring=input.split(' ')
    IDdibaca=False
    kode=False
    returnValue=""
    for i in arraystring:
```

```
        if deteksiKodeKuliah(i)!=None:
            kode=True
            kodeDibaca=i
        pattern = re.compile("l[0-9][0-9][0-9][0-9]")
        if len(re.findall(pattern,input))>0:
            IDdibaca = True
            IDdicari = re.findall(pattern,input)[0]
    if (deteksiPerintah(input)==False):
        if (kode==True and deteksiKataPenting(input)!=None and
deteksiKeywords(input)!=None and tanggal==True):
            if(len(listOfDeadlinesComponent)>=10):
                if(len(listOfDeadlinesComponent)>=100):
                    if(len(listOfDeadlinesComponent)>=1000):
                        ID="L"+str(len(listOfDeadlinesComponent)+1)
                    else:
                        ID="L0"+str(len(listOfDeadlinesComponent)+1)
                else:
                    ID="L00"+str(len(listOfDeadlinesComponent)+1)
            else:
                ID="L000"+str(len(listOfDeadlinesComponent)+1)
            tanggal=tanggalDibaca
            kode=kodeDibaca
            kataPenting=deteksiKataPenting(input)
            topik=deteksiKeywords(input)
            return (addTask(ID, tanggal,kode,kataPenting,topik))
        else:
            return "Maaf, pesan tidak dikenali"
    else:
        if(kode!=True and carikmpgak(input,"fitur")):
            return help()
        elif((IDdibaca!=True or kode!=True) and
carikmpgak(input,"tampilkan")):
            return tampilkanDeadlines(input)
        elif(kode==True and carikmpgak(input,"kapan")):
            #print(kodeDibaca)
            return tanyakanDeadline(kodeDibaca)
        elif(IDdibaca==True and readDate(input)!=None and
carikmpgak(input,"diundur")):
            #print(readDate(input))
            tanggalDibaca=readDate(input)
            return perbaharuiTask(IDdicari,tanggalDibaca)
        elif(IDdibaca==True and carikmpgak(input,"selesai")):
            return deleteTask(IDdicari)
    return "GAGAL"
```

Fungsi untuk menyimpan *deadline* yang sudah tercatat.

```
def saveDeadlinesComponent() :
    data = listOfDeadlinesComponent
    with open("deadline.txt", "w") as txt_file:
        for line in data:
            txt_file.write(" ".join(line) + "\n")
```

Menghubungkan ke *flask*.

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


@app.route('/', methods=['GET', 'POST'])
def upload_all():
    if request.method == 'POST':
        name = request.form['name']
        listOfChat = {}
        global listOfChatAll
        listOfChat[0]=name
        x = getChatBotResponse(name)
        listOfChat[1]=x
        listOfChatAll.append(listOfChat)
        saveDeadlinesComponent()
        return
render_template('showmessage.html',listOfChatAll=listOfChatAll,listOfChat
=listOfChat)

def getChatBotResponse(name):
    return pilihanInput(name)

if __name__ == "__main__":
    app.run(threaded=True)
```

4.2. Penjelasan tata cara penggunaan program (interface program, fitur yang disediakan, dan sebagainya)

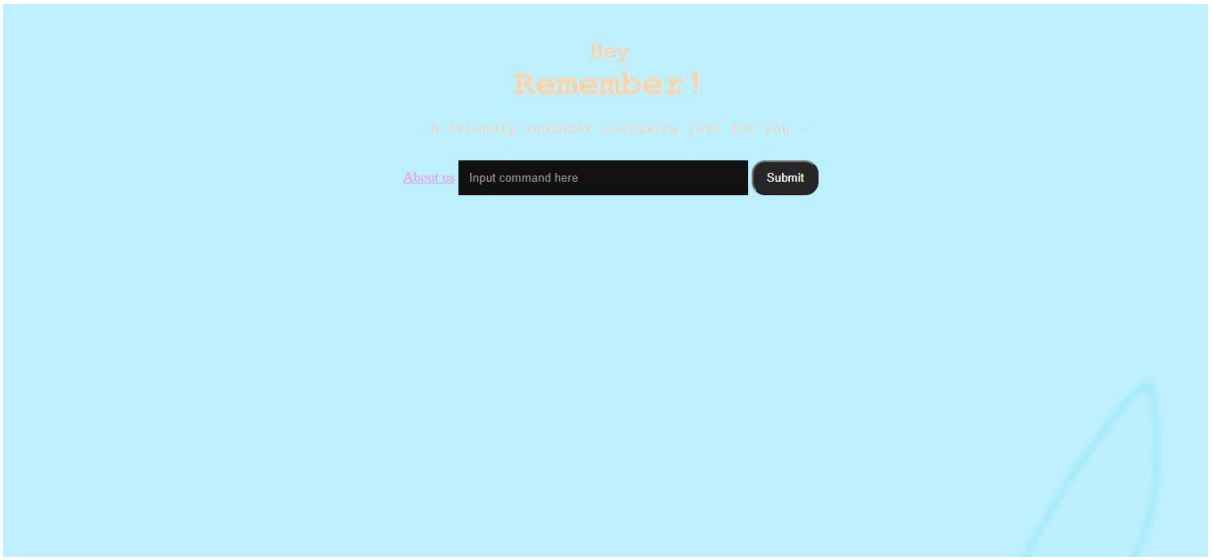
Program yang kelompok kami buat dapat dijalankan dengan menginstall library tambahan yang digunakan, yaitu flask, wtforms, werkzeug.utils, dan sastrawi. Penginstallan library dapat menggunakan “pip install <nama library>”. Tidak lupa untuk memastikan telah menginstall python pada perangkat masing - masing. Program

ini dapat dijalankan dengan menggunakan syntax “python chatbot.py” atau “python3 chatbot.py”. Setelah program dijalankan, kita dapat menggunakan chatbot dengan menuliskan kata kunci pada input box yang ada pada tampilan awal dan menekan tombol submit untuk menjalankan program.

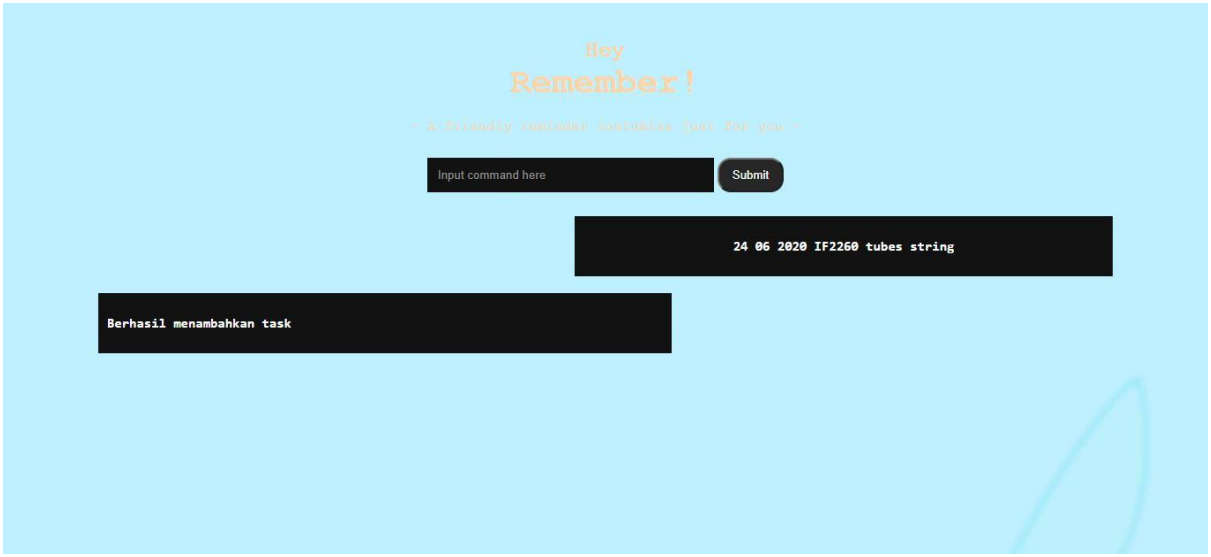
Selain dapat menjalankan chatbot, kita juga dapat mengakses laman “About us” dengan mengklik hyperlink yang ada pada tampilan program. Program ini menyediakan fitur untuk menambahkan task baru, melihat daftar task yang harus dikerjakan, menampilkan deadline dari suatu task tertentu, memperbaharui task tertentu, dan menampilkan opsi help.

4.3. Hasil pengujian

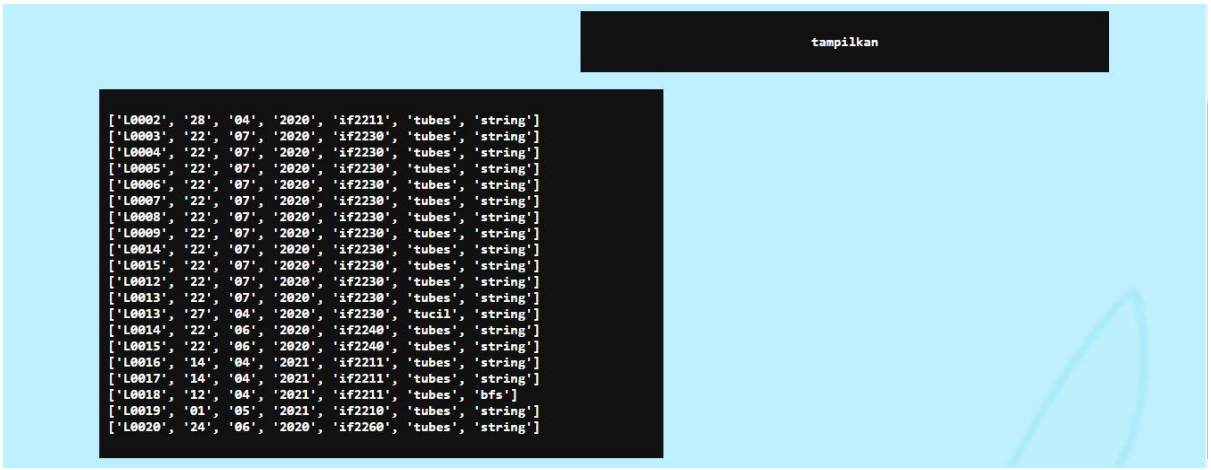
Tampilan awal chat bot



Menambahkan tugas

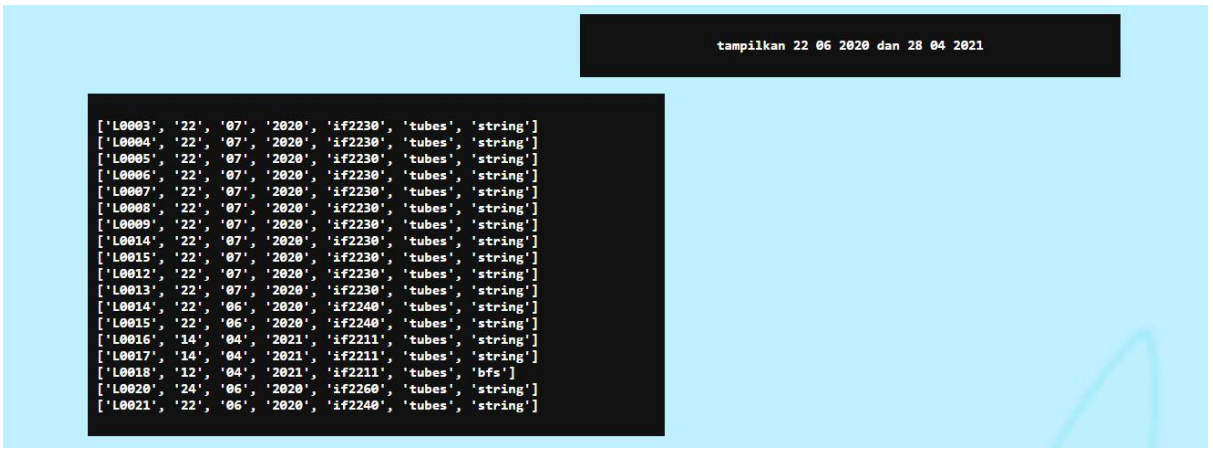


Menampilkan semua deadline



Menampilkan semua fitur yang tersedia pada chatbot

Menampilkan tugas antara 2 tanggal

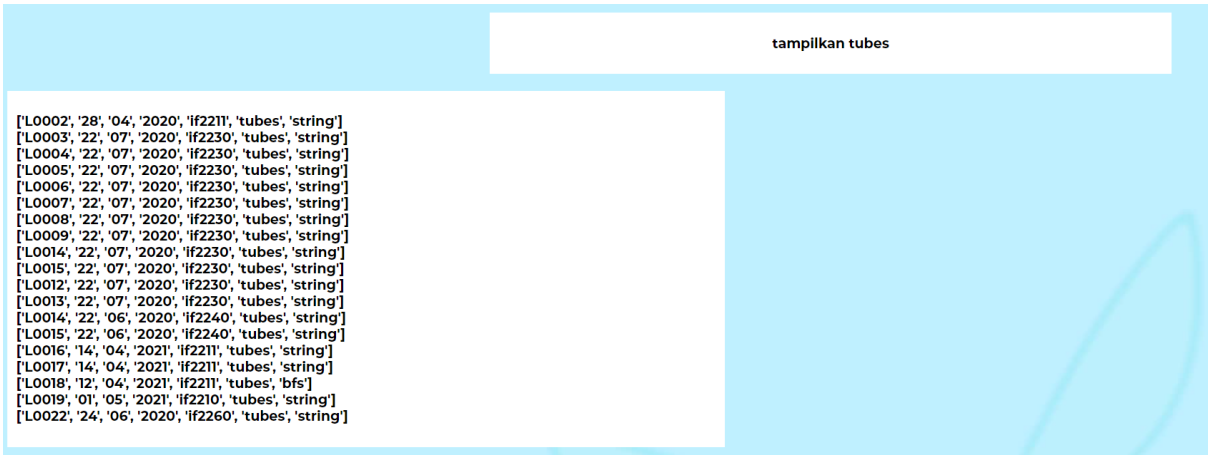


Menampilkan tugas apa saja dalam 1 hari

Menghapus tugas yang telah selesai dikerjakan

Mengundur tanggal suatu tugas

Menampilkan semua deadline dengan jenis tugas tertentu



Menampilkan seluruh deadline pada mata kuliah tertentu

4.4. Analisis hasil pengujian

Program chatbot kita dapat melakukan beberapa fitur, yang pertama adalah menambahkan task baru dengan memasukan kata kunci “<tanggal> <kata penting> <topik>”. Berdasarkan hasil pengujian yang dilakukan, dengan memasukkan input 24 06 2020 IF2260 string, program dapat mengembalikan pesan bahwa task berhasil ditambahkan dan hal tersebut diperkuat dengan saat menampilkan seluruh task yang ada, task yang baru telah ditambahkan. Namun, karena ada dua angka pada kata kunci yang dimasukkan, yaitu pada kode matakuliah dan tanggal, program akan salah mendeteksi tanggal ketika penulisan input dimulai dengan kode mata kuliah terlebih dahulu.

Fitur kedua adalah menampilkan seluruh task yang ada dengan memasukkan kata kunci “tampilkan”. Berdasarkan dari hasil pengujian yang telah dilakukan, program mampu menampilkan seluruh task yang ada dengan baik.

Fitur berikutnya adalah menampilkan semua fitur yang dengan memasukkan kata kunci “fitur”. Fitur ini dapat digunakan dengan baik seperti yang dapat dilihat pada hasil pengujian.

Fitur keempat adalah memperbaharui *task* dengan kata kunci “diundur”, dengan ID . Fitur ini dapat digunakan dengan baik, tetapi masih terbatas karena hanya tanggal yang dapat diperbaharui.

Fitur kelima adalah menampilkan tugas apa saja dalam satu hari. Fitur ini dapat digunakan dengan memasukkan input berupa “tampilkan 1 hari”. Seperti yang dapat dilihat pada hasil pengujian, program mampu menghasilkan hasil yang baik dengan menampilkan seluruh informasi tugas pada 1 hari.

Fitur berikutnya adalah menghapus tugas yang sudah selesai. Fitur ini akan menghapus deadline yang sudah selesai dikerjakan saat pengguna memasukkan input “<id deadline> selesai”. Berdasarkan hasil pengujian, fitur ini dapat dijalankan dengan baik dengan program menampilkan pesan berupa list berhasil diperbaharui.

Fitur ketujuh adalah menampilkan semua deadline dengan jenis task tertentu. Fitur dapat digunakan dengan memasukkan input berupa “tampilkan <jenis task>”. Fitur ini dapat dijalankan dengan baik, seperti yang dapat dilihat pada hasil pengujian.

Fitur yang terakhir adalah menampilkan seluruh deadline pada mata kuliah tertentu, yaitu yang dapat digunakan dengan memasukkan input “<kode mata kuliah> kapan”. Berdasarkan hasil pengujian yang dilakukan, fitur ini dapat dijalankan dengan baik.

Dari hasil pengujian yang kami lakukan, program ini masih memiliki kekurangan, yaitu pada tampilan hasil pencarian berdasarkan kata kunci. Tampilan yang ditampilkan masih dalam bentuk array, bukan sebuah kalimat utuh.

BAB V
KESIMPULAN

5.1. Kesimpulan

Program chatbot yang kelompok kami buat mampu menghasilkan tampilan yang sesuai dengan input user dengan menggunakan kata kunci tertentu.

5.2. Saran

Semoga chatbot ini dapat dikembangkan lebih lanjut lagi dengan menambahkan berbagai fitur lainnya serta membuat tampilan chatbot lebih menarik lagi.

5.3. Komentar

Tugas Besar ini sangat menarik karena aplikasi ini sangat dibutuhkan oleh mahasiswa IF yang banyak sekali deadline namun mahasiswanya juga sangat deadliner, sehingga membutuhkan reminder tugas seperti ini.

BAB VI

DAFTAR PUSTAKA

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