

Laporan Tugas Besar Strategi Algoritma – IF 2211

Penerapan *String Matching* dan *Regular Expression* dalam
Pembangunan *Deadline Reminder Assistant*

Semester II Tahun 2020/2021



Disusun oleh:

Reinaldo Antolis	13519015
Jeane Mikha Erwansyah	13519116
Josep Marcello	13519164

PROGRAM STUDI TEKNIK INFORMATIKA
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG
2020

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.

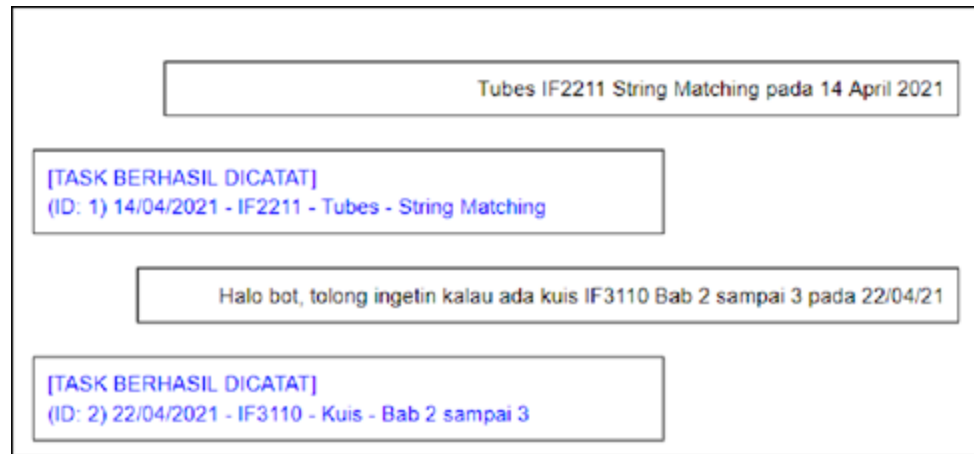
Fitur-Fitur Aplikasi:

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
 - 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



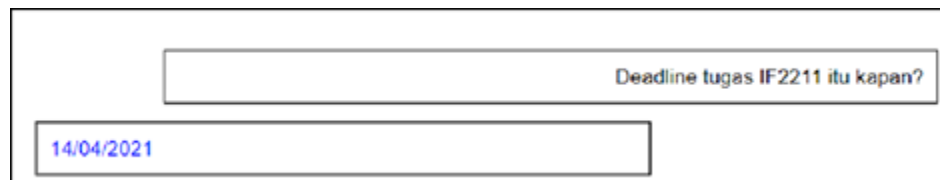
2. Melihat daftar *task* yang harus dikerjakan
 - a. Seluruh *task* yang sudah tercatat oleh assistant
 - b. Berdasarkan periode waktu
 - i. Pada periode tertentu (DATE_1 until DATE_2)
 - ii. N minggu ke depan
 - iii. N hari ke depan
 - iv. Hari ini
 - c. Berdasarkan jenis task (kata penting)
 - i. Sesuai dengan daftar task yang didefinisikan
 - ii. User dapat melihat daftar task dengan jenis 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.

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

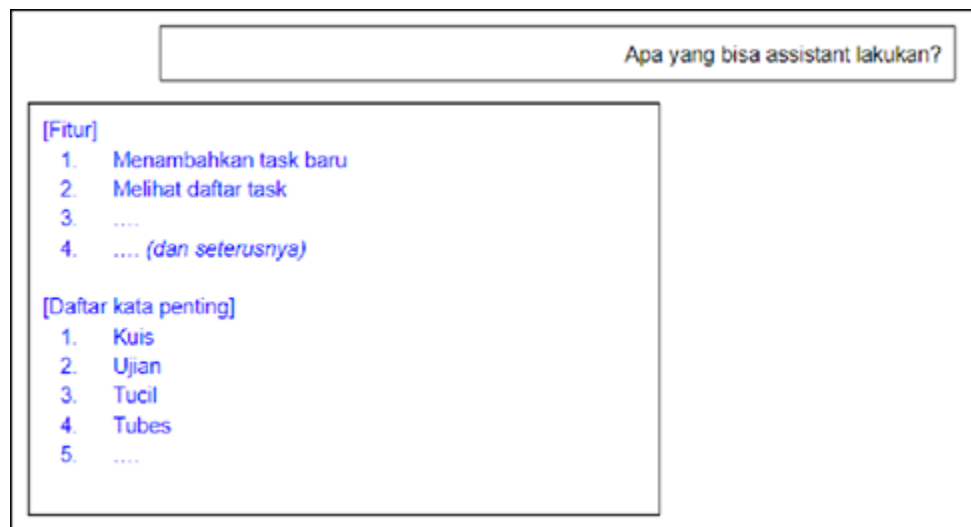
- Apa saja **deadline** antara **03/04/2021** sampai **15/04/2021**?
- **03/04/2021** dan **15/04/2021** ada **deadline** apa saja ya?

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?”



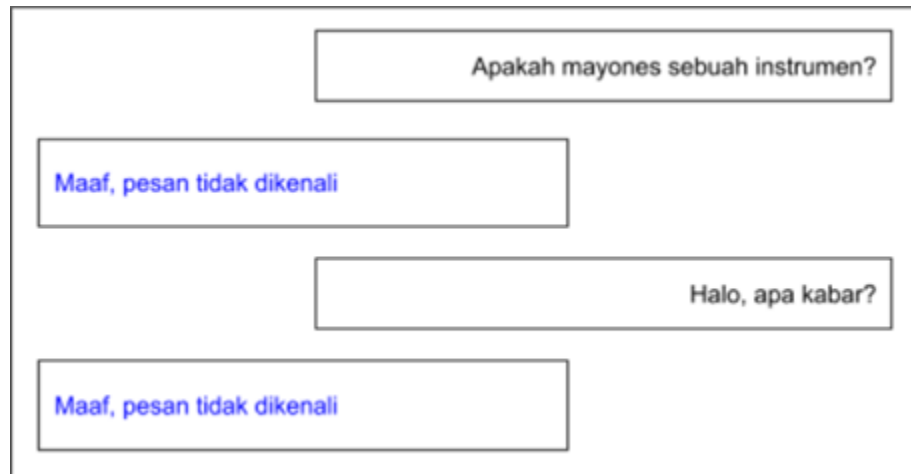
4. Memperbaharui task tertentu

- 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. Apabila task berhasil diperbaharui, Chatbot akan menampilkan pesan sukses memperbaharui suatu task. Sebaliknya, Chatbot akan menampilkan pesan error apabila task yang dimaksud tidak dikenali oleh Chatbot (belum masuk ke dalam Daftar Task)
5. Menandai bahwa suatu task sudah selesai dikerjakan
 - a. Apabila user sudah menyelesaikan suatu task, maka task tersebut bisa ditandai bahwa task tersebut sudah selesai dan tidak perlu lagi ditampilkan pada Daftar Task selanjutnya.
 - b. Apabila perintah yang dimasukkan user bisa dieksekusi, Chatbot akan menampilkan pesan sukses. Sebaliknya, Chatbot akan menampilkan pesan error apabila task yang dimaksud tidak dikenali oleh Chatbot (belum masuk ke dalam Daftar Task)
6. Menampilkan opsi *help* yang difasilitasi oleh assistant
 - a. Berisikan command-command yang dapat digunakan oleh user
 - b. 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

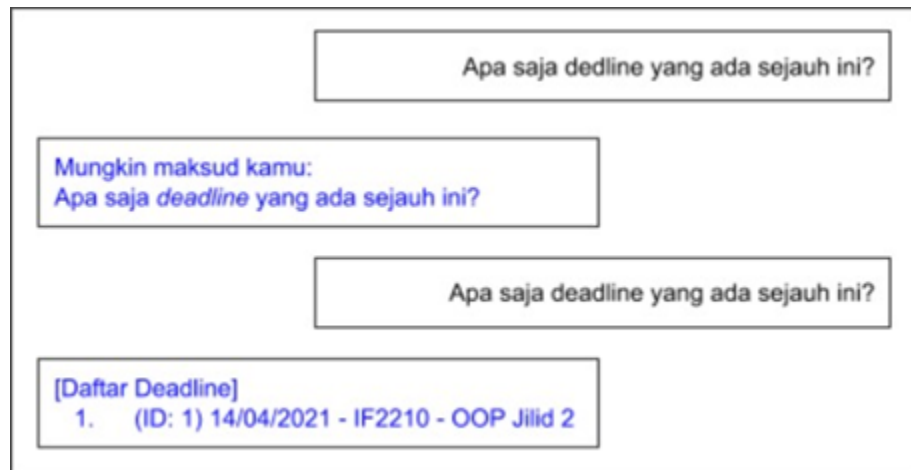


7. Mendefinisikan list kata penting terkait apakah itu merupakan suatu *task* atau tidak
 - a. Minimal terdapat 5 kata penting berbeda, contohnya adalah: ["Kuis", "Ujian", "Tugil", "Tubes", "Praktikum"]
 - b. Kata penting akan digunakan pada penentuan jenis tugas dari suatu *task*.
 - c. Daftar kata 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



- 9. **(Bonus)** Chatbot dapat memberikan rekomendasi kata jika terdapat kesalahan kata (*typo*) pada perintah yang ditulis pengguna
 - a. Berikan rekomendasi kata jika perintah masukan pengguna *mismatch* dengan daftar kata yang diterima chatbot, namun masih memiliki tingkat kemiripan di atas 75%.
 - b. Contoh interaksi



- c. 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 Knutt-Morris-Pratt (KMP)

Algoritma Knutt-Morris-Pratt (KMP) merupakan algoritma pencocokan string yang mencari pola kata dalam teks dengan penentuan pergeseran pencocokan pola kata dalam teks ketika terjadi ketidakcocokan pola dapat dilakukan dengan lebih efisien sedemikian rupa sehingga pencocokan pola kata dalam teks yang berikutnya dapat menghindari pencocokan yang telah dilakukan sebelumnya.

Contoh:

Awal:

Teks	a b b a b b b x
Pola	a b b b

Setelah pergeseran:

Teks	a b b a b b b x
Pola	a b b b

2.2 Algoritma Boyer-Moore (BM)

Algoritma Boyer-Moore (BM) merupakan algoritma pencocokan string efisien yang memanfaatkan dua teknik, yaitu teknik *looking-glass* dan *character-jump*. Teknik *looking-glass* melakukan pencarian pola P di dalam teks T dengan cara mencocokkan P dari bagian kanan. Teknik *character-jump* melakukan pergeseran karakter berdasarkan kondisi-kondisi tertentu yang dibagi menjadi tiga kasus.

2.3 Regular Expression (Regex)

Regular expression merupakan teknik yang dikembangkan dalam ilmu komputer teoritis dan teori bahasa formal. Regular expression (biasanya disingkat regex) merupakan sederetan karakter yang menentukan suatu pola pencarian dan biasanya dipakai dalam algoritma pencocokan string.

2.4 *Chatbot*

Chatbot merupakan perangkat lunak yang dapat melakukan percakapan seperti manusia dengan pengguna melalui pesan teks pada *chat*. Peran utama dari *chatbot* adalah untuk membantu pengguna dengan menyediakan tanggapan berdasarkan pertanyaan pengguna.

BAB III

ANALISIS PEMECAHAN MASALAH

3.1 Langkah-Langkah Pemecahan Masalah

1. Menambahkan *task* baru
 1. Merima input dari pengguna.
 2. Melakukan pencocokan kata dengan komando yang lain.
 3. Melakukan regex untuk mencari *deadline*, kode mata kuliah, topik, dan jenis *task*.
 - a. *Deadline*: regex dilakukan dengan menggunakan fungsi `extract_date(msg:str)`.
 - b. Kode mata kuliah: regex dilakukan dengan mencari 2 huruf yang diakhiri dengan 4 angka.
 - c. Topik: regex dilakukan dengan mengambil kata-kata yang berada di antara dua tanda kutip (“”).
 - d. Jenis *task*: regex dilakukan dengan mencari kata yang sesuai dengan jenis-jenis *task* yang ada.
 4. Jika ada semua kriteria yang dibutuhkan maka *backend* akan menyimpan data *task* baru ke *database* dan *frontend* akan menampilkan pesan berisi id serta input pengguna yang telah di-regex. Sebaliknya jika ada satu atau lebih kriteria yang tidak dipenuhi maka *frontend* akan menampilkan pesan gagal.
2. Melihat daftar *task*
 1. Menerima input dari pengguna
 2. Melakukan pencocokan dengan kata kunci: “apa aja” atau “apa saja”.
 3. Jika ditemukan kata kunci pada masukan pengguna, maka akan di-*trigger* fitur melihat *task*
 4. Tentukan jenis tugas apa yang pengguna minta. Jika tidak ditentukan, artinya pengguna meminta semua
 5. Dilakukan pencarian kata kunci: “dari” dan “antara” lalu “hingga” dan “sampai.”
 - a. Jika ditemukan pasangan dari pencarian pertama dan kedua, dilanjutkan pencarian tanggal, tapi jika hanya ditemukan satu saja maka dianggap tidak mencari dari tanggal,
 - b. Jika hanya ditemukan satu tanggal tapi dua kata kunci menentukan jarak tanggal, maka akan dikeluarkan pesan bot gagal memproses.
 - c. jika ditemukan dua tanggal akan dilakukan pemeriksaan pada tanggal, selain itu tampilkan pesan bot gagal memproses pesan pengguna.
 - d. Jika tanggal kedua lebih kecil dari tanggal pertama, maka tampilkan pesan bot gagal memproses pesan pengguna.

6. Jika tidak ada, maka dicari kata kunci “ke depan”, “berikutnya”, “selanjutnya”, atau “lagi” pada masukan pengguna.
 - a. Jika ada, akan dicari waktu penanda durasi, yaitu “hari”, “minggu”, “bulan”, atau “tahun.”
 - b. Jika tidak ditemukan maka akan dikeluarkan pesan gagal memproses *query*.
 - c. Jika berhasil maka akan dihitung durasinya.
7. Selanjutnya akan ditentukan pengguna ingin kategori suatu tugas tertentu atau tidak. Caranya dengan mencari pola kata “tubes,” “tucil,” “tugas besar,” “tugas kecil,” “praktikum,” “prak,” “uas,” “uts,” atau “latihan.” Jika tidak ada artinya pengguna mencari semua jenis tugas.
8. Selanjutnya akan diiterasi semua jenis tugas yang ada
9. Setiap tugas akan diperiksa deadline dan jenisnya, jika sesuai dengan kriteria pengguna maka akan ditambahkan ke daftar tugas yang akan ditunjukkan.
3. Menampilkan *deadline* dari suatu task tertentu
 1. Menerima input dari pengguna
 2. Cari ID mata kuliah dari masukan pengguna
 3. Cari jenis tugas dari pengguna
 4. Jika ID atau jenis tugas tidak ditemukan, tampilkan pesan gagal memproses permintaan ke pengguna
 5. Jika ditemukan, iterasi semua tugas lalu dicari tugas yang ID-nya sesuai dan jenisnya sesuai lalu tambahkan ke daftar tugas yang akan ditampilkan
4. Memperbaharui *task* tertentu
 1. Menerima input dari pengguna
 2. Melakukan pencocokan kata penting “ubah”, “diundur”, “ganti” atau “jadi” dan pengecekan keberadaan ID tugas serta tanggal pada string input pengguna
 3. Bila semua syarat pada poin 2 dipenuhi, dilakukan pencarian ID tugas dari input pengguna pada database tugas. Bila ditemukan, deadline tugas di database diperbaharui sesuai dengan tanggal pada input pengguna. Bila tidak, dikembalikan pesan tugas tidak terdapat dalam daftar tugas.
 4. Bila salah satu syarat pada poin 2 tidak dipenuhi, maka dikembalikan pesan bot tidak dapat mengenali input pengguna
5. Menandai bahwa suatu *task* sudah selesai dikerjakan
 1. Menerima input dari pengguna
 2. Melakukan pencocokan kata penting “selesai” dan pengecekan keberadaan ID tugas pada string input pengguna
 3. Bila kedua syarat pada poin 2 dipenuhi, dilakukan pencarian ID tugas dari input pengguna pada database tugas. Bila ditemukan, tugas ditandai sebagai selesai di database (*record* dihapus dari database). Bila tidak, dikembalikan pesan tugas tidak terdapat dalam daftar tugas.

4. Bila salah satu syarat pada poin 2 tidak dipenuhi, maka dikembalikan pesan bot tidak dapat mengenali input pengguna
6. Menampilkan opsi *help* yang difasilitasi oleh *chatbot*
 1. Menerima input dari pengguna
 2. Melakukan pencocokan kata penting “help”, “tolong”, “tasukete” atau “助けて” pada string input pengguna
 3. Bila pencocokan kata penting pada poin 2 berhasil, dikembalikan pesan berisikan fitur-fitur dan list kata penting yang diterima chatbot
7. Memberikan rekomendasi kata ketika pengguna melakukan kesalahan pengetikan kata
 1. Menerima input dari pengguna.
 2. Melakukan *matching* antara masukan pengguna dengan daftar kata kunci/*triggers* untuk fitur-fitur lainnya.
 3. Jika masukan pengguna tidak ada yang men-*trigger* fitur lainnya, maka akan digunakan perhitungan jarak Levenshtein untuk mencocokkan semua kata milik pengguna dengan semua *trigger* yang mungkin.
 4. Jika kemiripan Levenshtein-nya lebih besar dari atau sama dengan 60%, maka *trigger* akan direkomendasikan ke pengguna.

3.2 Fitur Fungsional dan Arsitektur *Chatbot*

3.2.1 Fitur Fungsional

Berikut merupakan fitur fungsional dari *chatbot* Bot Wangy:

1. Menambahkan tugas baru
2. Melihat daftar tugas yang harus dikerjakan
3. Menampilkan deadline dari suatu tugas tertentu
4. Memperbarui deadline tugas tertentu
5. Menandai suatu tugas sudah selesai dikerjakan
6. Menampilkan opsi *help* yang difasilitasi oleh assistant
7. Merekomendasikan kata ketika pengguna melakukan saltik

3.2.2 Arsitektur

3.2.2.1 Frontend

Frontend pada *chatbot* Bot Wangy memanfaatkan kerangka kerja Svelte.js untuk mempermudah pemrograman. Kode utama untuk *frontend* terdapat pada berkas App.svelte. Berkas ini terhubung dengan index.html yang merupakan perantara antara kode pada App.svelte dengan peramban internet pengguna

3.2.2.2 Backend

Backend pada *chatbot* Bot Wangy menggunakan bahasa pemrograman *python*. Terdapat tiga modul dalam backend *chatbot* ini, yaitu:

- a. app.py

Berisi main program yang menerima input pengguna dan melakukan pencocokan kata penting dengan input pengguna untuk menentukan command yang ingin diberikan pengguna.

b. `response.py`

Berisi fungsi-fungsi yang memberikan tanggapan terhadap command dari input pengguna.

c. `matching.py`

Berisi fungsi-fungsi yang menggunakan algoritma pencocokan string.

3.2.2.3 Database

Database pada *chatbot* Bot Wangy memanfaatkan Google Firestore *database*. Pada database tersebut terdapat dua *collection*, yaitu 'keywords' yang menyimpan seluruh kata penting yang digunakan untuk mengklasifikasikan perintah/*query* dari pengguna dan 'tugas' yang menyimpan *record* semua tugas yang akan diingatkan ke pengguna.

BAB IV

IMPLEMENTASI DAN PENGUJIAN

4.1 Spesifikasi Teknis Program

4.1.1 Struktur Data

Berikut merupakan struktur data yang digunakan pada Google Firestore *database*:

a. Collection: Keywords

Menyimpan kata penting

- jenis_tugas: “tubes”, “tucil”, “kuis”, “praktikum”, “uts”, “uas”, “latihan”
- lihat_deadline: “kapan”
- lihat_help: “help”, “tolong”, “助けて”, “tasukete”
- lihat_task: “apa saja”, “apa aja”
- nandain_task_selesai: “selesai”
- tambah_task: “ingatkan”, “tambah”
- update_task: “ubah”, “diundur”, “ganti”, “jadi”

b. Collection: Tugas

Menyimpan tugas yang akan diingatkan ke pengguna

Field:

- deadline: timestamp. Mencatat waktu deadline tugas
- id_matkul: string. Mencatat kode mata kuliah
- jenis: string. Mencatat jenis tugas
- topik: string. Mencatat topik tugas

Tugas yang sudah diselesaikan dihapus dari *database*

4.1.2 Fungsi dan Prosedur

- `respond()`
Fungsi untuk menerima dan mengirimkan respon input pengguna
- `find_last_occurance(s: str) -> 'dict[str, int]'`
Fungsi untuk mendapatkan kemunculan terakhir dari semua karakter pada sebuah string
- `boyer_moore(text: str, pattern: str) -> int`
Fungsi untuk pencocokan string dengan algoritma Boyer-Moore
- `levenshtein_distance(s1: str, s2: str) -> int`
Fungsi untuk menghitung jarak levenshtein antara 2 string
- `extract_date(msg:str) -> 'list[datetime]'`
Fungsi untuk mengekstrak tanggal dari string input pengguna dengan memanfaatkan regex

- `extract_jenis(msg: str, db) -> str`
Fungsi untuk mengekstrak jenis tugas dari string input pengguna dengan memanfaatkan regex
- `extract_course_id(msg: str) -> str`
Fungsi untuk mengekstrak kode mata kuliah dari string input pengguna dengan memanfaatkan regex
- `extract_topic(msg: str) -> str`
Fungsi untuk mengekstrak topik tugas dari string input pengguna dengan memanfaatkan regex
- `extract_task_id(msg: str) -> str`
Fungsi untuk mengekstrak ID tugas dari string input pengguna dengan memanfaatkan regex
- `load_keywords(db) -> 'dict[str, list[str]]'`
Fungsi untuk loading keywords dari *database*
- `lihat_tugas(msg: str, db) -> str`
Fungsi untuk mendapatkan list tugas dari *database*
- `lihat_deadline(msg: str, db) -> str`
Fungsi untuk mendapatkan deadline tugas-tugas dari suatu mata kuliah
- `tambah_tugas(msg: str, db) -> str`
Fungsi untuk menambahkan list tugas ke *database*
- `update_tugas(msg: str, db) -> str`
Fungsi untuk memperbaharui deadline suatu tugas di *database*
- `clear_tugas(msg: str, db) -> str`
Fungsi untuk menandakan suatu tugas telah selesai dikerjakan di *database*
- `handle_bingung()`
Fungsi untuk mengirimkan pesan bahwa *chatbot* tidak dapat mengenali command di input pengguna
- `help_msg(db) -> str`
Fungsi untuk mengembalikan pesan yang menampilkan seluruh command dan kata penting yang dapat digunakan oleh pengguna “help”, “tolong”, “助けて”, atau “tasukete”.

4.2 Tata Cara Penggunaan Program

4.2.1 Interface Program

Interface program berbentuk aplikasi berbasis *web messenger* dengan beberapa *chat bubble* di sebelah kiri dan kanan aplikasi yang berisi pesan oleh *chatbot* dan pengguna. Pengguna mengirimkan pesan yang berisikan command untuk *chatbot* dengan mengisi *textbox* dan menekan tombol “Enter” pada papan ketik atau tombol *send* pada aplikasi. Kemudian, *chatbot* akan mengirimkan pesan balasan sesuai dengan pesan yang dikirimkan pengguna.

4.2.2 Fitur-Fitur yang Disediakan Program

4.2.2.1 Menambahkan Tugas Baru

Mengirimkan pesan yang mengandung kata penting “ingatkan” atau “tambah”, tanggal, kode mata kuliah, jenis tugas, dan topik tugas.

4.2.2.2 Melihat Daftar Tugas yang Harus Dikerjakan

Mengirimkan pesan yang mengandung kata penting “apa saja” atau “apa aja”, rentang waktu deadline tugas (opsional), dan jenis task (opsional). Kata penting untuk jenis task adalah “tubes”, “tucil”, “kuis”, “praktikum”, “uts”, “uas”, dan “latihan”.

4.2.2.3 Menampilkan *Deadline* dari Suatu Tugas Tertentu

Mengirimkan pesan yang mengandung kata penting “kapan”, jenis tugas, dan kode mata kuliah.

4.2.2.4 Memperbarui *Deadline* Tugas Tertentu

Mengirimkan pesan yang mengandung kata penting “ubah”, “diundur”, “ganti”, atau “jadi”, tanggal, dan ID tugas yang ingin diperbarui *deadline*-nya.

4.2.2.5 Menandai Suatu Tugas Sudah Selesai Dikerjakan

Mengirimkan pesan yang mengandung kata penting “selesai” dan ID tugas yang ingin ditandai selesai dikerjakan.

4.2.2.6 Menampilkan Opsi *Help* yang Difasilitasi oleh *Assistant*

Mengirimkan pesan yang mengandung kata penting “help”, “tolong”, “助けて”, atau “tasukete”.

4.2.2.7 Merekomendasikan Kata ketika Pengguna Melakukan Saltik

Mengirimkan pesan yang mengandung kata yang mirip dengan kata penting.

4.3 Hasil Pengujian

4.3.1 Menambahkan *Task* Baru

Bot Wangy

Bot Wangy 21:56
Halo! Selamat datang di Bot Wangy! Silahkan masukkan pesan Anda!

Mahasiswa 21:56
Ada tugas apa saja?

Bot Wangy 21:56
Ga ada deadline yang akan datang

Mahasiswa 21:56
Ingatkan ada tugas besar IF2211 "Chatbot", deadline-nya tanggal 28 April 2021

Bot Wangy 21:56
[Task berhasil dicatat]
ID: 1
Materi: IF2211
Deadline (yyyy/mm/dd): 2021-04-28
Jenis: tugas
Topik: chatbot

Enter your message...



Bot Wangy

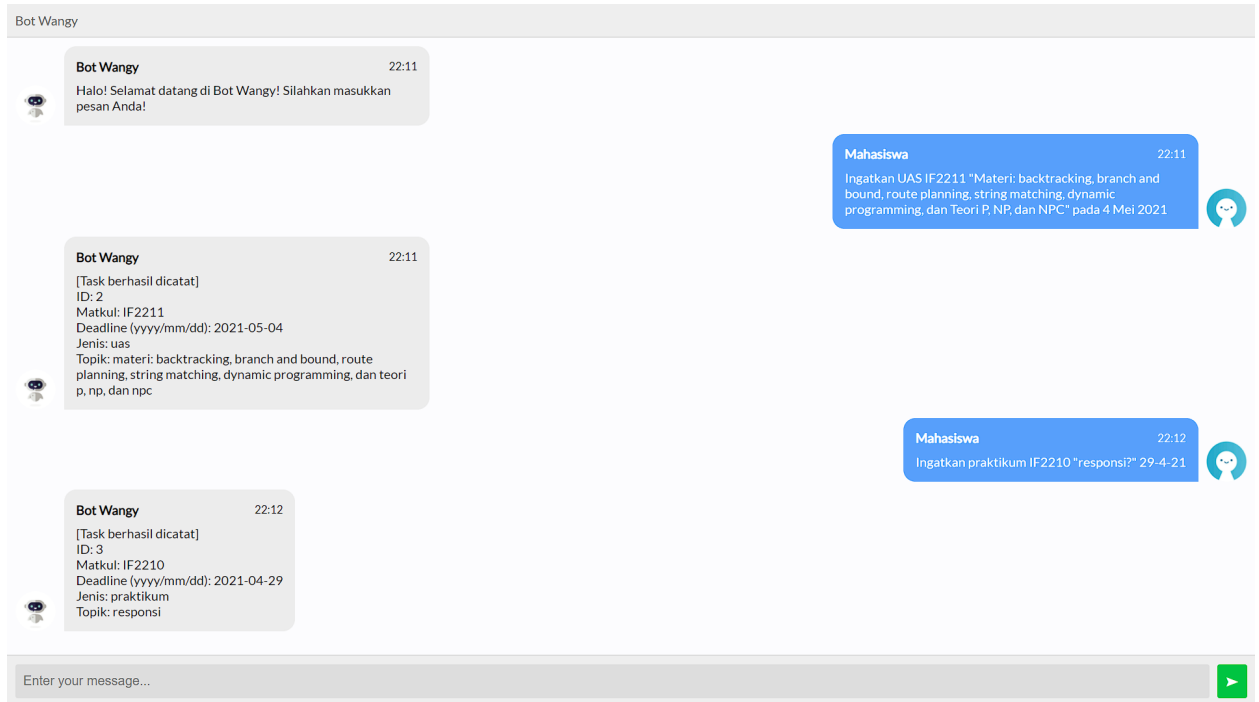
Bot Wangy 22:11
Halo! Selamat datang di Bot Wangy! Silahkan masukkan pesan Anda!

Mahasiswa 22:11
Ingatkan UAS IF2211 "Materi: backtracking, branch and bound, route planning, string matching, dynamic programming, dan Teori P, NP, dan NPC" pada 4 Mei 2021

Bot Wangy 22:11
[Task berhasil dicatat]
ID: 2
Materi: IF2211
Deadline (yyyy/mm/dd): 2021-05-04
Jenis: uas
Topik: materi: backtracking, branch and bound, route planning, string matching, dynamic programming, dan teori p, np, dan npc

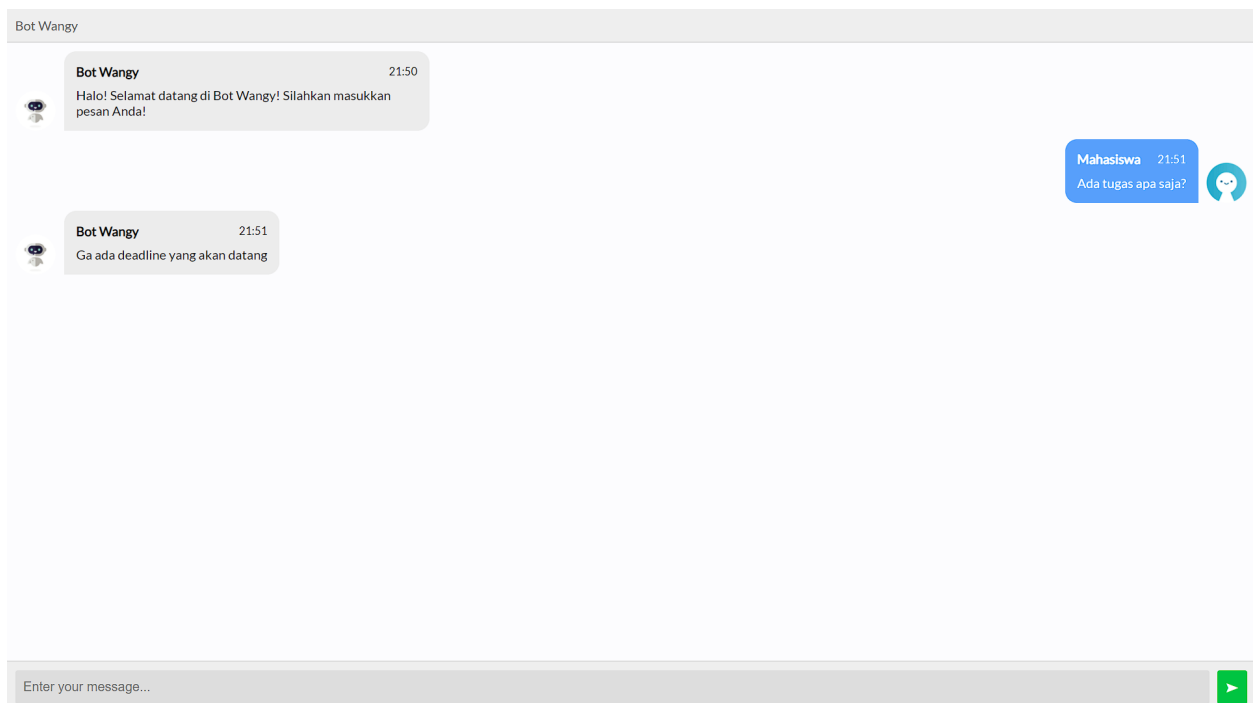
Enter your message...





4.3.2 Melihat Daftar *Task* yang Harus Dikerjakan

4.3.2.1 Tidak ada *task*



4.3.2.2 Ada beberapa *task*

Bot Wangy

Bot Wangy 22:12

[Task berhasil dicatat]
ID: 3
Matkul: IF2210
Deadline (yyyy/mm/dd): 2021-04-29
Jenis: praktikum
Topik: responsi

Mahasiswa 22:12

Ingatkan praktikum IF2210 "responsi?" 29-4-21

Bot Wangy 22:13

[Daftar tugas IF'19]
1. ID: 1
Matkul: IF2211
Deadline (yyyy-mm-dd): 2021-04-28
tugas besar: chatbot

2. ID: 2
Matkul: IF2211
Deadline (yyyy-mm-dd): 2021-05-04
uas: materi: backtracking, branch and bound, route planning, string matching, dynamic programming, dan teori p, np, dan npc

3. ID: 3
Matkul: IF2210
Deadline (yyyy-mm-dd): 2021-04-29
praktikum: responsi

Mahasiswa 22:13

Ada tugas apa saja?

Enter your message...

4.3.2.3 Pada *range* tanggal tertentu

Bot Wangy

Bot Wangy 22:13

[Daftar tugas IF'19]
1. ID: 1
Matkul: IF2211
Deadline (yyyy-mm-dd): 2021-04-28
tugas besar: chatbot

2. ID: 2
Matkul: IF2211
Deadline (yyyy-mm-dd): 2021-05-04
uas: materi: backtracking, branch and bound, route planning, string matching, dynamic programming, dan teori p, np, dan npc

3. ID: 3
Matkul: IF2210
Deadline (yyyy-mm-dd): 2021-04-29
praktikum: responsi

Mahasiswa 22:14

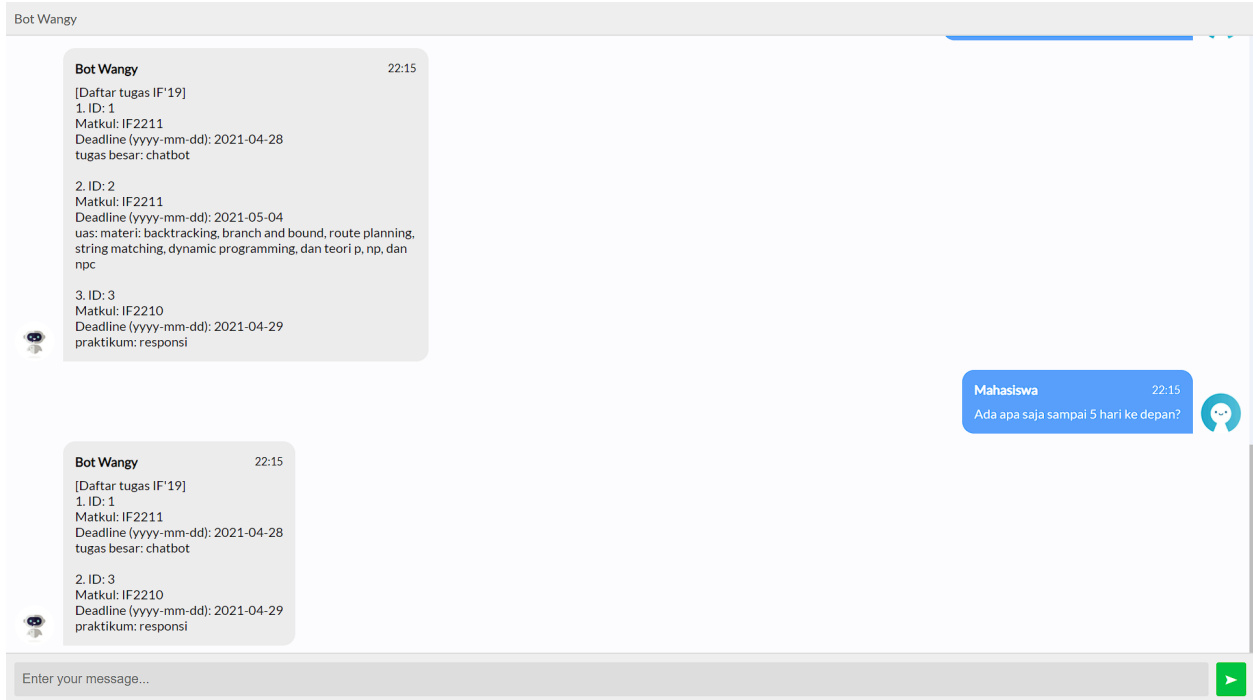
Ada apa saja dari tanggal 20 Januari 2021 sampai 3/05/21

Bot Wangy 22:14

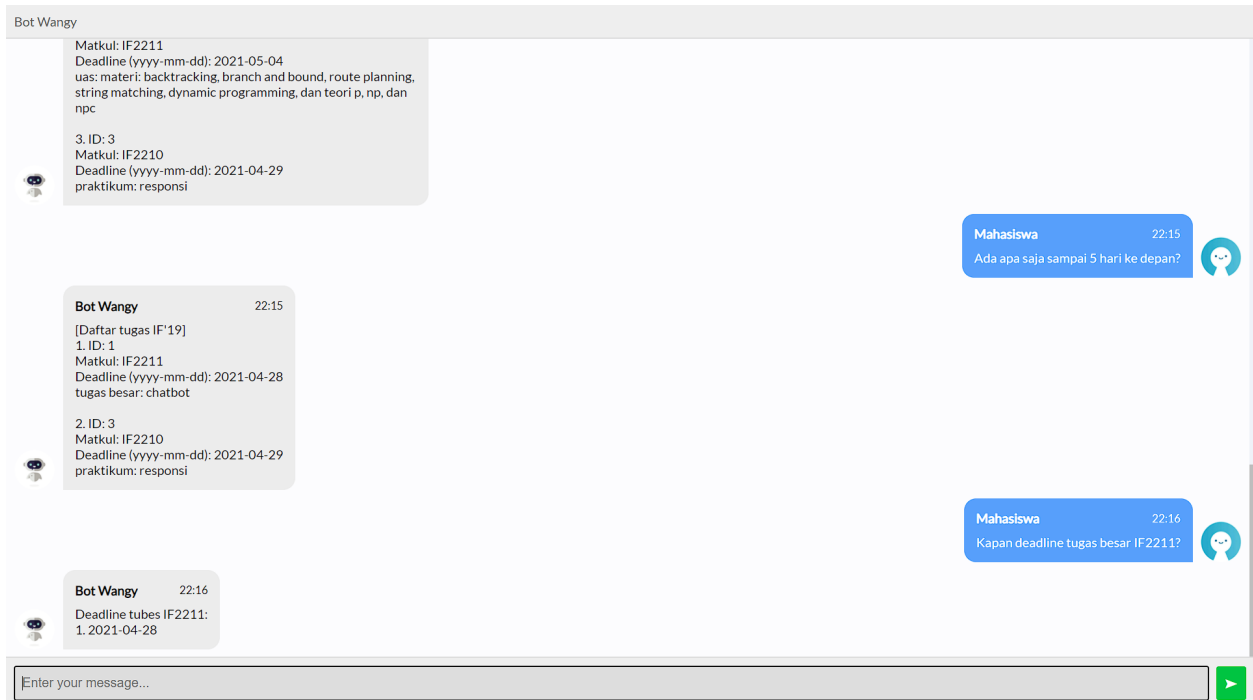
[Daftar tugas IF'19]
1. ID: 1
Matkul: IF2211
Deadline (yyyy-mm-dd): 2021-04-28
tugas besar: chatbot

2. ID: 3
Matkul: IF2210
Deadline (yyyy-mm-dd): 2021-04-29
praktikum: responsi

Enter your message...



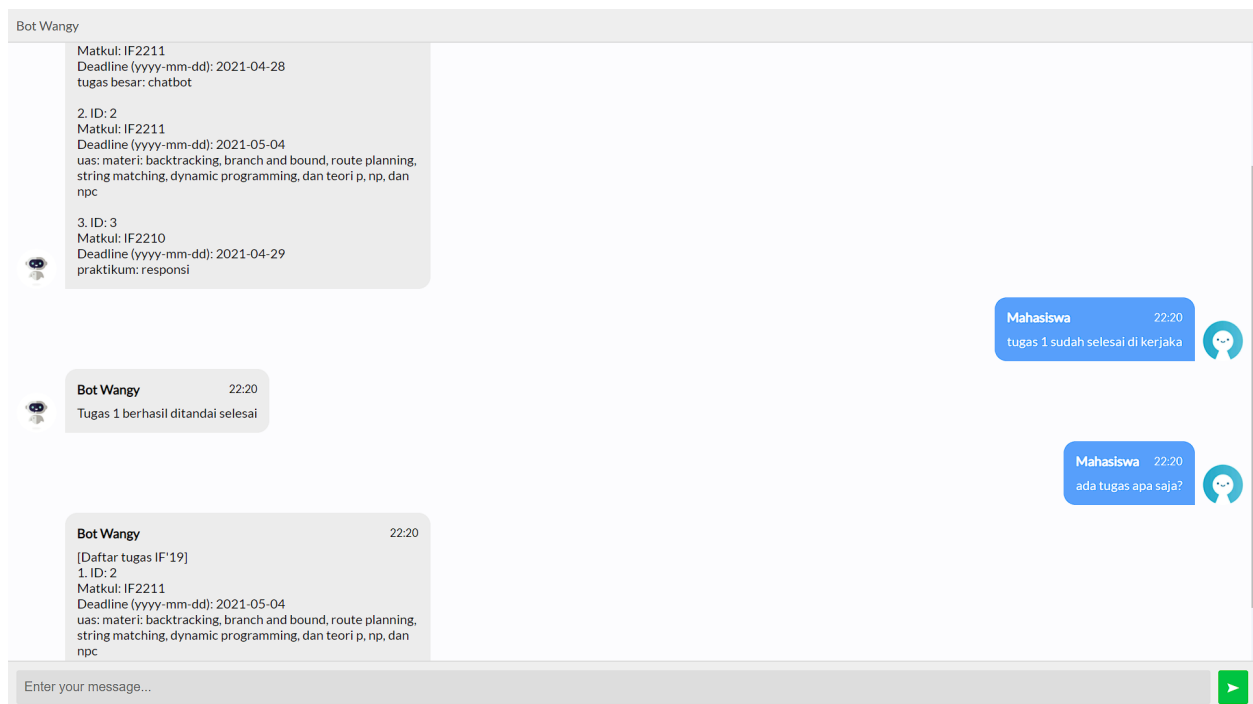
4.3.3 Menampilkan *Deadline* dari Suatu *Task* Tertentu



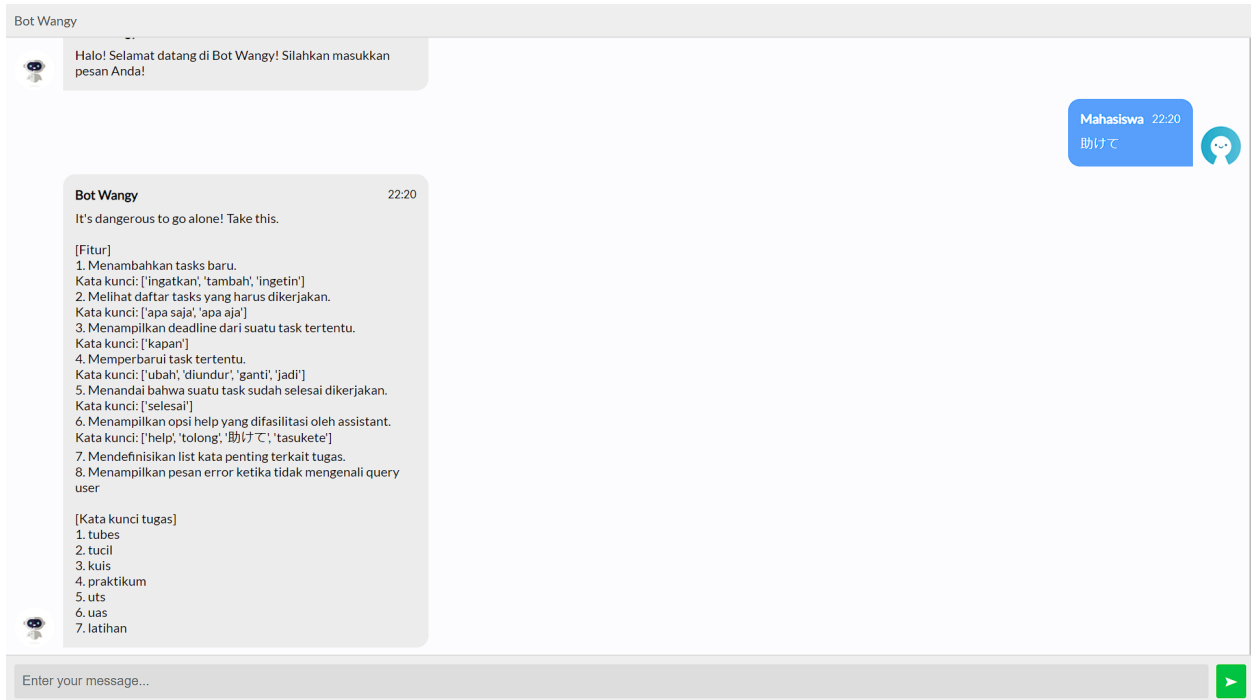
4.3.4 Memperbarui *Deadline Task* Tertentu



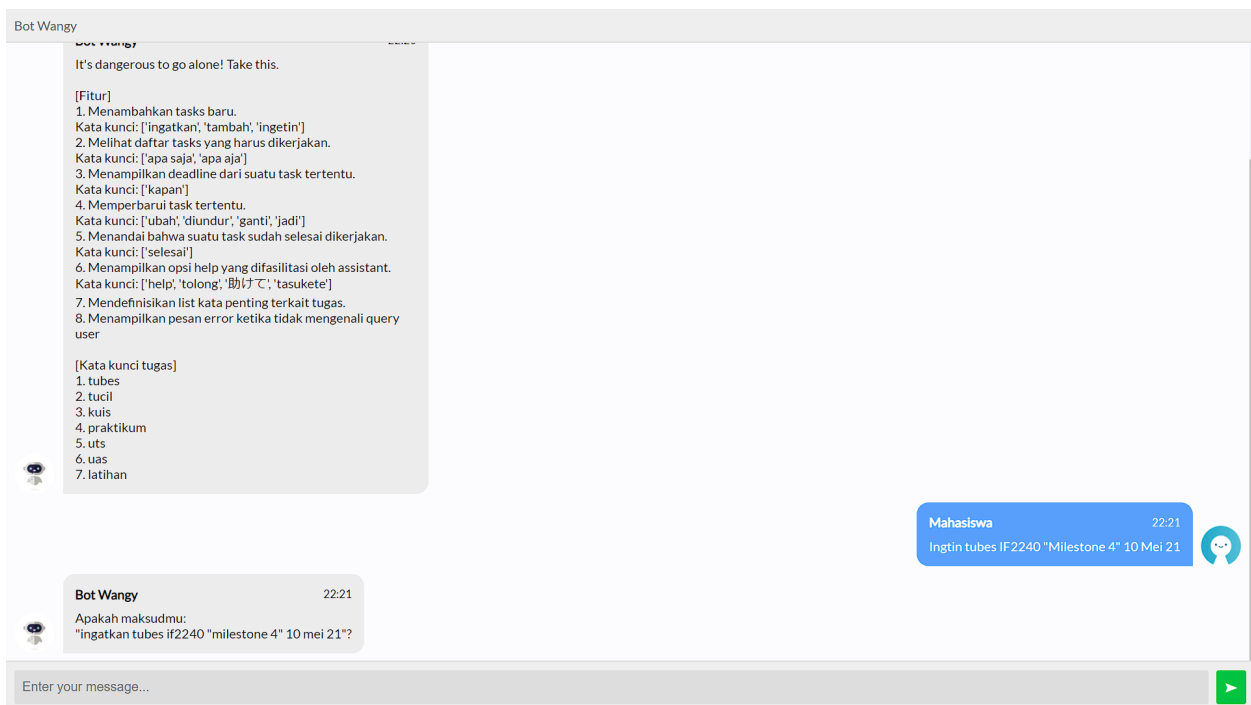
4.3.5 Menandai Suatu *Task* Sudah Selesai Dikerjakan



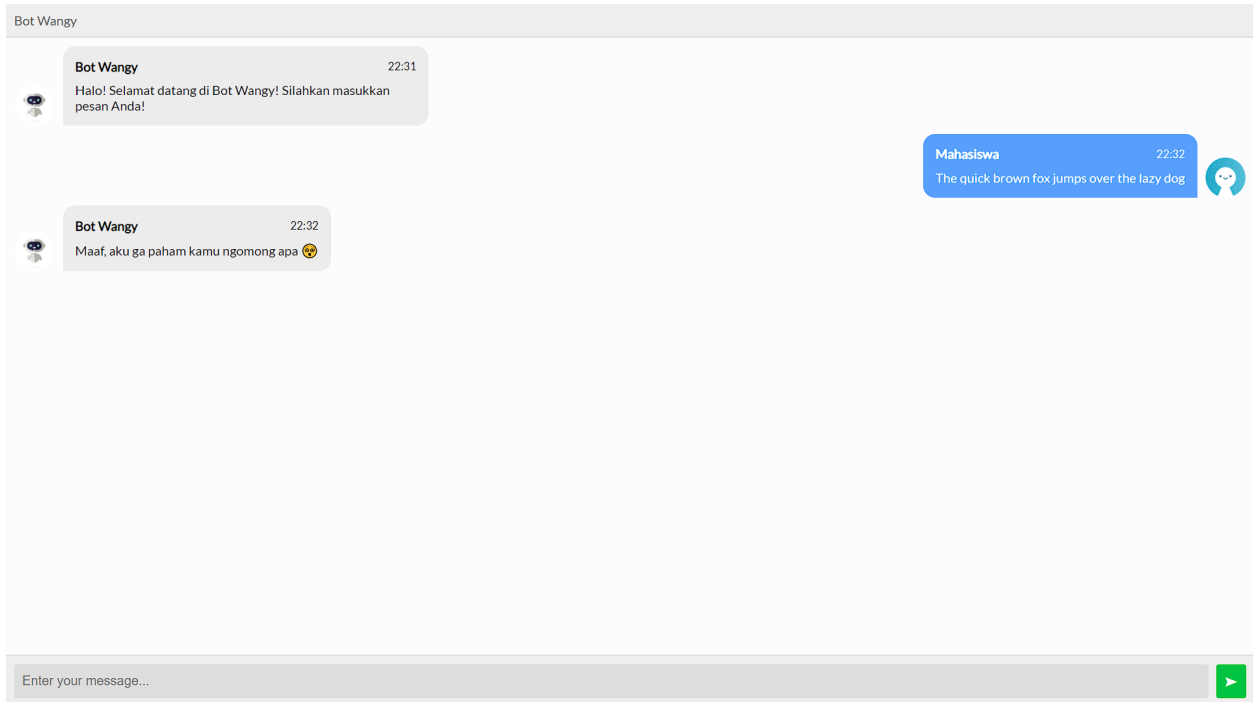
4.3.6 Menampilkan Opsi *Help* yang Difasilitasi oleh *Assistant*



4.3.7 Merekomendasikan Kata ketika Pengguna Melakukan Saltik



4.3.8 Pesan Tidak Dapat Dikenali



4.4 Analisis Hasil Pengujian

Seluruh fitur fungsional dari *chatbot* berjalan sesuai yang diharapkan dalam setiap skenario pengujian. *Chatbot* dapat mengembalikan pesan *error* dengan baik apabila terdapat kasus di mana input pengguna tidak sesuai dengan command yang dikenali oleh *chatbot*. Data pada Google Firestore *database* terbaharui secara otomatis sesuai dengan perintah-perintah yang dijalankan pengguna, seperti menambah task akan menambah *record* tugas dan menandakan tugas selesai akan mengubah atribut 'selesai' dari tugas yang ditandai menjadi *true*.

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Pembangunan *deadline reminder assistant* dalam bentuk *chatbot* dengan memanfaatkan algoritma pencocokan string dan *regular expression* telah berhasil dilaksanakan dengan baik. Seluruh fitur fungsional *chatbot* yang dibangun dapat bekerja sesuai dengan yang diharapkan pada seluruh skenario pengujian yang dilaksanakan.

5.2 Saran

Deadline reminder assistant dalam bentuk *chatbot* yang dibangun masih dapat dikembangkan lagi dalam beberapa aspek. Desain dari layout *chatbot* masih dapat dibuat lebih menarik. Fungsionalitas dari *chatbot* dapat ditambahkan lagi untuk pengembangan lebih lanjut, seperti fitur alarm yang mengingatkan sehari sebelum deadline tugas.

5.3 Komentar/Refleksi

- ~~Mana sempat keburu telat~~
- Seru
- *Bye bye stima, you will be !remembered*
- #BatalKantugasMakalah

DAFTAR PUSTAKA

1. “Pencocokan String (String/Pattern Matching)”. Munir, Rinaldi
<https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Pencocokan-string-2021.pdf>
2. “String Matching dengan Regular Expression”. Khodra, Masayu Leylia
<https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2018-2019/String-Matching-dengan-Regex-2019.pdf>
3. “Chatbot”. LiveChat, Inc.
<https://www.chatbot.com/>
4. “Simple Chat UI”. Hashemian, Sajad
<https://codepen.io/sajadhsm/pen/odaBdd>
5. “re — Regular expression operations”. Python Software Foundation
<https://docs.python.org/3/library/re.html>
6. “Svelte.js + Flask — Combining Svelte with a Backend Server”. Cabrera, Alex
<https://cabreraalex.medium.com/svelte-js-flask-combining-svelte-with-a-simple-backend-server-d1bc46190ab9>
7. “Understanding the Levenshtein Distance Equation for Beginners”. Nam, Ethan
<https://medium.com/@ethannam/understanding-the-levenshtein-distance-equation-for-beginners-c4285a5604f0>
8. “The Boyer-Moore Fast String Searching Algorithm”, Moore, J Strother
<https://www.cs.utexas.edu/~moore/best-ideas/string-searching/index.html>
9. “Spesifikasi Tugas Besar III IF2211 Strategi Algoritma”. Asisten Laboratorium IRK STEI ITB
<https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Tugas-Besar-3-IF2211-Strategi-Algoritma-2021.pdf>