SISTEM INFORMASI SENTIMEN *TUBERCULOSIS* (SIS-TUBES)

DOKUMENTASI PENGEMBANGAN & MANUAL PEMAKAIAN

Penyusun:

Ronaldo Cristover Octavianus

Dzikri Robbi

Laras Ervintyana

Hapnes Toba

Mewati Ayub

PROGRAM STUDI MAGISTER ILMU KOMPUTER
FAKULTAS TEKNOLOGI INFORMASI
UNIVERSITAS KRISTEN MARANATHA

2022

Daftar Isi

1. Pengantar

Tuberculosis atau dikenal juga dengan TB atau TBC, adalah adalah suatu penyakit menular yang disebabkan oleh kuman *Mycobacterium Tuberculosis* [1]. Penyakit ini termasuk salah satu penyakit yang mematikan dan perlu diobati dalam jangka panjang. Berdasarkan data WHO pada tahun 2015, Indonesia memiliki jumlah penderita TB terbanyak di dunia dengan jumlah populasi sebanyak 10,4 Juta jiwa.

Salah satu cara untuk mengurangi dan mencegah penyebaran TBC adalah dengan melakukan *tracing*. Dengan melakukan *tracing*, penyebaran TBC diharapkan dapat diketahui lebih dini dan lebih cepat untuk ditanggulangi. Saat ini di Indonesia, sudah terdapat sistem pendataan penyakit TBC bernama SITB (Sistem Informasi *Tuberculosis*) yang merupakan aplikasi yang digunakan oleh semua pemangku kepentingan mulai dari Fasilitas Pelayanan Kesehatan, Dinas Kesehatan, Kementrian Kesehatan untuk melakukan pencatatan dan pelaporan kasus TBC [2]. Akan tetapi sistem ini baru dapat melakukan *tracing* jika pasien sudah positif TBC dan datanya di-*input*-kan ke dalam sistem. Oleh karena itu, untuk dapat mendeteksi potensi penyebaran TBC yang lebih cepat maka diperlukan sistem lain yang dapat memberikan deteksi dini penyebaran TBC di suatu wilayah di Indonesia.

Media sosial saat ini telah menjadi bagian hidup masyarakat Indonesia dengan jumlah pengguna media sosial di Indonesia telah mencapai 50% dari jumlah keseluruhan total penduduk. Penggunaan media sosial sendiri memberikan banyak manfaat, salah satunya adalah penyebaran informasi yang cepat dan mudah diakses. Salah satu media sosial yang paling berpengaruh dalam penyebaran informasi adalah Twitter. Berdasarkan latar belakang situasi di atas, dikembangkanlah sebuah sistem untuk menentukan sentimen penyebaran penyakit TBC melalui media sosial Twitter yang sekaligus dapat membantu *tracing* penyebaran penyakit TBC di Indonesia.

2. Konsep dan Teknologi Pendukung

A. *Tuberculosis*

Tuberculosis merupakan penyakit menulari yang disebabkan oleh bakteri *Mycrobacterium tuberculosis*. Gejala utama pasien TBC yaitu batuk berdahak selama dua minggu atau lebih. Batuk dapat diikuti dengan gejala tambahan yaitu dahak bercampur darah, batuk darah, sesak nafas, badan lemas, nafsu makan menurun, berat badan menurun, malaise, berkeringat malam hari tanpa kegiatan fisik, demam, dan meriang lebih dari satu bulan [1].

B. Analisis Sentimen

Analisis sentimen, yang disebut juga dengan *opinion mining*, merupakan salah satu cabang ilmu dari *data mining* yang bertujuan untuk menganalisis, memahami, mengolah, dan mengekstrak data tekstual yang berupa opini terhadap entitas seperti produk, servis, organisasi, individu, dan topik tertentu. Analisis ini digunakan untuk mendapatkan suatu bentuk informasi tertentu dari suatu kumpulan data yang ada.

C. *Twitter*

Media sosial adalah ssitem berbasis teknologi internet yang dibangun berdasarkan ideologi dan teknologi dari Web 2.0, serta memungkinkan pengguna membuat konten [3]. Menurut Wijanto, Twitter merupakan sebuah situs media sosial berbasis *microblogging*, dimana penggunanya mengirimkan sebuah pesan yang disebut dengan *tweets*.

Sejak tahun 2008, konsep dan sistem untuk memonitor *disease outbreaks* dan *emergencies* dengan menggunakan *Twitter* sudah dikembangkan. *Twitter* dianggap sangat *up-to-date*, memiliki lebih dari lima ratus juta pengguna dan lebih dari tiga ratus empat puluh juta *tweets* yang di-*posting* setiap harinya. Kebanyakan dari *tweets* tersebut bersifat *public* sehingga memungkinkan untuk menarik data selain *tweet* seperti lokasi. Dikarenakan banyaknya *tweet* yang tersedia dan bersifat *real-time*, sehingga menjadikan *Twitter* ideal untuk pengklasifikasian sentimen untuk pemantauan penyakit secara lebih luas [4].

D. *Twitter Data Streaming*

Data *Twitter* menjadi salah satu sumber data penelitian yang paling sering digunakan. Hal ini dikarenakan pengguna *Twitter* dapat membagikan informasi secara *realtime*, melampirkan konten dari berbagai macam sumber, termasuk melampirkan gambar dan video. Dari komponen *Twitter* tersebut dapat digunakan untuk mengekstrak informasi seperti:

1. *Username* : Merupakan identifikasi dari pengguna atau pemilik akun *Twitter*
2. *Time Stamp*: Setiap twitter yang dikirimkan selalu mengandung dan menampilkan kapan tweet tersebut dikirimkan
3. *Tweet Text*: Isi dari *tweet* berupa *text* dimana di dalamnya bisa terkandung gambar, *link*, *hashtag* dan lainnya.
4. *Hashtag*: Teks yang didahului simbol "#" serta dikaitkan dengan topik tertentu.
5. *Reply*: Balasan dari teks yang di tweet
6. *Retweet*: Suatu fungsi untuk membagikan *tweet* kepada pengikut / *followers* mereka.

Untuk melakukan *stream* pada data yang ada pada *Twitter*, dapat digunakan API REST yang sudah disediakan resmi oleh *Twitter* itu sendiri. Dari *data stream* inilah dapat dilakukan pengumpulan data *tweet* yang ada secara real time.

E. *Text Mining*

Text mining atau *text data mining*, adalah bidang pengetahuan yang mencakup area *information retrieval*, *text analysis*, *information extraction*, *clustering*, *categorization*, *visualization*, *database technology*, *machine learning*, dan *data mining* [4].

Dalam beberapa tahun terakhir, obyek dari *text mining* yang banyak diteliti adalah *website* atau *world wide web*. *World wide web* memiliki banyak konten dokumen teks yang dapat diolah lebih lanjut menggunakan *text mining*, antara lain berita, media sosial, *e-commerce*, dan lainnya.

Tahap pertama dari *text mining* adalah *text preprocessing*, dimana *text preprocessing* mengolah data tidak terstruktur menjadi data terstruktur. Keluaran dari *text preprocessing* digunakan untuk masukan ke algoritma *text mining*. Keluaran dari *algoritma text mining* dianalisis untuk menjadi sebuah pengetahuan.

F. *Text Preprocessing*

Tahapan dalam *text mining* dimulai dengan *text preprocessing*. *Text preprocessing* menyiapkan data teks menjadi kata/token yang siap diolah lebih lanjut. *Text preprocessing* berpengaruh terhadap keberhasilan algoritma *text mining* yang digunakan [4]. Beberapa proses yang dilakukan dalam *text preprocessing* adalah tokenisasi, menghilangkan *stop word*, *stemming* & *lemmatization* [4].

Pada tahapan *preprocessing*, semua kata yang ada dalam *tweet* diubah menjadi *lowercase* dan menghilangkan beberapa konten yang dianggap tidak penting, seperti [5]:

- *Space pattern*

- *Internet Addresses (Links, URL)*
- *Twitter Mentions*
- *Retweet Symbols*
- *Stopwords*

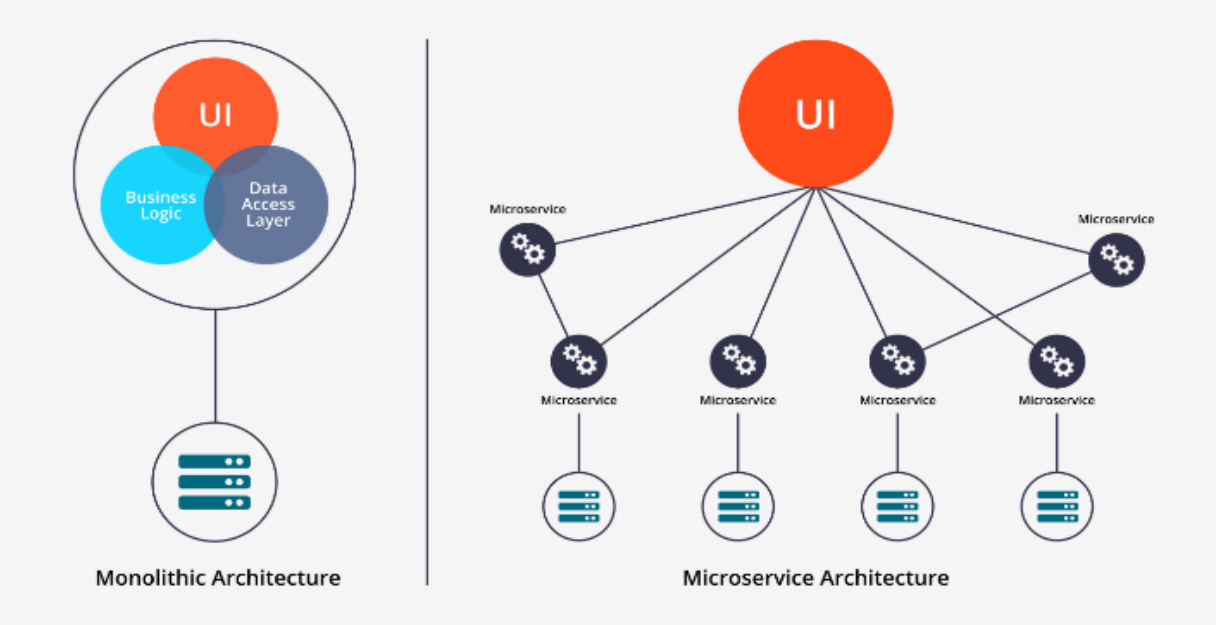
G. *Information Extraction*

Information extraction merupakan identifikasi kata kunci dari sebuah dokumen teks dan relasi antar dokumen teks secara otomatis. *Information extraction* memproses data yang tidak terstruktur menjadi data terstruktur yang siap digunakan algoritma *text mining*. *Information extraction* melihat pola sekuensial dari kata kunci yang sudah ditentukan, ini disebut dengan *pattern matching*. Information extraction bertujuan menghasilkan informasi penting dari kelompok dokumen dengan jumlah yang banyak [4].

H. *Term Frequency - Inverse Document Frequency*

Terms Frequency - Inverse Document Frequency (TF-IDF) merupakan metode pembobotan kata secara statistik yang menunjukkan seberapa pentingnya sebuah kata pada suatu dokumen pada sekumpulan / koleksi dokumen [4]. Metode pembobotan *TF-IDF* biasanya digunakan dalam *text mining*. *Term frequency* adalah jumlah sebuah kata pada dokumen, sedangkan *inverse document frequency* atau *IDF* adalah nilai yang digunakan untuk mengukur seberapa penting sebuah kata pada koleksi dokumen.

Dalam suatu bahasa, terdapat kata-kata yang tidak memiliki makna penting, contohnya dalam bahasa Inggris adalah kata '*the*' dan '*of*', kata-kata tersebut harus dihilangkan dengan cara menentukan *threshold* yang dipakai dalam *TF-IDF* sehingga terdapat *filter* awal terlebih dahulu sebelum kata-kata tersebut mulai untuk diproses [6].



Gambar 1. Arsitektur Berbasis *Microservices*

I. *Microservices*

Microservices atau disebut dengan layanan mikro adalah suatu pendekatan arsitektur asli dari *cloud* dimana satu aplikasi terdiri dari banyak komponen atau layanan kecil yang digabungkan secara

independen (*loose coupling*) dan dapat digunakan secara mandiri. Layanan ini biasanya memiliki *stack* teknologi yang berbeda-beda termasuk dengan model *database* dan juga manajemen data.

Microservices ini juga dapat berkomunikasi antara satu dengan yang lainnya melalui *Application Programming Interface (API)* atau *Event Stream* atau *Message Broker* dan lainnya. *Microservices* itu sendiri diatur sesuai dengan kemampuan bisnis dimana layanan-layanan yang ada dibagi dan dipetakan sesuai dengan kebutuhan bisnis. Dengan menggunakan *microservices*, pengelolaan aplikasi menjadi lebih muda dimana setiap layanan tidak memiliki hubungan satu dengan lain dan terpisah (*loose coupled*) sehingga sangat memungkinkan untuk melakukan *scaling* secara mandiri tanpa mengganggu layanan lain. Secara umum arsitektur *microservices* dapat digambarkan seperti pada Gambar 1.

J. *Python*

Python adalah suatu bahasa pemrograman *interpretative* yang dapat digunakan di berbagai macam *platform* dengan filosofi perancangan yang berfokus pada tingkat keterbacaan kode dan merupakan salah satu bahasa yang populer yang berkaitan dengan *Data Science, Machine Learning, Internet of Things (IoT)*. Keunggulan *Python* yang bersifat *interpretative* juga banyak digunakan untuk *prototyping, scripting* dalam pengelolaan struktur, hingga pembuatan aplikasi *web* skala besar.

Python pada pengembangan sistem ini digunakan untuk melakukan pengolahan data seperti *processing data, cleaning data*, dan lain-lain. Peran utamanya adalah melakukan pengelolahan data sehingga data tersebut bisa digunakan oleh layanan lain dengan tujuan tertentu.

K. *PHP*

PHP adalah suatu bahasa pemrograman yang sangat umum untuk digunakan dan juga *powerful*. *PHP* sangat umum dan cukup mudah digunakan dalam pembuatan suatu perangkat lunak. Dalam pembuatan proyek ini, bahasa pemrograman *PHP* digunakan sebagai *API Gateway* dimana pada prosesnya setiap *request* dari layanan depan (*front end*) akan masuk ke dalam *API Gateway* dan nantinya akan dialirkan ke layanan-layanan lain sesuai dengan fungsinya.

L. *JSON*

JSON (Javascript Object Notation) adalah format untuk menyimpan dan menukar informasi yang dapat dibaca oleh manusia dan digunakan sebagai format transfer data antara *server* dan *client*. JSON merupakan alternatif dari XML dimana ukuran file JSON lebih kecil dibandingkan XML. Dalam JSON terdapat dua inti dari objek yaitu *Key* dan *Value*. Contoh dari JSON adalah sebagai berikut:

```
1.  {"city":"New York", "country":"United States "}
```

M. *MySQL*

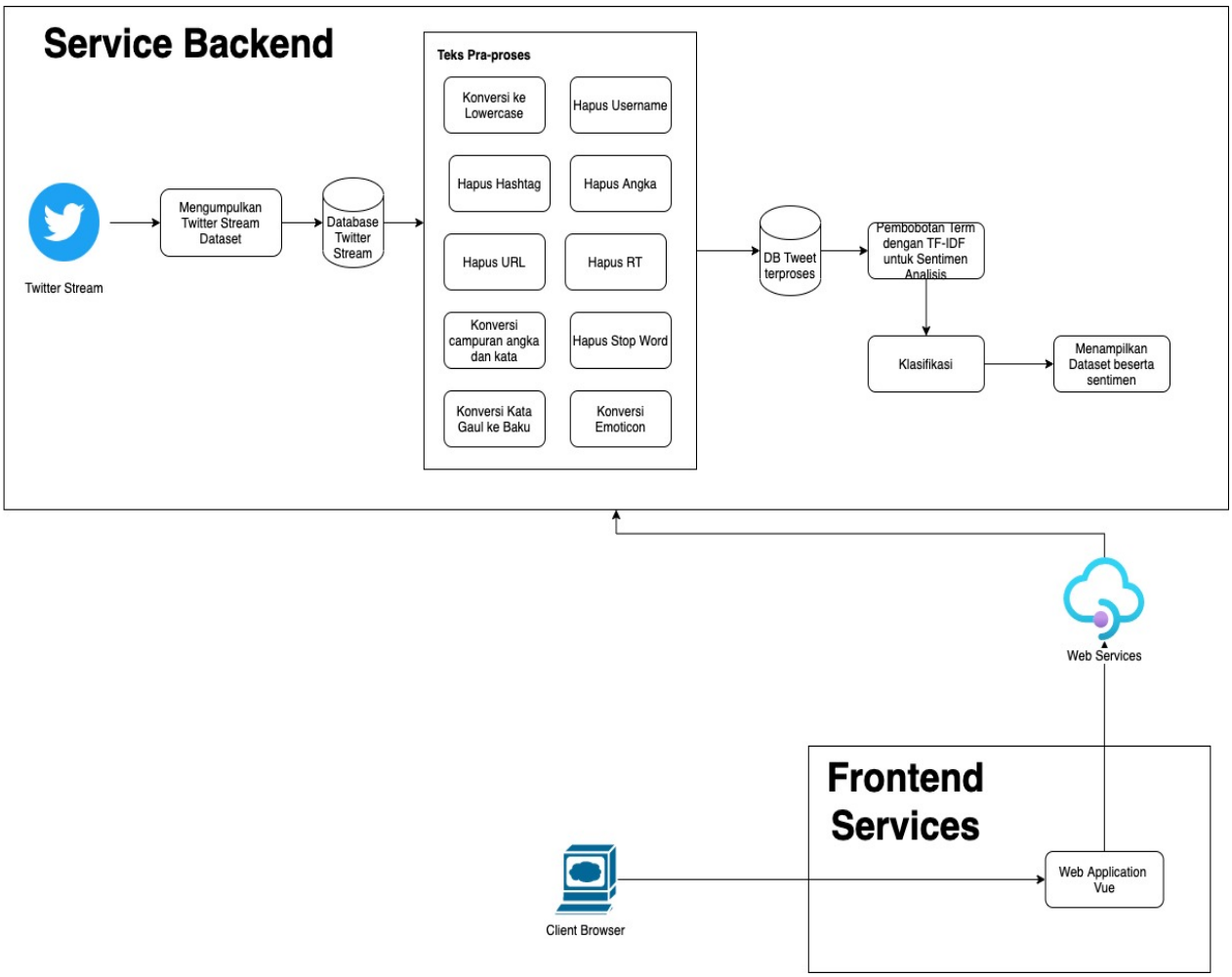
MySQL merupakan sebuah *database management system* yang menggunakan perintah dasar *SQL*. *MySQL* sendiri merupakan sebuah aplikasi *opensource* sehingga penggunaannya gratis. *MySQL* sendiri dalam pengembangan sistem ini digunakan sebagai tempat penyimpanan data dari *twitter streaming*.

3. Analisis dan Perancangan

A. Analisis Singkat

Sistem Informasi Sentimen *Tuberculosis* (SIS-TUBES) merupakan sebuah sistem yang memiliki beberapa proses. Proses utama yang tersedia adalah: proses melakukan *stream data* secara *real-time*,

proses untuk melakukan perhitungan dan kalkulasi terhadap data yang ada, proses untuk menampilkan data agar *user* bisa melihat secara visual. Dalam tiga bagian besar, proses-proses ini perlu digabungkan sebagai suatu sistem yang dapat membantu *user* dan tentunya *developer* dalam mengembangkan sistem ke depannya.



Gambar 2. Aristektur Sistem Secara Keseluruhan

B. Diagram Perancangan

Aplikasi ini dibuat dengan menggunakan arsitektur *microservices* dimana *service* atau layanan dipecah sesuai dengan tujuannya masing-masig. Alasan dibuat arsitektur tersebut adalah:

- Mempermudah *development* karena berbedanya bahasa yang digunakan untuk setiap proses
- Mempermudah pengembangan aplikasi karena setiap *service* terpaut namun dalam prosesnya bisa berjalan masing-masing
- Dapat menangani *request* dengan jumlah yang lebih banyak karena fungsi yang dipanggil bersifat mikro dan tidak memakan banyak *resource* karena *package*, *library* dan *server* dapat menampung *request* yang cukup tinggi.
- Dalam proses ke depannya, jumlah data akan semakin tinggi oleh karena itu proses pengolahan data akan semakin tinggi juga. Dengan menggunakan arsitektur ini, proses *scaling service* dapat dilakukan pada *service* yang memiliki *load* tinggi saja dan menjadi lebih hemat *resource*.

Arsitektur sistem selengkapnya diberikan pada Gambar 2.

Pada Gambar 2 dapat dilihat bahwa arsitektur dibuat menjadi 2 bagian yaitu:

- *Service Backend*
  Service backend adalah *service* yang bertugas untuk melakukan pengolahan data di belakang layar dengan setiap data yang masuk akan diolah dan nantinya akan dikembalikan apabila *service frontend* meminta suatu data. Pada *service* ini, dibuat menjadi beberapa *microservices* dimana terdapat *services* seperti berikut:
  a. *Twitter Stream Service*
     Service ini berfungsi untuk melakukan *stream API Twitter* dan menyimpannya ke dalam *database*. *API Data Stream* yang di-*collect* memiliki format JSON. Dari *response stream* tersebut dipilih objek mana saja yang akan disimpan dan dipisah serta sisanya akan disimpan sebagai *raw data* atau data acuan dalam suatu kolom. Data yang dilakukan *stream* tentunya perlu dilakukan *filtering* sehingga tidak semua data akan masuk, namun hanya data-data yang berhubungan dengan *tuberculosis* saja. Dalam proses *filtering data stream*, diberikan kata kunci sebagaimana dituliskan dalam Tabel 1.

Tabel 1. Kata Kunci untuk Pemrosesan *Data Stream*

Kata Kunci
Batuk
Tuberkulosis
Tuberculosis
TB
TBC
TB Paru
Batuk Berdarah

  b. *Preprocessing Service*
     Tahap *preprocessing* atau praproses data merupakan suatu proses untuk mempersiapkan data mentah sebelum dilakukan proses lainnya. Pada umumnya, praproses data dilakukan dengan cara menghilangkan data yang tidak sesuai dan mengubahnya menjadi bentuk yang mudah diproses oleh sistem. Praproses ini perlu dilakukan untuk melakukan analisis sentimen khususnya pada media sosial yang sebagian besar berisi kata-kata atau kalimat yang tidak formal dan tidak terstruktur dan memiliki *noise* yang besar.
     Terdapat tiga model praproses untuk kalimat atau teks dengan *noise* besar. Tiga model tersebut adalah:
     a. *Orthographic Model*
        Model ini dipergunakan untuk memperbaiki kata atau kalimat yang memiliki kesalahan dari segi bentuk kata atau kalimat.
     b. *Error Model*
        Model ini dipergunakan untuk memperbaiki kesalahan dari segi kesalahan eja atau kesalahan penulisan. Ada dua jenis kesalahan yang dikoreksi dengan model ini yaitu kesalahan penulisan dan kesalahan eja. Kesalahan penulisan dapat mengacu pada kesalahan

pengetikan sedangkan kesalahan eja muncul ketika penulis tidak mengetahui ejaan yang benar.

c. *White Space Model*

Model ini mengacu pada pengoreksian tanda baca. Contoh kesalahan dalam model ini adalah tidak menggunakan titik di akhir kalimat namun model ini tidak terlalu efektif untuk diterapkan pada sosial media yang tidak mengindahkan tanda baca.

Tahap praproses ini bisa disebut juga dengan proses ekstraksi data dimana data yang sebelumnya telah dikumpulkan dalam database akan diproses melalui proses ekstraksi. Dalam proses ini, akan dilakukan beberapa proses seperti:

a. *Case Folding*, yaitu membuat semua teks menjadi huruf kecil
b. *Remove Punctuation*, yaitu menghapus semua karakter non alfabet misalnya simbol, spasi dan lainnya.
c. *Remove Username*, yaitu menghapus nama user yang diawali dengan simbol @ karena dianggap tidak penting
d. *Remove Hashtag*, yaitu menghapus karakter "#" yang biasanya dijadikan judul topik.
e. *Clean Number*, yaitu menghapus angka yang ada didepan atau dibelakang suatu kata, seperti: jalan2, makan2, 22nya dan lainnya.
f. *Clean One Character*, yaitu menghapus jika terdapat hanya satu huruf saja seperti g, a, f, dan lain-lain.
g. *Remove URL*, yaitu menghapus *URL* yang terdapat pada twit atau kata.
h. *Remove RT*, yaitu menghapus *prefix* RT yang menandakan bahwa teks tersebut merujuk pada suatu *tweet* dan *username*.
i. *Convert Number*, yaitu melakukan proses perubahan number menjadi suatu kata khususnya pada kata-kata yang memiliki campuran seperti s4y4ng, b4tuk dimana bila diubah menjadi sayang dan batuk.
j. *Remove Stop Word.* Stop word diproses pada sebuah kalimat jika mengandung kata-kata yang sering keluar dan dianggap tidak penting seperti waktu, penghubung, dan lain-lain.
k. *Convert Word*, yaitu melakukan konversi dari kalimat baku, kalimat dengan bahasa *alay*, gaul menjadi bahasa baku, sebagai contoh konversi diberikan pada Tabel 2.

Tabel 2. Contoh Konversi Kata

Sebelum	Sesudah
Akyu	Aku
Akuwh	Aku
Akku	Aku
Aq	Aku
Aquwh	Aku
Awak	Aku
Amaca	Ah masa
Alluw	Hallo
Alo	Hallo
Atw	Atau

l. *Convert Emotion*, yaitu merubah *emoticon* dalam *Twitter* menjadi suatu kata sentimen. Contoh konversi diberikan dalam Tabel 3.

Tabel 3. Contoh Konversi *Emoticon*

Emoticon	Konversi	Masuk Kelas
:] :-) :) :o) :] :3 :c) :> =] 8) =) :} :^)	Senang	Positif
>:D :-D :D 8- D 8D x-D xD XD XD =-D =D =-3 =3	Tertawa	Positif
> :[:-(:(:-c :c :-< :< :-[:[:{ > .><> .< :'(	Sedih	Negatif
D :< D : D 8 D ; D = D X v.v D-':	Horror	Netral
> :P :-P :P X-P x-p xp XP :- p :p =p :-Þ :Þ	*Tongue*	Netral
>:o>:O :-O :O °o° °O° :O o_O o.O 8-0	*Shock*	Positif
> :\ >:/ :-/ :-. :/ :\ =/ =\ :S	Kesal	Negatif
:| :-|	Ekspresi Datar	Negatif

c. *Term and Classification Service*

Service ini memiliki fungsi untuk melakukan perhitungan seperti:

- *Indexing* dan Tokenisasi

  Pada proses ini, data yang sudah diclean akan dilakukan indexing dan tokenisasi yang selanjutnya akan dilakukan proses pembobotan, TF-IDF.
- Proses TF-IDF

  Pembobotan *term* dilakukan dengan mendapatkan *context of discussion* dari *dataset*. *Context of Discussion* dapat digunakan untuk analisis tren percakapan yang terjadi di *Twitter* terkait dengan penyakit TBC.
- Proses Klasifikasi

  Dari hasil pengolahan seperti tokenisasi, pembobotan TF-IDF maka dilakukan proses klasifikasi dimana setiap *tweet* akan dinilai dan diklasifikasi apakah tweet tersebut masuk kategori positif atau negatif.

d. *API Gateway Services*

Service ini digunakan sebagai penghubung antara *Frontend Services* dan *Backend Services*. Setiap *request* yang dikirimkan dari *frontend* akan diterima oleh *API Gateway* terlebih dahulu sebelum melakukan pengambilan data yang sudah diolah.

- *Service Frontend*

  Service Frontend adalah *service* yang bertemu langsung dengan pengguna dimana *service* ini berbasis web dengan teknologi CSR (*Client Side Rendering*). Service ini akan melakukan pembangkitan *script* dan melakukan *render* pada *browser user*, sehingga *load* di sisi server akan menjadi lebih rendah dan interaksi *user* pada aplikasi menjadi lebih baik juga.

C. Perancangan Basis Data

Data yang ditarik dan di-*stream* dari API Twitter, perlu disimpan di dalam suatu *database* MySQL. Sebelum menyimpan *data stream*, perlu dilakukan analisis terhadap *response* balikan dari *API Stream*. Untuk struktur JSON yang digunakan diberikan dalam Gambar 3.

```
object ▶ in_reply_to_screen_name
▼ object {33}
    created_at : Wed May 26 09:18:17 +0000 2021
    id : 1397482208193958000
    id_str : 1397482208193957890
    text : https://t.co/AsXO4JuVdT\nBCG covid治験が5件ほど（欧が多いが家加等も。医療従事者
           被験が多）（件のギリシャは完了扱い）だいたいこの夏までにそれなりの進展を示す予定（延期
           はありうるが）のようなので、それ… https://t.co/0atYSPNTW9
    source : <a href=\"https://mobile.twitter.com\" rel=\"nofollow\">Twitter Web
             App</a>
    truncated : ☑ true
    in_reply_to_status_id : null
    in_reply_to_status_id_str : null
    in_reply_to_user_id : null
    in_reply_to_user_id_str : null
    in_reply_to_screen_name : null
  ▶ user {39}
    geo : null
    coordinates : null
    place : null
    contributors : null
    quoted_status_id : 1396828416083517400
    quoted_status_id_str : 1396828416083517441
  ▶ quoted_status {28}
  ▶ quoted_status_permalink {3}
    is_quote_status : ☑ true
  ▶ extended_tweet {3}
    quote_count : 0
    reply_count : 0
    retweet_count : 0
    favorite_count : 0
  ▶ entities {4}
    favorited : ☐ false
    retweeted : ☐ false
    possibly_sensitive : ☐ false
    filter_level : low
    lang : ja
    timestamp_ms : 1622020697054
```


Gambar 3. Struktur JSON saat Menampung *Data Stream*

Dari struktur objek yang ada pada Gambar 3, dibuatlah suatu struktur tabel sementara (penampung) sebagaimana diberikan pada Tabel 4.

Tabel 4. Struktur Tabel Penampung *Data Stream*

Nama Kolom	Tipe Data	Keterangan
idx	int(11) primary, auto_increment	*auto generated* id
id	varchar(255)	nomor otomatis
json	text	*response* asli dari twitter stream api
tweet_text	text	*tweet text*
tweet_text_clean	text	*tweet text* setelah dilakukan pra proses.
user_id	varchar(255)	id dari *user*
user_screen_name	varchar(255)	nama *screen twitter*
user_avatar_url	varchar(255)	gambar profil *user*
geo	varchar(255)	alamat geo lokasi
coordinates	varchar(255)	*coordinate user update*
places	varchar(255)	tempat
created_at	timestamp	tanggal *insert table*

Nama Kolom	Tipe Data	Keterangan
updated_at	timestamp	tanggal *update table*
sentiment_analysis	varchar(25)	hasil dari perhitungan sentimen
lang	varchar(10)	bahasa aplikasi yang digunakan *user*
source	varchar(255)	perangkat yang digunakan user untuk melakukan *tweet*

4. Dokumentasi Pengembangan

A. *Twitter Stream Dataset*

Proses pengumpulan data dilakukan oleh *service* yang dibangun menggunakan *framework Laravel* dengan melakukan *streaming* pada media sosial *Twitter*. Untuk dapat melakukan *streaming* pada media sosial *Twitter* tersebut, dibutuhkan *API Key* yang diperoleh secara resmi melalui laman pengembang aplikasi pada media sosial *Twitter*. Selain itu, untuk mendukung layanan *streaming* juga digunakan *library Phirehose*, yaitu sebuah *library* berbasis PHP yang dapat melakukan koneksi dan *streaming* data dari media sosial Twitter.

Service yang telah berhasil dibuat tersebut bekerja secara *real time* dan berkelanjutan selama periode pengumpulan data, yaitu antara tanggal 26 Mei 2021 hingga 6 Juni 2021. *Service* ini bekerja mengumpulkan data *tweet* yang mengandung kata kunci yang telah ditentukan. *Tweet* yang telah diperoleh kemudian disimpan di dalam basis data MySQL dengan struktur data yang diberikan dalam Tabel 5.

Tabel 5. Struktur Tabel untuk Penyimpanan Hasil *Data Stream*

Object	Value
id	ID *tweet*
json	Semua data *tweet* dalam format JSON
tweet_text	Isi *text* dalam *tweet* yang belum dilakukan proses *cleaning*
tweet_text_clean	Berisi nilai kosong yang kemudian akan diisi oleh *text* dalam *tweet* yang sudah dilakukan proses *cleaning*
user_id	ID pengguna yang melakukan *tweet*
user_screen_name	*Username* pengguna yang melakukan *tweet*
user_avatar_url	URL avatar pengguna yang melakukan *tweet*
geo	Izin pengguna untuk pelacakan *geolocation* pada tweet, berisikan nilai *TRUE* atau *FALSE*
coordinates	Koordinat lokasi saat *tweet* dibuat, berisikan nilai *latitude* dan *longitude*
places	Tempat/lokasi saat *tweet* dibuat (jika kolom ini tidak memiliki nilai, maka kolom akan berisikan nilai keterangan lokasi pada data *user_location* pengguna)
sentiment_analysis	Nilai kosong yang kemudian akan diisi oleh nilai sentimen yang telah diproses oleh layanan pengolahan sentimen
created_at	*Timestamp* input data
updated_at	*Timestamp* update data

Proses *streaming* ini menghasilkan data sebanyak 31.787 baris yang kemudian akan dilakukan teks praposes untuk membersihkan kolom *tweet_text* dari *stopwords* yang tidak relevan untuk kebutuhan proses analisis selanjutnya. Dalam proses streaming, terdapat data yang tidak konsisten dimana data ini menjadi salah satu data yang penting dalam pemetaan sentimen ke depannya. Dalam *object* data yang didapatkan dari data *stream*, 90% dari data tidak memiliki *location* dan geo. Untuk meng-*handle* masalah ini, apabila *tweet* tersebut tidak memiliki *location* dan geo maka data yang disimpan di dalam *database* ada user *location* yang terdapat pada *object user*.

B. *Text Preprocessing*

Proses text processing dilakukan pada data stream twitter yang disimpan di dalam *database*, proses *batch* dilakukan dalam kurun waktu tertentu dimana *service* mengambil data *twitter* yang ada pada baris dan akan melakukan proses *preprocessing* seperti:

a. *Case Folding*
b. *Remove Punctuation*
c. *Remove Username*
d. *Remove Hashtag*
e. *Clean Number*
f. *Clearn On Character*
g. *Remove URL*
h. *Remove RT*
i. *Convert Number*
j. *Remove Stop Word*
k. *Convert Word*
l. *Convert Emotion*

Kode untuk melakukan *preprocessing* ini diberikan dalam Tabel 6.

Tabel 6. Kode Program untuk *Preprocessing*

```
#Case Folding
tweet = tweet.lower()

# Remove Punctuation
table = str.maketrans(dict.fromkeys(string.punctuation))
# OR
{key: None for key in string.punctuation}
tweet = tweet.translate(table)

# Remove username
tweet = re.sub("@[A-Za-z0-9]+","",tweet) #Remove @ sign

# Remove Hashtag
```

```
tweet = tweet.replace("#", "").replace("_", " ")
#Remove hashtag sign but keep the text

# Clean Number
# Clean One Character
# Remove URL
tweet = re.sub(r"(?:\@|http?\://|https?\://|www)\S+", "",
tweet)
#Remove http links
tweet = " ".join(tweet.split())

# Remove RT
retweet_key = ['rt', 'RT', 'RT@', 'rt@', '☑', '...']
retweet_query = tweet.split()
resultwords = [word for word in retweet_query if word.lower()
not in retweet_key]
tweet = ' '.join(resultwords)
L = []
for word in tweet.split():
    if not word.endswith('…'):
        L.append(word)
tweet = ' '.join(L)

# Convert Number
numberPattern = r'[0-9]'
tweet = re.sub(numberPattern, '', tweet)

# Remove Stop Word
text_tokens = word_tokenize(tweet)
tokens_without_sw = [word for word in text_tokens if not word
in stopwords.words('indonesian')]
tweet = (" ").join(tokens_without_sw);

# Convert Word
# Convert Emotion
regrex_pattern = re.compile(pattern = "["
    u"\U0001F600-\U0001F64F"  # emoticons
```

```
        u"\U0001F300-\U0001F5FF"  # symbols & pictographs
        u"\U0001F680-\U0001F6FF"  # transport & map symbols
        u"\U0001F1E0-\U0001F1FF"  # flags (iOS)
                                "]+", flags = re.UNICODE)
    tweet =  regrex_pattern.sub(r'',tweet)
```

Dalam proses ini setiap kata akan dirubah sesuai dengan tahapan sebagaimana diberikan pada Tabel 6.

places
New York, NY
(NULL)
(NULL)
ورقة كتاب.
(NULL)
Murcia (España)
Kogi, Nigeria
(NULL)
(NULL)
Lagos
◆ s/h
Lagos,Nigeria.
(NULL)
Jos, Nigeria
Mumbai
she/her ¦fan account¦anikpoptw¦read carrd byf
she/her
Mi 力 Sa % (^O^☆♪
(NULL)
yachi simp
: 21+
(NULL)
(NULL)
rivamika; gojohime; fushikugi
(NULL)
(NULL)
??
(NULL)
(NULL)
Abuja, Nigeria
#DéViaggio
Abuja, Nigeria
NSA - #BAEKSANG - #disneyfams
(NULL)
地球
(NULL)
Masaka, Uganda

Gambar 4. Pembersihan Data Lokasi di Luar Indonesia

Di bawah ini adalah contoh perubahan *tweet* sebelum di-*preprocess* dan setelah di-*preprocess*:

```
Batuk-batuk mulu ampun, deg-degan banget tapi hamdallah dah antigen
2x pun ya negatif huhu takut bgt :")

Ok susah nak batuk susah nak bersin sebab perut aku sakit. Sakit
sangat ye muscle pain ni.
```

Menjadi:

```
batuk batuk mulu ampun deg degan banget tapi hamdallah dah antigen
2x pun ya negatif huhu takut bgt

ok susah nak batuk susah nak bersin sebab perut aku sakit sakit
sangat ye muscle pain ni
```

Process *cleaning* dan *preprocessing* dilakukan pada data-data yang sudah disimpan di dalam *database* MySQL. Setiap *tweet* akan dilakukan proses setiap detik sekali. Pada proses ini, setiap data yang sudah diolah akan dimasukan kembali ke dalam kolom *tweet_text_clean*, nantinya akan dilakukan kalkulasi dan proses perhitungan tfi-idf dan klasifikasi.

Pada implementasi, proses *preprocessing* dan juga *cleaning* serta *filtering* perlu dilakukan pada data *location* yang tersimpan pada *database*. Contoh data-data yang perlu dilakukan *clean* dan *preprocess* adalah data-data sebagaimana diberikan dalam Gambar 4. Lokasi-lokasi tersebut perlu dilakukan *cleaning* sehingga nantinya bisa dilakukan *filtering* data dengan baik.

C. TF-IDF

Setelah praposes data selesai, maka proses selanjutnya adalah proses *indexing*, perhitungan bobot dengan TF-IDF kemudian klasifikasi. Proses Indexing dan TF-IDF dilakukan bila terdapat *trigger* dari *user*, sedangkan proses klasifikasi dilakukan pada interval waktu tertentu yaitu setiap satu menit sekali. Ketiga proses ini dibuat dalam bahasa Java menggunakan Maven. Berikut pembahasan untuk masing-masing proses:

- Proses *Indexing*

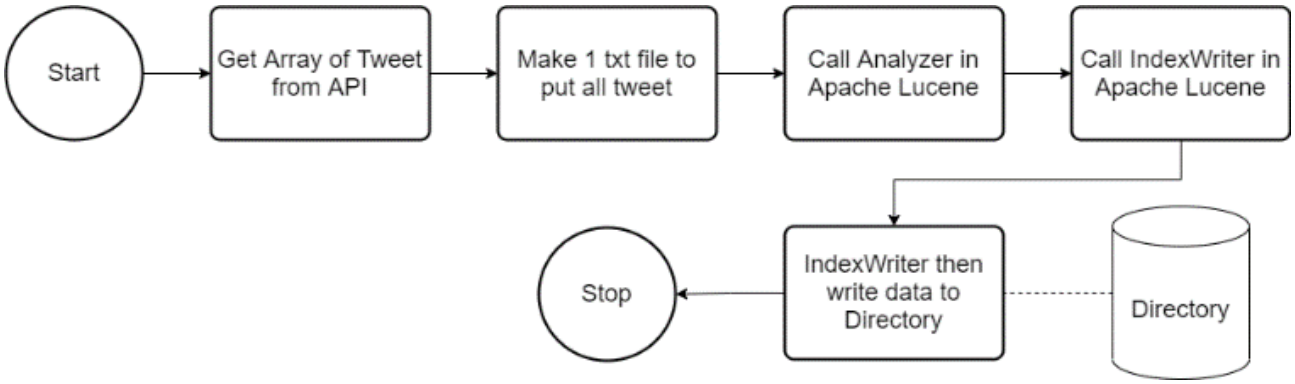
  Proses *indexing* dilakukan bila terdapat *trigger* dari *user*. Pada proses ini, *indexing* dilakukan dengan menggunakan library *apache lucene*. Ilustrasi penggunaan Lucene untuk proses *indexing* ini dapat dilihat pada Gambar 5 [7].

  Dokumen yang akan diolah didapatkan dari *tweet* yang diberikan oleh *API*. Dokumwn dalam kebutuhan ini dibentuk dalam satu txt-file yang berisi *array of tweet* yang kemudian data-data *tweet* tersebut diberikan kepada *Analyzer* untuk menganalisis dokumen dan akan memberikan perintah apakah harus meng-*update* indeks atau membuat indkes baru kepada *IndexWriter*.

  IndexWriter sendiri berfungsi untuk membuat dan meng-update indeks sesuai dengan perintah *Analyzer*, kemudian *IndexWriter* ini akan membuat sebuah direktori yang berisikan data-data (kata-kata) yang telah dibentuk indeksnya.

  Format JSON yang dipakai untuk proses ini adalah:

```
[
    {"tweet_text" : "kenalilah gejala tb"},
    {"tweet_text" : "tb adalah sakit menular"},
    {"tweet_text" : "hilangkan tb indonesia"}
]
```



Gambar 5. Proses Pembentukan Indeks pada Perangkat Lucene

Contoh *output* yang akan dikeluarkan dijelaskan pada proses TF-IDF.

- Proses TF-IDF

Proses selanjutnya adalah pembobotan *term* dari *tweet* menggunakan TF-IDF, pembobotan *term* ini masih menggunakan *Apache Lucene*, pembobotan *term* dilakukan untuk mendapatkan *context of discussion* dari *dataset*.

Dalam pengolahannya, data yang telah diindeks sebelumnya kemudian dipanggil kembali untuk dibaca dan dihitung TF dan IDF nya. *Apache Lucene* memiliki perhitungan tersendiri terhadap TF dan IDF. Untuk perhitungan TF menurut *Apache Lucene* [8] adalah **`tf(t in d) = frequency`**$^{1/2}$, akar pangkat dua dari banyaknya *term* tersebut muncul. Untuk perhitungan IDF menurut *Apache Lucene* [8] adalah **`idf(t) = 1 + log (docCount+1 / docFreq+1)`**, dimana `docFreq` adalah banyaknya dokumen dimana *term* yang dimaksud muncul dan `docCount` adalah banyaknya semua dokumen.

Dalam kebutuhan penelitian ini, semua *tweet* yang masuk akan dijadikan dalam satu dokumen, sehingga untuk `docCount` dan `docFreq` akan selalu statis bernilai 1. Hal ini dikarenakan perlu dihitungnya seberapa bernilainya sebuah *term* dalam satu kumpulan tertentu (Gambar 6).

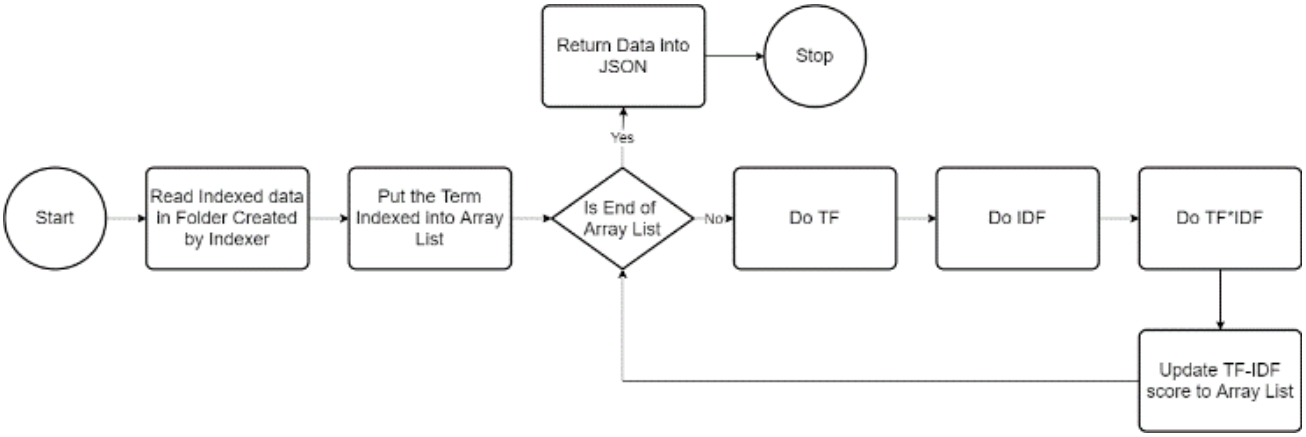
TF-IDF proses dilakukan dalam *loop condition* bergantung kepada banyaknya data yang telah diindeks pada proses sebelumnya, selanjutnya dilakukan proses perhitungan TF, lalu IDF lalu pengalian TF*IDF, kemudian di akhir *loop condition* data tersebut di-*return* dalam format JSON.

Format JSON yang dipakai pada proses ini untuk input sama seperti proses *indexing* karena proses *indexing* sampai TF-IDF dilakukan pada satu waktu, untuk format JSON output yang dikeluarkan adalah:

```
[
        {
          "term": "yang",
          "document": "Doc_1.txt",
          "score": 1.0
        },
        {
```

```
        "term": "untuk",
        "document": "Doc_1.txt",
        "score": 1.0
      },
]
```

Dimana *term* adalah katanya, *document* adalah variable statis yang selalu berisikan nilai "Doc_1.txt" dan *score* adalah hasil perhitungan TF dikali IDF untuk *term* tersebut. Bila sebuah *term* memiliki nilai yang besar, berarti *term* tersebut bermakna.

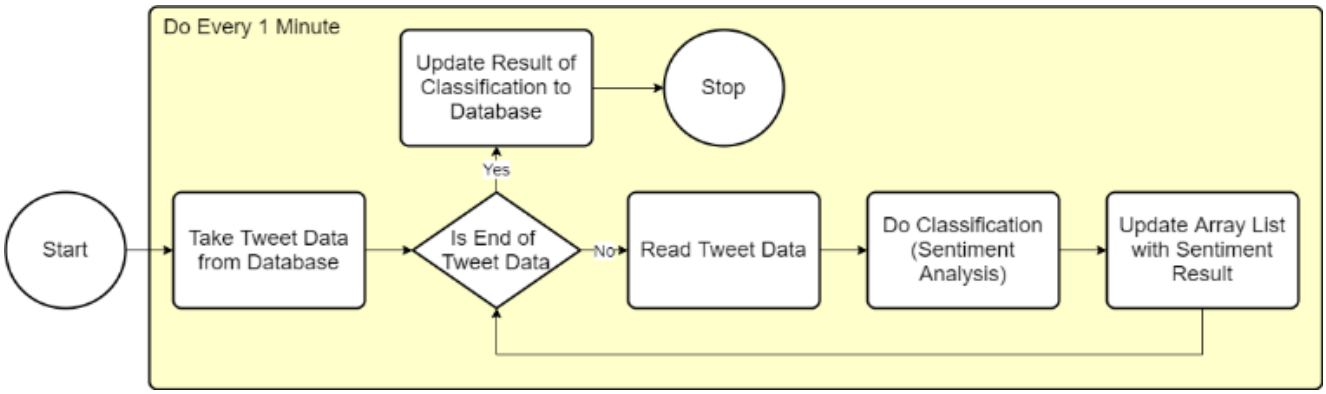


Gambar 6. Proses Penghitungan TF-IDF

D. *Classification*

Proses klasifikasi dilakukan dalam rentang waktu tertentu, waktu yang ditentukan adalah setiap satu menit sekali. Dalam proses klasifikasi ini, dikelompokkan setiap *tweet* ke dalam sentimen positif, negatif atau netral. Untuk proses ini, tidak dibangun dari awal, melainkan menggunakan Sengon Project. Sengon Project adalah aplikasi analisis sentimen berbasiskan Bahasa Indonesia yang ditulis dalam bahasa Java.

Proyek ini dibangun menggunakan Maven dan mengandung beberapa *library* yaitu OpenNLP, Apache Lucene dan Language Detector. Sengon Project ini menggunakan *lexicon classification* untuk teknik klasifikasinya [9].



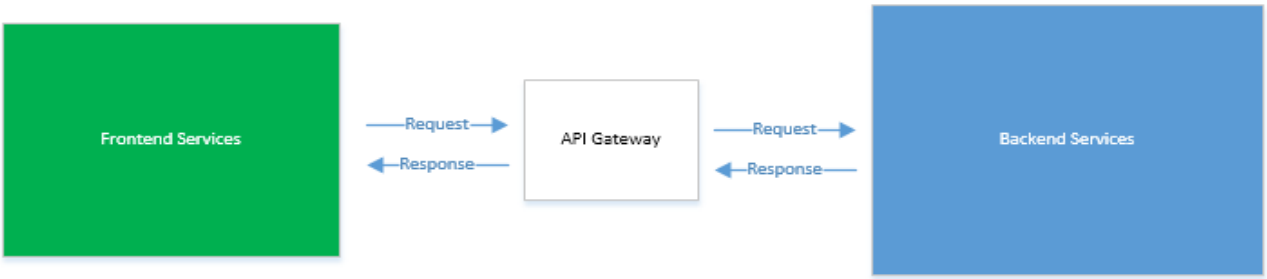
Gambar 7. Proses Penghitungan TF-IDF

Proses klasifikasi dimulai dengan mengambil data dari *database*, data yang diambil adalah data yang belum pernah diproses sebelumnya (memiliki nilai sentimen `null`) dan telah dilakukan proses teks

praproses. Data yang diambil adalah maksimal lima puluh data untuk menghindari terjadinya *deadlock*. Data yang telah diambil tersebut kemudian disimpan dalam *array list* yang akan dilakukan proses *looping* untuk dilakukannya klasifikasi dan setiap hasil klasifikasi yang diterima akan dilakukan *update* terhadap *array list* tersebut, yang kemudian secara bersamaan akan dilakukan *update* hasil sentimen ke dalam *database*. Proses selengkapnya untuk klasifikasi dapat dilihat dalam Gambar 7.

E. *API Gateway Services*

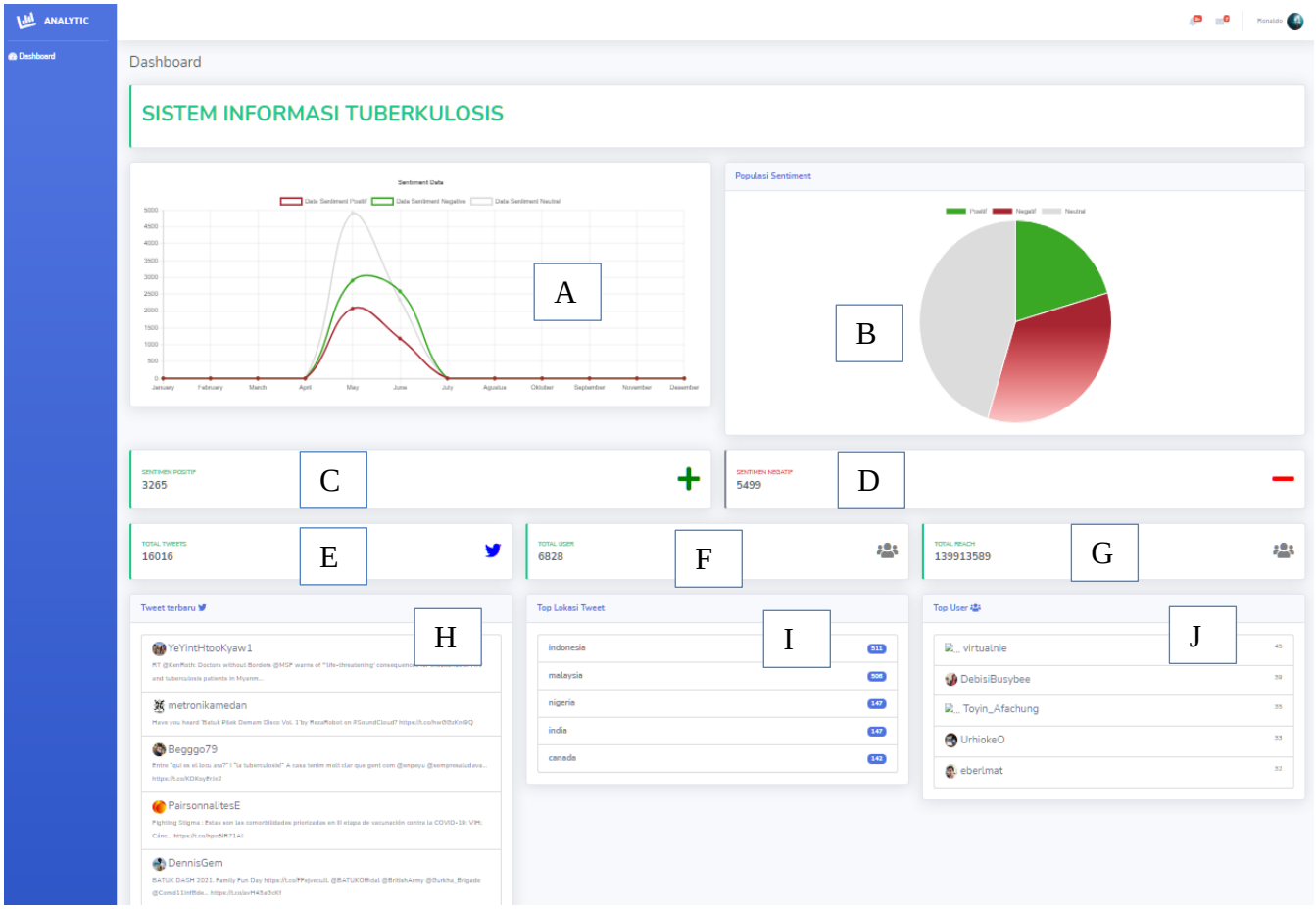
API Gateway Service dibangun dengan menggunakan *Framework* Laravel sebagai *base framework*. *Service* yang dibuat ini memiliki fungsi untuk melakukan pemanggilan data ke dalam *dataset* yang telah dilakukan proses *preprocessing*, TF-IDF dan juga klasifikasi (Gambar 8). *Service* yang telah dibuat akan mengirimkan *response* dengan format JSON yang nantinya oleh *service frontend* akan ditampilkan kepada *user*.



Gambar 8. Konsep Eksekusi API Ditinjau dari *Frontend* dan *Backend*

F. *Frontend Services*

Frontend Service dibangun dengan menggunakan *framework* VueJS, dimana *framework* ini melakukan *rendering script* di sisi *client* sehingga proses *load* suatu aplikasi menjadi lebih cepat karena sebagian prosesnya dilakukan di sisi *browser*. Konsep interaksi dengan pengguna melalui *request API* pada *server* pada *web browser* dan hasil pengembalian dari *frontend service* diberikan dalam Gambar 9.



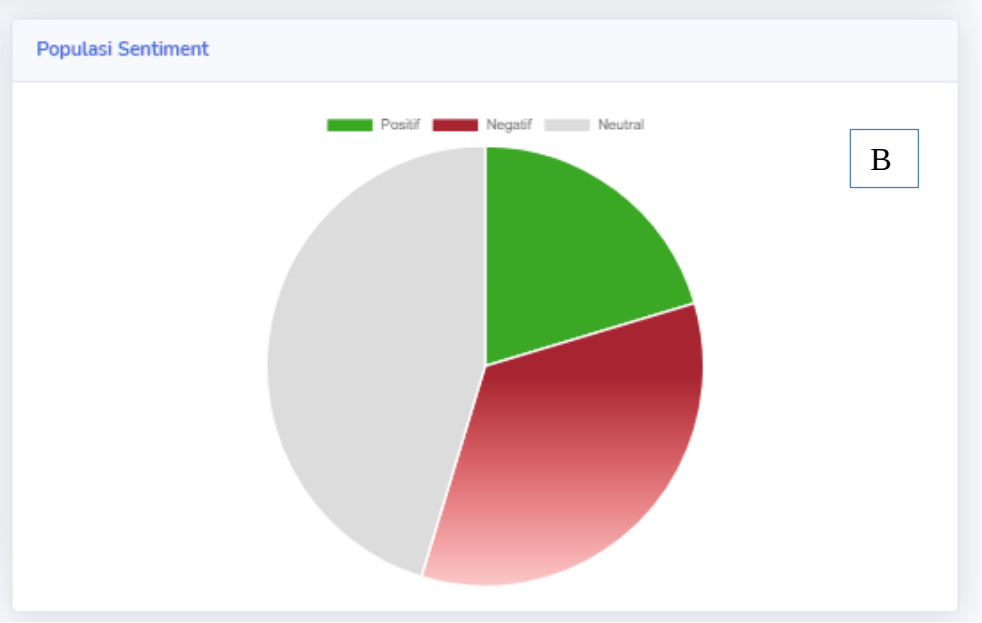
Gambar 9. Hasil Tampilan pada *Frontend*

Pada *service frontend* ini, *user* dapat melihat beberapa fitur seperti:

- Grafik sentimen berdasarkan bulan kemunculan
  Grafik ini akan menampilkan kurva jumlah *tweet* yang mengandung kata-kata yang telah di-*filter* sebelumnya (Gambar 10).

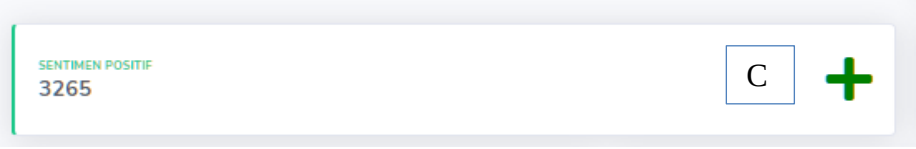
Gambar 10. Hasil Tampilan pada *Frontend*

- *Pie Chart* Sebaran Sentimen
  Grafik ini akan menampilkan persentase sentimen positif, negatif dan juga netral dalam bentuk *chart* sehingga *user* mudah untuk melihat persentase sentimen yang ada pada *tweet* saat ini (Gambar 11).



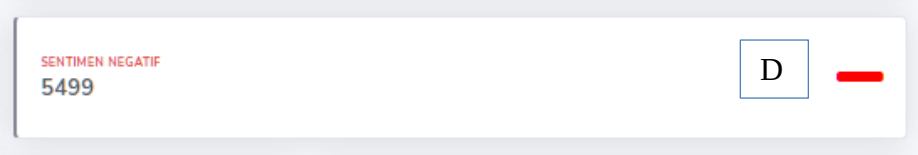
Gambar 11. Persentase Perbandingan Hasil Analisis Sentimen Positif dan Negatif

- Jumlah sentimen positif
  Bagian ini menampilkan jumlah sentimen positif dari *tweet* yang ada (Gambar 12).



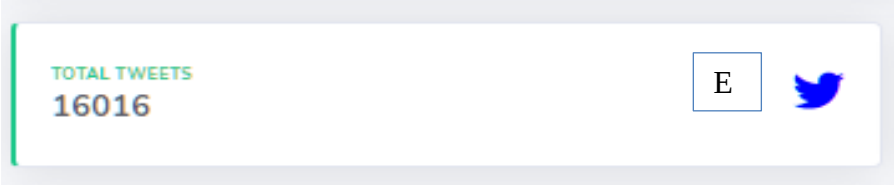
Gambar 12. Jumlah Sentimen Positif

- Jumlah sentimen negatif
  bagian ini menampilkan jumlah sentimen negatif dari *tweet* yang ada (Gambar 13).



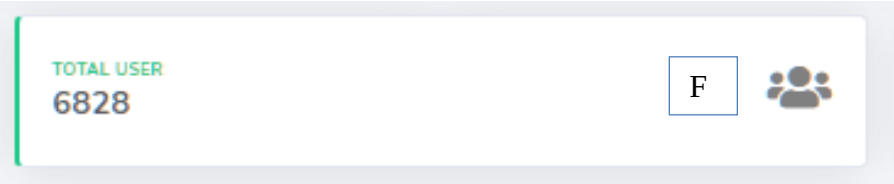
Gambar 13. Jumlah Sentimen Negatif

- Total *tweet* terkait *tuberculosis*
  Menampilkan jumlah total *tweet* yang mengandung kata yang telah di-*filter* (Gambar 14).



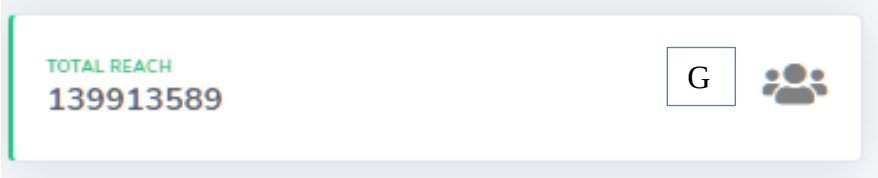
Gambar 14. Jumlah *Tweet* Terkait dengan *Tuberculosis*

- Total *user* yang aktif
  Menampilkan jumlah total *user* yang paling sering mencuitkan tentang topik yang di-*filter* (Gambar 15).



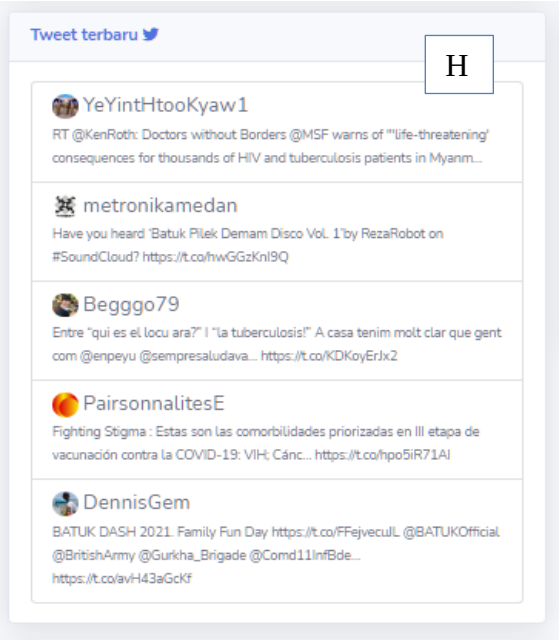
Gambar 15. Jumlah *User* yang Sering Mencuitkan Topik *Tuberculosis*

- Total *reach user*
  Menampilkan total jumlah *reach* dari *user* (Gambar 16).



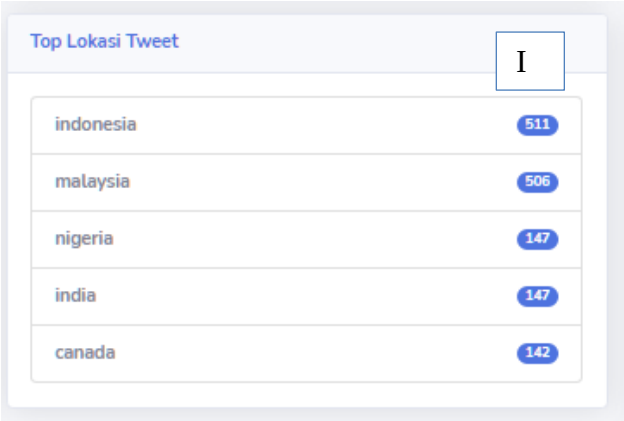
Gambar 16. Jumlah Keseluruhan *User Reach* yang *Tweet*-nya Dianalisis

- *Tweet* terbaru
  Menampilkan *tweet* terbaru tentang topik *tuberculosis* (Gambar 17).



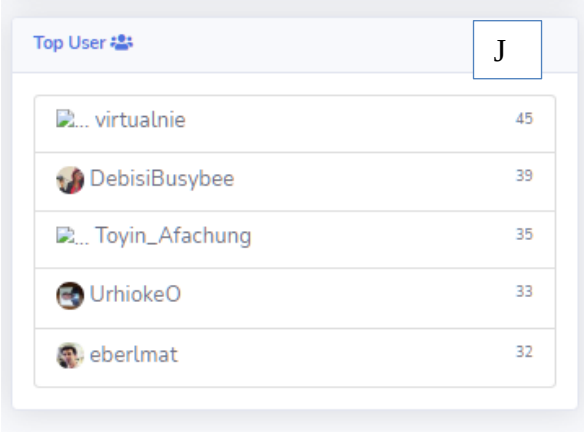
Gambar 17. Lima *Tweet* Terbaru Terkait Topik *Tuberculosis*

- Top lokasi *tweet*
  Menampilkan kata kunci lokasi yang paling sering melakukan *tweet* terkait *tuberculosis* (Gambar 18).



Gambar 18. Jumlah Kata Kunci pada Lokasi yang Memunculkan Topik *Tuberculosis*

- Top *user*
  Menampilkan *user* teratas dalam mengirimkan *tweet* mengenai *tuberculosis* (Gambar 19).



Gambar 19. Jumlah Kata Kunci pada Lima *User* yang Memunculkan Topik *Tuberculosis* Terbanyak

5. Pengujian Sistem

Sistem Informasi *Tuberculosis* ini memiliki keunggulan seperti:

1. Sistem ini dapat memberikan informasi tambahan terkait *tuberculosis*. Dengan adanya *dashboard* ini, *user* dapat mengatahui sentimen-sentimen yang muncul dengan kata kunci *tuberculosis* dan sebarannya.
2. Sistem ini memiliki potensi sangat besar dimana setiap komponen dan juga fungsinya bisa dilakukan optimasi yang dapat meningkatkan akurasi dari sentimen terhadap suatu cuitan / *tweet*.
3. Sistem ini dibangun dengan menggunakan arsitektur *microservices*, arsitektur ini mendukung peningkatan skala menjadi lebih besar seiring dengan meningkatnya jumlah informasi yang diberikan dan jumlah informasi yang akan diolah.

4. Dengan arsitektur yang ada, sangat memungkinkan untuk melakukan penambahan sumber data tanpa harus merombak seluruh aplikasi. Untuk menambahkan sumber data, hanya diperlukan menambahkan *service* baru yang nantinya bisa langsung diproses.

6. Rencana Versi Selanjutnya

Proses pengolahan informasi terkait *tuberculosis* pada media sosial sangat membantu dalam penambahan informasi terkait penyebaran dan juga sentimen masyarakat. Dengan menggunakan fitur-fitur yang ada di dalam sistem ini, *user* dapat melihat sebaran data, jumlah sentimen positif atau negatif. Dengan adanya informasi terkait sentimen yang ada, *user* dapat menambahkan dan melakukan identifikasi lebih dalam terhadap sebaran yang diinformasikan pada sistem ini.

Seiring dengan berkembangnya informasi yang ada pada media sosial, maka semakin banyak data yang perlu diolah dan juga diproses. Oleh karena itu, diperlukan proses pembelajaran data, dimana data yang masuk perlu dilakukan pengecekan dengan metode *Machine Learning*. Data yang masuk akan dilatih dengan model tertentu, dimana model tersebut akan dijadikan suatu perhitungan dan acuan dalam menentukan sentimen-sentimen yang ada. Lalu secara berkala model tersebut selalu dilatih dan diolah agar tingkat akurasi sentimen menjadi sangat tinggi. Salah satu isu penting yang perlu ditanggulangi adalah penanganan bahasa-bahasa daerah dan kata-kata informal yang seringkali tercampur dengan bahasa Indonesia baku. Semakin tinggi akurasi dari suatu data sentimen, maka data yang dihasilkan akan semakin membantu *user* dalam mengambil keputusan.

Pustaka Pendukung

[1] (2018) InfoDATIN Pusat Data dan Informasi Kementrian Kesehatan RI. [Online]. Tersedia di: https://pusdatin.kemkes.go.id/resources/download/pusdatin/infodatin/infodatin-tuberkulosis-2018.pdf

[2] (2021) Sistem Informasi Tuberkulosis. [Online]. Tersedia di: http://sitb.id/sitb/about

[3] Nugraha Kristian Adi, Danny Sebastian, "Pembentukan Dataset Topik Kata Bahasa Indonesia pada Twitter Menggunakan TF-IDF & Cosine SImilarity," *Jurnal Teknik Informatika dan Sistem Informasi*., vol. 4, pp. 376-386, Dec. 2018.

[4] Ji. Xiang, Soon Ae Chun, Zhi Wei, James Geller, "Twitter sentiment classification for measuring public health concerns." *Social Network Analysis and Mining, Springer*., vol. 11, pp. 1-25, May. 2015.

[5] Gaydhani Aditya, Vikrant Doma, Shrikant Kendre, Laxmi Bhagwat, "Detecting Hate Speech and Offensive Language on Twitter using Machine Learning: An N-gram and TFIDF based Approach," *IEEE International Advance Computing Converence*., Sep. 2018.

[6] O'Dea Bridianne, Stephen Wang, Philip J. Batterham, Alison L. Calear, Cecile Paris, Helen Christensen, "Detecting suicidality on Twitter," *Internet Interventions, Elsevier*, vol. 2, pp. 183-188, May. 2015.

[7] (-) Lucene – Indexing Process. [Online]. Tersedia di: https://www.tutorialspoint.com/lucene/lucene_indexing_process.htm

[8] (-) Class TFIDFSimilarity. [Online]. Tersedia di: https://lucene.apache.org/core/7_6_0/core/org/apache/lucene/search/similarities/TFIDFSimilarity.html

[9] (2017) Sengon Project. [Online]. Tersedia: https://github.com/masasdani/sengon

[10] Adityawan E. 2014. Analisis sentimen dengan klasifikasi naïve bayes pada pesan Twitter menggunakan data seimbang [skripsi]. Bogor (ID): IPB

[11] Guerini M, Gatti L, Turchi M. 2013. Sentiment analysis: How to derive prior polarities from SentiWordNet. Di dalam: Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing; 2013

okt 18-21; Washington, Amerika Serika (US): The Association for Computational Linguistics. hal 1259-1269.