TUGAS AKHIR – IS184853

PENERAPAN ALGORITMA *COSINE SIMILARITY* DAN PEMBOBOTAN TF-IDF PADA SISTEM CHATBOT UNTUK CALON MAHASISWA ITS

APPLICATION OF COSINE SIMILARITY ALGORITHM AND TF-IDF WEIGHTING IN CHATBOT SYSTEMS FOR ITS PROSPECTIVE STUDENTS

FIRMAN HIDAYAT
NRP 05211540000149

Dosen Pembimbing
Ahmad Muklason, S.Kom., M.Sc., Ph.D.

DEPARTEMEN SISTEM INFORMASI
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya 2019

TUGAS AKHIR – IS184853

PENERAPAN ALGORITMA *COSINE SIMILARITY* DAN PEMBOBOTAN TF-IDF PADA SISTEM CHATBOT UNTUK CALON MAHASISWA ITS

FIRMAN HIDAYAT
NRP 05211540000149

Dosen Pembimbing
Ahmad Muklason, S.Kom., M.Sc., Ph.D.

DEPARTEMEN SISTEM INFORMASI
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya 2019

Halaman ini sengaja dikosongkan

FINAL PROJECT – IS184853

APPLICATION OF COSINE SIMILARITY ALGORITHM AND TF-IDF WEIGHTING IN CHATBOT SYSTEMS FOR ITS PROSPECTIVE STUDENTS

FIRMAN HIDAYAT
NRP 05211540000149

Supervisor
Ahmad Muklason, S.Kom., M.Sc., Ph.D.

INFORMATION SYSTEMS DEPARTMENT

Faculty of Information and Communication Technology

Sepuluh Nopember Institut of Technology

Surabaya 2019

Halaman ini sengaja dikosongkan

LEMBAR PENGESAHAN

PENERAPAN ALGORITMA *COSINE SIMILARITY* DAN PEMBOBOTAN TF-IDF PADA SISTEM CHATBOT UNTUK CALON MAHASISWA

TUGAS AKHIR

Disusun Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Departemen Sistem Informasi
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember

Oleh:

FIRMAN HIDAYAT
NRP. 0521 15 40000 149

Surabaya, Juli 2019

KEPALA
DEPARTEMEN SISTEM INFORMASI

Mahendrawathi ER, S.T., M.Sc., Ph.D
NIP. 19761011 200604 2 001

Halaman ini sengaja dikosongkan

LEMBAR PERSETUJUAN

PENERAPAN ALGORITMA *COSINE SIMILARITY* DAN PEMBOBOTAN TF-IDF PADA SISTEM CHATBOT UNTUK CALON MAHASISWA ITS

TUGAS AKHIR

Disusun Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Departemen Sistem Informasi
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember

Oleh :

FIRMAN HIDAYAT
NRP. 05211540000149

Disetujui Tim Penguji : Tanggal Ujian : 8 Juli 2019
Periode Wisuda : September 2019

Ahmad Muklason, S.Kom., M.Sc., Ph.D. **(Pembimbing I)**

Wiwik Anggraeni, S.Si., M.Kom. **(Penguji I)**

Faizal Mahananto, S.Kom, M.Eng., Ph.D. **(Penguji II)**

DEPARTEMEN SISTEM INFORMASI

Halaman ini sengaja dikosongkan

PENERAPAN ALGORITMA *COSINE SIMILARITY* DAN PEMBOBOTAN TFIDF PADA SISTEM CHATBOT UNTUK CALON MAHASISWA ITS

Nama Mahasiswa	**:**	**Firman Hidayat**
NRP	**:**	**05211540000149**
Departemen	**:**	**Sistem Informasi**
Dosen Pembimbing	**:**	**Ahmad Muklason, S.Kom., M.Sc., Ph.D**

ABSTRAK

Akhir-akhir ini penggunaan chatbot sedang marak digunakan pada organisasi. Hal ini dikarenakan chatbot merupakan salah satu solusi untuk menjawab pertanyaan yang sering ditanyakan oleh pelanggan, atau frequently asked questions (FAQ) bagi suatu organisasi. Dengan adanya chatbot ini membuat kemampuan CRM pada suatu organisasi berjalan dan bisa digunakan untuk berhubungan dengan customer secara realtime dan aktif selama 24/7. Salah satu cara mengembangkan aplikasi chatbot adalah dengan menggunakan pendekatan information retrieval yaitu menggunakan metode pembobotan Term Frequency – Inverse Document Frequency dan Algoritma Cosine Similarity. Berdasarkan hasil penelitian pada tugas akhir ini, metode pembobotan TF-IDF dan algoritma Cosine Similarity memiliki kemampuan yang baik untuk digunakan dalam aplikasi chatbot karena sistem chatbot dapat menfidentifikasi pertanyaan dengan rata – rata kemiripan 0.662550216 untuk pertanyaan yang belum pernah ditanyakan maupun yang pernah ditanyakan sebelumnya.

Kata Kunci: Chatbot, Information retrieval, Term Frequency, Inverse Document Frequency, FAQ, Cosine Similarity

APPLICATION OF COSINE SIMILARITY ALGORITHM AND TF-IDF WEIGHTING IN CHATBOT SYSTEMS FOR ITS PROSPECTIVE STUDENTS

Name : Firman Hidayat
NRP : 05211540000149
Department : Information Systems
Supervisior : Ahmad Muklason, S.Kom., M.Sc., Ph.D

ABSTRACT

On this day, the use of chatbots is being used in much organizations. This is because chatbots are one of the solution to answering frequently asked questions (FAQs) from customers, for an organization. Chatbot makes CRM capabilities in an organization can running and can be used to connect with customers in real time and be active 24/7. One of ways to develop chatbot applications is using an information retrieval approach that is using the Term Frequency weighting method - Inverse Document Frequency and the Cosine Similarity Algorithm. Based on the results of this research, the weighting method of TF-IDF and the Cosine Similarity algorithm have good ability to be used in chatbot applications because chatbot can specify questions with an average similarity of 0.662550216 for questions that have never been asked whether they have been asked before.

Keywords: Chatbot, Information retrieval, Term Frequency, Inverse Document Frequency, FAQ, Cosine Similarity

KATA PENGANTAR

Alhamdulillahi robbil 'alamin, segala puji bagi Allah SWT atas limpahan nikmat, rahmat, petunjuk, dan karunia-Nya, sehingga penulis dapat menyelesaikan laporan penelitian tugas akhir dengan judul, "**PENERAPAN ALGORITMA *COSINE SIMILARITY* DAN PEMBOBOTAN TF-IDF PADA SISTEM CHATBOT UNTUK CALON MAHASISWA ITS**" yang menjadi salah satu syarat kelulusan pada Departemen Sistem Informasi, Fakultas Teknologi Informasi dan Komunikasi, Institut Teknologi Sepuluh Nopember Surabaya.

Pengerjaan tugas akhir ini tidak lepas dari bantuan, bimbingan, dukungan, maupun doa dari berbagai pihak. Oleh karena itu, izinkan penulis menyampaikan rasa terima kasih yang sebesar-besarnya kepada,

1. Kedua orang tua penulis, Bapak Hanipan dan Ibu Surkanah yang tidak pernah lelah dan tak pernah henti memberikan wejangan, dukungan, motivasi, serta doa yang tulus dan ikhlas demi yang terbaik untuk penulis.
2. Kakak penulis, Heni Wahyuni, yang telah memberikan dukungan, baik secara moril maupun materil, serta arahan untuk kebaikan penulis.
3. Bapak Ahmad Muklason, S.Kom., M.Sc., Ph.D. selaku dosen pembimbing yang telah meluangkan banyak waktu dan selalu sabar membimbing serta mengarahkan penulis dalam pengerjaan tugas akhir ini.

4. Ibu Wiwik Anggraeni, S.Si., M.Kom. dan Bapak Faizal Mahananto, S.Kom., M.Eng., Ph.D. selaku dosen penguji yang telah memberikan kritik dan saran yang membangun dalam proses pengerjaan tugas akhir ini.
5. Rekan-rekan pejuang Laboratorium RDIB yang selalu menemani dan mendukung serta menjadi tempat diskusi penulis dalam pengerjaan tugas akhir hingga tak kenal waktu saat di Lab RDIB.
6. Teman-teman Lannister yang selalu memberikan dukungan dan hiburan tersendiri bagi penulis. Tetaplah menjadi *beton terbang bersinar*.
7. Teman-teman Jambanozta yang selalu memberikan hiburan tersendiri bagi penulis. Tetaplah menjadi seperti itu.
8. Gamal Akbar dan Abdul Azizun selaku teman kos penulis yang selalu memberikan dukungan dan bantuan bagi penulis.

Semoga semua bentuk dukungan maupun doa menjadi catatan amal kebaikan di hadapan Tuhan Yang Maha Esa. Penulis juga menyadari pengerjaan penelitian tugas akhir ini masih jauh dari kata sempurna. Oleh karena itu, penulis menerima pertanyaan, saran, dan kritik yang membangun untuk menjadi masukan penulis dan masukan penelitian selanjutnya. Semoga penelitian tugas akhir dapat memberikan manfaat bagi pembaca.

Surabaya, Juli 2019

Penulis

DAFTAR ISI

DAFTAR GAMBAR

Halaman ini sengaja dikosongkan

DAFTAR TABEL

Halaman ini sengaja dikosongkan

DAFTAR KODE

BAB I
PENDAHULUAN

Pada bab ini, akan dijelaskan tentang Latar Belakang Masalah, Perumusan Masalah, Batasan Masalah, Tujuan Tugas Akhir, Manfaat Kegiatan Tugas Akhir dan Relevansi dengan laboratorium RDIB.

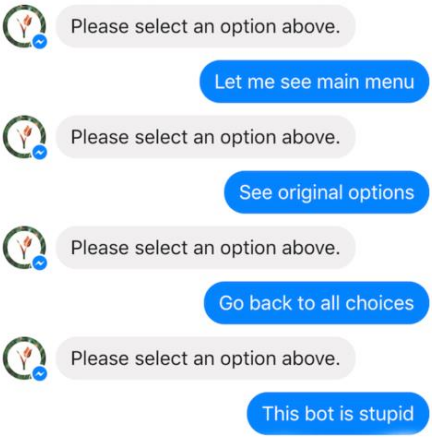
1.1. Latar Belakang

Chatbot adalah sebuah program komputer yang ditujukan untuk melakukan percakapan dengan orang lain menggunakan kecerdasan buatan [1]. *Chatbot* didesain untuk menstimulasi percakapan dengan pengguna manusia, khususnya lewat internet. Jadi *chatbot* bertindak seolah olah menjadi manusia yang dibuat untuk memfasilitasi komunikasi antara manusia dan komputer, mengerti tentang pertanyaan dengan bahasa natural dan menjawab sesuai dengan konteksnya. Menurut Zadrozny et al. setuju bahwa cara yang paling bagus dalam menjembatani antara manusia dan komputer adalah dengan memberikan kebebasan pengguna akan menyampaikan kepentingannya, keinginannya atau *Query* secara langsung dengan bahasa natural, dengan berbicara, mengetik atau menunjuknya [2].

Chatbot saat ini menjadi topik yang sedang hangat dibicarakan pada berbagai industri saat ini. Menurut survei yang telah dilakukan Oracle, 6 dari 10 top aplikasi terpopuler di dunia adalah *messaging app* dan memiliki *user* sebanyak 4,1 Miliar [3].

Sebagai bagian terdepan dalam industri, chabot berperan penting dalam bisnis. *Chatbot* berperan penting dalam menghubungkan pelanggan dan organisasi. Pada survei yang telah dilakukan oleh Oracle, sebanyak 65% orang lebih memilih menggunakan *messaging app* untuk melakukan bisnis [3]. Hal ini membuktikan bahwa budaya orang dalam melakukan bisnis sudah bergeser. Kemudian terdapat fakta [3] bahwa sebanyak 50% pelanggan memilih untuk memesan sebuah produk atau jasa melalui *messaging app*. Data terakhir adalah lebih dari 50% pelanggan menginginkan bisnis terbuka dalam 24/7 [3]. Hal ini tentu menjadi alasan positif bagi *chatbot* untuk menjadi trend karena *chatbot* memberikan semua ekspektasi pelanggan seperti pada survei yang telah dilakukan oleh Oracle.

Bahasa yang digunakan dalam *chatbot* konvensional masih sangat terbatas seperti menggunakan kata kunci tertentu atau biasa disebut dengan desain pixel centric *chatbot* seperti yang terjadi pada Gambar 1.1 [4].



Gambar 1.1 ***Contoh percakapan pada pixel centric chatbot***

Pada kejadian realitanya banyak pengguna merasa tidak nyaman dengan desain tersebut. Hal ini tepat seperti yang dikatakan oleh Zadrozny et al. mengenai setuju bahwa cara yang paling bagus dalam menjembatani antara manusia dan komputer adalah dengan memberikan kebebasan pengguna akan menyampaikan kepentingannya dengan bahasa yang natural [2]. Jika hal ini terus berlanjut maka konsep *conversational interface* [5] tidak bisa tercapai dan user engagement juga akan gagal diraih. Hal ini juga akan mengakibatkan kerugian seperti kepercayaan pelanggan menurun dan tentu akan merugikan organisasi tersebut.

Solusi yang dikemukakan pada penelitian tugas akhir ini adalah dengan menggunakan teknik *information retrieval* dengan metode pembobotan *Term Frequency – Inverse Document Frequency* (TF-IDF) [6]. Metode TF-IDF

merupakan gabungan dari metode *Term Frequency* (TF) dan *Inverse Document Frequency* (IDF).

Metode *Term Frequency* digunakan untuk mencari berapa banyak munculnya setiap kata dalam satu dokumen [7]. Semakin banyak kata yang muncul pada suatu dokumen maka semakin besar kemungkinan relevansi dokumen terhadap *Query* pengguna. Metode *Term Frequency* dipilih karena bisa mengatasi dataset yang sedikit dan implementasi metode yang mudah.

Kemudian metode *Inverse Document Frequency* adalah metode perhitungan berapa banyak informasi dokumen yang disediakan oleh kata tersebut [7]. Contohnya adalah berapa sering atau jarangnya kata yang muncul pada semua dokumen. *Inverse Document Frequency* muncul dengan konsep mengurangi bobot kata yang sering muncul dan meningkatkan bobot pada kata yang jarang muncul pada dokumen. Metode *Inverse Document Frequency* dipilih karena mengatasi permasalahan dari metode *Term Frequency*. Kelemahan dari metode *Term Frequency* adalah cenderung melakukan kesalahan dalam melihat dokumen dengan penggunaan kata yang berulang-ulang seperti kata "*the*" [7]. Hal ini dapat diatasi oleh metode *Inverse Document Frequency* sehingga mendapatkan performa yang maksimal.

Untuk memanfaatkan pembobotan dari metode TF-IDF maka dipilihlah algoritma *Cosine Similarity* untuk membandingkan kemiripan dokumen terhadap *Query* yang dimana akan menjadi jawaban dari aplikasi *chatbot*. Algoritma *Cosine Similarity* dipilih karena bisa memanfaatkan hasil dari

pembobotan TF-IDF dan mampu untuk membandingkan kemiripan dokumen [6].

Dari hal-hal sebelumnya bisa disimpulkan bahwa metode TF-IDF dan *Cosine Similarity* sangat cocok dalam membuat closed-domain *chatbot* dengan konsep retrieval-based. Aplikasi yang digunakan dalam implementasi penelitian tugas akhir ini adalah aplikasi *LINE Messenger*. Melalui penelitian tugas akhir ini juga diharapkan dapat membantu memberikan solusi untuk menjawab pertanyaan yang sering ditanyakan oleh pelanggan, atau *frequently asked questions* (FAQ) dengan menggunakan teknologi *chatbot* berbasis *information retrieval* khususnya menggunakan metode TF-IDF dan algoritma *Cosine Similarity* .

1.2. Perumusan Masalah

Berdasarkan latar belakang yang telah dijabarkan di atas, berikut adalah rumusan masalah yang dijadikan acuan dalam penelitian tugas ini adalah sebagai berikut:

1. Bagaimana menerapkan algoritma *Cosine Similarity* dan metode pembobotan TF-IDF pada aplikasi *chatbot*?
2. Bagaimana performa algoritma *Cosine Similarity* dan metode pembobotan TF-IDF pada aplikasi *chatbot*?

1.3. Batasan Pengerjaan Tugas Akhir

Batasan permasalahan penelitian ini adalah,

1. Data yang digunakan dalam aplikasi *chatbot* adalah dataset *frequently asked question* yang ditanyakan oleh calon mahasiswa baru ITS.

2. Dataset yang digunakan berbahasa Indonesia.
3. Aplikasi *chatbot* yang dibangun merupakan tipe closed domain dan *retrieval-based chatbot*.
4. Aplikasi yang dikembangkan menggunakan aplikasi *LINE Messenger*.Tujuan Tugas Akhir

1.4. Manfaat Tugas Akhir

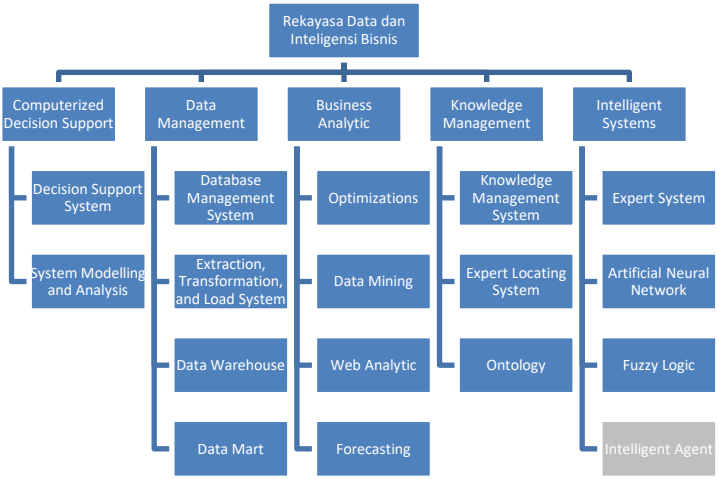
Beberapa manfaat yang diharapkan dapat tercapai pada penelitian ini adalah,

1. Menjadi bahan referensi atau rujukan tambahan dalam bidang aplikasi *chatbot* khususnya menggunakan pendekatan konsep *Information retrieval*.
2. Membantu menyelesaikan permasalahan aplikasi *chatbot* dengan menggunakan pendekatan *Information retrieval* khususnya menggunakan algoritma *Cosine Similarity* dan metode pembobotan TF-IDF.
3. Menghasilkan aplikasi *chatbot* yang memiliki performa yang tinggi.

1.5. Relevansi

Penelitian tugas akhir ini berkaitan dengan roadmap penelitian Laboratorium Rekayasa Data dan Intelegensi Bisnis (RDIB) yang *term*asuk pada pokok penelitian *Intelligent System*, khususnya di bidang *Intelligent Agent*. Hal ini menunjukkan bahwasannya penelitian ini akan berkaitan dan sejalan dengan tujuan Laboratorium RDIB yaitu menjadi pusat penelitian dan pusat rujukan terkait pemanfaatan data yang mendukung analisis bisnis dan

organisasi untuk dapat ditransformasi menjadi informasi yang bermakna serta pengetahuan sehingga dapat mendukung proses pengambilan keputusan. Gambar 1.2 adalah bidang keilmuan yang terdapat pada Laboratorium Rekayasa dan Intelegensi Bisnis.



Gambar 1.2 Bidang Keilmuan Laboratorium Rekayasa Data dan Inteligensi Bisnis

BAB II
TINJAUAN PUSTAKA

Pada bab ini akan dibahas mengenai tinjauan pustaka terkait penelitian sebelumnya dan penjelasan dasar teori yang akan menjadi acuan dalam pengerjaan tugas akhir ini.

2.1. Studi Sebelumnya

Pada subbab ini akan diterangkan mengenai beberapa penelitian terdahulu yang telah dilakukan dan memiliki relevansi dengan tugas akhir ini. Berikut beberapa penelitian yang telah dilakukan sebelumnya terkait dengan topik penelitian tugas akhir ini.

Tabel 2.1 Penelitian Terdahulu

No	Penelitian Terdahulu	
1.	Judul Penelitian	*DocChat: An Information retrieval Approach for Chatbot Engines Using Unstructured Documents* (2016) [8]
	Tahun	2016
	Nama Peneliti	Zhao Yan, Nan Duan, Junwei Bao, Peng Chen , Ming Zhou, Zhoujun Li, Jianshe Zhou
	Permasalahan	*Chatbot Engine*
	Dataset	FAQ dataset
	Algoritma	*Novel response retrieval approach*

No	Penelitian Terdahulu	
	Deskripsi	Penelitian [8] membahas mengenai penggunaan pendekatan *information retrieval* sebagai mesin dalam *chatbot*. Produk yang dikembangkan pada penelitian ini adalah DocChat. Penelitian ini menggunakan dua jenis dataset yaitu dataset dengan menggunakan bahasa inggris dan bahasa mandarin.
	Keterkaitan dengan Tugas Akhir	Penggunaan pendekatan *information retrieval* sebagai chatbot engine menjadi referensi penulis.
	Gap Penelitian	Pada penelitian menggunakan dua jenis dataset yaitu dataset dengan bahasa inggris dan dataset berbahasa mandarin sedangkan penelitian pada tugas akhir menggunakan dataset FAQ dengan bahasa Indonesia.
2.	Judul Penelitian	*Document Language Models, Query Models, and Risk Minimization for Information retrieval (2001) [9]*
	Tahun	2001
	Nama Peneliti	John Lafferty, Chengxiang Zhai
	Permasalahan	*Information retrieval*
	Dataset	*TREC testing collections: the AP*

No	Penelitian Terdahulu	
		collection on disk 1 (topics 1-50), the TREC 8 ad hoc task collection (topics 401-450) on disk 4&5, dan the TREC 8 web track collection (topics 401-450 on web data).
	Algoritma	*Query* Model
	Deskripsi	Penelitian [9] membandingkan dua metode *information retrieval* yaitu metode *vector space model* (TF-IDF) dan metode yang diajukan oleh penulis yaitu gabungan *Query model* dan pseudo. Penelitian ini menggunakan tiga dataset yaitu dataset AP89, TREC 8 dan web dengan hasil presisi peningkatan masing-masing sebesar +23%, +22%, +25%
	Keterkaitan dengan Tugas Akhir	Penelitian menggunakan topik *information retrieval* dan menggunakan metode TF-IDF sebagai perbandingan dalam melakukan uji presisi.
	Gap Penelitian	Penelitian tugas akhir ini menggunakan TF-IDF sebagai mesin *chatbot* sedangkan penelitian menggunakan TF-IDF untuk membandingkan performa dengan metode yang penulis ajukan pada paper ini.

No	Penelitian Terdahulu	
3.	Judul Penelitian	APLIKASI CHATBOT BERBASIS WEB PADA SISTEM INFORMASI LAYANAN PUBLIK KESEHATAN DI MALANG DENGAN MENGGUNAKAN METODE TF-IDF (2018) [6]
	Tahun	2018
	Nama Peneliti	Dhebys Suryani Hormansyah, Yoga Putera Utama
	Permasalahan	*Bot engine*
	Dataset	Data layanan publik kesehatan di Kota Malang (data praktek dokter umum,data puskesmas dan data rumah sakit umum)
	Algoritma	TF-IDF, *Cosine Similarity*
	Deskripsi	Penelitian [10] menggabungkan dua metode yaitu metode TF-IDF dan algoritma *Cosine Similarity* untuk digunakan pada sistem *chatbot*. *Chatbot* yang digunakan pada penelitian ini berbasis web. Penelitian menggunakan dataset dari data layanan publik kesehatan di kota malang yang meliputi data praktek dokter umum,data puskesmas dan data rumah sakit umum.

No	Penelitian Terdahulu	
		Kesimpulan yang didapat dari sistem penelitian ini dapat dikatakan berhasil dalam mengimplementasikan metode TF-IDF dan *Cosine Similarity* pada aplikasi *chatbot*.
	Keterkaitan dengan Tugas Akhir	Penelitian menggunakan dua metode yaitu metode TF-IDF dan *Cosine Similarity* yang dimana memiliki konsep metode yang sama dengan penelitian tugas akhir sehingga menjadi rujukan untuk penulis.
	Gap Penelitian	Penelitian tugas akhir menggunakan aplikasi *chatbot* berbasis aplikasi *LINE Messenger* sedangkan pada penelitian [6] ini menggunakan aplikasi *chatbot* berbasis web. Peneliti tidak menampilkan nilai akurasi dari sistem yang telah dibuat sehingga tidak diketahui performanya.
4.	Judul Penelitian	*The MiPACQ Clinical Question Answering System (2011)* [11]
	Tahun	2011
	Nama Peneliti	Brian L. Cairns, MS, Rodney D. Nielsen, PhD, James J. Masanz, MS, James H. Martin,PhD, Martha S. Palmer, PhD, Wayne H. Ward, PhD, Guergana K. Savova, PhD

No	Penelitian Terdahulu	
	Permasalahan	*Question Answering System*
	Dataset	Medpedia, Pertanyaan Klinis
	Algoritma	*Information retrieval, Rule-based Re-ranking, ML Based Re-ranking*
	Deskripsi	Penelitian [11] menggunakan machine learning sebagai mesin dari Clinical Question Answering System. Peneliti membandingkan performa dari metode *information retrieval* dan *machine learning* pada dataset Medpedia dan pertanyaan klinis yang dimana menghasilkan peningkatan presisi sebesar +133% dengan menggunakan machine learning. Menurut peneliti hal ini terjadi karena metode *information retrieval* tidak cocok dengan teks dalam paragraph level.
	Keterkaitan dengan Tugas Akhir	Penelitian menggunakan konsep *Question Answering System* yang bisa menjadi rujukan bagi penelitian tugas akhir.
	Gap Penelitian	Penelitian menggunakan machine learning sebagai dasar dari aplikasi MiPACQ *Clinical Question Answering* System. Penelitian menggunakan *Database* medpedia dan pertanyaan

No	Penelitian Terdahulu	
		klinis.
5.	Judul Penelitian	*An Intelligent Discussion-Bot for Answering Student Queries in Threaded* Discussions (2005) [12]
	Tahun	2005
	Nama Peneliti	Donghui Feng, Erin Shaw, Jihie Kim, Eduard Hovy
	Permasalahan	Discussion-Bot
	Dataset	Thread diskusi dan dokumen course
	Algorithm	*Novel response retrieval approach*
	Deskripsi	Penelitian [12] menggunakan gabungan teknik *information retrieval* dan machine learning dalam aplikasinya. *Information retrieval* digunakan untuk membalas thread yang dibuat oleh murid seperti pada *Question Answering System*. *Machine learning* digunakan untuk menganalisa tugas maupun pekerjaan rumah dari murid sehingga tidak perlu mengoreksi satu per satu. Penelitian menggunakan data thread dari semester sebelumnya yang berjumlah 1236 arsip thread diskusi and 279 dokumen course.
	Keterkaitan dengan	Penelitian menggunakan *information retrieval* dengan TF-IDF sehingga bisa

No	Penelitian Terdahulu	
	Tugas Akhir	menjadi rujukan dalam penelitian tugas akhir.
	Gap Penelitian	Penelitian menyelesaikan Penelitian menggunakan gabungan dari dua Teknik yaitu teknik *information retrieval* dan *machine learning*. Dataset yang digunakan adalah thread diskusi dan dokumen *course* sedangkan pada penelitian tugas akhir ini menggunakan dataset FAQs Institut Teknologi Sepuluh Nopember.

2.2. Dasar Teori

Pada subbab ini akan menjelaskan konsep-konsep atau teori yang sekiranya dibutuhkan oleh pembaca untuk lebih memahami laporan tugas akhir dan juga digunakan untuk untuk mendukung pengerjaan tugas akhir.

2.2.1. *Chatbot*

Chatbot adalah sebuah program komputer yang ditujukan untuk melakukan percakapan dengan orang lain menggunakan kecerdasan buatan [13]. Jadi *chatbot* bertindak seolah olah menjadi manusia yang dibuat untuk memfasilitasi komunikasi antara manusia dan komputer, mengerti tentang pertanyaan dengan bahasa natural dan menjawab sesuai dengan konteksnya.

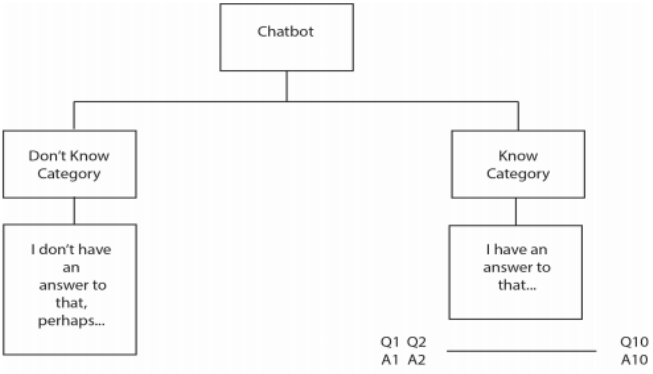
Permasalahan *Chatbot* menjadi topik yang hangat untuk sekarang, namun *chatbot* sudah menjadi objek penelitian sejak 50 tahun yang lalu [14]. *Chatbot* pertama dikemukakan oleh Joseph Weizenbaum di M.I.T pada tahun 1966 [15]. Joseph mengimplementasikan *chatbot* yang bernama ELIZA untuk meniru seorang *psychotherapist* [14]. Pada masa sekarang *chatbot* sering digunakan untuk menyampaikan suatu informasi. Kane D. (2016) mengatakan bahwa *chatbot* memiliki kemampuan untuk berhubungan dengan beberapa pola dalam satu waktu sehingga menjadi alat yang cocok untuk memberikan referensi dan pengarahan bantuan yang dibutuhkan oleh pengguna [16].

Pada perkembangannya *chatbot* tercipta dalam banyak kategori. *Chatbot* sebagai media hiburan seperti ELIZA [15] dan ALICE, *chatbot* sebagai *information retrieval* seperti Siri dan *Google Assistant*, *chatbot* sebagai *Happy Assistant System* seperti *chatbot* pada e-commerce [2]. Jadi pada tahap ini *chatbot* sudah menjadi alternatif pengganti pada banyak pekerjaan strategis.

Secara umum *chatbot* yang digunakan dalam customer service dapat dibagi menjadi dua kategori berdasarkan pengembangnya yaitu *first-party* dan *third-party*. *First party chatbot* lebih merujuk pada mesin percakapan yang dikembangkan oleh perusahan besar untuk mengembangkan bisnis mereka dengan

mengembangkan kualitas *customer service* dan mengurangi biaya keseluruhan *customer service*. *Chatbot* dengan kategori *first-party* biasanya digunakan oleh industri yang berorientasi pada *customer* seperti bank, telekomunikasi dan *e-commerce*. *Third party chatbot* merujuk pada basis *open source* untuk membantu pada *developer* membangun mesin percakapan mereka seperti *Microsoft Bot Framework*, *Facebook Messenger*, *Google Assistant* dan *Amazon Lex* [17].

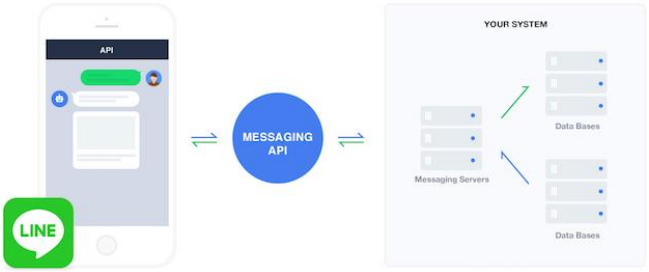
Bagian penting dari *conversational chatbot* adalah membangun alur dialog. Hal ini bisa dilihat pada Gambar 2.1. Pada proses pembangunan alur dialog, harus berasumsi bahwa ada contoh yang tidak bisa direspon oleh *chatbot* dan ada yang bisa *chatbot* jawab [13].



Gambar 2.1 Contoh Kategori respon pada *chatbot*

2.2.2. *LINE*

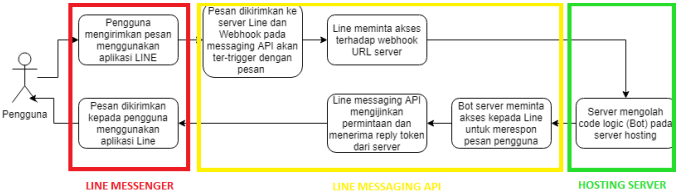
Sistem yang digunakan pada penelitian tugas akhir ini akan dibangun pada aplikasi *LINE* messenger dengan memanfaatkan fitur *messaging API* yang telah disediakan oleh *LINE*. *LINE messaging API* berfungsi untuk memungkinkan adanya komunikasi dua arah antara sistem yang dibuat oleh pengembang dengan pengguna *LINE* secara otomatis [18]. Sistem yang digunakan pada penelitian tugas akhir ini menggunakan *hosting* dari website Heroku. Heroku berkolaborasi dengan GitHub dan menggunakan GitHub sebagai repositori untuk memudahkan pengembang dalam meluncurkan aplikasinya. Heroku dipilih karena bisa mendukung banyak bahasa pemrograman populer seperti python, java, php, node.js, ruby, dll. Ilustrasi arsitektur aplikasi *LINE chatbot* bisa dilihat pada Gambar 2.2 [18].



Gambar 2.2 Ilustrasi arsitektur *LINE chatbot*

Pada gambar 2.2 terdapat tiga bagian utama dalam arsitektur *chatbot* yang dibuat pada penelitian tugas akhir ini yaitu *LINE* messenger, *messaging API* dan

hosting server. *LINE* messenger berfungsi sebagai gerbang pertama atau user *interface* dalam berhubungan dengan pengguna. *LINE* messenger juga berfungsi untuk menjadi tempat mengambil *Query* pengguna dan juga menampilkan jawaban yang relevan dengan *Query* pengguna. Kemudian pada bagian *messaging API* berfungsi untuk menghubungkan antara mesin bot dengan pengguna secara otomatis. Kemudian pada bagian terakhir yaitu *hosting* server berfungsi sebagai mesin dari *chatbot*. Semua bagian *Code logic* untuk membuat *chatbot* akan diunggah dalam *server* kemudian *server* akan menjalankan *Code* secara otomatis. Alur kerja dari arsitektur *LINE chatbot* digambarkan pada gambar 2.3.



Gambar 2.3 Ilustrasi alur kerja arsitektur *LINE chatbot*

Gambar 2.3 merupakan ilustrasi alur kerja arsitektur *LINE chatbot* yang telah diusulkan. Aplikasi *LINE* Messenger digunakan dalam proses mengirimkan pesan (*Query*) kepada sistem dan juga menerima respon dari sistem. Kemudian pesan akan diolah oleh *LINE* messaging API. Ketika pesan diolah oleh *LINE* messaging API maka *webhook* akan ter-*trigger* dan meminta akses terhadap *webhook URL*. Pada saat meminta akses terhadap mesin bot maka otomatis akan mendapatkan *reply token* sebagai

penanda identifikasi pada saat mesin bot mengirimkan respon balasan. Setelah akses diterima maka pesan dari aplikasi *LINE* messenger akan diolah oleh mesin bot pada *hosting* server. Server akan menjalan *Code* logic dari mesin *chatbot* dan akan menghasilkan respon berupa HTTPS dengan format json dan dikirim sesuai dengan reply token-nya. Kemudian mesin bot akan meminta akses untuk memberikan respon pada *LINE messaging API* dan setelah disetujui pesan akan diteruskan oleh *LINE messaging API* kepada pengguna aplikasi *LINE messenger* yang telah diidentifikasi berdasarkan *reply token* yang telah dibuat.

2.2.3. *Information retrieval*

Definisi *Information retrieval* (IR) memiliki makna yang sangat luas [19]. Contoh dari *information retrieval* adalah ketika kita mengambil kartu kredit dari dompet kemudian menuliskan nomor kartu kredit tersebut. Namun sebagai bidang studi akademik, *information retrieval* didefinisikan [19] sebagai berikut: *Information retrieval* (IR) adalah mencari material (biasanya dokumen) dari hal yang yang bersifat tidak terstruktur (biasanya text) yang dapat memenuhi kebutuhan dalam mencari informasi dari koleksi yang besar (biasanya disimpan dalam komputer) [19].

Information retrieval (IR) merupakan cabang dari computer science yang berurusan dengan pemrosesan dari koleksi dokumen yang mengandung "*free text*" seperti makalah ilmiah ataupun buku elektronik [20]. Objektif dari pemrosesan ini adalah memfasilitasi pencarian yang cepat dan akurat berdasar pada kata

kuncinya. IR mulai digunakan pada bidang lain pada computer science, seperti teknologi *Database* dan *natural language processing* (NLP).

2.2.4. *Term Frequency* (TF)

Term frequency pertama kali dikemukakan oleh Hans Peter Luhn (1957) [21]. Menurut Hans (1957) *term frequency* adalah bobot dalam suatu "*term*" yang muncul dalam dokumen sebanding dengan frekuensinya. Pada *term frequency* (tf (t, d)), hal yang paling sederhana adalah dengan menghitung *term* pada dokumen. Contohnya yaitu *term* t muncul berapa kali dalam dokumen d. Persamaan metode *Term Frequency* ditunjukkan dengan Eq 2.1.

$$tf(t,d) = 0.5 + 0.5.\frac{f_{t,d}}{\max\{f_{t',d}: t' \in d\}} \qquad \textbf{\textit{Eq 2.1}}$$

Dimisalkan ketika kita mempunyai teks dokumen dan ingin mengurutkan dokumen yang paling relevan dengan *Query* "*the brown cow*". Langkah pertama adalah dengan mengeliminasi dokumen yang tidak mengandung kata "*the*", "*brown*", "*cow*" tapi dengan hal ini masih terdapat beberapa dokumen yang mengandung ketiga kata tersebut. Kemudian langkah selanjutnya adalah dengan menghitung berapa kali setiap kata dari *Query* muncul dalam setiap dokumen, jumlah kata yang muncul pada setiap dokumen disebut dengan *term frequency*.

2.2.5. *Inverse Document Frequency* (IDF)

Inverse document frequency adalah mengukur bagaimana indeks *term* membedakan dokumen yang relevan dan dokumen yang tidak relevan dengan memberikan bobot yang tinggi untuk *term* yang langka [22]. Dimisalkan terdapat contoh *term* "the" yang sangat umum, maka *term frequency* cenderung melakukan kesalahan dalam melihat dokumen dengan penggunaan kata "the" yang berulang kali digunakan tanpa memberikan bobot yang cukup untuk kata "brown" dan "cow". Karenanya *inverse document frequency* muncul dengan konsep mengurangi bobot kata yang sering muncul dan meningkatkan bobot pada kata yang jarang muncul pada dokumen.

Karen Spark Jones (1972) pertama kali mengemukakan *inverse document frequency* yang dimana menjadi landasan pembobotan kata atau *term*. Karen juga berkata bahwa suatu *term* dapat dikuantifikasi sebagai *inverse* function dari dokumen yang ada [23]. Nilai dari *inverse document frequency* didapatkan dari total semua dokumen N dibagi dengan jumlah dokumen yang berisi dengan *term* [24]. Persamaan IDF ditunjukkan dengan persamaan Eq 2.2.

$$idf(t, D) = log \frac{N}{|\{d \in D : t \in d\}|} \qquad \textbf{Eq 2.2}$$

Keterangan:

$N : jumlah\ total\ dokumen\ dalam\ korpus\ N = |D|$

$|\{d \in D : t \in d\}| : jumlah\ dokumen\ yang\ muncul\ term\ t$

2.2.6. TF-IDF

Term Frequency dikombinasikan dengan *Inverse Document Frequency* sehingga menghasilkan metode *Term Frequency-Inverse Document Frequency* (TF-IDF). TF-IDF merupakan perhitungan statistik yang dimana merefleksikan bagaimana pentingnya kata pada dokumen dalam koleksi korpus. Metode ini sering digunakan dalam pembobotan dalam *information retrieval*, text mining dan user modeling. Nilai dari TF-IDF meningkat secara proporsional yang berkaitan dengan berapa kali kata muncul dalam dokumen dan diimbangi dengan jumlah dokumen yang ada dalam korpus yang berisi dengan kata tersebut. Hal ini membantu dalam mencari fakta bahwa beberapa kata muncul lebih sering secara umum. TF-IDF sangat populer dalam pembobotan kata, sebanyak 83% dari sistem rekomendasi berbasis teks di perpustakaan digital menggunakan TF-IDF. Persamaan TF-IDF dapat dilihat pada Eq 2.3

$$tfidf(t, d, D) = tf(t, d).idf(t, D) \qquad \textbf{\textit{Eq 2.3}}$$

2.2.7. *Cosine Similarity*

Algoritma *Cosine Similarity* adalah algoritma yang digunakan dalam menghitung similarity (tingkat kemiripan) terhadap dua buah vektor. Algoritma ini secara umum dikembangkan dari *vector space similarity Measure*. Algoritma *Cosine Similarity* menghitung tingkat kesamaan antara dua buah vektor dengan

menggunakan kata kunci (keywords) dari sebuah dokumen sebagai ukuran [25]. Algoritma *Cosine Similarity* dipilih karena dapat memanfaatkan pembobotan TF-IDF dengan baik. Metode perangkingan dari hasil metode TF-IDF menjadi masalah karena TF-IDF hanya mengukur bobot pada setiap katanya sedangkan algoritma *Cosine Similarity* melakukan proses perangkingan dengan mengukur kemiripan dari setiap dokumen yang dimana memiliki lingkungan yang sama. Pemilihan dari algortima *Cosine Similarity* juga didasarkan pada performa yang telah dilakukan pada penelitian sebelumnya [26] seperti yang telah digambarkan pada tabel 2.2.

Similarity measures	Term weighting	N-grams	Number of retrieved documents (highest ranking documents	
			1	5
Cosine similarity	Binary/TF/TFIDF	10	0.97	1.0
KLD	Binary/TFIDF/TF	12	0.97	1.0
Dice coefficient	Binary/TF	12	0.96	1.0
Jack -index	Binary/TF	12	0.96	1.0
Bhayttacharyan	BinaryTF/TFIDF	12	0.95	1.0
PCC(R)	TF/Binary	10	0.94	1.0
JSD	Binary/TF/TFIDF	10	0.83	0.897
Euclidean distance	TF/Binary	8	0.68	0.73

Tabel 2.2 Perbandingan Algoritma Text Similarity

Pada tabel 2.2 menunjukkan bahwa algoritma *Cosine Similarity* dan metode pembobotan TF-IDF berada pada urutan teratas pada hasil percobaan pada penelitian sebelumnya. Persamaan algoritma *Cosine Similarity* ditunjukkan dengan persamaan Eq 2.4. [25]

$$\text{CosSim}(d_i, q_i) = \frac{q_i . d_i}{|q_i||d_i|} = \frac{\sum_{j=1}^{t}(q_{ij} . d_{ij})}{\sqrt{\sum_{j=1}^{t}(q_{ij})^2 . \sum_{j=1}^{t}(d_{ij})^2}} \quad \textbf{Eq 2.4}$$

2.2.8. *Evaluation*

Validasi yang dilakukan pada penelitian tugas akhir adalah menggunakan *standard error*. *Standard Error* adalah indeks yang menggambarkan sebaran rata-rata sampel terhadap rata-rata dari rata-rata keseluruhan kemungkinan sampel. Untuk menghitung nilai *standard error* maka dibutuhkan urutan perhitungan yaitu mencari rataan (*mean*) kemudian mencari standar deviasi lalu mencari *standard error*.

1. *Mean*

   Mean adalah teknik penjelasan kelompok data didasarkan pada nilai rata-rata dari kelompok data tersebut. Untuk menghitung nilai *Mean* digunakan persamaan Eq 2.5.

   $$Mean = \frac{X_1+X_2+\ldots+X_n}{n} \qquad \textbf{Eq 2.5}$$

2. *Standard Deviation*

   Standar deviasi adalah suatu indeks yang menggambarkan sebaran data terhadap rata-ratanya. Untuk menghitung nilai *Standard Deviation* digunakan persamaan Eq 2.6.

   $$SD = \sqrt{\frac{\sum(x-mean)^2}{n}} \qquad \textbf{Eq 2.6}$$

3. *Standard Error*

   Standard Error adalah indeks yang menggambarkan sebaran rata-rata sampel terhadap rata-rata dari rata-rata

keseluruhan kemungkinan sampel. Untuk menghitung nilai *Standard Error* digunakan persamaan Eq 2.7.

$$SE = \frac{SD}{\sqrt{n}} \qquad \textbf{\textit{Eq 2.7}}$$

Kemudian skenario yang dilakukan pada ekperimen aplikasi *chatbot* sesuai dengan skenario yang telah digunakan pada penelitian sebelumnya [22]. Skenario yang dimaksud dijelaskan pada tabel 2.3.

Tabel 2.3 Skenario Eksperimen

Skenario	Penjelasan
Skenario 1 (S1)	Pertanyaan yang diajukan sama dengan pertanyaan yang telah direkam oleh FAQs *Database*
Skenario 2 (S2)	Pertanyaan yang diajukan berbeda dengan pertanyaan yang telah direkam oleh FAQs *Database* (pertanyaan yang belum ditanyakan sebelumnya)
Skenario 3 (S3)	Pertanyaan yang diajukan menggunakan skenario 1 dan skenario 2

Halaman ini sengaja dikosongkan

BAB III
METODOLOGI PENELITIAN

Pada bab ini akan dijelaskan mengenai alur metodologi yang akan dilakukan dalam tugas akhir ini. Metodologi ini juga digunakan sebagai pedoman untuk melaksanakan tugas akhir agar terarah dan sistematis.

3.1. Metodologi Penelitian

Diagram metodologi pengerjaan Tugas Akhir dapat dilihat pada gambar 3.1.

Gambar 3.1 Metodologi

3.2. Tahapan Pelaksanaan Tugas Akhir

Tahapan pelaksanaan tugas akhir akan menjelaskan terkait segala sesuatu yang akan dikerjakan oleh penulis

atau merupakan langkah-langkah pengerjaan tugas akhir. Berdasarkan bagan Gambar 3.1, berikut adalah langkah langkah dalam pengerjaan penelitian ini.

3.2.1. Identifikasi Masalah

Identifikasi masalah dilakukan dengan melakukan analisis studi kasus untuk mendapatkan informasi mengenai suatu permasalahan yang diangkat. Hasil atau luaran yang didapatkan dari tahap pertama ini adalah permasalahan atau problem yang akan diangkat dalam topik penelitian tugas akhir ini.

3.2.2. Studi Literatur

Studi literatur dilakukan setelah menemukan topik permasalahan, kemudian mempelajari kembali penelitian-penelitian terkait yang telah dilakukan oleh para peneliti sebelumnya. Penelitian terkait dengan topik permasalahan telah diidentifikasi diperoleh dari paper, jurnal, buku maupun sumber lainnya. Pada tahap studi literatur, hasil yang didapatkan atau luaran yang diperoleh adalah algoritma atau metode yang digunakan untuk memecahkan topik permasalahan yang diangkat pada penelitian tugas akhir ini.

3.2.3. Pengambilan dan Pemahaman Data

Pengambilan dan pemahaman data dilakukan setelah topik permasalahan dan algoritma telah ditentukan sebelumnya. Data yang dipilih dan digunakan dalam penelitian tugas akhir ini adalah data pertanyaan yang sering ditanyakan oleh calon mahasiswa baru ITS atau bisa disebut dengan data frequently asked questions

(FAQ). Model data yang digunakan adalah model Question-Answers Pairs. Model data yang digunakan digambarkan dengan kode 3.1.

```
{
   "message":"this is an evening in my timezone"
  ,"response":" here is afternoon !"
},
{
   "message":"how do you feel today ?  tell me something about yourself"
  ,"response":"my name is ishika,i hope we can be virtual friends !"
},
{
   "message":"how many virtual friends have you got ?"
  ,"response":"i have many !  but not enough to fully understand humans beings "
},
```

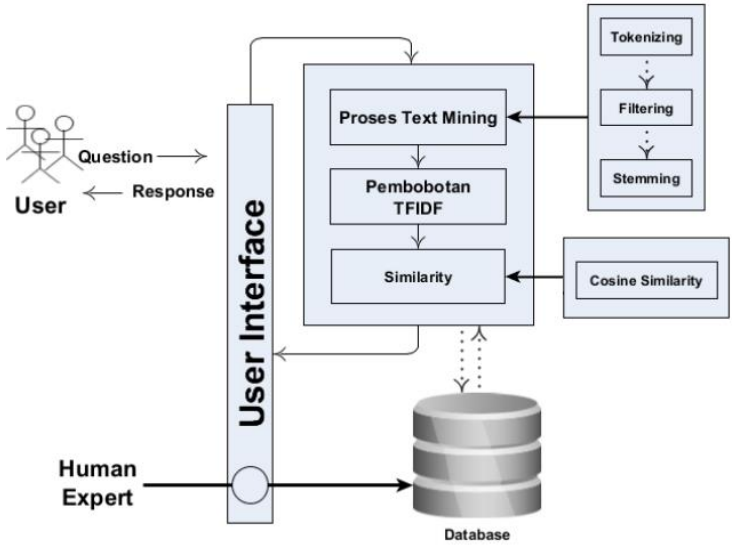
Kode 3.1 *Code* Data FAQ pada Json

Metode yang digunakan dalam mencari data yaitu dengan melakukan wawancara terhadap divisi hubungan masyarakat atau humas ITS dan melakukan crawling data pada media sosial ITS seperti Youtube, Instagram dan Twitter serta website ITS. Selanjutnya dilakukan pemahaman mendalam terhadap format dataset, Batasan yang digunakan dan model dataset. Pemahaman dilakukan dengan melakukan studi literatur terhadap dataset sejenis.

3.2.4. Desain Metode

Desain metode dilakukan setelah mengetahui algoritma yang digunakan dan dataset yang digunakan. Desain metode didasarkan pada bagaimana nanti aplikasi akan berjalan pada aplikasi *LINE*. Desain metode akan disesuaikan sesuai dengan kebutuhan penelitian tugas akhir ini seperti batasan masalah, tujuan pembuatan aplikasi dan kebutuhan pengguna. Hasil atau luaran yang dihasilkan pada tahap ini adalah desain atau

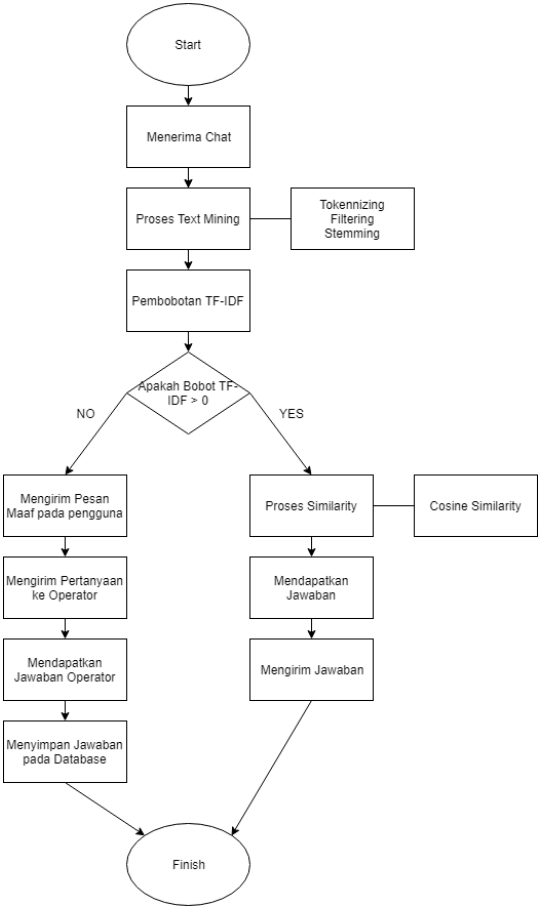
workflow dari aplikasi *chatbot*. Desain arsitektur dari aplikasi *chatbot* yang diusulkan digambarkan pada gambar 3.2.



Gambar 3.2 Desain Alur Aplikasi Chatbot

Pada gambar 3.2 terdapat dua aktor yaitu pengguna dan operator (Human Expert). Pengguna berhadapan langsung dengan user interface atau pada penelitian tugas akhir ini adalah aplikasi *LINE* messenger. Pengguna dapat memberikan *Query* pada sistem dan mendapatkan respon dari sistem dengan melalui aplikasi *LINE* messenger. *Query* dari pengguna akan diolah oleh sistem dengan proses *text mining*, pembobotan TF-IDF dan similarity. Operator dapat melakukan masukan terhadap *Database* berupa jawaban pada pertanyaan

yang belum bisa terjawab oleh sistem. Detail dari arsitektur ini akan dijelaskan dengan desain *flowchart* yang digambarkan oleh gambar 3.3.



Gambar 3.3 Ilustrasi alur kerja arsitektur *chatbot*

Ketika *chatbot* menerima pesan dari pengguna maka pesan akan dianggap sebagai *Query* oleh sistem. *Query*

akan diolah dengan proses *text mining* dengan tiga tahapan yaitu proses *Tokenizing*, *stopwords* dan *Stemming*. Proses *tokenizing*adalah proses pemisahan string input pada tiap kata yang menyusunnya. Proses *stopwords* adalah proses membuang kata yang tidak diperlukan seperti kata “yang”, “dan”, “di”, “dari” dan seterusnya. Kemudian proses *Stemming* merupakan proses menghilangkan kata imbuhan sehingga menjadi kata dasar. Proses selanjutnya adalah melakukan pembobotan TF-IDF pada *Query* dan akan mendapatkan daftar dokumen yang mengandung kata yang sama dengan *Query* pengguna. Kemudian jika terdapat dokumen yang relevan akan dilakukan proses ranking dengan menggunakan algoritma *Cosine Similarity*. Algortima *Cosine Similarity* akan mencari kemiripan antara dokumen dan *Query* dengan nilai tertinggi. Setelah melakukan proses pengurutan maka akan didapatkan dokumen yang dimana akan menjadi jawaban dan dikirimkan kepada pengguna. Jika pembobotan TF-IDF tidak memiliki dokumen yang relevan sama sekali atau 0 dokumen maka pertanyaan tidak bisa terjawab dan akan menampilkan pesan “Tidak bisa Menjawab” kepada pengguna. Kemudian pertanyaan tersebut akan dikirimkan kepada operator untuk mendapatkan jawaban. Jawaban yang telah dikirimkan oleh operator akan direkam pada *Database* dan akan digunakan pada pertanyaan berikutnya.

3.2.5. Implementasi Metode

Input atau masukan pada tahap ini adalah metode pembobotan TF-IDF dan algoritma *Cosine Similarity* dengan menggunakan dataset pertanyaan yang sering ditanyakan (FAQ) oleh calon mahasiswa baru ITS. Metode TF-IDF merupakan gabungan dari metode TF dan metode IDF sehingga terjadi proses terjadi dalam dua tahap. Algoritma TF akan mencari kata dengan kata yang sama dan menghitung secara statistikal setiap kata pada dataset. Kemudian metode TF akan mengelompokkan kata yang sama dan melakukan perhitungan. Metode TF tidak mampu membedakan kata yang sering muncul sehingga memiliki kemungkinan akan memiliki interpretasi dokumen yang salah. Oleh karena itu digunakan metode IDF untuk menutupi kelemahan tersebut. Metode IDF akan mengambil output dari metode TF kemudian melakukan perhitungan terhadap dokumen dengan mengurangi bobot pada kata yang sering muncul dan menambah bobot pada kata yang jarang muncul. Kemudian dengan bobot tersebut akan dijadikan input atau masukan pada algoritma *Cosine Similarity* . Algoritma *Cosine Similarity* akan mencari kemiripan (similarity) berdasarkan jarak dari dua buah vektor. Vektor yang dibandingkan adalah vektor pada *Query* dan vektor pada dokumen.

3.2.6. Pengujian dan Evaluasi

Pengujian dilakukan dengan memasukkan pertanyaan ke dalam aplikasi *chatbot* sesuai dengan skenario pada

tabel 2.4. Pada setiap pengujian akan dilakukan analisis perhitungan *standard error* sesuai dengan subbab 2.2.8 sehingga dapat diketahui performa dari aplikasi *chatbot* yang telah dibuat.

3.2.7. Analisis Hasil dan Kesimpulan

Setelah dilakukan pengujian dan evaluasi pada aplikasi *chatbot*, analisis hasil dilakukan dengan mengamati akurasi pada setiap pengujian. Analisis juga meliputi perbandingan hasil akurasi setiap pengujian dan performa dalam membalas pesan pada aplikasi.

3.2.8. Penyusunan Laporan Tugas Akhir

Penyusunan laporan dilakukan dengan cara mengumpulkan semua dokumentasi masukan, proses dan keluaran dari penelitian tugas akhir. Hasil berupa Buku Laporan Tugas Akhir diharapkan dapat menjadi rujukan dan dapat dikembangkan pada penelitian yang akan datang.

BAB IV
PERANCANGAN

Dalam bab ini akan dijelaskan persiapan perancangan terkait dengan data yang digunakan dan aplikasi yang akan dibuat sebagai tahap persiapan implementasi.

4.1. Pengolahan Data

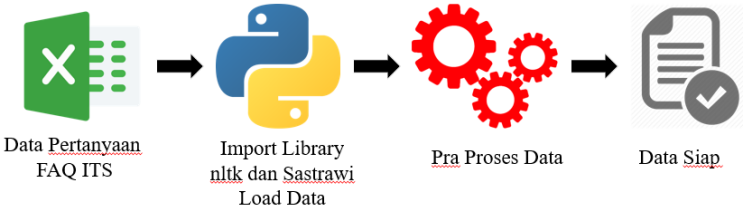
Dalam penelitian tugas akhir ini, data yang digunakan adalah data pertanyaan yang ditanyakan oleh calon mahasiswa ITS atau FAQ dataset. FAQ dataset dapat diperoleh dari situs info.its.ac.id dimana dataset tersebut disediakan oleh Organisasi Pengelola Informasi dan Dokumentasi yang ditanyakan oleh calon mahasiswa ITS melalui formulir permohonan informasi dan dokumentasi.

4.1.1. Pengambilan Data

Proses pengambilan data dilakukan pada dashboard admin pada website info.its.ac.id/admin. Pengambilan data dilakukan secara manual dengan menyalin semua pertanyaan yang ditanyakan oleh calon mahasiswa ITS.

4.1.2. Pra-Proses Data

Pada proses ini, data yang telah didapatkan dari proses pengambilan data diolah untuk dirubah menjadi huruf kecil, menghilangkan tanda baca atau karakter tertentu, merubah ke bentuk kata dasar hingga menghilangkan kata yang sering sekali muncul dengan memanfaatkan *library* Python yaitu sastrawi. Tahapan yang akan dilalui dataset pertanyaan ditunjukkan pada gambar 4.1.



Gambar 4.1 Ilustrasi alur kerja pra-proses data

Pada gambar 4.1 bisa dilihat bahwa terdapat proses pra-proses data yang dilakukan pada dataset pertanyaan. Proses tersebut adalah

1. **Tanda Petik**

   Pada tahap ini, pada kalimat pertanyaan akan dilakukan penghapusan tanda petik. Hal ini dikarenakan tanpa petik bisa merusak tatanan dari sebuah *query Database* dan Bahasa pemrograman.

Tabel 4.1 Contoh Tahap Mengilangkan Tanda Petik

Sebelum Penghilangan Tanda Petik	Sesudah Penghilangan Tanda Petik
1. Apakah SPI Rp. 75.000.000 bisa 'dicicil' ..??	1. Apakah SPI Rp. 75.000.000 bisa dicicil ..??

2. ***Tokenization***

   Pada tahap *Tokenizing*, kalimat pertanyaan akan dipisahkan menjadi setiap kata penyusun dari kalimat pertanyaan tersebut. Proses ini

menggunakan *library nltk* untuk Bahasa pemrograman python. Hal ini ditujukan untuk menilai setiap kata lebih dalam.

Tabel 4.2 Contoh Tahap *Tokenization*

Sebelum *Tokenization*	**Sesudah *Tokenization***
1. Apakah SPI Rp. 75.000.000 bisa dicicil ..??	['1', '.', 'Apakah', 'SPI', 'Rp', '.', '75.000.000', 'bisa', 'dicicil', '..', '?', '?']

3. Casefolding

Tahap Pada tahap ini, pertanyaan pada dataset dirubah semua menjadi ke bentuk huruf kecil atau *lowercase* untuk menyamakan struktur katanya secara keseluruhan.

Tabel 4.3 Contoh Tahap *Casefolding*

Sebelum *Casefolding*	**Sesudah *Casefolding***
['1', '.', 'Apakah', 'SPI', 'Rp', '.', '75.000.000', 'bisa', 'dicicil', '..', '?', '?']	['1', '.', 'apakah', 'spi', 'rp', '.', '75.000.000', 'bisa', 'dicicil', '..', '?', '?']

4. Remove Punctuation

Pada tahap *remove punctuation*, pertanyaan dataset akan diproses untuk dihilangkan karakter-karakter berupa tanda baca, simbol dan angka yang masih ditemukan pada pertanyaan.

Tabel 4.4 Contoh Tahap *Remove Punctuation*

Sebelum *Remove Punctuation*	**Sesudah *Remove Punctuation***
['1', '.', 'apakah', 'spi', 'rp', '.', '75.000.000', 'bisa', 'dicicil', '..', '?', '?']	['1', 'apakah', 'spi', 'rp', '75000000', 'bisa', 'dicicil']

5. Remove Single Character

Tahap ini merupakan tahap dimana *token* yang hanya memiliki 1 huruf akan dihapus karena dianggap tidak mengandung kata yang berbobot.

Tabel 4.5 Contoh Tahap *Remove Single Character*

Sebelum *Remove Single Character*	**Sesudah *Remove Single Character***
['1', 'apakah', 'spi', 'rp', '75000000', 'bisa', 'dicicil']	['apakah', 'spi', 'rp', '75000000', 'bisa', 'dicicil']

6. Stop Words

Tahap ini merupakan tahapan untuk mengidentifikasi setiap kata dalam kalimat pertanyaan untuk menghilangkan kata yang tidak bermakna penting. Tahap ini menggunakan *library* sastrawi dengan Bahasa pemrograman python.

Tabel 4.6 Contoh Tahap *Stopwords*

Sebelum *Stopwords*	**Sesudah *Stopwords***
['apakah', 'spi', 'rp', '75000000', 'bisa', 'dicicil']	[' ', 'spi', 'rp', '75000000', ' ', 'dicicil']

7. Stemming

Proses *Stemming* digunakan untuk merubah bentuk kata yang memiliki kata dasar yang sama dengan arti yang serupa namun memiliki bentuk kata yang berbeda karena mendapat imbuhan yang berbeda. Pada tahap ini, digunakan *library* sastrawi sebagai pendukungnya.

Tabel 4.7 Contoh Tahap *Stemming*

Sebelum *Stemming*	**Sesudah *Stemming***
[' ', 'spi', 'rp', '75000000', ' ', 'dicicil']	[' ', 'spi', 'rp', '75000000', ' ', 'cicil']

8. DeTokenizer

Proses *DeTokenizer* digunakan untuk mengubah kalimat yang sudah menjadi *token* kembali menjadi bentuk kalimat. Hal ini dikarenakan dapat berbenturan dengan library pada pembobotan TF-IDF.

Tabel 4.8 Contoh *Detokenizer*

Sebelum *DeTokenizer*	Sesudah *DeTokenizer*
[' ', 'spi', 'rp', '75000000', ' ', 'dicicil']	spi rp 75000000 cicil

4.1.3. Penyimpanan Data

Data yang telah melalui proses pra-proses data kemudian disimpan kedalam basis data PostgreSQL karena mendukung format data Json. Kemudian data pertanyaan yang belum terjawab akan disimpan pada basis data PostgreSQL dengan tabel yang berbeda dan sedangkan data perhitungan TF-IDF akan dimasukkan kedalam *pickle* yang telah disediakan oleh Python setelah dilakukan proses *Training*. Berikut adalah penjelasan terkait data yang dikumpulkan dan disimpan.

Tabel 4.9 Alur Data

No	Input	Proses	Output
1	Data pertanyaan Mentah (Info.its.ac.id/admin)	Mengambil data FAQ ITS dari *Website* Info.its.ac.id	Daftar Pertanyaan FAQ ITS yang diajukan oleh claon mahasiswa ITS (Excel)

No	Input	Proses	Output
2	Daftar Pertanyaan FAQ (Excel)	Pra-Proses Data	Daftar Petanyaan FAQ setelah pra-proses data (Json)
3	Daftar Petanyaan FAQ setelah pra-proses data (Json)	Penyimpanan data FAQ ITS	Daftar Petanyaan FAQ (PostgreSQL)

Berikut adalah bentuk tabel yang digunakan pada setiap data

a. Data yang sudah dilakukan proses pra-proses data yang disimpan pada basis data PostgreSQL

Tabel 4.10 Tabel FAQ ITS

Tabel	Atribut	Keterangan
faq	Id (int)	Id untuk setiap pertanyaan dan jawaban
	Info (json)	Daftar pasangan pertanyaan dan jawaban

b. Data pertanyaan yang belum dijawab pada basis data PostgreSQL

Tabel 4.11 Tabel Pertanyaan

Tabel	Atribut	Keterangan
pertanyaan	Id (int)	Id untuk setiap pertanyaan dan jawaban
	Pertanyaan (varchar)	Pertanyaan yang belum pernah diajukan sebelumnya
	Jawaban (varchar)	Jawaban yang diberikan oleh admin atau badan yang berwenang

4.2. Implementasi Pembobotan TF-IDF dan Cosine Similarity

Implementasi algoritma TF-IDF akan digunakan untuk mencari bobot setiap kata pada kalimat pertanyaan yang diajukan. Implementasi dilakukan setelah tahap pra proses data telah dilakukan. Pembobotan ini merupakan proses training pada data dan menghasilkan bobot pada

setiap kata dan kalimat pada dataset dan akan disimpan pada pickle. Saat ada pertanyaan pada input aplikasi dari pengguna maka pertanyaan akan dilakukan pra proses data yang kemudian akan dilakukan pembobotan TF-IDF. Kemudian hasil pembobotan akan dilakukan perhitungan kemiripan oleh algoritma *Cosine Similarity* dengan nilai bobot pada pickle. Nilai tertinggi pada algoritma *Cosine Similarity* atau memiliki nilai kemiripan tertinggi akan dijadikan kandidat jawaban yang dicari oleh pengguna. Kemudian akan dilakukan optimasi pada nilai minimum untuk *Cosine Similarity* untuk mendapat luaran atau output yang sesuai. Adapun *pseudoCode* yang digunakan untuk pembobotan TF-IDF dengan algoritma *Cosine Similarity* ditunjukkan dengan gambar 4.2.

```
GET daftar pertanyaan
GET query pengguna
DO Training TF-IDF daftar pertanyaan
DO Training TF-IDF query pengguna
FOR setiap Pertanyaan
        DO Cosine Similarity (query pengguna,daftar pertanyaan)
        MAX = Index
        RETURN index
READ Index Jawaban
ENDFOR
```

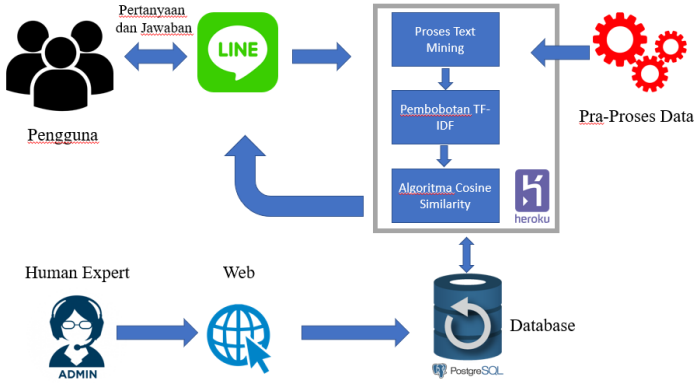
Gambar 4.2 Pseudo*Code* Pembobotan TF-IDF dan Cosine Similarity

4.3. Aplikasi *Chatbot*

Pembuatan aplikasi *chatbot* merupakan proses dalam membuat aplikasi yang berfungsi untuk menggabungkan algoritma TF-IDF dan *Cosine Similarity* dengan aplikasi messenger yang telah ditentukan. Aplikasi ini dibuat menggunakan bahasa pemrograman Python dan

menggunakan basis data PostgreSQL dengan menggunakan *hosting* dari heroku.com.

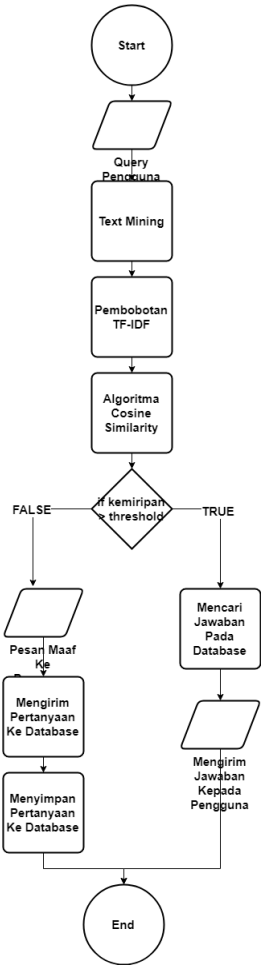
Bahasa pemrograman Python digunakan untuk membuat mesin *chatbot* yaitu algoritma TF-IDF dan *Cosine Similarity* . Kemudian PostgreSQL digunakan untuk menyimpan dataset pertanyaan dari website info.its.ac.id yang telah dilakukan proses pra-proses data dan data pertanyaan yang belum pernah ditanyakan sebelumnya. Kemudian *hosting* heroku.com akan digunakan sebagai *hosting* dari mesin *chatbot* tersebut karena bisa mendukung dan menjalankan bahasa pemrograman pyton secara *online*. Berikut adalah alur kerja dari aplikasi *chatbot* ditunjukkan dengan gambar 4.3.



Gambar 4.3 Ilustrasi Alur Aplikasi Chatbot

Pada gambar 4.3 terdapat dua aktor yaitu pengguna dan *operator (Human Expert*). Pengguna berhadapan langsung dengan *user interface* atau pada penelitian tugas akhir ini adalah aplikasi *LINE* messenger. Pengguna

dapat memberikan *Query* pada sistem dan mendapatkan respon dari sistem dengan melalui aplikasi *LINE* messenger. *Query* dari pengguna akan diolah oleh sistem dengan proses *text mining*, pembobotan TF-IDF dan similarity. Operator dapat melakukan masukan terhadap *Database* berupa jawaban pada pertanyaan yang belum bisa terjawab oleh sistem. *Detail* dari arsitektur ini akan dijelaskan dengan desain *flowchart* yang digambarkan oleh gambar 4.4.



Gambar 4.4 Ilustrasi Alur Aplikasi Chatbot

Ketika *chatbot* menerima pesan dari pengguna maka pesan akan dianggap sebagai *Query* oleh sistem. *Query* akan diolah dengan proses text mining dengan tahapan

seperti yang digunakan pada proses pra-proses data. Proses selanjutnya adalah melakukan pembobotan TF-IDF pada *Query* dan akan mendapatkan daftar dokumen yang mengandung kata yang sama dengan *Query* pengguna. Kemudian jika terdapat dokumen yang relevan akan dilakukan proses ranking dengan menggunakan algoritma *Cosine Similarity*. Setelah melakukan proses pengurutan maka akan didapatkan dokumen yang dimana akan menjadi jawaban dan dikirimkan kepada pengguna. Jika pembobotan TF-IDF tidak memiliki dokumen yang relevan sama sekali (bernilai 0) atau kurang dari nilai threshold yang telah ditentukan maka pertanyaan tidak bisa terjawab dan akan menampilkan pesan "maaf" kepada pengguna. Kemudian pertanyaan tersebut akan dikirimkan kepada operator untuk mendapatkan jawaban. Jawaban yang telah dikirimkan oleh operator akan direkam pada *Database* dan akan digunakan pada pertanyaan berikutnya.

BAB V
IMPLEMENTASI

Bab ini menjelaskan proses mengenai pelaksanaan perancangan yang telah dijelaskan pada bab IV Perancangan. Penjelasan dalam bagian implementasi meliputi penjelasan lingkungan implementasi, implementasi *Code* untuk setiap tahapan dan hasil pembuatan aplikasi *chatbot* dan dashboard.

5.1. Lingkungan Implementasi

Pada subbab ini akan dijelaskan spesifikasi perangkat yang digunakan dalam pembuatan aplikasi dalam proses implementasi. Perangkat yang dimaksud meliputi perangkat keras dan perangkat lunak yang digunakan untuk pengembangan aplikasi pada penelitian tugas akhir ini. Berikut merupakan daftar perangkat yang digunakan dalam penelitian ini:

Tabel 5.1 Spesifikasi Perangkat Keras

Hardware	**Spesifikasi**
Jenis	Asus A46CB
Processor	Intel Core I7 3537u
Memory	DDR3 4 GB
Storage	- 240GB SSD - 1TB HDD

Tabel 5.2 Spesifikasi Perangkat Lunak

Fungsi	Software / Teknologi
Sistem Operasi	Windows 10 Education 64-Bit
Bahasa Pemrograman	- PHP 7.1.8 x86 - Python 3.6.4 x64
IDE	Thonny Intellij
Text Editor	Notepad++ 7.7.1
Darabase	PostgreSQL
Data Labeling	Microsoft Excel 365
Web Browser	Google Chrome
Python Library	nltk Sklearn Sastrawi
CLI	Heroku Git

5.2. Implementasi Pengolahan Data

Bagian ini akan menjelaskan bagaimana cara untuk mendapatkan data FAQ ITS mulai dari pengambilan data, proses pra-proses data hingga penyimpanan data.

5.2.1. Pengambilan Data

Tahap ini merupakan tahap dalam pengambilan data FAQ ITS. Data diperoleh dari *website* info.its.ac.id namun dengan pengguna sebagai admin. *Website* info.its.ac.id dibawahi oleh Subunit Promosi dan Humas dan dikelola oleh Pejabat Pengelola Informasi dan Dokumentasi (PPID). Pengambilan data dilakukan pada tanggal 24 Maret 2019 dan didapatkan total daftar pasangan pertanyaan seperti ditunjukkan pada tabel 5.3.

Kemudian data tersebut disaring secara manual dengan memisahkan data pertanyaan antara pertanyaan yang diajukan oleh calon mahasiswa ITS dan pertanyaan yang lainnya yang tidak sesuai dengan batasan pada penelitian tugas akhir ini. Setelah dilakukan penyaringan data maka didapatkan total daftar pasangan pertanyaan yang ditunjukkan pada tabel 5.3.

Tabel 5.3 Jumlah Data Pertanyaan

No	Pertanyaan	Jumlah
1	Pertanyaan sebelum disaring	328

No	Pertanyaan	Jumlah
2	Pertanyaan setelah disaring	213

5.2.2. Implementasi Pra-Proses Data

Pada proses pra-proses data, dibutuhkan input yaitu data pertanyaan yang sudah disaring sebelumnya, namun khusus untuk proses pembersihan nama dilakukan juga pada jawaban pada dataset. Data yang digunakan dalam proses dilakukan dalam bentuk format json. Proses ini dilakukan menggunakan bahasa pemrograman python dengan memanfaatkan *library* yaitu *nltk* dan sastrawi. *Library* nltk digunakan untuk proses *tokenization* sedangkan *library* sastrawi digunakan untuk proses *Stopwords* dan *Stemming* karena mendukung bahasa Indonesia. Berikut adalah langkah-langkah dalam melakukan pra-proses data beserta *Code*.

1. **Tanda Petik**

   Tahapan tanda petik menggunankan permrograman python sederhana. Proses penghapusan tanda petik ditujukkan dengan kode 5.1.

```
def tandapetik(words):
    new_word = words.replace("'", "")
    return new_word
```

Kode 5.1 *Code* Proses Penghapusan Tanda Petik

Berikut adalah hasil implementasi penghapusan tanda petik.

Tabel 5.4 Hasil Implementasi Penghapusan Tanda Petik

Pertanyaan Sebelum Penghapusan Tanda Petik	**Pertanyaan Sesudah Penghapusan Tanda Petik**
Apakah surat peringatan yang dikirim ke sekolah berarti pemblacklistan sekolah dari ITS atau hanya pengurangan kuota saja?	Apakah surat peringatan yang dikirim ke sekolah berarti pemblacklistan sekolah dari ITS atau hanya pengurangan kuota saja?
Salam Sejahtera Saya lulusan D3 Teknik Komputer Universitas Telkom. Apakah saya dapat Lanjut Jenjang di ITS dengan mengambil program studi S1 Teknik Komputer? Jika Iya, saya dapat melihat informasi dimana?	Salam Sejahtera Saya lulusan D3 Teknik Komputer Universitas Telkom. Apakah saya dapat Lanjut Jenjang di ITS dengan mengambil program studi S1 Teknik Komputer? Jika Iya, saya dapat melihat informasi dimana?

Pertanyaan Sebelum Penghapusan Tanda Petik	Pertanyaan Sesudah Penghapusan Tanda Petik
Terima Kasih	Terima Kasih

2. *Tokenization*

Tahapan *tokenization* digunakan untuk mengubah pertanyaan menjadi *token*. Proses *tokenization* ditujukkan dengan kode 5.2.

```
def tokenization(words):
    new_word = nltk.word_tokenize(words)
    return new_word
```

Kode 5.2 *Code* Proses *Tokenization*

Berikut adalah hasil implementasi *Tokenization*.

Tabel 5.5 Hasil Implementasi *Tokenization*

Pertanyaan Sebelum *Tokenization*	Pertanyaan Sebelum *Tokenization*
Apakah surat peringatan yang dikirim ke sekolah berarti pemblacklistan sekolah dari ITS atau hanya pengurangan kuota saja?	['Apakah', 'surat', 'peringatan', 'yang', 'dikirim', 'ke', 'sekolah', 'berarti', 'pemblacklistan', 'sekolah', 'dari', 'ITS', 'atau', 'hanya', 'pengurangan', 'kuota', 'saja', '?']

Pertanyaan Sebelum *Tokenization*	**Pertanyaan Sebelum *Tokenization***
Salam Sejahtera Saya lulusan D3 Teknik Komputer Universitas Telkom. Apakah saya dapat Lanjut Jenjang di ITS dengan mengambil program studi S1 Teknik Komputer? Jika Iya, saya dapat melihat informasi dimana? Terima Kasih	['Salam', 'Sejahtera', 'Saya', 'lulusan', 'D3', 'Teknik', 'Komputer', 'Universitas', 'Telkom', '.', 'Apakah', 'saya', 'dapat', 'Lanjut', 'Jenjang', 'di', 'ITS', 'dengan', 'mengambil', 'program', 'studi', 'S1', 'Teknik', 'Komputer', '?', 'Jika', 'Iya', ',', 'saya', 'dapat', 'melihat', 'informasi', 'dimana', '?', 'Terima', 'Kasih']

3. *Casefolding*

Tahapan *casefolding* digunakan menyamakan kalimat menjadi bentuk yang sama (*lowercase*). Proses *casefolding* ditujukkan dengan kode 5.3.

```
1  def to_lowercase(words):
2      new_words = []
3      for word in words:
4          new_word = word.lower()
5          new_words.append(new_word)
6      return new_words
```

Kode 5.3 *Code* Proses *Casefolding*

Berikut adalah hasil implementasi *casefolding*.

Tabel 5.6 Hasil Implementasi *Casefolding*

Pertanyaan sebelum *Casefolding*	**Pertanyaan setelah *Casefolding***
['Apakah', 'surat', 'peringatan', 'yang', 'dikirim', 'ke', 'sekolah', 'berarti', 'pemblacklistan', 'sekolah', 'dari', 'ITS', 'atau', 'hanya', 'pengurangan', 'kuota', 'saja', '?']	['apakah', 'surat', 'peringatan', 'yang', 'dikirim', 'ke', 'sekolah', 'berarti', 'pemblacklistan', 'sekolah', 'dari', 'its', 'atau', 'hanya', 'pengurangan', 'kuota', 'saja', '?']
['Salam', 'Sejahtera', 'Saya', 'lulusan', 'D3', 'Teknik', 'Komputer', 'Universitas', 'Telkom', '.', 'Apakah', 'saya', 'dapat', 'Lanjut', 'Jenjang', 'di', 'ITS', 'dengan', 'mengambil', 'program', 'studi', 'S1', 'Teknik', 'Komputer', '?', 'Jika', 'Iya', ',', 'saya', 'dapat', 'melihat', 'informasi', 'dimana', '?', 'Terima', 'Kasih']	['salam', 'sejahtera', 'saya', 'lulusan', 'd3', 'teknik', 'komputer', 'universitas', 'telkom', '.', 'apakah', 'saya', 'dapat', 'lanjut', 'jenjang', 'di', 'its', 'dengan', 'mengambil', 'program', 'studi', 's1', 'teknik', 'komputer', '?', 'jika', 'iya', ',', 'saya', 'dapat', 'melihat', 'informasi', 'dimana', '?', 'terima', 'kasih']

4. *Remove Punctuation*

Tahapan *remove punctuation* digunakan untuk menghilangkan tanda baca. Proses *remove punctuation* ditujukkan dengan kode 5.4.

```
def remove_punctuation(words):
    new_words = []
    for word in words:
        new_word = re.sub(r'[^\w\s]', '', word)
        if new_word != '':
            new_words.append(new_word)
    return new_words
```

Kode 5.4 *Code* Proses *Remove Punctuation*

Berikut adalah hasil implementasi *remove punctuation*.

Tabel 5.7 Hasil Implementasi *Remove Punctuation*

Pertanyaan sebelum *Remove Punctuation*	**Pertanyaan setelah *Remove Punctuation***
['apakah', 'surat', 'peringatan', 'yang', 'dikirim', 'ke', 'sekolah', 'berarti', 'pemblacklistan', 'sekolah', 'dari', 'its', 'atau', 'hanya', 'pengurangan', 'kuota', 'saja', '?']	['apakah', 'surat', 'peringatan', 'yang', 'dikirim', 'ke', 'sekolah', 'berarti', 'pemblacklistan', 'sekolah', 'dari', 'its', 'atau', 'hanya', 'pengurangan', 'kuota', 'saja']
['salam', 'sejahtera', 'saya', 'lulusan', 'd3',	['salam', 'sejahtera', 'saya', 'lulusan', 'd3',

Pertanyaan sebelum *Remove Punctuation*	**Pertanyaan setelah *Remove Punctuation***
'teknik', 'komputer', 'universitas', 'telkom', '.', 'apakah', 'saya', 'dapat', 'lanjut', 'jenjang', 'di', 'its', 'dengan', 'mengambil', 'program', 'studi', 's1', 'teknik', 'komputer', '?', 'jika', 'iya', ',', 'saya', 'dapat', 'melihat', 'informasi', 'dimana', '?', 'terima', 'kasih']	'teknik', 'komputer', 'universitas', 'telkom', 'apakah', 'saya', 'dapat', 'lanjut', 'jenjang', 'di', 'its', 'dengan', 'mengambil', 'program', 'studi', 's1', 'teknik', 'komputer', 'jika', 'iya', 'saya', 'dapat', 'melihat', 'informasi', 'dimana', 'terima', 'kasih']

5. *Remove Single Character*

Tahapan *remove single character* digunakan untuk menghapus token yang hanya memiliki 1 huruf atau 1 angka. Proses *remove single character* ditujukkan dengan kode 5.5.

```
def remove_singlechar(words):
    new_words = []
    for word in words:
        if len(word) > 1:
            new_words.append(word)
    return new_words
```

Kode 5.5 *Code* Proses *Remove Single Character*

Berikut adalah hasil implementasi *remove single character.*

Tabel 5.8 Hasil Implementasi *Remove Single Character*

Pertanyaan sebelum *Remove Single Character*	**Pertanyaan setelah *Remove Single Character***
['apakah', 'surat', 'peringatan', 'yang', 'dikirim', 'ke', 'sekolah', 'berarti', 'pemblacklistan', 'sekolah', 'dari', 'its', 'atau', 'hanya', 'pengurangan', 'kuota', 'saja']	['apakah', 'surat', 'peringatan', 'yang', 'dikirim', 'ke', 'sekolah', 'berarti', 'pemblacklistan', 'sekolah', 'dari', 'its', 'atau', 'hanya', 'pengurangan', 'kuota', 'saja']
['salam', 'sejahtera', 'saya', 'lulusan', 'd3', 'teknik', 'komputer', 'universitas', 'telkom', 'apakah', 'saya', 'dapat', 'lanjut', 'jenjang', 'di', 'its', 'dengan', 'mengambil', 'program', 'studi', 's1', 'teknik', 'komputer', 'jika', 'iya', 'saya', 'dapat', 'melihat', 'informasi', 'dimana', 'terima', 'kasih']	['salam', 'sejahtera', 'saya', 'lulusan', 'd3', 'teknik', 'komputer', 'universitas', 'telkom', 'apakah', 'saya', 'dapat', 'lanjut', 'jenjang', 'di', 'its', 'dengan', 'mengambil', 'program', 'studi', 's1', 'teknik', 'komputer', 'jika', 'iya', 'saya', 'dapat', 'melihat', 'informasi', 'dimana', 'terima', 'kasih']

6. *Stop Words*

Tahapan *stopwords* digunakan untuk menghilangkan kata-kata yang sering muncul. Proses *stopwords* ditujukkan dengan kode 5.6.

```
def stop_words(words):
    new_words = []
    for word in words:
        factory = StopWordRemoverFactory()
        stopword = factory.create_stop_word_remover()
        new_word = stopword.remove(word)
        new_words.append(new_word)
    return new_words
```

Kode 5.6 *Code* Proses *Stopwords*

Berikut adalah hasil implementasi *stopwords*.

Tabel 5.9 Hasil Implementasi *StopWords*

Pertanyaan Sebelum Proses *Stopwords*	**Pertanyaan Setelah Proses *Stopwords***
['apakah', 'surat', 'peringatan', 'yang', 'dikirim', 'ke', 'sekolah', 'berarti', 'pemblacklistan', 'sekolah', 'dari', 'its', 'atau', 'hanya', 'pengurangan', 'kuota', 'saja']	['', 'surat', 'peringatan', '', 'dikirim', '', 'sekolah', 'berarti', 'pemblacklistan', 'sekolah', '', 'its', '', '', 'pengurangan', 'kuota', '']
['salam', 'sejahtera', 'saya', 'lulusan', 'd3',	['salam', 'sejahtera', '', 'lulusan', 'd3', 'teknik',

Pertanyaan Sebelum Proses *Stopwords*	**Pertanyaan Setelah Proses *Stopwords***
'teknik', 'komputer', 'universitas', 'telkom', 'apakah', 'saya', 'dapat', 'lanjut', 'jenjang', 'di', 'its', 'dengan', 'mengambil', 'program', 'studi', 's1', 'teknik', 'komputer', 'jika', 'iya', 'saya', 'dapat', 'melihat', 'informasi', 'dimana', 'terima', 'kasih']	'komputer', 'universitas', 'telkom', '', '', '', 'lanjut', 'jenjang', '', 'its', '', 'mengambil', 'program', 'studi', 's1', 'teknik', 'komputer', '', 'iya', '', '', 'melihat', 'informasi', '', 'terima', 'kasih']

7. Stemming

Tahapan stemming digunakan untuk merubah kata menjadi bentuk kata dasarnya. Proses *stemming* ditujukkan dengan kode 5.7.

```
def stem_words(words):
    new_words = []
    for word in words:
        factorystemming = StemmerFactory()
        stemmer = factorystemming.create_stemmer()
        new_word = stemmer.stem(word)
        new_words.append(new_word)
    return new_words
```

Kode 5.7 *Code* Proses *Stemming*

Berikut adalah hasil implementasi *stemming*.

Tabel 5.10 Hasil Implementasi *Stemming*

Pertanyaan sebelum Stemming	**Pertanyaan setelah pra-Stemming**
['', 'surat', 'peringatan', '', 'dikirim', '', 'sekolah', 'berarti', 'pemblacklistan', 'sekolah', '', 'its', '', '', 'pengurangan', 'kuota', '']	['', 'surat', 'ingat', '', 'kirim', '', 'sekolah', 'arti', 'pemblacklistan', 'sekolah', '', 'its', '', '', 'kurang', 'kuota', '']
['salam', 'sejahtera', '', 'lulusan', 'd3', 'teknik', 'komputer', 'universitas', 'telkom', '', '', '', 'lanjut', 'jenjang', '', 'its', '', 'mengambil', 'program', 'studi', 's1', 'teknik', 'komputer', '', 'iya', '', '', 'melihat', 'informasi', '', 'terima', 'kasih']	['salam', 'sejahtera', '', 'lulus', 'd3', 'teknik', 'komputer', 'universitas', 'telkom', '', '', '', 'lanjut', 'jenjang', '', 'its', '', 'ambil', 'program', 'studi', 's1', 'teknik', 'komputer', '', 'iya', '', '', 'lihat', 'informasi', '', 'terima', 'kasih']

8. DeTokenizer

Tahapan detokenizer digunakan untuk merubah kalimat yang berbentuk token menjadi bentuk kalimat. Proses detokenizer ditujukkan dengan kode 5.8.

```
1 words = TreebankWordDetokenizer().detokenize(words)
```

Kode 5.8 *Code* Proses *DeTokenizer*

Berikut adalah hasil implementasi *detokenizer*.

Tabel 5.11 Hasil Implementasi *DeTokenizer*

Pertanyaan sebelum *DeTokenizer*	**Pertanyaan setelah *DeTokenizer***
['', 'surat', 'ingat', '', 'kirim', '', 'sekolah', 'arti', 'pemblacklistan', 'sekolah', '', 'its', '', '', 'kurang', 'kuota', '']	surat ingat kirim sekolah arti pemblacklistan sekolah its kurang kuota
['salam', 'sejahtera', '', 'lulus', 'd3', 'teknik', 'komputer', 'universitas', 'telkom', '', '', '', 'lanjut', 'jenjang', '', 'its', '', 'ambil', 'program', 'studi', 's1', 'teknik', 'komputer', '', 'iya', '', '', 'lihat', 'informasi', '', 'terima', 'kasih']	salam sejahtera lulus d3 teknik komputer universitas telkom lanjut jenjang its ambil program studi s1 teknik komputer iya lihat informasi terima kasih

Secara keseluruhan dari tahapan pra-proses data tersebut maka berikut merupakan contoh luaran atau output dari pra-proses data yang telah dilakukan:

Tabel 5.12 Hasil Implementasi Pra-Proses Data

Pertanyaan sebelum pra-proses data	**Pertanyaan setelah pra-proses data**
Apakah surat peringatan yang dikirim ke sekolah berarti pemblacklistan sekolah dari ITS atau hanya pengurangan kuota saja?	surat ingat kirim sekolah arti pemblacklistan sekolah its kurang kuota
Salam Sejahtera Saya lulusan D3 Teknik Komputer Universitas Telkom. Apakah saya dapat Lanjut Jenjang di ITS dengan mengambil program studi S1 Teknik Komputer? Jika Iya, saya dapat melihat informasi dimana? Terima Kasih	salam sejahtera lulus d3 teknik komputer universitas telkom lanjut jenjang its ambil program studi s1 teknik komputer iya lihat informasi terima kasih

Kemudian data yang telah melalui proses pra-proses data akan disimpan pada *file* dengan format .xlsx.

5.2.3. Penyimpanan Data

Untuk mempersiapkan data dalam bentuk *format* data json maka dibuatlah Bahasa pemrograman python

sederhana untuk menkonversikan data yang sudah diolah (melalui pra-proses data) kedalam *format* json. Kode yang digunakan untuk menkoversikan data kedalam *format* json adalah sebagai berikut.

```
import xlrd
from collections import OrderedDict
import simplejson as json

wb = xlrd.open_workbook('datascenario.xlsx')
sh = wb.sheet_by_index(0)

data_list = []
for rownum in range(0, sh.nrows):
    data = OrderedDict()
    row_values = sh.row_values(rownum)
    data['message'] = row_values[0]
    data['response'] = row_values[1]
    data_list.append(data)
j = json.dumps(data_list)
with open('datafaq-pra-proses.json', 'w') as f:
    f.write(j)
```

Kode 5.9 *Code* Penyimpanan Data Json

5.3. Implementasi Pembuatan Aplikasi *Chatbot*

Pembuatan aplikasi *chatbot* bertujuan untuk membuat mesin *chatbot* dengan menggunakan pendekatan Infomation Retrieval. Aplikasi ini menggunakan pembobotan TF-IDF pada pertanyaan pengguna dan mencocokkan kemiripan dengan data pertanyaan pada PostgreSQL dengan menggunakan algoritma *Cosine Similarity* . Aplikasi *chatbot* yang dibuat merupakan aplikasi yang menggunakan bahasa pemrograman python dan menggunakan aplikasi *LINE* messenger sebagai medianya dengan *hosting* dari Heroku.com.

Informasi yang diberikan oleh aplikasi *chatbot* ini metupakan jawaban dari dataset dengan kemiripan atau nilai *Cosine Similarity* tertinggi.

5.3.1. Pembuatan Aplikasi *Chatbot*

Bagian ini merupakan langkah-langkah dalam melakukan implementasi pembuatan aplikasi *chatbot*. Berikut adalah beberapa bagian penting dari aplikasi *chatbot* dengan dilengkapi kode yang digunakan.

a. Pembacaan *Database*

Pada bagian ini pembacaan *Database* dilakukan dan *Database* yang digunakan adalah PostgreSQL. Sebelum dilakukan pembacaan *Database* maka dilakukan dahulu proses penghubungan aplikasi dengan *Database* ditunjukkan dengan kode 5.10.

Kemudian untuk pembacaan *Database* untuk data pertanyaan ditunjukkan dengan kode 5.11 sedangkan pembacaan *Database* untuk data jawaban ditunjukkan dengan kode 5.12.

```
connection = psycopg2.connect(
    user="xkgtwybjavpmvv",
    password="a2bd5d1f8aa594389d8ce410c68d8a9c1498729575a599d2699c556f424b4768",
    host="ec2-174-129-227-80.compute-1.amazonaws.com",
    port="5432",
    database="dbfmqd43g48h9s" , sslmode="require")
```

Kode 5.10 *Code* Koneksi *Database*

```
        cursor = connection.cursor()
        postgreSQL_select_Query = "SELECT info ->> 'message' AS pesan FROM faq;"
        cursor.execute(postgreSQL_select_Query)
        reader = cursor.fetchall()
        for row in reader:
            sentences.append(row[0])
            i += 1
```

Kode 5.11 *Code* Mengambil Data Pertanyaan

```
    cursor = connection.cursor()
    postgreSQL_select_Query = "SELECT info ->> 'response' AS pesan FROM faq;"
    cursor.execute(postgreSQL_select_Query)
    reader = cursor.fetchall()
    for row in reader:
        j = j + 1
        if j == response_index:
            return row[0], max
            break
```

Kode 5.12 *Code* Mengambil Data Jawaban

b. *Train Database*

Kemudian langkah berikutnya adalah *training* data dengan menggunakan pembobotan TF-IDF. Bagian ini digunakan untuk mencari bobot nilai untuk setiap pertanyaan pada dataset FAQ ITS. Proses ini menggunakan *library* sklearn karena mendukung perhitungan TF-IDF. Proses ini ditunjukkan dengan kode 5.13. Hasil dari proses training *Database* akan disimpan dalam file berekstensi .pickle. Hasil output dari proses ini ditunjukkan dengan gambar 5.1.

```
tfidf_vectorizer = TfidfVectorizer()
tfidf_matrix_train = tfidf_vectorizer.fit_transform(sentences)

f = open(tfidf_vectorizer_pikle_path, 'wb')
pickle.dump(tfidf_vectorizer, f)
f.close()

f = open(tfidf_matrix_train_pikle_path, 'wb')
pickle.dump(tfidf_matrix_train, f)
f.close()
```

Kode 5.13 *Code* Train Data Pertanyaan

```
(0, 751) 0.6557805284519534
(0, 1234) 0.7549515868605594
(1, 751) 0.6557805284519534
(1, 1234) 0.7549515868605594
(2, 1064) 0.24109633550391027
(2, 449) 0.27783836313930355
(2, 570) 0.2532603558354177
(2, 965) 0.4937474718946108
(2, 133) 0.34443275144810065
(2, 803) 0.34443275144810065
(2, 468) 0.09675384235043508
(2, 404) 0.27783836313930355
(2, 595) 0.2468737359473054
(2, 593) 0.34443275144810065
(2, 927) 0.23097004736988924
(3, 711) 0.24503923471006206
(3, 451) 0.15064709655434844
(3, 1086) 0.10999461538425748
(3, 940) 0.13280536320176564
(3, 434) 0.19820745064590065
(3, 1124) 0.24503923471006206
(3, 239) 0.30947189292592914
(3, 1211) 0.30947189292592914
(3, 930) 0.22347249517900877
```

Gambar 5.1 Hasil Luaran Data *Training*

c. Pembacaan Input pengguna

Tahapan ini adalah tahapan untuk meminta masukan *Query* yang diberikan oleh pengguna. *Query* yang dimaksud adalah pertanyaan yang diajukan oleh pengguna. Tahapan ini ditunjukkan dengan menggunakan kode 5.14.

```
def chats(query):
    minimum_score = 0.1
    querypraproses = stemming(stopwords(casefolding(query)))
    query_response, score = chatbot(querypraproses , minimum_score ,
                                    file , tfidf_vectorizer_pikle_path ,
                                    tfidf_matrix_train_path)
    print(score)
    return query_response

while 1:
    sent = input("bot ITS : ")
    print(chats(sent))
```

Kode 5.14 *Code* Mengambil *Query* Pengguna

d. Pembobotan TF-IDF pada *Query* pengguna

Setelah dilakukan pembacaan pertanyaan dari pengguna maka pertanyaan tersebut akan melaui pra-proses data. Setelah dilakukan pra-proses data maka pertanyaan tersebut akan dicari bobotnya dengan menggunakan metode pembobotan TF-IDF. Proses pembobotan ini ditunjukkan dengan menggunakan kode 5.15.

```
#train data input user
tfidf_matrix_test = tfidf_vectorizer.transform(query)
```

Kode 5.15 *Code* Mengambil Data Pertanyaan

e. *Cosine Similarity*

Proses dilakukan setelah proses training *Database* dan pengolahan data dari *Query* pengguna. Input dari proses ini adalah bobot yang telah didapatkan sebelumnya dari proses pembobotan TF-IDF. Proses ini akan menghitung kemiripan antara dataset FAQ ITS dengan data *Query* pengguna. Output dari proses ini adalah nilai indeks yang

memiliki nilai kemiripan tertinggi. Proses ini ditunjukkan dengan menggunakan kode 5.16.

```
#Cosine Similarity
cosine = cosine_similarity(tfidf_matrix_test, tfidf_matrix_train)

#Ranking Cosine Similarity
cosine = np.delete(cosine, 0)

max = cosine.max()
```

Kode 5.16 *Code* Algortima Cosine Similarity

f. Output

Pada tahap ini merupakan tahap pembacaan data dengan input dari proses sebelumnya yaitu nilai indeks yang memiliki nilai kemiripan tertinggi. Nilai indeks tersebut adalah indeks yang merujuk pada pertanyaan yang memiliki nilai kemiripan tertingi. Setelah menemukan indeks tersebut maka pasangan jawaban dari pertanyaan tersebut akan digunakan sebagai output dari proses ini. Proses ini ditunjukkan dengan kode 5.17.

```
response_index = 0

#cek lolos minimum score
if (max > minimum_score):
    new_max = max
    list = np.where(cosine == new_max)
    #Cari Indeks
    response_index = np.amax(list[0])
else :
    return "Tidak Bisa Menjawab" , 0

j = 0
cursor = connection.cursor()
postgreSQL_select_Query = "SELECT info ->> 'response' AS pesan FROM faq;"
cursor.execute(postgreSQL_select_Query)
reader = cursor.fetchall()
for row in reader:
    j = j + 1
    if j == response_index:
        return row[0], max
        break
```

Kode 5.17 *Code* Ouput

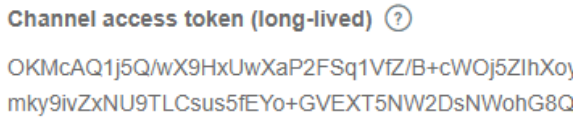
5.4. Implementasi Pembuatan Aplikasi *LINE Chatbot*

Pembuatan aplikasi *LINE chatbot* bertujuan untuk mengabungkan aplikasi *LINE* messenger dengan mesin *chatbot* yang telah dibuat dengan menggunakan *hosting* dari heroku. Heroku merupakan *hosting* yang telah bekerja sama secara resmi dengan pihak *LINE* messenger sehingga bisa terintegrasi dengan baik. Berikut adalah langkah-langkah dalam membuat aplikasi *LINE chatbot*.

a. Membuat akun provider *LINE*

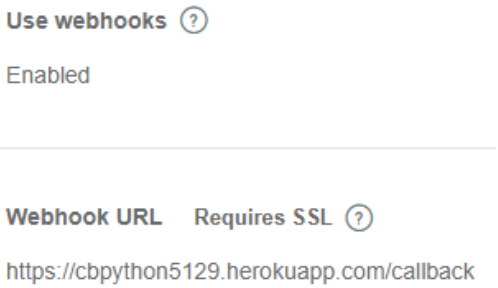
Hal yang pertama adalah mendaftar pada *website* developers.*LINE.biz* untuk membuat akun *Official Account* (OA) *LINE* yang akan digunakan sebagai akun media sosial. Bagian penting untuk mengidentifikasi *LINE* ditunjukkan pada gambar 5.2 dan gambar 5.3.

Gambar 5.2 Channel Secret



Gambar 5.3 Channel Access Token

Terdapat dua token yaitu "*Channel secret*" dan "*Channel access token*". *Token - token* tersebut akan digunakan untuk menyambungkan *hosting* Heroku dengan aplikasi *LINE Messenger*. Kemudian aktifkan mode *webhook* yang berfungsi untuk menyalurkan *Query* pengguna kedalam *hosting* Heroku. *Webhook* yang digunakan pada aplikasi *chatbot* ditunjukkan dengan menggunakan gambar 5.4.



Gambar 5.4 webhook Heroku

Alamat *webhook* didapatkan dari alamat aplikasi yang telah kita buat pada *hosting* Heroku.

b. Membuat Project Aplikasi di Heroku

Kemudian pada tahap ini adalah membuat project baru kedalam *hosting* Heroku. Pada penelitian tugas akhir ini nama project yang dibuat adalah “cbpython5129”.

c. Membuat *Database* di Heroku

Proses ini merupakan proses untuk menyiapkan dataset agar bisa digunakan oleh aplikasi *chatbot*. Basis data yang digunakan dalam melakukan penyimpanan adalah PostgreSQL dengan memiliki 2 tabel utama yaitu tabel untuk menyimpan data FAQ ITS dan yang kedua adalah data pertanyaan yang belum pernah diajukan sebelumnya. Berikut adalah langkah langkah dalam membuat *Database* untuk aplikasi *chatbot*.

- **Membuat Tabel**

  Pada tahap ini merupakan proses untuk membuat tabel pada PostgreSQL. Pembuatan tabel FAQ ITS dilakukan dengan menggunakan kode 5.18. Sedangkan tabel untuk pertanyaan yang belum pernah diajukan menggunakan kode 5.19.

```
1  CREATE TABLE faq (
2      ID serial NOT NULL PRIMARY KEY,
3      info json NOT NULL
4  );
```

Kode 5.18 *Code* Membuat Tabel faq

```
CREATE TABLE pertanyaan(
   ID serial NOT NULL PRIMARY KEY,
   info json NOT NULL
);
```

Kode 5.19 *Code* Membuat Tabel pertanyaan

- **Mengisi Data**

  Proses ini merupakan proses untuk mengisi data kedalam tabel. Proses pengisian data untuk tabel FAQ ITS ditunjukkan dengan kode 5.20.

```
INSERT INTO faq (info)
VALUES
    ('{
        "message": "Apakah surat peringatan yang dikirim ke sek
        "response": "Terkait pertanyaan yang diajukan, setelah
    }'),
    ('{
        "message": "min, saya ingin bertanya. Jika saya mengiku
        "response": "Menjawab pertanyaan Mbak Rufaidah, silahka
    }')
```

Kode 5.20 *Code* Mengisi Tabel faq

d. Menggabungkan mesin *chatbot* dengan Heroku dan *LINE*

Proses ini merupakan proses integrase aplikasi *chatbot* antara *LINE* messenger dan *hosting* Heroku. Berikut adalah bagian-bagian dalam aplikasi *chatbot*.

- **Koneksi ke *Database***

  Kode 5.21 digunakan untuk menyambungkan *Database* PotsgreSQL terhadap aplikasi chatbot.

```
connection = psycopg2.connect(
    user="xkgtwybjavpmvv",
    password="a2bd5d1f8aa594389d8ce410c68d8a9c1498729575a599d2699c556f424b4768",
    host="ec2-174-129-227-80.compute-1.amazonaws.com",
    port="5432",
    database="dbfmqd43g48h9s" , sslmode="require")
```

Kode 5.21 *Code* Koneksi ke *Database* PostgreSQL

- **Koneksi ke *LINE* Messenger**

  Kode 5.22 digunakan untuk menyambungkan aplikasi *chatbot* dengan aplikasi *Line Messenger*. Pada kode tersebut digunakan *token* yang diperoleh dari gambar 5.2 dan gambar 5.3.

```
# Channel Access Token
line_bot_api = LineBotApi('OKMcAQ1j5Q/wX9HxUwXaP2FSq1VfZ/B+c'+
'WOj5ZIhXoy4Tki6dqe92l+TjPFcRFX05EpZOu3tm27KEqfEt+cNsy63Jz5uOLBS'+
'FLd5v8liNq7Xlmky9ivZxNU9TLCsus5fEYo+GVEXT5NW2Ds'+
'NWohG8QdB04t89/1O/w1cDnyilFU=')
# Channel Secret
handler = WebhookHandler('bd42b12ab0f81591ef044c73b25d24db')
```

Kode 5.22 *Code* Koneksi Ke Line Messenger

- **Pembuatan App Route**

  Kode 5.32 merupakan kode untuk membuat *app route*. Kode ini digunakan sebagai alamat aplikasi kita dan digunakan pada *webhook url* pada gambar 5.4.

```
@app.route("/callback", methods=['POST'])
def callback():
    signature = request.headers['X-Line-Signature']
    body = request.get_data(as_text=True)
    app.logger.info("Request body: " + body)
    try:
        handler.handle(body, signature)
    except InvalidSignatureError:
        abort(400)
    return 'OK'
```

Kode 5.23 *Code* Membuat *App Route* Heroku

- **Pembacaan *Query* Pengguna**

  Tahap ini merupakan tahap untuk menangkap query dari pengguna aplikasi Line Messenger. Pembacaan query pengguna ditunjukkan dengan kode 5.24. Pada baris 7 query dari pengguna akan dilakukan proses training dengan aplikasi chatbot yang telah dibuat.

```
@handler.add(MessageEvent, message=TextMessage)
def handle_message(event):
    text = event.message.text #simplify for receove message
    sender = event.source.user_id #get user_id
    gid = event.source.sender_id #get group_id
    print(chats(text))
    jawaban = previous_chats(text)
```

Kode 5.24 *Code* Membaca *Query* Pengguna *Line Messenger*

- Mengirimkan Jawaban Kepada Pengguna

  Tahap ini merupakan tahap terkahir yaitu menampilkan jawaban balasan dari aplikasi *chatbot* berdasarkan *query* pengguna. Tahap ini ditunjukkan dengan kode 5.25.

```
line_bot_api.reply_message(event.reply_token, TextSendMessage(jawaban))
```

Kode 5.25 *Code* Mengirimkan Jawaban Ke Pengguna

e. Upload *chatbot* ke Heroku

Perintah CLI untuk aplikasi *LINE Chatbot*

- *heroku login*

  Perintah ini digunakan untuk *login* kedalam akun *hosting* Heroku.

- *git init*

  Perintah ini digunakan untuk inisialisasi git repository karena Heroku menggunakan git sebagai repositorinya.

- *heroku git:remote -a cbpython5129*

  Peintah ini digunakan untuk memegang kendali aplikasi cbpython5129 yang telah dibuat sebelumnya.

- *git add .*

  Perintah ini digunakan untuk menambahkan semua file yang ingin diunggah kedalam *repository git*.

- *git commit -m "1"*

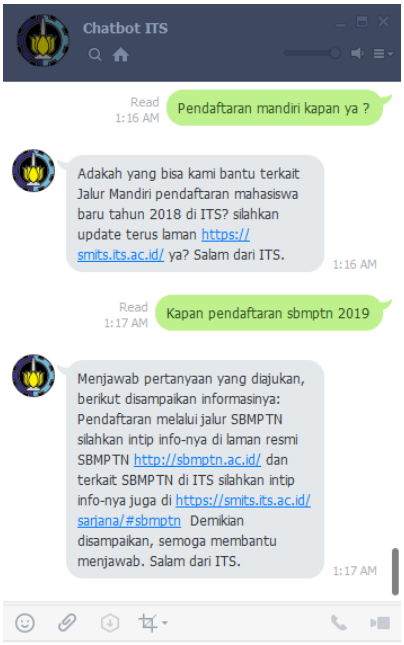
  Perintah ini digunakan memberikan keterangan atau penanda saat akan melakukan penggunggahan data ke dalam *git repository*.

- *git push heroku master*

  Perintah ini digunakan untuk mengunggah file ke dalam *git repository* aplikasi cbpyhton5129 pada *hosting* Heroku.

f. Hasil Visualisasi Aplikasi *LINE* Chabot

Contoh visualisasi aplikasi *Line* yang terhubung dengan aplikasi *chatbot* yang telah dibuat ditunjukkan dengan gambar 5.5.



Gambar 5.5 Hasil Visualisasi *Line Messenger*

BAB VI
HASIL DAN PEMBAHASAN

Pada bab ini akan menjelaskan mengenai proses dan hasil uji coba eksperimen serta analisis terhadap hasil yang diperoleh dari proses implementasi pembobotan TF-IDF dan algoritma *Cosine Similarity*, termasuk eksperimen nilai minimum yang digunakan.

6.1. Data Uji Coba

Data yang digunakan sebagai uji coba adalah FAQ ITS dengan menggunakan 3 skenario yaitu :

Tabel 6.1 Skenario Eksperimen

Skenario	Penjelasan
Skenario 1 (S1)	Pertanyaan yang diajukan sama dengan pertanyaan yang telah direkam oleh FAQs *Database*
Skenario 2 (S2)	Pertanyaan yang diajukan berbeda dengan pertanyaan yang telah direkam oleh FAQs *Database* (pertanyaan yang belum ditanyakan sebelumnya)
Skenario 3 (S3)	Pertanyaan yang diajukan menggunakan skenario 1 dan skenario 2

6.2. Lingkungan Uji Coba

Pada subbab Lingkungan Uji Coba ini akan menjelaskan terkait lingkungan pengujian dalam melakukan implementasi penelitian tugas akhir terkait evaluasi performa aplikasi *chatbot*. Spesifikasi perangkat keras yang digunakan dalam implementasi ditunjukkan pada Tabel 6.2.

Tabel 6.2 Spesifikasi Perangkat Keras

Perangkat Keras	Spesifikasi
Jenis	Laptop Asus A46CB
Processor	Intel(R) Core i7-3537U
RAM	4 GB
Storage	240GB SSD 1000GB HDD

Untuk spesifikasi perangkat lunak yang digunakan dalam pengerjaan penelitian tugas akhir ditunjukkan pada Tabel 6.3.

Tabel 6.3 Spesifikasi Perangkat Lunak

Perangkat Lunak	Fungsi
Windows 10 Education 64 bit	Sistem Operasi
Thonny	Python IDE (Implementasi Algoritma)
Database	PostgreSQL
Notepad++	Pengolahan Data dan Hasil
Microsoft Excel 365	Pengolahan Hasil Uji Coba
Microsoft Word 365	Penulisan Laporan

6.3. Analisis Hasil Pra-Proses Data

Dataset pertanyaan FAQ ITS yang mentah berjumlah 213 pertanyaan yang memiliki jumlah kata sebanyak 8835 kata berkurang menjadi 6165 kata setelah melalui pra-proses data seperti pada tahapan sub bab 5.2.2. Pengurangan jumlah kata dalam dataset ini dikarenakan proses *Remove Punctuation , Remove Single Character, Stopwords* dan *Stemming* dengan melakukan pembuangan kata dan karakter yang tidak penting dalam data pertanyaan. Berikut adalah rincian jumlah kata yang tersisa pada setiap tahapan pra-proses data:

Tabel 6.4 Hasil Implementasi Pra-Proses Data

Tahap Pra-Proses Data	Jumlah Kata
Data Mentah	8835 Kata
Tanda Petik	8835 Kata
Tokenization	8835 Kata
Casefolding	8835 Kata
Remove Punctuation	8761 Kata
Remove Single Character	8613 Kata
Stopwords	6165 Kata
Stemming	6165 Kata

Tahap Pra-Proses Data	Jumlah Kata
DeTokenizer	6165 Kata

Berikut merupakan ringkasan hasil dari pra-proses data:

Tabel 6.5 Ringkasan Pra-Proses Data

Jumlah	Sebelum Pra-Proses	Setelah Pra-Proses
Pertanyaan	213	213
Total Kata	8835 Kata	6165 Kata

6.4. Analisis Hasil Eksperimen

Evaluasi performa untuk aplikasi *chatbot* dilakukan berdasarkan skenario pada tabel 6.1. Eksperimen dilakukan dengan menggunakan pertanyaan untuk setiap skenario sebanyak 1 pertanyaan karena menggunakan perhitungan manual untuk setiap perhitungannya dan dirasa mewakili performa sistem yang dibuat. Dataset FAQ ITS yang digunakan pada *Database* berisi sebanyak 213 pasang FAQ dengan rincian data training sebanyak 173 pasang data FAQ dan data testing 40 pasang data FAQ. Berikut adalah hasil eksperimen untuk setiap skenario.

6.4.1. Analisis Hasil Eksperimen Skenario 1

Subbab ini menjelaskan mengenai performa pembobotan TF-IDF dan Algoritma *Cosine Similarity* dalam mengeksekusi skenario 1. Pada skenario 1 pertanyaan yang digunakan adalah pertanyaan yang sama persis dengan pertanyaan yang pernah diajukan sebelumnya. Skenario 1 mengukur bagaimana performa aplikasi *chatbot* terhadap pertanyaan yang sama persis. Bagian ini juga menjelaskan hasil eksperimen pada tabel 6.6.

Tabel 6.6 Hasil Eksperimen Skenario 1

	Skenario
	Skenario 1
Rata - Rata	1.00000
Standar Deviasi	0.00000
Standar *Error*	0.00000

Seperti pada tabel 6.6, pada kondisi skenario 1 performa dari aplikasi *chatbot* memiliki nilai standar deviasi dan standar *error* sebesar 0. Hal ini mengindikasikan bahwa nilai rata - rata pada tabel 6.6 merupakan nilai rata – rata kemiripan pada sistem dengan nilai standar *error* sebesar 0. Dengan nilai *standard error* sebesar 0 maka bisa disimpulkan bahwa performa rata – rata kemiripan yang diberikan oleh sistem adalah sebesar 1.

6.4.2. Analisis Hasil Eksperimen Skenario 2

Pada Subbab ini menjelaskan mengenai performa pembobotan TF-IDF dan Algoritma *Cosine Similarity* dalam mengeksekusi skenario 2. Pada skenario 2 pertanyaan yang digunakan adalah pertanyaan yang berbeda dengan pertanyaan yang pernah diajukan sebelumnya namun jawaban akan pertanyaan tersebut masih ada dalam *Database* FAQ ITS. Skenario 2 mengukur bagaimana performa aplikasi *chatbot* terhadap pertanyaan yang berbeda namun tetap tersedia jawaban yang diiginkan dalam *Database*. Bagian ini juga menjelaskan hasil eksperimen pada tabel 6.7.

Tabel 6.7 Hasil Eksperimen Skenario 2

	Skenario
	Skenario 1
Rata - Rata	0.32544733
Standar Deviasi	0.092962009
Standar *Error*	0.014698584

Seperti pada tabel 6.7, pada kondisi skenario 2 performa dari aplikasi *chatbot* memiliki nilai standar *error* sebesar 0.014698584. Hal ini mengindikasikan bahwa nilai rata - rata pada tabel 6.7 merupakan nilai rata – rata kemiripan pada sistem dengan nilai standar *error* sebesar 0.014698584. Dengan nilai standar *error* sebesar 0.014698584 atau ~1% maka bisa disimpulkan bahwa performa rata – rata kemiripan yang diberikan

oleh sistem adalah sebesar 0.32544733 karena memiliki tingkat standar *error* sebanyak 1% dan bisa dianggap dekat dengan nilai rata – rata kemiripan keseluruhan data FAQ untuk pertanyaan yang belum direkam pada *database*.

6.4.3. Analisis Hasil Eksperimen Skenario 3

Subbab ini menjelaskan mengenai performa pembobotan TF-IDF dan Algoritma *Cosine Similarity* dalam mengeksekusi skenario 3. Pada skenario 3 pertanyaan yang digunakan adalah pertanyaan yang sama persis dengan pertanyaan yang pernah diajukan sebelumnya. Skenario 3 mengukur bagaimana performa aplikasi *chatbot* terhadap pertanyaan yang sama persis dan sedikit berbeda namun tetap memiliki jawaban yang diinginkan pada *Database*. Bagian ini juga menjelaskan hasil eksperimen pada tabel 6.8.

Tabel 6.8 Hasil Eksperimen 3

	Skenario
	Skenario 1
Rata - Rata	0.662550216
Standar Deviasi	0.340620551
Standar *Error*	0.053856838

Seperti pada tabel 6.8, pada kondisi skenario 3 performa dari aplikasi chatbot memiliki nilai standar *error* sebesar 0.053856838. Hal ini mengindikasikan bahwa nilai rata - rata pada tabel 6.8 merupakan nilai rata – rata

kemiripan pada sistem dengan nilai standar *error* sebesar 0.053856838. Dengan nilai standar *error* sebesar 0.053856838 atau ~5% maka bisa disimpulkan bahwa performa rata – rata kemiripan yang diberikan oleh sistem untuk skenario 3 adalah sebesar 0.662550216 karena memiliki tingkat standar *error* sebanyak ~5% dan bisa dianggap dekat dengan nilai rata – rata kemiripan keseluruhan data FAQ untuk pertanyaan dengan skenario 3.

6.5. Perbandingan Hasil Eksperimen

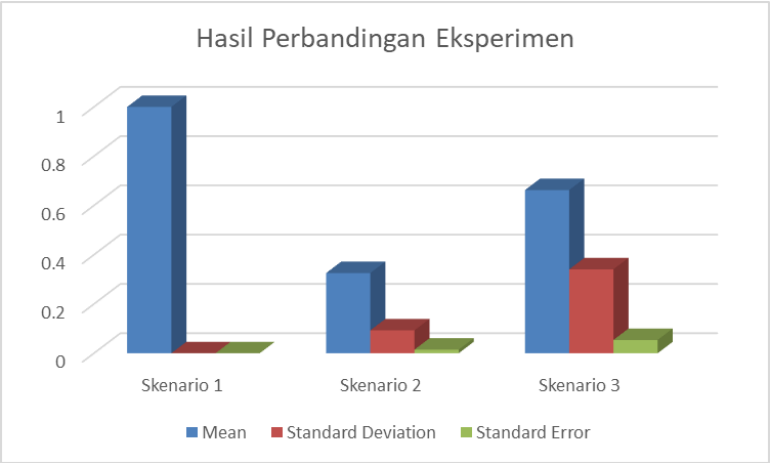
Pada subbab ini dilakukan pembandingan performa melalui nilai *mean, standard deviation* dan *standard error*.

6.5.1. Perbandingan Hasil Eksperimen Penelitian

Perbandingan hasil eksperimen penelitian dilakukan terhadap hasil yang telah didapatkan pada bagian sebelumnya. Perbandingan dilakukan pada hasil eksperimen skenario 1, skenario 2 dan skenario 3. Adapun perbandingan nilai rata-rata, standar deviasi dan *standard error* ditunjukkan pada tabel 6.9 sedangkan visualisasi perbandingan eksperimen ditunjukkan dengan gambar 6.1.

Tabel 6.9 Perbandingan Hasil Skenario

	Skenario		
	Skenario 1	**Skenario 2**	**Skenario 3**
Rata - Rata	1.00000	0.32544733	0.662550216
Standar Deviasi	0.00000	0.092962009	0.340620551
Standar *Error*	0.00000	0.014698584	0.053856838



Gambar 6.1 Perbandingan Hasil Eksperimen

Berdasarkan pada hasil perbandingan pada tabel 6.9, nilai *mean* merupakan nilai rata – rata kemiripan untuk setiap pertanyaan yang diajukan pada setiap skenario yang digunakan. Nilai *standard error* digunakan untuk melihat seberapa akurat nilai rata – rata untuk merepresentasikan performa sistem chatbot dengan melihat kemiripannya untuk setiap skenario yang digunakan.

Pada perbandingan nilai *standar error* pada tabel 6.9 bisa nilai *standard error* untuk skenario 1 adalah 0. Hal ini mengindikasikan bahwa nilai rata – rata pada skenario 1 merupakan nilai rata – rata sangat dekat dengan nilai rata – rata yang sebenarnya (keseluruhan data). Hal ini juga mengindikasikan bahwa sistem *chatbot* pada evaluasi skenario 1 telah mampu memberikan nilai kemiripan sebesar 1(nilai maksimal kemiripan) dengan nilai standard *error* sebesar 0% atau memiliki akurasi kedekatan sebesar 100% dari rata – rata

keseluruhan data. Dengan nilai rata – rata kemiripan sebesar 1 maka sistem dengan sempurna dapat mencari dokumen pertanyaan yang sesuai pada data FAQ.

Pada perbandingan nilai *standar error* pada tabel 6.9 bisa nilai *standard error* untuk skenario 2 adalah 0.014698584. Hal ini mengindikasikan bahwa nilai rata – rata pada skenario 2 merupakan nilai rata – rata sangat dekat dengan nilai rata – rata yang sebenarnya (keseluruhan data). Hal ini juga mengindikasikan bahwa sistem *chatbot* pada evaluasi skenario 2 telah mampu memberikan nilai kemiripan sebesar 0.32544733 (nilai maksimal kemiripan adalah 1) dengan nilai standard *error* sebesar 1% atau memiliki akurasi kedekatan sebesar 99% dari rata – rata keseluruhan data. Hal ini dikarenakan pada skenario 2, pertanyaan yang digunakan adalah pertanyaan yang belum direkam oleh data FAQ sehingga hanya memiliki rata – rata kemiripan sebesar 0.32544733.

Pada perbandingan nilai *standar error* pada tabel 6.9 bisa nilai *standard error* untuk skenario 3 adalah 0.053856838. Hal ini mengindikasikan bahwa nilai rata – rata pada skenario 3 merupakan nilai rata – rata sangat dekat dengan nilai rata – rata yang sebenarnya (keseluruhan data). Hal ini juga mengindikasikan bahwa sistem *chatbot* pada evaluasi skenario 2 telah mampu memberikan nilai kemiripan sebesar 0.662550216 (nilai maksimal kemiripan adalah 1) dengan nilai standard *error* sebesar 5% atau memiliki akurasi kedekatan sebesar 95% dari rata – rata keseluruhan data.

BAB VII
KESIMPULAN DAN SARAN

Pada bab ini akan dijelaskan rangkuman singkat yang dapat disimpulkan dari penelitian ini. Terdapat saran dari penulis yang diharapkan dapat membantu dalam meningkatkan hasil pada penelitian selanjutnya.

7.1. Kesimpulan

Berdasarkan hasil yang telah diuraikan pada bagian sebelumnya, kesimpulan yang dapat diambil adalah,

1. Algoritma *cosine simlarity* dan pembobotan TF-IDF bisa diterapkan sebagai sistem *chatbot* pada data FAQ ITS.
2. Hasil dari metode pembobotan TF-IDF dan Cosine Similarity menghasilkan nilai rata-rata kemiripan 1 pada skenario 1. Hal ini mengindikasikan bahwa *sistem chatbot* bisa dengan sempurna mencari pertanyaan yang telah direkam sebelumnya.
3. Hasil dari metode pembobotan TF-IDF dan algoritma *Cosine Similarity* menghasilkan nilai rata-rata kemiripan 0.32544733 pada skenario 2. Hal ini mengindikasikan bahwa sistem *chatbot* memiliki performa yang buruk untuk pertanyaan dengan skenario 2. Hal ini dikarenakan pertanyaan yang digunakan adalah pertanyaan yang belum pernah ditanyakan dalam data FAQ.

4. Hasil dari metode pembobotan TF-IDF dan algoritma *Cosine Similarity* menghasilkan nilai rata-rata kemiripan 0.662550216 pada skenario 3. Hal ini mengindikasikan bahwa sistem *chatbot* memiliki performa yang baik. Hal ini dikarenakan terdapat kombinasi dari pertanyaan skenario 1 dan skenario 2.
5. Dari hasil perbandingan 3 skenario yang telah dilakukan, bisa disimpulkan bahwa sistem chatbot mampu dengan baik menjawab pertanyaan yang telah direkam oleh data FAQ namun memiliki performa yang buruk untuk pertanyaan yang belum pernah direkam sebelumnya.
6. Adapun pada pendekatan konsep Information Retrieval, proses pra-proses data sangat penting karena mempengaruhi bobot kata pada metode pembobotan TF-IDF sehingga mempengaruhi performa aplikasi chatbot.

7.2. Saran

Berdasarkan hasil dan kesimpulan tersebut, saran yang dapat diberikan untuk penelitian selanjutnya adalah,

1. Penelitian ini hanya melakukan pra-proses data berdasarkan dengan *library* yang digunakan. Sehingga bisa ditambahkan proses pra-proses data dengan menambahkan proses pra-proses data yang belum dilakukan pada penelitian tugas akhir ini.

2. Penelitian ini hanya menggunakan *stopwords* dari *library* yang digunakan. Penambahan kata pada stopwords dapat juga diimplementasikan untuk menemukan hasil yang lebih optimum.
3. Pengembangan algoritma *text similarity* yang digunakan pada penelitian ini hanya menggunakan algoritma cosine similarity sehingga penemuan pengembangan lain juga dapat diimplementasikan untuk menemukan hasil yang lebih optimum.

Halaman ini sengaja dikosongkan

DAFTAR PUSTAKA

[1] Y. Zhao, D. Nan, B. Junwei, C. Peng, Z. Ming, L. Zhoujun dan Z. Jianshe, "DocChat: An Information Retrieval Approach for Chatbot Engines Using Unstructured Documents," *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics,* p. 516–525, 2016.

[2] Z. M. B. Wlodek dan C. J, "NATURAL LANGUAGE DIALOGUE for Personalized Interaction," *COMMUNICATIONS OF THE ACM,* vol. 43, no. 8, pp. 116-120, 2000.

[3] U. T. Victor, P. Christo dan O. Michael, "Performance Evaluation of Similarity Measures on Similar and Dissimilar Text Retrieval," *Proceedings of the 7th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2015),* vol. 1, pp. 577-584, 2015.

[4] A. Shtok, G. Dror, Y. Maarek dan I. Szpektor, "Learning from the Past: Answering New Questions with Past Answers," *WWW 2012 – Session: Leveraging User-Generated Content,* 2012.

[5] R. W. Schmidt, "Learning System Customer Service Chatbot," 2018.

[6] M. N. P, "An introduction to information retrieval: applications in genomics," *Pharmacogenomics J,* pp. 96-102, 2002.

[7] Oracle, “CHATBOTS 101,” Oracle, 2017.

[8] N. Ogie, Jumadi dan N. Dian, “PERBANDINGAN METODE COSINE SIMILARITY DENGAN METODE JACCARD SIMILARITY PADA APLIKASI PENCARIAN TERJEMAH AL-QUR’AN DALAM BAHASA INDONESIA,” *JOIN*, vol. 1, no. 1, 2016.

[9] A. Mukhlason, “Semantic Web Aware-Knowledge Management Driven e-Learning System: An Enhanced e-Learning Management System Using Semantic Web and Knowledge Management Technology,” dalam *Semantic Web Aware-Knowledge Management Driven e-Learning System: An Enhanced e-Learning Management System Using Semantic Web and Knowledge Management Technology*, PERAK, 2009, pp. 52-56.

[10] G. Lestarina, “LINE Chatbot,” BEKRAF Developer Day, Surabaya, 2017.

[11] J. Lafferty dan C. Zhai, “Document Language Models, Query Models, and Risk Minimization for Information Retrieval,” *SIGIR conference on Research and development in information retrieval*, pp. 111-119, 2001.

[12] S. KAREN, “A STATISTICAL INTERPRETATION OF TERM SPECIFICITY AND ITS APPLICATION IN RETRIEVAL,” *Journal of Documentation*, vol. 28, no. 1, pp. 11-21, 1972.

[13] D. Kane, “The Role of Chatbots in Teaching and

Learning," 2016.

[14] W. Joseph, "ELIZA—a computer program for the study of natural language communication between man and machine," *Communications of the ACM,* vol. 9, no. 1, pp. 36-45 , 1966 .

[15] I. Hanif, Z. A. Agus dan A. N. Dini, "Klasifikasi Perintah Bahasa Natural Menggunakan Global Vectors for Word Representations (GloVe), Convolutional Neural Networks, dan Teknik Transfer Learning pada Aplikasi Chatbots," *Jurnal Teknik ITS,* 2018.

[16] P. L. H, "A Statistical Approach to Mechanized Encoding and Searching of Literary Information," *IBM Journal of Research and Development,* vol. 1, no. 4, pp. 309-317 , 1957 .

[17] D. Feng, E. Shaw, J. Kim dan E. Hovy, "An Intelligent Discussion-Bot for Answering Student Queries in Threaded Discussions," *Proceedings of the 11th international conference on Intelligent user interfaces,* pp. 171-177, 2006.

[18] Fabricio, "Why chatbots fail," [Online]. Available: chatbot.fail. [Diakses 26 February 2019].

[19] S. H. Dhebys dan P. U. Yoga, "APLIKASI CHATBOT BERBASIS WEB PADA SISTEM INFORMASI LAYANAN PUBLIK KESEHATAN DI MALANG DENGAN MENGGUNAKAN METODE TF-IDF," *Jurnal Informatika Polinema,* vol. 4, no. 3, pp. 224-228,

2018.

[20] L. Cui, S. Huang, F. Wei, C. Tan, C. Duan dan M. Zhou, SuperAgent: A Customer Service Chatbot for E-commerce Websites, Association for Computational Linguistics, 2017.

[21] D. M. Christopher, R. Prabhakar dan S. Hinrich, Introduction to Information Retrieval, New York, USA: Cambridge University Press New York, 2008, pp. 100-123.

[22] J. Y. Chai, V. Horvath, N. Nicolov, M. Stys, N. Kambhatla, W. Zadrozny dan P. Melville, "Natural Language Assistant: A Dialog System for Online Product Recommendation," *AI Magazine,* vol. 23, pp. 63-76, 2002.

[23] M. Brian L. Cairns, P. Rodney D. Nielsen, M. James J. Masanz, P. James H. Martin, P. Martha S. Palmer, P. Wayne H. Ward dan P. Guergana K. Savova, "The MiPACQ Clinical Question Answering System," *Annual Symposium proceedings,* pp. 171-180, 2011.

[24] M. Alessandro, M. Vincenzo dan G. Angelo, "Automated Self-Learning Chatbot Initially Built as a FAQs Database Information Retrieval System: Multi-level and Intelligent Universal Virtual Front-Office Implementing Neural Network," *Informatica,* vol. 42, p. 515–525, 2018.

[25] Wikipedia, "Wikipedia," [Online]. Available: https://en.wikipedia.org/wiki/Tf%E2%80%93idf.

[Diakses 5 Maret 2019].

[26] wikipedia, “Wikipedia,” [Online]. Available: https://en.wikipedia.org/wiki/Tf%E2%80%93idf. [Diakses 05 Maret 2019].

BIODATA PENULIS

Penulis yang memiliki nama lengkap Firman Hidayat ini, lahir di Banyuwangi pada tanggal 07 Februari 1997. Penulis merupakan anak terakhir dari orang tua bernama Hanipan dan Surkanah. Penulis menempuh pendidikan dasar di SD Negeri Kepatihan 2 Banyuwangi pada tahun 2003 hingga 2009. Setelah lulus dari sekolah dasar, penulis melanjutkan pendidikan formal di bangku sekolah menengah pertama di SMP Negeri 1 Banyuwangi dan lulus pada tahun 2012. Pada tahun yang sama, penulis melanjutkan pendidikan formal di SMA Negeri 1 Glagah Banyuwangi dan lulus pada tahun 2015. Pada tahun 2015, penulis melanjutkan pendidikan tinggi di Departemen Sistem Informasi, Fakultas Teknologi Informasi dan Komunikasi, Insitut Teknologi Sepuluh Nopember (ITS) Surabaya, melalui jalur Program Kemitraan dan Mandiri (PKM) tahun 2015.

Selama menjalani pendidikan tinggi di ITS, penulis aktif dalam berbagai kegiatan, baik organisasi maupun kepanitiaan. Penulis pernah menjadi staf Bidang Mentoring Kajian Islam Sistem Informasi (KISI) ITS pada tahun 2016 dan 2017. Pada tahun 2016, penulis juga pernah menjadi Staf Divisi Perkap Information System Expo (ISE! 2016).

Selain aktif dalam organisasi dan kepanitiaan, penulis juga mengikuti beberapa pelatihan diantaranya SAP University Alliance Course yang diadakan Departemen Sistem Informasi. Penulis juga telah mendapatkan sertifikasi IC3 (Internet Core Competency Certification) pada tahun 2017.

Untuk mengetahui informasi lebih lanjut mengenai penelitian ini maupun terkait dengan penulis, dapat menghubungi melalui email projectfirman15@gmail.com.

LAMPIRAN