Institut
Teknologi
Sepuluh Nopember

TUGAS AKHIR - KS141501

INTEGRASI DATA LEMBAGA SERTIFIKASI HALAL DI DUNIA DENGAN DATA HALAL NUTRITION FOOD

INTEGRATION OF HALAL CERTIFICATION BODIES DATA FROM AROUND THE WORLD WITH HALAL NUTRITION FOOD DATA

ALKAUTSAR
05211540000048

Dosen Pembimbing
Nur Aini Rakhmawati, S.Kom., M.Sc., Eng., Ph.D

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

TUGAS AKHIR - KS141501

INTEGRASI DATA LEMBAGA SERTIFIKASI HALAL DI DUNIA DENGAN DATA HALAL NUTRITION FOOD

Alkautsar
05211540000048

Dosen Pembimbing
Nur Aini Rakhmawati, S.Kom., M.Sc., Eng., Ph.D

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

UNDERGRADUATE THESIS - KS141501

INTEGRATION OF HALAL CERTIFICATION BODIES DATA FROM AROUND THE WORLD WITH HALAL NUTRITION FOOD DATA

ALKAUTSAR
05211540000048

Supervisor
Nur Aini Rakhmawati, S.Kom., M.Sc., Eng., Ph.D

INFORMATION SYSTEM DEPARTMENT
Information Technology and Communication Faculty
Sepuluh Nopember Institute of Technology
Surabaya 2019

LEMBAR PENGESAHAN

INTEGRASI DATA LEMBAGA SERTIFIKASI HALAL DI DUNIA DENGAN DATA HALAL NUTRITION FOOD

TUGAS AKHIR
Disusun Sebagai Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Departemen Sistem Informasi
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember

Oleh:

ALKAUTSAR
NRP. 0521 15 4000 0048

Surabaya, Juli 2019

KEPALA
DEPARTEMEN SISTEM INFORMASI

Mahendrawati ER, ST, M.Sc, Ph.D
NIP 19761011 200604 2 001

LEMBAR PERSETUJUAN

INTEGRASI DATA LEMBAGA SERTIFIKASI HALAL DI DUNIA DENGAN DATA HALAL NUTRITION FOOD

Disusun Sebagai Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Departemen Sistem Informasi
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember

Oleh:

ALKAUTSAR
NRP. 0521 15 4000 0048

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

Nur Aini Rakhmawati, S.Kom, M.Sc.Eng, Ph.D **(Pembimbing I)**

Faizal Johan Atletiko, S.Kom, M.T **(Penguji I)**

Irmasari Hafidz, S.Kom, M.Sc **(Penguji II)**

INTEGRASI DATA LEMBAGA SERTIFIKASI HALAL DI DUNIA DENGAN DATA HALAL NUTRITION FOOD

Nama Mahasiswa	**: Alkautsar**
NRP	**: 05211540000048**
Departemen	**: Sistem Informasi FTIK-ITS**
Pembimbing I	**: Nur Aini Rakhmawati, S.Kom., M.Sc. Eng., Ph.D**
Pembimbing II	**: -**

ABSTRAK

Indonesia merupakan salah satu negara dengan mayoritas penduduk muslim terbesar di dunia. Prosentase Muslim Indonesia mencapai 12,7 persen dari populasi dunia. Dari 205 juta penduduk Indonesia, dilaporkan sedikitnya 88,1 persen beragama Islam. Meskipun begitu kita ketahui bahwa negara selain Indonesia juga memiliki banyak penduduk muslim. Oleh karena umat muslim sangat memperhatikan hal mengenai makanan apakah makanan tersebut halal atau tidak sehingga, dibutuhkan sesuatu yang dapat membantu umat muslim mengenai hal tersebut. Saat ini telah ada aplikasi Halal Nutrition Food yang mana pengguna dapat melakukan pengecekan produk halal yang terdapat dalam aplikasi tersebut serta hal-hal lainnya yang disediakan aplikasi Halal Nutrition Food. Terdapat beberapa penelitian yang dilakukan untuk mengembangkan aplikasi ini, salah satunya tugas akhir ini. Tugas akhir ini merupakan bagian dari pengembangan website Halal Nutrition Food dimana pengembangan yang dilakukan merupakan pengumpulan data dari 10 lembaga sertifikasi dari berbagai negara, yang memiliki list produk pada website masing-masing lembaga tersebut. Data yang di dapat akan dihitung similaritasnya dan diintegrasikan dengan data yang sudah ada pada website Halal Nutrition Food.

Tugas akhir ini bermanfaat untuk meningkatkan kepercayaan pengguna terhadap produk-produk yang ada pada website Halal Nutrition Food.

Kata kunci: Halal Nutrition Food, lembaga sertifikasi, integrasi, similarity

INTEGRATION OF HALAL CERTIFICATION BODIES DATA FROM AROUND THE WORLD WITH HALAL NUTRITION FOOD DATA

Name : Alkautsar
NRP : 5211540000048
Department : Information System FTIK-ITS
Supervisor : Nur Aini Rakhmawati, S.Kom., M.Sc. Eng., Ph.D

ABSTRACT

Indonesia is one of the countries with the largest Muslim population in the world. The percentage of Indonesian Muslims reaches 12.7 percent of the world population. Of the 205 million people in Indonesia, it is reported that at least 88.1 percent are Muslim. Even so we know that countries other than Indonesia also have many Muslim populations. Because Muslims are very concerned about the matter of whether food is halal or not, something that can help Muslims in this regard is needed. Currently there is a Halal Nutrition Food application where users can check halal products recorded in the application as well as other things provided by the Halal Nutrition Food application. There are several studies conducted to develop this application, one of them is this final project. This final project is part of the development of the Halal Nutrition Food website where the development carried out is collecting data from 10 certification bodies from various countries, which have a list of products on the websites of each of these institutions. Similarity of the collected data will be calculated and the data will be integrated with data already on the Halal Nutrition Food website. This final project is useful to increase user trust in the products on the Halal Nutrition Food website.

Keywords: Halal Nutrition Food, Certification bodies, Integration, similarity

KATA PENGANTAR

Dengan mengucapkan rasa syukur kepada Tuhan Yang Maha Pengasih dan Maha Penyayang atas izin-Nya penulis dapat menyelesaikan buku yang sederhana ini dengan judul INTEGRASI DATA LEMBAGA SERTIFIKASI HALAL DI DUNIA DENGAN DATA HALAL NUTRITION FOOD

. Dalam penyelesaian Tugas Akhir ini, penulis diiringi oleh pihak-pihak yang selalu memberi dukungan, saran, dan doa sehingga penelitian berlangsung dengan lancar. Secara khusus penulis mengucapkan terima kasih dari lubuk hati terdalam kepada:

1. Tuhan, yang selalu menemani dan membimbing penulis dalam segala aspek kehidupan.
2. Ibu Mahendrawati ER, ST, M.Sc, Ph.D selaku Ketua Departemen Sistem Informasi ITS Surabaya.
3. Nur Aini Rakhmawati, S.Kom., M.Sc., Eng., Ph.D selaku dosen pembimbing yang telah mencurahkan segenap tenaga, waktu dan pikiran dalam penelitian ini, serta memberikan motivasi yang membangun.
4. Bapak Faizal Johan Atletiko, S.Kom, M.T dan Ibu Irmasari Hafidz, S.Kom, M.Sc selaku dosen penguji yang telah memberikan kritik dan saran yang membuat kualitas penelitian ini lebih baik lagi.
5. Segenap dosen dan karyawan Departemen Sistem Informasi.
6. Orang tua penulis, yang tiada hentinya mendoakan dan memberikan dukungan kepada penulis.
7. Teman-teman yang telah membantu mengajari hal-hal baru untuk membantu peneletian ini menjadi berkualitas.
8. Pihak lainnya yang berkontribusi dalam tugas akhir yang belum dapat penulis sebutkan satu per satu.

Penyusunan tugas akhir ini masih jauh dari kata sempurna, untuk itu penulis menerima segala kritik dan saran yang membangun sebagai upaya menjadi lebih baik lagi ke depannya.

Semoga buku tugas akhir ini dapat memberikan manfaat untuk pembaca.

Surabaya, Juli 2019

Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

Halaman ini sengaja dikosongkan

DAFTAR KODE

BAB I
PENDAHULUAN

Bab ini akan menjelaskan tentang pendahuluan pengerjaan tugas akhir yang meliputi latar belakang, rumusan masalah, batasan masalah, tujuan penelitian, dan manfaat yang akan diperoleh dari penelitian tugas akhir ini.

1.1 Latar Belakang

Indonesia merupakan salah satu negara dengan mayoritas penduduk muslim terbesar di dunia. Prosentase Muslim Indonesia mencapai 12,7 persen dari populasi dunia. Dari 205 juta penduduk Indonesia, dilaporkan sedikitnya 88,1 persen beragama Islam[1]. Dalam agama Islam terdapat banyak hal-hal yang mengatur penganutnya, diantaranya yaitu mengenai makanan dan minuman yang dikonsumsi. Setiap makanan dan minuman yang dikonsumsi oleh umat Islam sendiri harus bersifat halal, sehingga banyak Negara yang mayoritas rakyatnya penganut agama Islam membentuk lembaga khusus untuk menangani masalah ini seperti melakukan pengawasan terhadap produk-produk yang tersebar di Negara mereka masing-masing. Indonesia sendiri memilki lembaga khusus yang bernama LPPOM MUI (Lembaga Pengawasan Pangan, Obat-obatan, dan Komestika Majelis Ulama Indonesia) dalam mengawasi produk makanan dan minuman halal untuk masyarakat penganut Agama Islam di Indonesia.

LPPOM MUI ini merupakan lembaga yang dinaungi oleh departemen agama yang memiliki kewenangan untuk mengeluarkan sertifikat halal pada produk-produk yang dikonsumsi oleh rakyat Indonesia. Disamping itu LPPOM MUI memiliki tanggung jawab lain yang berupa menyediakan iformasi produk halal, akan tetapi hal ini sering kali menjadi kontroversi karena banyak beredarnya berita-berita terkait produk-produk yang belum terjamin halal haramnya.

Beredarnya berita-berita seperti itu yang tidak jelas asalusulnya biasanya kita menyebutnya berita HOAX. Di era

perkembangan teknologi seperti sekarang ini masyarakat khususnya di Indonesia sendiri dapat dengan mudah membagikan sebuah berita hanya melihat dari judulnya saja tanpa melihat isinya bahkan sumber berita tersebut asalnya dari mana, baik itu dari WhatApp, facebook maupun media social lainnya[2]. Berita hoax terus meningkat dari tahun 2002 hingga 2016. Dalam kurun waktu tersebut, terdapat 8.617 kasus dengan sekitar 4.600 kasus terjadi di tahun 2016[3]. Karena isu hoax mengenai halal dan haramnya suatu produk pastinya membuat masyarak muslim yang ada di Indonesia resah. Dan apabila hal ini terus di perbincangkan maka akan menyudutkan agama atau kelompok lain. Oleh karena itu dibutuhkannya aplikasi Halal Nutrition food ini untuk mendapatkan informasi yang akurat dari berbagai lembaga yang mengawasi produk halal dari berbagai Negara.

Di sisi lain aplikasi ini ingin dapat digunakan secara meluas misalnya seseorang yang sedang berada Negara yang mayoritas penduduknya non-muslim sehingga sulit mencari produkproduk halal-dari Negara tersebut. Oleh karena itu aplikasi ini akan dikembangkan dimana data-data dari *Halal Certification Bodies* dikumpulkan dari berbagai Institusi yang terdapat dari seluruh negara sehingga data-data produk tersebut memiliki nilai ataupun terpercaya bagi pengguna untuk dimanfaatkan.

Pengembangan data ataupun integrasi data pada *Halal Nutrition Food* web dari berbagai Institusi yang melakukan sertifikasi pastinya membutuhkan data yang terpercaya. Untuk itu dalam penelitian ini terlebih dahulu dilakukan pengecekan pada 43 institusi yang telah diidentifikasi , dan dari 43 institusi tersebut hanya sepuluh institusi yang memiliki list produk dan menggunakan bahasa latin. Hasil data yang telah dikumpulkan nantinya akan di hitung *similarity* nya setelah itu baru dilakukan

integrasi dengan data yang sudah ada pada website *Halal Nutrition Food.*

1.2 Rumusan Masalah

Berdasarkan uraian latar belakang diatas maka terdapat beberapa rumusan masalah antara lain:

1. Bagaimana mengumpulkan data dari setiap Institusi atau lembaga sertifikasi halal
2. Bagaimana cara mengintegrasikan data produk dari lembaga sertifikasi halal dengan data Halal Nutrition Food.

1.3 Batasan Permasalahan

Batasan untuk tugas akhir ini yaitu:

1. Data yang di *crawl* merupakan data yang terdapat pada website yang telah ditentukan sebelumnya.
2. Implimentasi data pada aplikasi *Halal Nutrition Food* hanya pada *platform* website.
3. Website yang akan di *crawl* merupakan website dengan bahasa inggris atau latin
4. Data yang dikumpulkan hanya berupa data produk makanan

1.4 Tujuan

Tujuan dikembangkannya Aplikasi Halal Food Nutrition dan *crawling data institution* ini pada dasarnya untuk membantu seluruh institusi agar data produk dari berbagai lembaga sertifikasi halal dapat terkumpul menjadi satu sehingga pengguna tidak membutuhkan mengunjungi website masingmasing dari berbagai institusi. Berikut tujuan lainnya:

1. Mengumpulkan data produk dari lembaga sertifikasi yang tersebar dari seluruh dunia
2. Mengintegrasikan data dari lembaga sertifikasi halal dengan Halal Nutrition Food

1.5 Manfaat

Manfaat yang akan diperoleh pada tugas akhir sebagai berikut.

Bagi pengguna:

1. Dapat mengetahui produk tersebut tersertifikasi dari lembaga halal dari negara mana.
2. Mengetahui lembaga-lembaga sertifikasi halal dari berbagai negara lain.
3. Meningkatkan kepercayaan pengguna terhadap produk yang di cari pada website.
4. Memudahkan pengguna mencari tahu mengenai produk halal di suatu negara yang minoritas *non*-muslim.

Bagi Penulis :

1. Dapat memiliki data dari berbagai Institusi sertifikasi yang nantinya dapat dilakukan penambahan data dari lembaga yang tidak tersedia datanya pada web.
2. Dapat mengetahui cara melakukan *crawl* data dan Integrasi.

1.6 Relevansi

Tugas akhir ini berkaitan dengan pengumpulan data atau *crawl data* dari website yang nantinya akan diintegrasikan dengan data yang sudah ada pada website *halal nutrition food*. Tugas akhir ini layak dijadikan sebagai tugas akhir karena memecahkan masalah masyarakat mengenai kepercayaan kepada suatu produk halal.

Berikutnya tugas akhir ini berkaitan dengan mata kuliah pemograman web, pengolahan Bahasa alami, system cerdas, dan teknologi web sehingga layak dijadikan sebagai tugas akhir departemen Sistem Informasi.

BAB II
TINJAUAN PUSTAKA

Bab tinjauan pustaka terdiri dari landasan-landasan yang akan digunakan dalam penelitian tugas akhir ini, mencakup penelitian-penelitian sebelumnya, kajian pustaka, dan metode yang digunakan selama pengerjaan.

1.7 Penelitian Sebelumnya

2.1.1. Peningkatan relevansi pencarian produk halal dalam Aplikasi Halal Nutrition Menggunakan Algoritma OKAPI BM25F.

Penelitian yang berjudul "*Peningkatan relevansi pencarian produk halal dalam Aplikasi Halal Nutrition Menggunakan Algoritma OKAPI BM25F"* melakukan pengembangan pada aplikasi Halal Nutrition Food dengan penambahan yaitu fitur pencarian produk halal memanfaatkan algoritma OKAPI BM25F sehingga pencarian produk menjadi relevan dan akurat. Penelitian ini hanya terbatas pada pencarian produk, meskipun begitu aplikasi ini pengguna juga dapat melakukan pencarian tidak hanya berupa produk melainkan dapat berupa zat aditif atau entitas-entitas lain yang terdapat pada data set[4].

Tugas akhir ini melanjutkan pengembangan pencarian ini dengan cara menambahkan data yang telah diakui oleh beberapa institusi dari berbagai negara di dunia.

2.1.2. A Mathematical Solution to String Matching for Big Data Linking.

Penelitian yang dilakukan oleh Kevin McCormack dan Mary Smyth menjelaskan bagaimana *data records* dapat dicocokkan ke seluruh dataset yang besar menggunakan teknik yang disebut *Identity Correlation Approach* (ICA)[5]. Teknik ICA kemudian dibandingkan dengan *string matching exercise. String matching exercise* dan teknik ICA digunakan untuk proyek data besar yang dilakukan oleh CSO[5]. Proyek ini disebut SESADP (Structure of Earnings Survey Administrative

Data Project) menghubungkan dataset Sensus Irlandia 2011 dengan Dataset Sektor Publik. Teknik ICA menyediakan *tool* matematika untuk menghubungkan dataset dan tingkat pencocokan. Penelitian ini membandingkan hasil dari solusi matematika untuk pencocokan string yang disebut Pendekatan Korelasi Identitas (ICA)[5]. Teknik ICA dikembangkan berdasarkan ukuran populasi dan jumlah variabel dalam dataset yang harus dicocokkan satu sama lain. Dari penjelasan diatas di ketahui bahwa penelitian ini akan menyajikan Identity Correlation Approach (ICA) dan aplikasinya serta hasil dari latihan pencocokan data. Kedua, hal ini akan merinci proses dan hasil dari pencocokan string dalam proyek *big data* SESADP.

Tugas akhir ini dapat melihat dasar teknik penelitian diatas mengenai pencocokan data serta perbandingan data dari penelitian yang dilakukan oleh Kevin dan Mary.

2.1.3. Pengembangan Pencarian Produk Terkait Menggunakan Euclidean Distance Dan Cosine Similarity Pada Aplikasi Halal Nutrition Food.

Penelitian dilakukan oleh Azmi Adi Firmansyah merupakan mengembangkan penelitian “Peningkatan relevansi pencarian produk halal dalam Aplikasi Halal Nutrition Menggunakan Algoritma OKAPI BM25F” yang dilakukan oleh Adnan Mauludin Fajriyadi yaitu menampilkan produk halal yang terkait berdasarkan komposisinya[6]. Produk terkait dicari berdasarkan kemiripan komposisi menggunakan euclidean distance dan cosine similarity. Produk yang memiliki banyak kemiripan dengan produk yang telah tersertifikasi, dapat menambah keyakinan pengguna walaupun produk yang sedang dicari belum tersertifikasi[6]. Aplikasi nantinya dapat menampilkan notifikasi kepada pengguna apabila terdapat produk halal yang mirip dengan produk yang sedang dilihat pengguna. Penelitian ini juga menunjukkan bahwa pencarian produk terkait menggunakan cosine similarity memiliki presisi sebesar 84%, sedangkan euclidean distance memiliki presisi sebesar 72%. Penelitian ini juga menguji fitur MoreLikeThis

dari Apache Lucene yang memiliki presisi sebesar 80%, sedikit lebih rendah dibandingkan dengan cosine similarity.

Tugas akhir ini dapat mengambil acuan dari penelitian diatas terkait dengan similarity, dan beberapa hal lain seperti kemiripan komposisi produk dan lainnya.

2.1.4. Integrasi Data dan Visualisasi Graf pada Aplikasi Halal Nutrition Food.

penelitian yang dilakukan oleh Berlian Fajar Yusuf merupakan open data produk makanan beserta informasinya yang berasal dari seluruh dunia[7]. Oleh karena itu, perlu adanya integrasi data antara Open Food Fact dan Halal Nutrition Food. Penelitian ini hanya mengambil data dari bahasa inggris dan perancis. Data tersebut dibersihkan melalui data cleansing yang di dalamnya terdapat pengukuran kesamaan bahan makanan menggunakan Levenshtein dan Jaccard distance untuk memperbaki kesalahan penulisan. Hasil data cleansing diintegrasikan ke Halal Nutrition Food. Untuk mengetahui kelompok makanan apa saja yang ada, data yang sudah dalam bentuk graf produk dan bahan makanan, dikelompokkan menggunakan METIS graph partitioning[7]. Hasil graph partitioning divisualisikan bersama dengan visualisasi graf hubungan produk dan bahan makanan, visualisasi graf produk makanan yang berstatus haram dan visualisasi graf produk makanan yang mengandung MSG

tugas akhir dapat mengacu pada penelitian diatas karena terdapat hubungan seperti integrasi antara open food fact dengan Halal Nutrition Food seperti halnya tugas akhir ini integrasi antara data dari lembaga sertifikasi dengan data Halal Nutrition Food.

1.8 Dasar Teori

2.2.1 Produk Halal

Definisi produk adalah barang dan/atau jasa yang terkait dengan makanan, minuman, obat, kosmetik, produk kimiawi, produk biologi, produk rekayasa genetik, serta barang gunaan

yang dipakai, digunakan, atau dimanfaatkan oleh masyarakat. Jika diartikan kedalam Produk Halal yaitu produk yang telah dinyatakan halal sesuai dengan syariat Islam oleh pihak-pihak yang memiliki wewenang dan memiliki kapabilitas dalam hal tersebut.

2.2.2 Web crawler

Web Crawler adalah program yang menelusuri World Wide Web dengan cara yang metodis, otomatis dan teratur. Istilah lain untuk web crawler adalah ant, automatic indexer, bots, web spiders atau web robots[8]. Web crawler adalah suatu program atau script otomat yang relatif simple, yang dengan metode tertentu melakukan scan atau "crawl" ke semua halamanhalaman Internet untuk membuat index dari data yang dicarinya[9]. Web crawler biasanya berupa bot atau agen perangkat lunak. Secara umum, proses crawling dimulai dengan list URL yang akan dikunjungi, disebut seeds. Kemudian web crawler akan mengunjungi URL tersebut satu per satu. Setiap page URL yang dikunjungi akan diidentifikasi apakah ada hyperlink di dalamnya. Jika ada maka akan ditambahkan ke dalam list URL yang akan dikunjungi. Ini disebut crawl frontier. URL yang didapat dari crawl frontier ajab dikunjungi secara rekursif dengan beberapa kebijakan tertentu[10].

2.2.3 Halal Intitution

Institusi merupakan suatu organisasi yang ada dan pendiriannya atas dasar tujuan yang nantinya akan langsung berhubungan dengan masyarakat. Kebanyakan institusi yang berdisi dengan tujuan untuk memberikan pendidikan pada kalangan umum[11].

Institusi merupakan segala daya tahap struktur yang mekanismenya berdasarkan tatanan sosial serta kerjasama dalam pembentukkan perilaku setiap individu yang terlibat dalam suatu institusi tertentu[11].

Institusi Halal ataupun sering disebut lembaga sertifikasi halal merupaakan sebuah organisasi yang memiliki hak untuk menentukan halal atau tidaknya sebuah produk. Syarat dan ketentuan produk tersebut halal apakah tidak tergantung pada institusi-institusi yang ada tergantung wilayah dan kebijakan pemerintah setempat akan tetapi tetap dalam ajaran Islam sendiri.

Untuk institusi atau lembaga halal yang ada didunia tercatat terdapat sekitar 43 lembaga yang memiliki wewenang melakukan sertifikasi pada produk. Akan tetapi dari 43 website lembaga tersebut hanya ada sekitar sepuluh lembaga yang memiliki daftar produk yang telah tersertifikasi oleh lembaga yang bersangkutan tersebut. Berikut seluruh lembaga sertifikasi yang terdaftar dan teridentifikasi:

No	Negara	nama	URL	Daftar Produk	Kandungan Makanan	Format data	Company	API
1	SINGAPURA	MAJLIS Ugama Islam Singapura (MUIS)	https://www.muis.gov.sg/	ada	ada	pdf	ada	-
2	MALAYSIA	Jabatan Kemajuan Islam Malaysia (JAKIM)	http://www.islam.gov.my/	ada		html	ada	-
3	BRUNEI DARUSSALAM	Bahagian Kawalan Makanan Halal Jabatan Hal Ehwal Syariah	http://www.kheu.gov.bn	-	-	-	-	-

4	JAPAN	Muslim Professional Japan Association (MPJA)	http://mpja.jp/	ada	-	html	ada	-
5	JAPAN	The Japan Moslem Association (JMA)	http://www.whfc-halal.com/	-	-	-	-	-
6	TAIWAN	Taiwan Halal Integrity Development Association (THIDA)	http://www.thida.org/	-	-	-	-	-
7	INDIA	Jamiat Ulama Halal Foundation	http://www.halalcommittee-jum.org/i	ada	-	html	-	-
8	INDIA	Jamiat Ulama I-Hind Halal Trust	http://www.jamiathalaltrust.org	-	-	-	-	-
9	HONGKONG	Asia Pacific Halal Council Co Ltd (APHC)	https://www.zawya.com	-	-	-	-	-
10	THAILAND	The Central Islamic Comitte of Thailand (CICOT)	http://www.cicot.or.th/	-	-	-	-	-
11	PHILLI	Halal Development	http://www.hdiphilippi	-	-	html	ada	

	PINES	Institute of the Phillipines (HDIP)	neshalal.com					
12	VIETNAM	Halal Certification Agency (HCA)	https://halalauthority.org	-	-	-	-	-
13	SRILANKA	Halal Accreditation Council (Guarantee) Limited	http://www.hac.lk/	ada	ada	html	ada	-
14	AUSTRALIA	The Islamic Coordinating Council of Victoria (ICCV)	crash	-	-	-	-	-
15	AUSTRALIA	Supreme Islamic Council of Halal Meat in Australia Inc. (SICHMA)	http://www.sichma.com.au	-	-	-	-	-
16	AUSTRALIA	Australian Halal Development & Accreditation (AHDAA)	http://www.ahdaa.com.au/	-	-	-	-	-
17	AUSTRALIA	Global halal Trade Center	http://www.globalhala	-	-	-	-	-

		Pty Ltd (GHTC Pty.Ltd)	ltrade.com					
18	AUSTRALIA	Western Australian Halal Authority (WAHA)	http://www.halalbooklet.com	-	-	-	-	-
19	AUSTRALIA	Australian Halal Authority & Advisers (AHAA)	https://www.ahaaservices.com	-	-	-	-	-
20	NEW ZEALAND	Asia Pasific Halal Service - New Zealand, Pty 2011 Limited (APHS-NZ-Pty 2011 ltd)	http://www.aphsnz.co.nz/	-	-	-	-	-
21	NEW ZEALAND	Al Kaussar Halal Food Authority	https://www.zawya.com	-	-	-	-	-
22	NEW ZEALAND	The Federation of Islamic Association of New Zealand, Inc (FIANZ)	https://fianz.com/	-	ada	-	-	-
23	BELGIUM	Halal Food Council of Europe (HFCE)	http://www.hfce.eu/	-	-	-	-	-

24	POLAND	The Muslim Religious Union of Poland (MRU)	-	-	-	-	-	-
25	NETHERLAND	Halal Quality Control (HQC),	-	-	-	-	-	-
26	SPAIN	Instituto Halal De Junta Islamica (Halal Institute of Spain)	http://www.institutohalal.com	-	-	-	-	-
27	ITALY	World Halal Authority (WHA)	https://www.wha-halal.org	-	-	-	-	-
28	NETHERLAND	Total Quality Halal Correct Certification (TQHCC)	https://www.salamadvisor.com	-	-	-	-	-
29	GERMANY	HALAL CONTROL	https://www.hqc-germany.com/	-	-	-	-	-
30	ENGLAND	Halal Certification Europe (HCE)	http://www.tmfb.net	-	-	-	-	-
31	ENGLAND	Halal Food Authority (HFA) – UK	https://www.halalfoodauthority.com/					
32	NETHER	Halal Feed and	-	-	-	-	-	-

	LAND	Food Inspection Authority (HFFIA)						
33	SWITZERLAND	Halal Certification Services (HCS)	https://halalcs.org					
34	TURKEY	Eurasia Halal Services Centre	http://www.eurasiahalal.com	ada	ada	html	-	-
35	TURKEY	HAFSA Halal Certification and Food Imp&Exp Ltd	http://www.hafsahalal.com/	-	-	-	-	-
36	UNITED STATES OF AMERICA	Islamic Services of America (ISA)	http://www.isaiowa.org/	-	-	-	-	-
37	UNITED STATES OF AMERICA	Halal Transaction of Omaha	https://www.halaltransactions.org	-	ada	html	-	-
38	UNITED STATES OF AMERICA	The Islamic Food and Nutrition Council of America (IFANCA)	https://www.ifanca.org	ada	ada	html	-	

39	UNITED STATES OF AMERICA	Halal Food Council USA (HFC USA)	https://www.halalfoodcouncilusa.com/	-	-	-	-	-
40	UNITED STATES OF AMERICA	American Halal Foundation (AHF)	https://www.halalfoundation.org/					
41	BRAZIL	Federation of Muslims Associations in Brazil (FAMBRAS)	http://www.fambras.org.br	-	-	-	-	-
42	BRAZIL	Islamic Dissemination Centre for Latin America (CDIAL) Brazil	http://www.cdialhalal.com.br					
43	SOUTH AFRICA	National Independent Halal Trust (NIHT)	http://www.halaal.org.za/	ada	-	html	-	-
44	UAE	Emirates Authority for Standardization and Metrology	http://halal.ae	ada	ada	html	-	-

Tabel 2. 1 Tabel Institusi yang Teridentifikasi

Oleh karena itu penelitian ini dilakukan pengumpulan data dari tujuh lembaga tersebut. Berikut lembaga yang memiliki data produk pada websitenya:

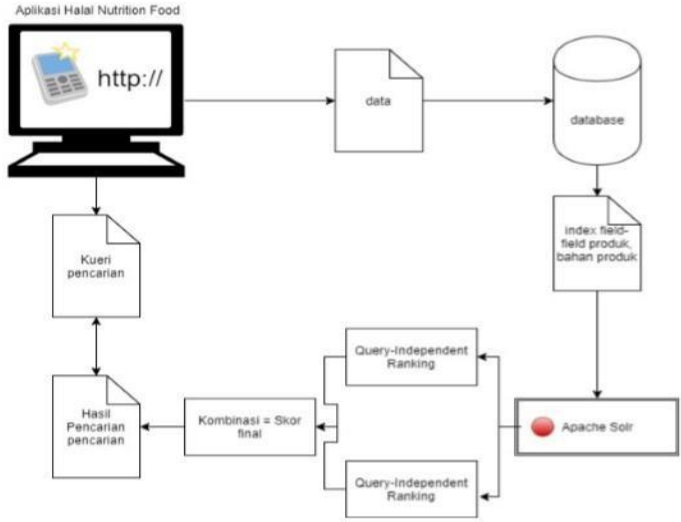
1. Jabatan Kemajuan Islam Malaysia (JAKIM).
2. Emirates Authority for Standardization and Metrology (EASM).
3. Muslim Consumer Group (MCG).
4. Supreme Islamic Council of Halal Meat in Australia Inc (SICHMA).
5. Taiwan Halal Integrity Development Association (THIDA).
6. South Afrika National Halaal Authority (SANHA).
7. The Islamic Food and Nutrition Council of Canada (IFANCC).

Pada penelitian ini data yang didapat dapat bertambah apabila dalam perngerjaan ditemukannya lembaga sertifikasi lain selain dari sepuluh lembaga yang telah diidentifikasi memiliki list produk sertifikasi lembaga tersebut.

2.2.4 Halal Nutrition Food Apps

Adalah aplikasi yang telah dibangun untuk membantu pengguna mencari produk halal maupun lainnya yang berhubungan dengan produk halal. Aplikasi ini berbasis android dan juga berbasis web. Halal Nutrition Food memungkinkan pengguna untuk mencari produk makanan halal melalui mesin pencarian dan mengetahui nutrisi berupa fakta nutrisi, komposisi, produsen, sertifikat, dan sebagainya[12]. Halal Nutrition Food menyediakan nutrisi produk halal dimana tidak ditampilkan di situs LPPOM MUI. Pada penelitian ini dilakukan pengembangan pada data yang ada pada aplikasi Halal Nutrition Food agar lebih. Berikut mengenai arsitektur Halal Nutrition Food:

Arsitektur sistem dari aplikasi Halal Nutrition Food dibuat melalui 2 layer server yang saling terkoneksi. Penjelasannya seperti Gambar 2.



Gambar 2. 1 Arsitektur Halal Food Nutrition

1. Apache Web Server, di dalamnya terdapat dua sistem yang berjalan yakni basis data dan aplikasi web Halal Nutrition Food.

- Sistem basis data yang digunakan adalah MySQL, skema basis datanya akan dijelaskan pada bagian desain basis data. Sistem kerjanya adalah ketika pengguna memasukkan data produk halal maka aplikasi web akan menyimpan data-data produk tersebut ke dalam basis data MySQL.
- Aplikasi android dan web Halal Nutrition Food yang berjalan pada server ini dibangun menggunakan framework PHP Laravel yang di dalamnya terdapat fungsi untuk menghubungkan website dengan server Apache Solr, proses penghubungan keduanya

digunakan ketika pengguna mengirimkan kueri pencarian produk melalui halaman aplikasi web kemudian aplikasi web akan mengirimkan kueri tersebut untuk di proses menggunakan metode kombinasi query-independent dan query dependent ranking di dalam server Apache Solr, selanjutnya server Apache Solr akan mengembalikan daftar hasil pencarian ke aplikasi web.

2. Apache Solr, digunakan untuk memproses pencarian yang dilakukan pengguna dan menyimpan indeks dari dokumen. Dalam Apache Solr akan dilakukan dua proses:

- Proses indexing field-field dari tabel dalam basis data agar pengguna bisa melakukan pencarain berdasarkan data dalam field tersebut.
- Prosesscoring menggunakan metodekombinasi query independent dan query-dependent rankings ehingga hasil pencarian akan diurutkan berdasarkan bobot penialian metode tersebut.

2.2.5 Scrapy

Scrappy atau biasa disebut dengan *Web Scrapping* merupakan sebuah proses ekstraksi data dari sebuah website[13]. Web scrapping merupakan kegiatan yang dilakukan untuk mengambil data tertentu dari sebuah halaman website sebelum pengambilan data biasanya proses akan menganalisis terlebih dahulu dokumen yang ada sebelum memulai pengumpulan atau pengambilan data. Melakukan *scrapping* biasanya menggunakan *web scrapper, bot, web spider, atau web crawler*. Berikut beberapa konsep dan fungsi dasar yang perlu diketahui untuk melakukan scrapy yaitu:

a. Spiders piders adalah kelas yang menentukan bagaimana situs tertentu (atau sekelompok situs) akan di *Scrap,* termasuk cara melakukan *crawl* dan cara mengekstraksi data terstruktur dari halaman mereka.

Dengan demikian spiders adalah tempat di mana Anda menentukan perilaku khusus untuk merayapi dan mengurai halaman untuk situs tertentu (atau, dalam beberapa kasus, sekelompok situs.

Konsep ini digunakan pada tugas akhir ketika membuat *scrapper* untuk sepuluh website lembaga sertifikasi.

b. Item Pipeline

Setelah item di *scrap* oleh spiders, item dikirim ke pipeline yang prosesnya melalui beberapa komponen yang di eksekusi secara berurutan. Pipeline ini biasanya sebuah kode python sederhana yang dijalankan secara berurutan dengan fungsi yang bermacam-macam.

Untuk kasus tugas akhir ini memungkinkan kita melakukan beberapa proses lainnya seperti membersihkan data HTML, memeriksa data yang duplikat, dan menyimpan data yang di *scrap* dalam database.

c. Selector

Merupakan mekanisme mengekstraksi data. Disebut *selector* karena hal ini memilih bagian tertentu dari dokumen html yang telah ditentukan.

Pada tugas akhir ini fungsi ini dapat digunakan pada halaman web yang memiliki list produk saja.

d. Telnet Console

Scrapy dilengkapi dengan konsol telnet *built-in* untuk memeriksa dan mengendalikan proses yang berjalan Scrapy. Konsol telnet hanyalah shell python biasa yang berjalan di dalam proses Scrapy, sehingga kita dapat melakukan apa saja pada *telnet console*.

Apabila dalam tugas akhir nantinya tidak digunakan maka konsol telnet ini dapat di non-aktifkan.

e. Settings

Settings pada scrapy memungkinkan Anda untuk menyesuaikan perilaku semua komponen Scrapy, termasuk inti, ekstensi, pipelines, dan spiders sendiri.

f. Link extractors

Link extractors adalah objek yang tujuannya hanya mengekstrak tautan dari halaman web.

g. Requests and Responses

Biasanya, *Request object* dihasilkan di spider dan melewati sistem sampai mereka mencapai Downloader, yang mengeksekusi permintaan dan mengembalikan objek Respons yang melakukan perjalanan kembali ke spider yang mengeluarkan permintaan.

h. Feed exports

Fungsi ini merupakan fitur yang sangat penting untuk kasus tugas akhir ini dimana ketika menerapkan *scraper* bagaimana ia mampu menyimpan data yang ter-s*crap* dengan benar dan, itu berarti menghasilkan "file ekspor" dengan data yang ter-*scrap* (biasanya disebut "export feed") untuk dikonsumsi oleh sistem lain .

Dalam tugas akhir ini memungkinkan kita menyimpan data yang dibutuhkan dengan format yang macammacam terutama CSV.

i. Item

Tujuan utama melakukan *scrapy* adalah mendapatkan atau mengekstraksi data terstruktur dari sumber yang tidak terstruktur. Pada umumnya spiders mengembalikan data yang telah diekstraksi menjadi *python dicts* akan tetapi, *python dicts* tidak memilki struktur dan mudah terjadi kesalahan ketik maupun data yang tidak konsisten.

Oleh karena itu scrapy menyediakan kelas dengan nama *item* yang menyediakan kamus dengan sintaks yang mudah untuk mendeklarasikan *available fields*.

Pada tugas akhir ini pastinya membutuhkan hal ini karena data yang diperlukan harus terstruktur agar mudah dilakukan similarity nantinya.

j. Stats Collection
Scrapy juga menyediakan fitur stats collection yang dapat memunculkan statistik dalam bentuk *key/value*, dimana nilai sering menjadi penghitung.

Jika sewaktu-waktu pada tugas akhir ini membutuhkan modul ini kita dapat mengimport nya dengan

menggunakan API nya

2.2.6 Jaccard Index

Jaccard Index, juga dikenal sebagai Intersection over Union dan Jaccard similarity coefficient (originally coined coefficient de communauté yang diciptakan oleh Paul Jaccard), adalah statistik yang digunakan untuk membandingkan kemiripan dan keragaman data set sample. Koefisien Jaccard mengukur kesamaan antara kumpulan data sampel yang terbatas dan didefinisikan sebagai ukuran perpotongan dibagi dengan ukuran penyatuan kumpulan sampel. Berikut formula untuk menghitung *Jaccard Index*:

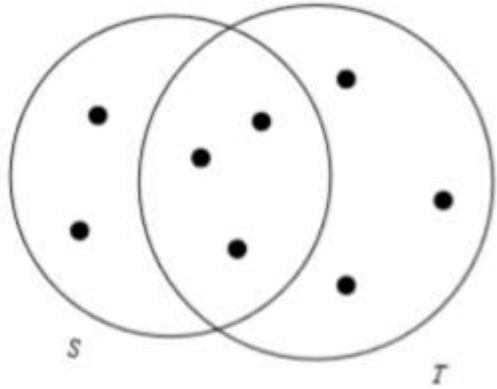
$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}$$

(If A and B are both empty, we define $J(A,B) = 1$.)

$$0 \leq J(A,B) \leq 1.$$

contoh:

pada gambar2.3 kita melihat dua set S dan T. Ada tiga elemen di persimpangan mereka dan total delapan elemen yang muncul di S atau T atau keduanya. Jadi, SIM (S, T) = 3/8.



Gambar 2. 2 Contoh Jaccard Index

2.2.7 Levenshtein Distance

Levenshtein distance adalah suatu model atau metode yang digunakan untuk mengukur nilai kemiripan antara dua buah kata (string). Nilai Levenshtein diperoleh dengan mencari cara termudah untuk mengubah suatu string. Secara umum, operasi mengubah yang diperbolehkan untuk keperluan ini adalah:

- memasukkan karakter ke dalam string.
- menghapus sebuah karakter dari suatu string.
- mengganti karakter string dengan karakter lain.
- Melakukan transposisi pada string.

Untuk menghitungnya, digunakan matriks *(n+1) x (m+1)* di mana *n* adalah panjang string *s1* dan *m* adalah panjang string *s2*. Dua buah string yang akan digunakan sebagai contoh adalah misalnua nama dari senuah produk yaitu COKE dengan COLA . Jika dilihat sekilas, kedua string tersebut memiliki jarak 4.

Berarti untuk mengubah string COKE menjadi COLA diperlukan 4 operasi, yaitu:

- Mensubtitusikan K dengan L (COKE → COLE)
- Mensubtitusikan E dengan A (COLE → COLA)

Dengan menggunakan representasi matriks dapat dilihat pada tabel di bawah.

		C	O	K	E
	0	1	2	3	4
C	1	0	1	2	3
O	2	1	0	1	2
L	3	2	1	1	2
A	4	3	2	1	2

Tabel 2. 2 Representasi Matrix Levenshtein

Elemen terakhir (kanan bawah) adalah elemen yang nilainya menyatakan jarak kedua string yang dibandingkan. Lalu untuk menghitung nilai kemiripan menggunakan rumus:

$$Sim = 1 - (\frac{Dis}{MaxLength})$$

- *Sim* = Similarity/ nilai kemiripan
- *Dis* = jarak Levenshtein
- *MaxLength* = nilai string terpanjang

Jika nilai similarity adalah 1, maka kedua string yang dibandingkan sama. Akan tetapi, jika similarity nya 0, maka kedua string yang dibandingkan tidak sama.

2.2.8 Cosine Similarity.

Metode *cosine* merupakan salah satu metode yang digunakan untuk menghitung nilai similaritas (nilai kemiripan) dari dua buah objek[14]. Pada dasarnya metode *cosine* memodelkan dua buah objek tersebut sebagai *vector of terms*[15]. *Cosine similarity* biasanya digunakan dalam praktink melakukan *text mining*, dan juga dalam mencari informasi yang sejenis[16]. Berikut formula untuk algoritma *cosine similarity*:

$$CosSim(A_i, B_i) = \frac{A_i . B_i}{|A_i||B_i|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \sqrt{\sum_{i=1}^{n} B_i^2}}$$

- A dan B = merupakan objek yang dibandingkan
- $A_i \; dan \; B_i$ = merupakan representatif dari vektor A dan B

2.2.8 Jaro Winkler Similarity.

Metode *jaro* juga merupakan metode pengukuran kesamaan pada dua buah objek. Metode *jaro* lebih baik digunakan pada objek atau string yang pendek karena, *jaro winkler* menilai objek berdasarkan *prefix length* yang menyatakan panjang awalan sebuah karakter merupakan panjang karakter yang dibandingkan hingga ditemukannya ketidak samaan lagi[17]. Karena membandingkan dengan cara melihat panjang string hingga tidak ditemukan lagi kesamaan maka metode *jaro* ini memiliki kemampuan membandingkan data dengan kecepatan yang sangat efektif dan efisien[18]. Dari pembahasan yang telah dijelaskan dasar dari metode *jaro winkler* dapat disimpulkan menjadi poin-poin sebagai berikut:

- Menghitung panjang *string* yang sama.
- Menghitung jumlah karakter yang sama antara dua objek.

Berikut ini merupakan formula yang digunakan *Jaro Winkler Similariy:*

$$JaroSim = \begin{cases} 0 & if\ m = 0 \\ \frac{1}{3}\left(\frac{m}{|s_1|} + \frac{m}{|s_2|} + \frac{m-t}{m}\right) & otherwise \end{cases}$$

- $|s_1|$: Panjang string
- m : Jumlah karakter yang sama
- t : Jumlah transposisi

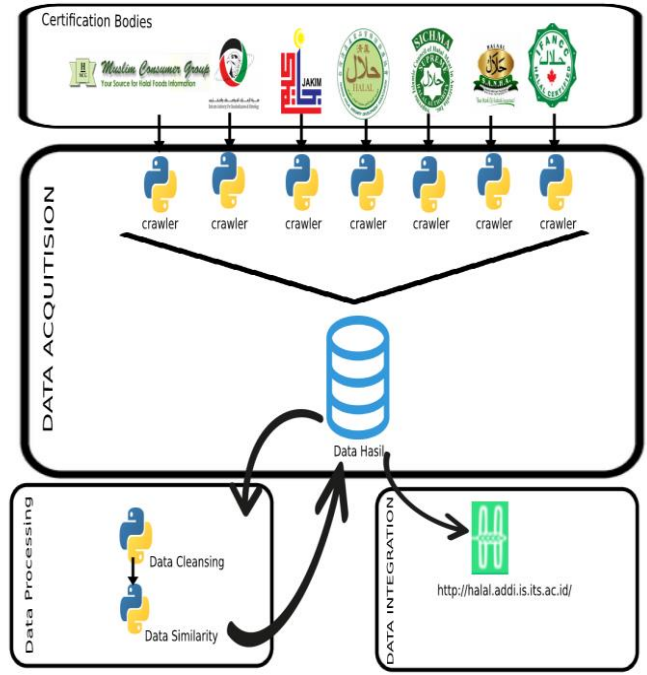
Halaman sengaja dikosongkan

BAB III
METODOLOGI

Pada bagian ini dijelaskan metodologi yang akan digunakan sebagai panduan untuk menyelesaikan penelitian tugas akhir ini.

3.1 Arsitektur Sistem

Berikut arsitektur sistem yang akan digunakan pada tugas akhir ini:



Gambar 3. 1 Arsitektur Sistem

Pada gambar 3.1 terdapat kotak-kotak yang didalamnya memiliki proses-proses tertentu. Berikut penjelasan mengenai proses-proses yang ada pada gambar 3.1.

a. Certification Bodies

Pada bagian ini sumber data yang akan diambil dan dikumpulkan merupakan dari berbagai lembaga sertifikasi halal dari berbagai negara.

b. Data Accuititon

Merupakan bagian dimana cara untuk mengumpulkan data dari sumber yanga ada dan mengumpulkan seluruh data yang didapat menjadi satu.

c. Data Processing

Pada bagian ini data yang telah dikupulkan akan dilakukan beberapa proses agar data dapat dimanfaatkan untuk kebutuhan lainnya seperti, integrasi data dan analisis-analisis lain. Pada bagian ini terdapat dua proses penting yaitu:

i. *Data Cleansing*

Merupakan ekstraksi informasi pada data agar data yang tidak terstruktur menjadi terstruktur sehingga dapat dilakukan proses lainnya seperti pengukuran *similarity, clustering,* dan lain-lain.

ii. *Similarity*

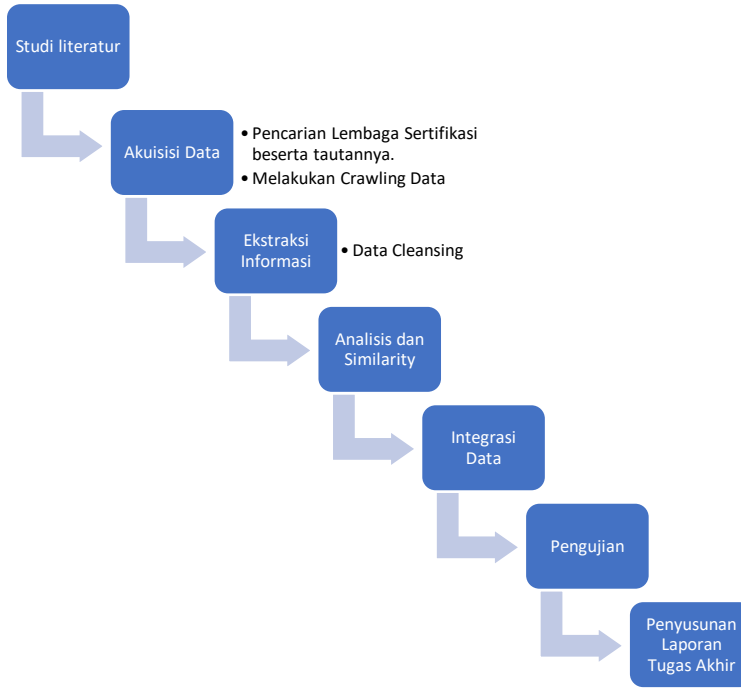
Similarity merupakan proses pengukuran kesamaan terhadap dua karakter yang dibandingkan. *Similarity* dilakukan setelah tahap pembersihan pada data.

d. Data Integration

Data integration merupakan integrasi terhadap suatu data yang telah tersedia. Pada tugas akhir ini data yang telah ada yang dimaksud merupakan data Halal Nuturition Food. Pada bagian ini merupakan tahap akhir setelah data melalui beberapa proses seperti *data cleansing,* dan *similarity.* Hasil dari kedua proses tersebut akan diintegrasikan dengan data yang ada pada *data base Halal Nutrition Food.*

3.2 Tahapan Pengerjaan Tugas Akhir

Pada Sub Bab ini akan dijelaskan mengenai metodologi dalam pelaksanaan tugas akhir. Berikut gambaran memtodologi pelaksanaan tugas akhir.



Gambar 3. 2 Metodologi Tugas Akhir

Gambar 3.2 merupakan langkah metodologi tugas akhir yang akan diimplementasikan dalam penelitian ini.

i. Studi Literatur

Pada tahap ini dilakukannya pengumpulan sebanyak mungkin literature yang berhubungan dengan Tugas akhir ini, khususnya dalam hal *Crawling.* Tujuannya untuk memahami konsep, metode, teori, teknik dan, teknologi yang digunakan sebagai acuan pada Tugas Akhir ini.

ii. Akuisisi data

Setelah mempelajari studi literatur data dapat diperoleh dari tujuh lembaga sertifikasi halal melalui website mereka. *Crawler* yang digunakan untuk mengumpulkan data memanfaatkan fitur yang ada pada bahasa pemograman *python* yaitu SCRAPY. *Scrapy* telah dilengkapi *item-item* yang berguna untuk mengambil data pada website-website yang ada di internet, sekaligus dapat mengatur data sesuai kebutuhan. Data yang diproleh berupa produk yang memiliki atribut sebagai berikut: nama product, manufacture, certification_id, certification bodies, dan negara.

iii. Ekstraksi Informasi

Ekstraksi informasi merupakan pengolahan data yang tidak terstruktur menjadi terstruktur. Pengolahan data menjadi terstruktur dilakukan dengan cara pembersihan data atau *Data Cleansing.* Data yang diolah merupakan data hasil *crawling* dari website lembaga sertifikasi.

iv. Analisis dan Similarity

Similarity merupakan pengukuran kesamaan antara dua karakter. Pada tugas akhir ini *similarity* dilakukan antara data hasil *crawling* dengan data *Halal Nutrition Food*. Similarity dilakukan agar dapat diketahui tingkat kesamaan antar produk, apakah data dapat diintegrasikan atau tidak.

v. Analisis dan Similarity

Setelah data yang ditargetkan telah ter-*crawling* dengan baik dan dikumpulkan menjadi satu selanjutnya data tersebut akan dilakukan *similarity* agar dapat diintegrasikan dengan data *Halal Nutrition Food*. Similarity menggunakan empat metode *similarity* yaitu *Jaccard, levenshtein, cosine,* dan *jaro.* Hasil dari salah satu metode tersebut nantinya akan dipilih untuk integrasi data. Pemilihan metode similarity terbaik dilakukan dengan beberapa cara salah satunya menguji *precission, recall, accuracy,* dan *f-mesure* pada hasil metode similarity tersebut.

vi. Pengujian

Pengujian dilakukan guna untuk memilih metode *similarity* terbaik yang akan digunakan untuk integrasi data. Jenis-jenis pengujian yang akan dilakukan terhadap data hasil *similarity* akan dijelaskan pada bab empat.

vii. Penyusunan Laporan Tugas Akhir

Penyusunan Laporan Tugas Akhir merupakan, pembuatan dokumentasi terhadap tahap-tahap yang dilakukan dalam tugas akhir ini. Laporan tugas akhir berisiskan seluruh tahap pengerjaan tugas akhir mulai dari permasalahan yang diangkat metodologi yang digunakan, perancangan, implementasi, hasil, serta kesimpulan dan saran. Tidak terkecuali pustaka rujukan yang menjadi acuan peneliti. Hasil dari tahap ini berupa buku tugas akhir.

Halaman sengaja dikosongkan

BAB IV
PERANCANGAN

Pada bab ini akan dijelaskan mengenai rancangan penelitian tugas akhir yang merupakan bagaimana proses penelitian mulai dari mengakuisisi data hingga integrasi data.

4.1 Akuisisi Data

Dalam akuisisi data terdapat dua tahap penting yang perlu diperhatikan yaitu:

- Pencarian Lembaga Sertifikasi beserta tautannya.
- Melakukan Crawling Data

4.1.1 Pencarian Lembaga Sertifikasi beserta tautannya.

Pencarian lembaga sertifikasi dan tautannya merupakan tahap awal dalam akuisisi data. Berikut tahap-tahap pencarian lembaga sertifikasi:

a. *Menemukan list Lembaga sertifikasi.*
Pencarian lembaga sertifikasi dapat dilakukan dengan cara mencari pada sebuah website resmi yang menjadi acuan untuk seluruh lebaga sertifikasi lainnya. Dalam tugas akhir ini lembaga yang menjadi acuan untuk mengidentifikasi lembaga sertifikasi lainnya adalah MUI (Majelis Ulama Indonesia).

b. *Pengecekan website Lembaga sertifikasi.*
Setelah mendapat list lembaga sertifikasi selanjutnya melakukan pengecekan apakah lembaga sertifikasi memiliki website ataupun tidak. Apabila lembaga sertifikasi memiliki website lalu selanjutnya yang dilakukan yaitu memastikan website tersebut menggunakan bahasa inggris ataupun bahasa latin.

c. *Validasi website Lembaga sertifikasi.*
Setelah kedua tahap diatas dilakukan selanjutnya dilakukan validasi akhir yaitu memastikan website tersebut resmi dari lembaga sertifikasi yang bersangkutan dan memiliki list data produk halal. Jika

semua sudah valid maka website tersebut dapat digunakan untuk pengumpulan data.

Dari tahap-tahap diatas diharapkan mendapatkan hasil seperti pada tabel 4.1

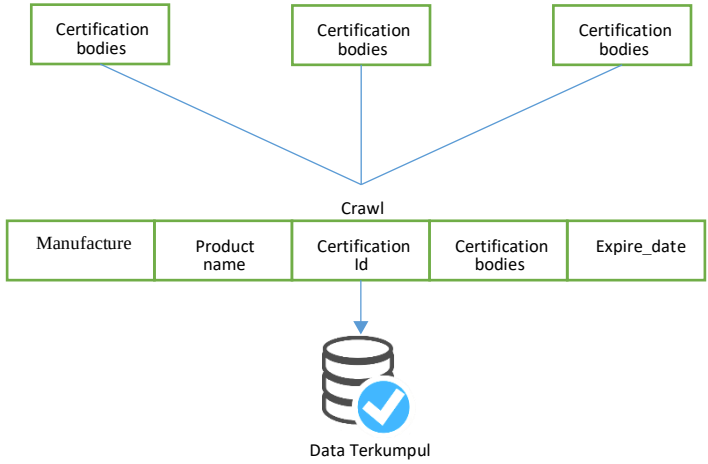
Tabel 4. 1 Tujuh Lembaga yang Menjadi Sasaran Penelitian

NO	Institusi	Negara	Url
1	Jakim	Malaysia	www.halal.gov.my
2	Easm	Emirate Arab	www.esma.gov.ae
3	Ifancc	Canada	www.ifancc.org

Pada tabel 4.1 merupakan contoh hasil yang diharapkan melalui tahap-tahap pencarian lembaga sertifikasi dan tautan yang dimiliki.

4.1.2 Melakukan Crawling Data

Setelah tahap pencarian lembaga sertifikasi dan tautannya selanjutnya merupakan tahap *crawling data* terhadap website lembaga sertifikasi yang telah di lakukan validasi akhir. Berikut arsitektur crawler yang akan digunakan pada tahap ini:



Gambar 4. 1 Arsitektur Crawler

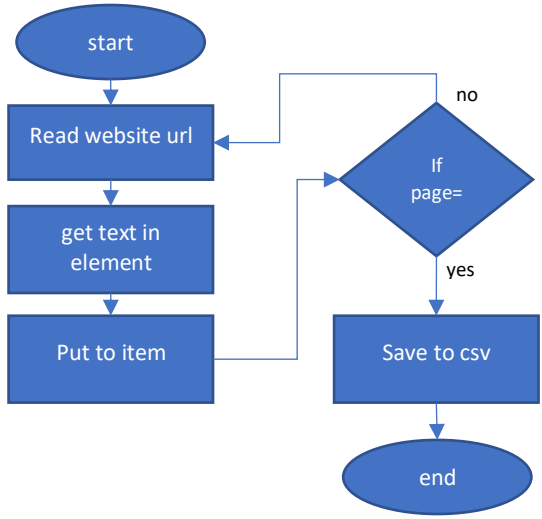
Dari gambar 4.1 dapat dilihat terdapat beberapa atribut yang akan di ambil datanya yaitu manuaktur, nama produk, no sertifikat, organisasi, dan kadaluarsa dari porduk-produk yang tersedia pada website lembaga sertifikasi.

Penjelasan Arsitektur Sistem :

- *Certification bodies.*
  merupakan website institusi-institusi dari berbagai negara yang melakukan sertifikasi dimana dibagian ini akan dilakukan pengumpulan data produk.
- *Crawl.*
  Bagian ini menjelaskan atribut apa saja yang akan diambil pada website lembaga sertifikasi.
- *Data Terkumpul.*
  Setelah data diambil dari setiap lembaga sertifikasi setelah itu akan digabung menjadi satu.

Berikut tahap-tahap crawling data yang dilakukan:

a. *Memebuat kode crawler.*
Pembuatan kode crawler dilakukan terpisah setiap website lembaga sertifikasi dikarenakan perbedaan strutktur *html* maupun pada *pagination* yang dimiliki oleh masing-masing website tersebut. Untuk *flowchart* penulisan kode *crawler* dapat dilihat pada gambar 4.2



Gambar 4. 2 Flowchart Kode Crawler

Dari gambar 4.2 kita lihat pada *flowchart* suatu crawler pertama melakukan pembacaan pada url lalu akan mengambil data dalam elemen *html* selanjutnya meletakkan pada item lalu hal tersebut akan dilakukan seterusnya hingga halaman website tidak lagi tersedia lalu seluruh data dalam item akan di impor ke csv yang nantinya akan di gabung menjadi satu.

b. *Melakukan Crawling Data.*
Melakukan crawling data merupakan implementasi dari kode crawler yang telah dibuat pada tahap sebelumnya. Kode crawler yang telah dibuat akan dijalankan untuk setiap website untuk mengambil data yang ada pada website-website yang telah diidentifikasi. Hasil dari tahap ini merupakan data-data dari setiap website lembaga sertifikasi.

c. *Menggabungkan data menjadi satu.*
Setalah menjalankan program untuk setiap lembaga sertifikasi selanjutnya, hasil dari *crawling data* dikumpulkan menjadi satu untuk dilakukan proses-

proses lainnya. Berikut contoh data yang diharapkan dari tahap *crawling data*:

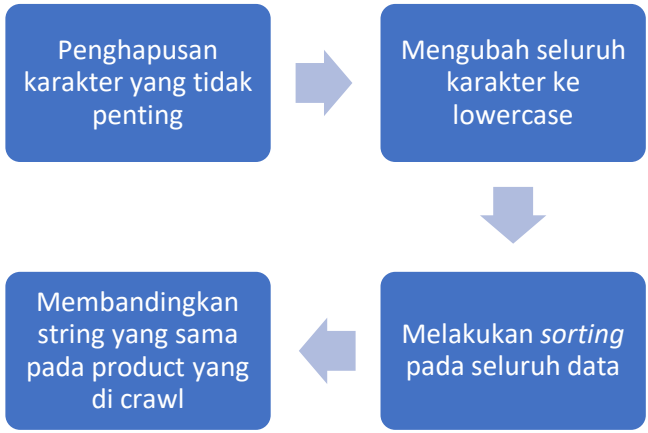
Tabel 4. 2 contoh hasil data crawling

Product_ Name	Manuf acture	Certific ation_n o	Certific atio_bo dies	Cou ntry	Expir e_date
Coca cola	Pt.coke	700982 1	Jauiht	India	2/9/20 15
Beef meat	EC THROSB Y PTY LTD	001217 08ECT0 1C	Sichma	Austr alia	28-04-16
CANNED JUICE DRINK	Rani	Q15-02-000057	Easm	Uni emira te rab	25-05-16

Dari tabel 4.2 merupakan contoh hasil data produk yang diharapkan dari tahap *crawling data.* Meskipun begitu dalam praktik penelitian tugas akhir ini tidak semua website yang dikumpulkan datanya memiliki entitas yang sama sehingga, website-website yang tidak memiliki atribut seperti yang disebutkan di atas maka akan dikosongkan.

4.2 Data Cleansing

Pembersihan data dalam kasus ini pada dasarnya tidak terlalu rumit dikarenakan, atribut data seperti nama produk dan manufaktur telah di crawl dengan baik. Berikut tahap pembersihan data yang dilakukan:



Gambar 4. 3 Alur Proses DATA CLEANSING

Pada gambar 4.3 menjelaskan alur proses pembersihan data atau biasa disebut dengan ekstraksi informasi. Berikut penjelasan tahap-tahap dari gambar 4.3.

4.2.1 Penghapusan karakter yang tidak penting

Karakter tidak penting yang dimaksud merupakan karakter-karakter yang bukan bagian dari atributnya sendiri. Dalam kasus tugas akhir ini data akan mengalami proses penghapusan karakter tidak penting. berikut contoh data yang membutuhkan proses penghapusan karakter tidak penting:

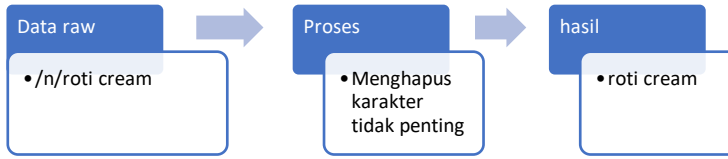
/n/roti cream “/n/t dan ‘’’’’’ice cream/t/n,

Data tersebut akan dibersihkan menjadi:

untuk menghasilkan data seperti diatas dilakukan dengan

roti cream dan ice cream

beberapa cara yaitu pertama ketika proses mengambil data digunakan *item* dan *pipeline* yang dimiliki oleh *scrapy*, yang kedua data tersebut di olah menggunakan *python.* Untuk lebih jelas alur penghapusan karakter tidak penting dapat di lihat pada gambar berikut:



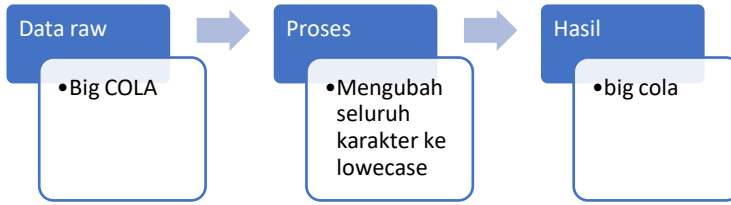
Gambar 4. 4 Proses penghapusan karakter yang tidak penting

Gambar 4.4 merupakan contoh proses penghapusan karakter tidak penting pada data. Dimana hasil yang didapat pada tahap ini diharapkan sama seperti pada gambar 4.4. list karakter tidak penting yang akan dihapus terdapat pada tautan berikut dengan nama file *list karakter yang dihapus.txt*:

https://github.com/kautsareko/Tugas-akhir

4.2.2 Mengubah seluruh karakter ke lower case

Data yang berhasil didapatkan ternyata masih banyak memiliki huruf kapital dan dalam kasus tugas akhir ini hal tersebut dapat berpengaruh ketika melakukan pengukuran *similarity* terhadap data, sehingga perlu dilakukan pengubahan seluruh karakter kedalam *lowercase.* Dalam hal ini memanfaatkan bahasa pemograman python adalah cara yang paling efektif dan efisien karena dapat dilakukan bersamaan dengan similarity data. Berikut gambaran mengubah karakter ke huruf kecil*:*

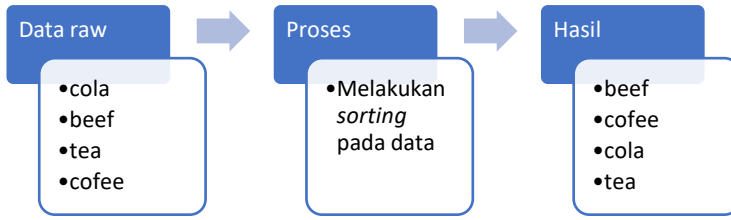


Gambar 4. 5 Proses pengubahan seluruh karakter ke lower case

Dari gambar 4.5 merupakan contoh pengubahan karakter yang masih memiliki huruf kapital menjadi huruf kecil. Diharapkan pada implementasi hasil sesuai dengan gambar 4.5.

4.2.3 Melakukan *Sorting* pada seluruh data

Pengurutan data dilakukan untuk menambah kecepatan eksekusi program ketika melakukan pengukuran *similarity*. *Sorting* dilakukan menggunakan *python* bersamaan dengan melakukan *preprocessing* yang lain seperti mengubah karakter menjadi *lowercase* dan menghapus karakter yang tidak penting. Berikut gambaran sederhana data yang ingin dihasilkan:



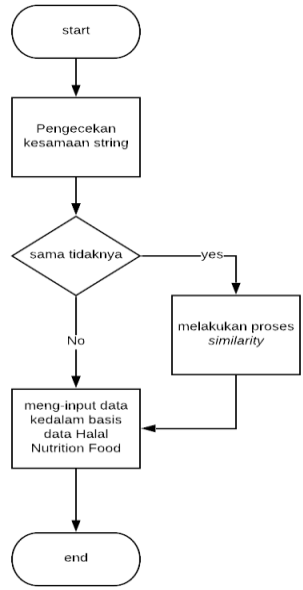
Gambar 4. 6 Proses sorting pada seluruh data

Gambar 4.6 menjelaskan alur *sorting* data produk dimana seluruh nama produk nantinya akan dilakukan pengurutan sesuai abjad seperti yang digambarkan pada gambar 4.6.

4.2.4 Membandingkan String yang sama pada product yang di crawl

Membandingkan string awal karakter hanyalah dilakukan untuk mempercepat waktu pengukuran *similarity*. Dalam hal ini melakukan perbandingan menggunakan bahasa

pemograman *python* dan, *string* yang diukur hanya lima karakter pertama pada nama produk dan manufaktur produk. Apabila terdapat kesamaan maka pengukuran *similarity* dapat dilakukan, jika tidak maka data langsung dimasukkan ke basis data Halal Nutrition Food. Berikut alur pengukuran string:



Gambar 4. 7 Flowchart perbandingan lima string pertama pada produk

Pada gambar 4.7 merupakan *flowchart* untuk penulisan kode program pada perbandingan lima string awal.

4.3 Similarity Data

Similarity dilakukan terhadap nama produk dengan nama produk dan manufaktur dengan manufaktur. *Similarity* dilakukan antara data yang dikumpulkan dari lembaga-lembaga sertifikasi dengan data yang telah ada pada Halal Nutrition Food. Hanya hasil *similarity* dengan nilai lebih besar dari 0.8 yang akan disimpan dan dianalisis, baik itu terhadap nama produk, manufaktur, ataupun keduanya. *Similarity* data dilakukan menggunakan empat metode yaitu *cosine, jaccard*,

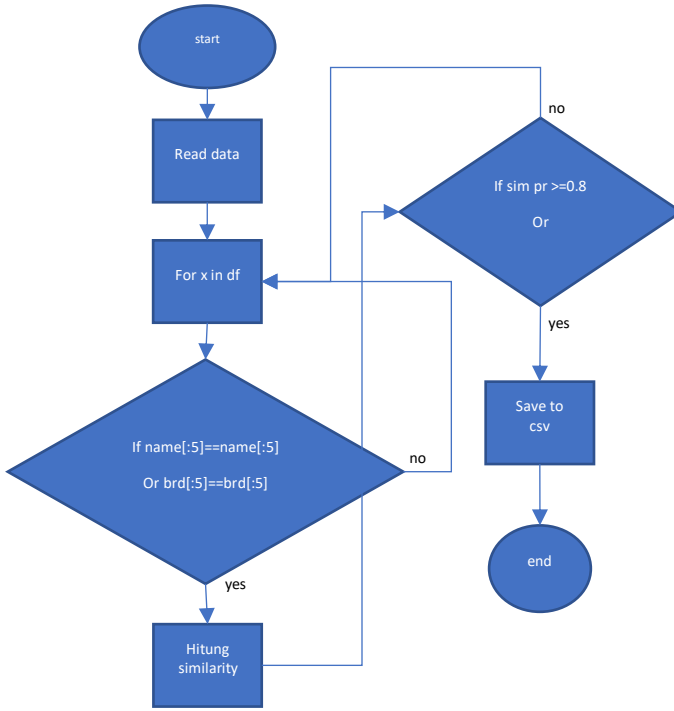
jaro, dan *levenshein.* Satu dari keempat metode tersebut nantinya yang terbaik dalam pengukuran kesamaan akan dipilih untuk melakukan integrasi data.

Similarity data dilakukan dengan mengacu pada sebuah referensi yang berjudul *"String Metrics and Word Similarity applied to Information Retrieval"*. Penelitian ini menjelaskan mengenai implementasi metode pengukuran *similarity* terhadap sebuah informasi yang diterima yang dibandingkan dengan informasi lain yang tersebar di internet[19]. Penelitian ini menjelaskan beberapa metode pengukuran diantaranya yaitu *levenshtein, Q_Grams, Cosine,* dan *Dice coefficient*. Mereka melakukan pengukuran kesamaan pada nama tempat yang mirip dan menurut dari aktualisasi secara manual dari penelitian mereka dari keempat metode tersebut *levenshtein* yang paling unggul dalam membandingkan nama tempat tersebut[19].

Dari penelitian yang dijelaskan pada paragraf sebelumnya dalam tugas akhir ini pada tahap *similarity* nama produk dan manufaktur mengacu pada penelitian tersebut. Yang nantinya *similiarity* terbaik akan dilihat melalui aktualisasi dan beberapa pengujian yang dilakukan.

4.3.1 Membandingkan String yang sama pada product yang di crawl

Dalam melakukan pengukuran kesamaan antar produk perlu dibuat kode program untuk membandingkan kedua objek tersebut karena menggunakan *library* tidak dapat digunakan secara instan butuh kode untuk membaca data dan mengarahkan data tersebut sehingga selanjutnya *library* dapat dipanggil pada kode program yang dibuat. Untuk rancangan penulisan kode program dalam pengukuran *similarity* dapat dilihat pada gambar 4.8.



Gambar 4. 8 Flowchart kode program pengukuran similarity

Gambar 4.8 merupakan alur rencana kode program yang akan dibuat untuk pengukuran *similarity*. Kode program yang akan dibuat nantinya diharapkan berjalan seperti *flowchart* pada gambar 4.8.

Pada tahap *similarity* ini pula dilakukan pembobotan antara nama produk dan manufaktur untuk melihat hubungan antar keduanya apakah terdapat keselarasan ataupun tidak. Pembobotan dilakukan sebanyak enam kali berikut pembobotan yang dilakukan:

- 0.6 utk nama produk & 0.4 utk manufakur

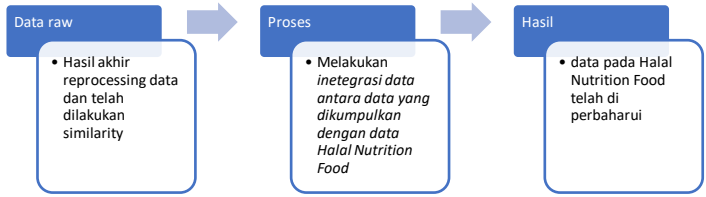
- 0.4 utk nama produk & 0.6 utk manufakur

- 0.7 utk nama produk & 0.3 utk manufakur
- 0.3 utk nama produk & 0.7 utk manufakur
- 0.8 utk nama produk & 0.2 utk manufakur
- 0.2 utk nama produk & 0.8 utk manufakur

Dari pembobotan tersebut nantinya dapat ditarik kesimpulan apakah terdapat hubungan antar keduanya ataupun tidak.

4.4 Integrasi Data

Sebelum memasuki kedalam tahap Integrasi data, data yang dikumpulkan dipastikan telah di bersihkan dan telah dilakukan perhitungan *similarity* agar, produk yang sama dapat dipisah dengan produk yang belum ada pada database Halal Nutrition Food. Produk yang sama nantinya dilakukan pembaharuan terhadap atribut yang belum dimiliki oleh Halal Nutrition Food. Untuk integrasi data nantinya akan diambil metode *similarity* yang paling bagus cara pngukurannya dalam hal akurasi, dan analisis-analisis lainnya. Berikut gambaran sederhana mengenai Integrasi Data:



Gambar 4. 9 Tahap pembaharuan data pada website Halal Nutrition Food

Pada gambar 4.9 menjelaskan alur integrasi data dimana, hasil data yang telah dilakukan *similarity* akan diperbaharui dengan data yang ada pada *data base* Halal Nutrition Food.

Pada bagian integrasi data tugas akhir ini mengacu pada sebuah referensi yang berjudul *"Data Integration: A Theoretical Perspective"* penelitian ini menjelaskan bagaimana menggabungkan data dari sumber yang berbeda yang memiliki skema yang berbeda dengan cara teori perspektif[20].

Dari penjelasan pada paragraf sebelumnya diketahui permasalahan pada penelitan diatas sangat mirip dengan masalah yang diangkat pada tugas akhir ini sehingga dalam melakukan integrasi data halal nutrition food, tugas akhir ini dapat mengacu pada *theorical perspective* tersebut. Berikut penjelasan cara integrasi data:

4.4.1 Impor Data ke Basis Data Halal Nutrition Food

Mengimpor hasil akhir dari data yang dikumpulkan yang telah di olah, kedalam basis data Halal Nutrition Food. Untuk melakukan hal tersebut dilakukan dengan dua tahap pertama mengimpor data baru yang belum ada pada basis data Halal Nutrition Food. Kedua memperbaharui data produk yang memiliki nama produk sama dan mirip berdasarkan hasil *similarity* yang dilakukan dengan data Halal Nutrition Food sehingga, hanya dilakukan pembaharuan atau penambahan atribut baru seperti kode produk, sertifikasi, dan lainnya.

4.5 Pengujian

Pada bagian ini dilakukan beberapa pengujian diantaranya seperti menghitung hasil dari setiap *similarity* dengan memberi batasan treshold lebih besar dari 0.8 dan lebih besar dari 0.9. lalu pengujian yang lain yaitu menghitung akurasi dari setiap *similarity* dengan cara menghitung *precision* dan *recall* dimana, penghitungan dilakukan hanya pada 100 produk sampel dan diberi batasan *treshold* sebesar 0.9. pengujian terakhir melakukan cek pada basis data halal nutrition food bahwa produk telah berhasil diperbaharui. Dari penjelasan diatas maka terdapat tiga pengujian yaitu:

4.4.1 Pengujian Jumlah data yang dihasilkan setiap *similarity.*

Pengujian ini dilakukan dengan cara melihat jumlah produk pada setiap similaritas yang memiliki nilai lebih besar dari 0.8 dan nilai yang lebih besar dari 0.9.

4.4.2 Penghitungan *precision* dan *recall.*

Menghitung akurasi dari setiap *similarity* dengan cara menghitung *precision* dan *recall* dimana, penghitungan dilakukan hanya pada 100 produk sampel dan diberi batasan *treshold* sebesar 0.9. Metode *similarity* yang memiliki *precision, recall* dan akurasi yang tinggi akan digunakan untuk integrasi data.

4.4.3 Pengujian pengecekan *data base* Halal Nutrition Food

Pengujian ini dapat dilakukan dengan banyak cara salah satunya mengecek pada sql yang digunakan pada *website* Halal Nutrition Food. Pengecekan yang dilakukan berupa memeriksa apakah data telah bertambah, selain itu pengecekan juga dapat dilakukan dengan cara menuliskan *query* dengan kondisi yang diinginkan.

BAB V
IMPLEMENTASI

5.1 Akuisisi Data

Akuisisi data yang dimaksudkan pada penelitian ini adalah cara bagaimana mengumpulkan data yang nantinya akan diolah menggunakan komputer.

5.1.1 Mencari Lembaga sertifikasi serta tautan yang dimiliki.

Dalam implementasi pencarian lembaga sertifikasi dalam tugas akhir ini menggunakan list yang ada pada website resmi MUI (Majelis Ulama Indonesia). Pada website tersebut terdapat 43 list lembaga sertifikasi resmi dari berbagai negara. Setelah itu seluruh lembaga tersebut dilakukan pencarian tautan dan melakukan validasi seperti tahap yang dijelaskan pada sub bab 4.1.1 sehingga website lembaga tersebut dapat digunakan untuk pengambilan data. Hasil dari tahap ini dapat dilihat pada tabel 5.1 dibawah ini:

Tabel 5. 1 Hasil Pencarian Lembaga Sertifikasi

Nama Organisasi	**Negara**	**Tautan**
Jabatan Kemajuan Islam Malaysia (JAKIM).	Malaysia	www.halal.gov.my
Emirates Authority for Standardization and Metrology (EASM).	UAE	www.halal.ae
Muslim Consumer Group (MCG).	USA	www.muslimconsumergroup.com
Supreme Islamic Council of Halal	Australia	www.sichma.com.au

Meat in Australia Inc (SICHMA).		
Taiwan Halal Integrity Development Association (THIDA).	Taiwan	www.halalverified.com
South African National Halaal Authority (SANHA).	Pakistan	www.sanha.org.pk
The Islamic Food and Nutrition Council of Canada (IFANCC).	Canada	www.ifancc.org

Tabel 5.1 merupakan hasil dari pencarian lembaga sertifikasi dan tautan website dari lembaga sertifikasi bersangkutan. Dari 43 daftar lembaga sertifikasi hanya tujuh lembaga yang memenuhi syarat seperti yang telah dijelaskan pada sub bab 4.1.1. Sehingga, pada tabel 5.1 merupakan organisasi serta website yang menjadi sasaran untuk pengumpulan data atau *crawling data.*

5.1.2 Melakukan Crawling Data

Seperti yang telah dijelaskan pada sub bab 4.1.2 crawling data memiliki beberapa tahap agar data dapat dikumpulkan berikut penjelasan *crawling data* untuk setiap website lembaga sertifikasi. Untuk tahap crawling data untuk semua website sama yang membedakannya hanya struktur, elemen *html* dan *pagination* yang dimiliki. Berikut tahap crawling data untuk seluruh website:

Gambar 5. 1 Proses crawling data

Pada gambar 5.1 terdapat alur proses *crawling data* yang selanjutnya akan dijelaskan satu persatu setiap lembaga sertifikasi.

a. Jakim

Jakim atau disebut juga dengan Jabatan Kemajuan Islam Malaysia, merupakan lembaga sertifikasi halal dibawah pemerintahan Malaysia yang bermula pada tahun 1965. Berikut gambar penampilan dari halaman *website* Jakim:

Gambar 5. 2 tampilan halaman website Jakim

gambar 5.2 merupakan gambar dari halaman website jakim dimana nantinya data yang ada pada tabel-tabel tersebut akan di ambil datanya.

Pada website Jakim terdapat perbedaan pada struktur *html* dan *pagination* dibandingkan dengan website-website lainnya

Untuk itu terdapat beberapa hal yang akan dilakukan agar data dari website ini dapat di ambil:

1. Menulis kode *crawler*.

i. Penulisan tautan.

Pada bagian ini cukup menuliskan tautan yang akan digunakan untuk pengambilan data lalu dituliskan pada bagian awal kode *crawler*, apabila terdapat tautan yang berbeda-berbeda untuk beberapa produk maka dapat langsung ditambahkan setelah penulisan tautan pertama dimana harus dipisahkan dengan tanda koma. Apabila terdapat sangat banyak tautan seperti *next_page* maka dalam *Scrapy* dapat memanfaatkan fitur *pagination* pada tautan yang di ambil datanya. Berikut penulisan tautan pada kode *crawl* Jakim:

```
1. import scrapy
2. from ..items import JakimItem
3.
4. class crawljakim(scrapy.Spider):
5.     name = 'jakim'
6.     page_number = 2
7.     #allowed_domains = ['http://www.halal.gov.my']
8.     start_urls = ['http://www.halal.gov.my/v4/in-
dex.php?data=ZGlyZWN0b3J5L2luZGV4X2RpcmVjdG9yeTs7Ozs=&negeri=14&cate-
gory=PR&page=1&ty=PR']
```

Kode 5. 1 penulisan tautan Jakim

Pada kode 5.1 tulisan pada baris ke 8 merupakan penulisan tautan website yang akan di lakukan pengambilan data.

ii. Penulisan *selector*.

```
1.  def parse(self, response):
2.          list_of_row = response.xpath("//li[@class='clearfix search-result-data']")
3.          for row in list_of_row:
4.              item = JakimItem()
5.
6.              product_name    = row.xpath(".//span[1]/text()").extract_first()
7.              product_brand   = row.xpath(".//span[3]//i/text()").extract_first()
8.              product_expire  = row.xpath(".//div[3]/text()").extract_first()
9.
10.             item['product_name'] = product_name
11.             item['product_brand'] = product_brand
12.             item['product_expire'] = product_expire
13.
14.             yield item
```

Kode 5. 2 pembuatan selector dan item

Kode 5.2 diatas menjelaskan cara iterasi untuk melakukan pengambilan data yang telah di pilah-pilah sesuai dengan posisinya pada *tag html* dimana hal tersebut dilakukan pada baris ke 3 hingga 8. Serta dibuatkan *item field* pada baris ke 10-12 agar data yang di dapat hasilnya terstruktur dengan baik.

```
1.  import scrapy
2.
3.  class JakimItem(scrapy.Item):
4.      # define the fields for your item here like:
5.      # name = scrapy.Field()
6.      product_name    = scrapy.Field()
7.      product_brand   = scrapy.Field()
8.      product_expire  = scrapy.Field()
9.      #pass
```

Kode 5. 3 penulisan item field pada file item

Kode 5.3 merupakan data file item yang telah ditulis *item field.* Penulisan item field terdapat pada baris ke 6-8 sesuai dengan kode *crawl* yang di buat untuk *website* Jakim.

iii. Ekstraksi menjadi csv.

Untuk melakukan hal tersebut dapat dilakukan penulisan kode pada *file pipeline* yang juga terdapat pada direktori *crawler* yang bersangkutan. Berikut contoh penulisan kode pada file *pipeline:*

```
1.  import csv
2.  # Define your item pipelines here
3.  #
4.  # Don't forget to add your pipeline to the ITEM_PIPELINES setting
5.  # See: https://doc.scrapy.org/en/latest/topics/item-pipeline.html
6.  class JakimPipeline(object):
7.      def open_spider(self, spider):
8.          self.file = open('outputData.csv', 'w')
9.          self.writer = csv.writer(self.file)
10.
11.     def close_spider(self, spider):
12.         self.file.close()
13.
14.     def process_item(self, item, spider):
15.         self.writer.writerow(
16.             [
17.                 item['product_name'],
18.                 item['product_brand'],
19.                 item['product_expire']
20.             ]
21.         )
22.         return item
```

Kode 5. 4 penulisan kode ekstraksi menjadi csv pada file pipelines

Untuk mengekstrak data menjadi csv terlebih dahulu mengimpor csv (pada baris 1) lalu melakukan edit csv seperti kode yang ada diatas (baris 7-15). Pada gambar datas terdapat beberapa fungsi dimana fungsi pertama (baris 7-9) merupakan membuka csv dan menulis sesuatu. Lalu di fungsi kedua (baris 11-12) merupakan *close* csv dimana csv akan di close setelah di edit. Dan fungsi ke tiga (baris 14-16) merupakan fungsi isi dari csv yaitu apa yang akan di tulis atau di ekstrak kedalam csv tersebut.

Pada tahap ini jangan lupa untuk mengaktifkan pipeline (baris 3) pada file *settings.py* pada directori crawler seperti gambar dibawah:

```
1.  # Configure item pipelines
2.  # See https://doc.scrapy.org/en/latest/topics/item-pipeline.html
3.  ITEM_PIPELINES = {
4.      'jakim.pipelines.JakimPipeline': 300,
5.  }
```

Kode 5. 5 menghilangkan komen item pipelines pada file setting

iv. Tahap *pagination.*

pada kasus Jakim data yang terdapat pada table ternyata tidak dalam satu halaman melainkan terdapat pada tautan-tautan berikutnya. Selain itu Jakim memiliki halaman yang banyak untuk data produk jumlahnya hingga ratusan. Sehingga pada tugas akhir ini dilakukan perulangan untuk tautan Jakim karena setelah dilihat tautannya hanya menambah angka di setiap pagenya. Jadi:

```
1.  next_page = 'http://www.halal.gov.my/v4/in-
    dex.php?data=ZGlyZWN0b3J5L2luZGV4X2RpcmVjdG9yeTs7Ozs=&negeri=14&cate-
    gory=PR&page=' + str(crawljakim.page_number) + '&ty=PR'
2.          if crawljakim.page_number <= 245:
3.              crawljakim.page_number += 1
4.              yield response.follow(next_page, callback = self.parse)
```

Kode 5. 6 penulisan kode untuk tautan selanjutnya

dari kode 5.6 *page_number* sebelumnya telah di tulis pada kode 5.3 baris ke 6 sehingga, langsung dapat digunakan. Selain itu disini hanya dilakukan perulangan lalu di lakukan *callback* agar tautan utamanya muncul lalu ditambah hingga page ke 245 (baris 1-4).

2. Melakukan Crawling Data.

Melakukan crawling data berarti menjalankan kode program yang telah ditulis sebelumnya. Pada tugas akhir ini kode program dijalankan secara terpisah karena struktur, elemen *html* untuk setiap website berbeda-beda. Setelah kode program untuk Jakim dijalankan terdapat 4.640 data yang berhasil di *crawling.* Seluruh data tersebut nantinya akan dikumpulkan menjadi satu setelah seluruh kode program selesai dijalankan.

b. Easm

EASM merupakan singkatan *dari Emirates Authority for Standardization and Metrology* yang merupakan sebuah organisasi lembaga sertifikasi yang didirikan oleh Sheikh Mohammed bin Rashid Al Maktoum. Pada tahun 2013 beliau

menjabat sebagai Wakil Presiden dan Perdana Menteri UEA di Dubai yang dikenal sebagai ibu kota Ekonomi Islam di dunia.

Berikut tampilan halaman *website* EASM yang terdapat produk dan propertinya:

List of Halal National Mark certified Products

You can download the list of certifited halal Products from here

Show 10 entries Search:

No	Certificate No	License No	Organization Name	Product Name	Brand Name	No Of Products	Issue Date	Location	Standards
1	Q15-01-000004	H0001-15	Al Ain Food And Beverages P.J.S.C	VARIOUS BREADS AND PASTRIES	GRAND MILLS	14	8/2/2015	UAE	UAE.S GSO 2055- 1
2	E15-01-000498	H0002-15	Global Food Industries LLC	FROZEN FOOD PRODUCTS	AL AMEER	1250	8/2/2015	UAE	UAE.S GSO 2055- 1
3	Q14-12-000208	H0004-15	Al Jazira Poultry Farm LLC.,	CHICKEN EGGS GRADE (A & AA)	GOLDEN EGGS	25	2/9/2015	UAE	UAE.S GSO 2055- 1
4	Q15-03-000068	H0005-15	Al Khaleej Sugar Company	WHITE REFINED CANE SUGAR	AL KHALEEJ SUGAR	9	3/9/2015	UAE	UAE.S GSO 2055- 1

Gambar 5. 3 tampilan halaman website EASM

Gambar 5.3 diatas mirip dengan tabel Jakim akan tetapi pada halaman ini properti dari suatu produk lebih banyak dibandingkan Jakim meskipun begitu, data yang ada sedikit jika dibandingkan dengan Jakim. Pada dasarnya semua kode *crawler* dibuat mirip akan tetapi, yang membedakannya tergantung pada struktur *html* yang dimiliki website dimana list produk tersedia. Perbedaan tersebut dapat berupa dari *selectornya* maupun pada halamannya.

Pada website EASM terdapat perbedaan hanya pada struktur *html* dibandingkan dengan website-website lainnya Untuk itu terdapat beberapa hal yang akan dilakukan agar data dari website ini dapat di ambil:

1. Menulis kode *crawler.*

i. Penulisan tautan

Penulisan tautan untuk kode *crawler* EASM sama halnya dengan Jakim dan begitu pula untuk insititusi lainnya. Untuk EASM sendiri penulisan kode untuk tautannya yaitu:

```
1.  import scrapy
2.  from ..items import EasmItem
3.
4.  class crawleasm(scrapy.Spider):
5.      name = 'easm'
6.      allowed_domains = ['http://halal.ae']
7.      start_urls = ['http://halal.ae/OpenData/ListOfHalalProducts']
```

Kode 5. 7 penulisan tautan pada kdoe crawler EASM

Pada kode 5.7 penulisan tautan terdapat pada baris 6 hingga 7. Tautan tersebut merupakan halaman dimana data produk tersedia

ii. Membuat *selector*.

```
1.  def parse(self, response):
2.          list_of_row = response.xpath("//tbody//tr")
3.          for row in list_of_row:
4.              item = EasmItem()
5.
6.              product_name        = row.xpath(".//td[5]/text()").extract_first()
7.              product_brand       = row.xpath(".//td[6]/text()").extract_first()
8.              product_expire      = row.xpath(".//td[8]/text()").extract_first()
9.              product_certificate = row.xpath(".//td[2]/text()").extract_first()
10.
11.             item ['product_name'] = product_name
12.             item ['product_brand'] = product_brand
13.             item ['product_expire'] = product_expire
14.             item ['product_certificate'] = product_certificate
15.
16.             yield item
```

kode 5. 8 kode pebuatan selector

kode 5.8 merupakan penulisan *selector* menggunakan *xpath* untuk mendapatkan data yang dibutuhkan (baris 6-9). Data produk yang akan di ambil disini terdapat empat item, masing-masing berupa nama produk, brand produk, expire produk, dan sertifikat produk. Ke empat atribut tersebut harus di inisiasi per item agar ketika di ekspor ke CSV datanya menjadi teratur (baris 11-14). Dalam *scrapy* ketika menulis program item seperti diatas maka harus membuat field item sesuai jumlah yang di inisiasi pada kode *crawler* yang dibuat. Berikut penulisan *field* pada file item:

```
1.  import scrapy
2.
3.  class EasmItem(scrapy.Item):
4.      # define the fields for your item here like:
5.      # name = scrapy.Field()
6.      product_name         = scrapy.Field()
7.      product_brand        = scrapy.Field()
8.      product_expire       = scrapy.Field()
9.      product_certificate  = scrapy.Field()
10.     #pass
```

Kode 5. 9 penulisan field item pada file item

Penulisan *field item* pada kode 5.9 berada pada baris 6 hingga baris 9.

iii. Ekstraksi menjadi csv.

Ekstraksi data ke csv dilakukan pada *file pipeline* berikut kode program yang di tulis:

```
1.  import csv
2.
3.  class EasmPipeline(object):
4.      def open_spider(self, spider):
5.          self.file = open('outputData.csv', 'w')
6.          self.writer = csv.writer(self.file)
7.
8.      def close_spider(self, spider):
9.          self.file.close()
10.
11.     def process_item(self, item, spider):
12.         self.writer.writerow(
13.             [
14.                 item['product_name'],
15.                 item['product_brand'],
16.                 item['product_expire'],
17.                 item['product_certificate'],
18.             ]
19.         )
20.         return item
```

Kode 5. 10 penulisan kode untuk ektraksi data ke csv

Kode 5.10 menjelaskan penulisan kode fungsi pertama (baris 4-6) bertujuan untuk membuka spider dan membuat csv, selanjutnya fungsi kedua (baris 8-9) merupakan fungsi untuk mengakhiri atau menutup spider, dan fungsi yang terakhir (baris 11-13) bertujuan untuk penulisan dalam csv. Dalam fungsi ini penulisan item yang diiginkan harus sesuai dengan spider utama yang telah ditulis sebelumnya.

Dan jangan lupa untuk mengaktifkan item pipeline pada file setting.py:

```
1. ITEM_PIPELINES = {
2.     'easm.pipelines.EasmPipeline': 300,
3. }
```

Kode 5. 11 menghilangkan komen pada item pipelines

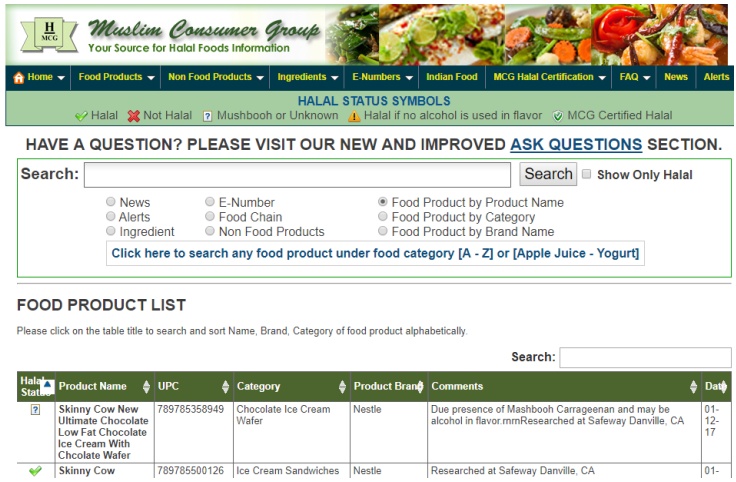
Pengaktifan *item pipelines* pada kode 5.11 terdapat pada bais 1 hingga 3.

2. Melakukan Crawling Data.

Melakukan crawling data berarti menjalankan kode program yang telah ditulis sebelumnya. Pada tugas akhir ini kode program dijalankan secara terpisah karena struktur, elemen *html* untuk setiap website berbeda-beda. Setelah kode program untuk EASM dijalankan terdapat 35 data yang berhasil di *crawling.* Seluruh data tersebut nantinya akan dikumpulkan menjadi satu setelah seluruh kode program selesai dijalankan.

c. MCG

Muslim Consumer Group atau biasa disebut MCG merupakan Lembaga sertifikasi dari negara Amerika. Syed Rasheeduddin Ahmed adalah penggagas MGC sejak tahun 1985. Berikut tampilan halaman website yang dimiliki oleh Lembaga sertifikasi MCG:



Gambar 5. 4 halaman utama website MCG

Gambar 5.4 sekaligus mewakili letak data produk yang akan di kumpulkan. Untuk pembuatan kode *crawler* MCG.

Pada website MCG terdapat perbedaan hanya pada struktur *html* dibandingkan dengan website-website lainnya Untuk itu terdapat beberapa hal yang akan dilakukan agar data dari website ini dapat di ambil:

1. **Menulis kode *crawler*.**

i. Penulisan tautan.

Penulisan tautan untuk MCG sama halnya dengan kode *crawler* untuk institusi lainnya. Untuk MCG sendiri penulisan kode untuk tautan yaitu:

```
1. import scrapy
2. from ..items import UsaItem
3.
4. class crawlusa(scrapy.Spider):
5.     name = 'usa'
6.     # allowed_domains = ['http://www.muslimconsumergroup.com']
7.     start_urls = ['http://www.muslimconsumergroup.com/products_list.html']
```

Kode 5. 12 penulisan tautan pada kode crawler

Penulisan tautan pada kode 5.12 terdapat pada baris ke tujuh.

ii. Membuat *selector*.

```
1.  def parse(self, response):
2.          list_of_row = response.xpath("//tbody//tr")
3.          for row in list_of_row:
4.              item = UsaItem()
5.
6.              product_status      = row.xpath(".//td[1]//img/@alt").extract_first()
7.              product_name        = row.xpath(".//td[2]//strong/text()").extract_first()
8.              product_brand       = row.xpath(".//td[5]/text()").extract_first()
9.              product_UPC         = row.xpath(".//td[3]/text()").extract_first()
10.
11.
12.             item ['product_status'] = product_status
13.             item ['product_name'] = product_name
14.             item ['product_brand'] = product_brand
15.             item ['product_UPC'] = product_UPC
16.
17.
18.             yield item
```

Kode 5. 13 penulisan item produk MCG

Pada kode 5.13 dilakukannya *xpath* untuk mendapatkan data yang dibutuhkan (baris 6-7). Data produk yang akan di ambil disini terdapat empat item, masing-masing berupa nama produk, brand produk, status produk, dan no UPC. Ke empat atribut tersebut harus di inisiasi per *item* agar ketika di ekspor ke csv datanya menjadi teratur (baris 12-15). Dalam *scrapy* ketika menulis program *item* seperti diatas maka harus membuat *field item* sesuai jumlah yang di inisiasi pada kode *crawler* yang dibuat. Berikut penulisan *field* pada file *item*:

```
1.  import scrapy
2.
3.  class UsaItem(scrapy.Item):
4.      # define the fields for your item here like:
5.      # name = scrapy.Field()
6.      product_status      = scrapy.Field()
7.      product_name        = scrapy.Field()
8.      product_brand       = scrapy.Field()
9.      product_UPC         = scrapy.Field()
10.     #pass
```

Kode 5. 14 penulisan field item pada file item

Penulisan *field item* pada kode 5.14 berada pada baris 6 hingga baris 9.

iii. Ekstraksi menjadi csv.

Penulisan kode untuk mengekstrak data ke csv dilakukan pada *pipelines* berikut kode program yang di tulis:

```
1.  import csv
2.
3.  class UsaPipeline(object):
4.      def open_spider(self, spider):
5.          self.file = open('outputData.csv', 'w')
6.          self.writer = csv.writer(self.file)
7.
8.      def close_spider(self, spider):
9.          self.file.close()
10.
11.     def process_item(self, item, spider):
12.         self.writer.writerow(
13.             [
14.                 item['product_status'],
15.                 item['product_name'],
16.                 item['product_brand'],
17.                 item['product_UPC'],
18.             ]
19.         )
20.         return item
```

Kode 5. 15 penulisan kode program untuk ektraksi data ke csv

Kode 5.15 diatas menjelaskan bahwa penulisan fungsi pertama (baris 4-6) bertujuan untuk membuka spider dan membuat csv, selanjutnya fungsi kedua (baris 8-9) merupakan fungsi untuk mengakhiri atau menutup spider, dan fungsi yang terakhir (baris 11-13) bertujuan untuk penulisan dalam csv. Dalam fungsi ini penulisan *item* yang diiginkan harus sesuai dengan spider yang telah ditulis sebelumnya.

Dan jangan lupa untuk mengaktifkan item pipeline pada file setting.py:

```
1.  ITEM_PIPELINES = {
2.      'usa.pipelines.UsaPipeline': 300,
3.  }
```

Kode 5. 16 menghilangkan komen pada item pipelines

Pengaktifan *item pipelines* pada kode 5.16 terdapat pada bais 1 hingga 3.

2. **Melakukan Crawling Data.**

   Melakukan crawling data berarti menjalankan kode program yang telah ditulis sebelumnya. Pada tugas akhir ini kode program dijalankan secara terpisah karena struktur, elemen *html* untuk setiap website berbeda-beda. Setelah kode program untuk MCG dijalankan terdapat 3.200 data yang berhasil di *crawling.* Seluruh data tersebut nantinya akan dikumpulkan menjadi satu setelah seluruh kode program selesai dijalankan.

d. SICHMA

SICHMA atau Supreme Islamic Council of Halal Meat in Australia Inc (SICHMA) adalah organisasi Islam nirlaba yang didedikasikan untuk mempromosikan makanan halal dan lembaga halal. SICHMA bermarkas di Auburn, New South Wales dan Queensland, di Australia. Produk halal bersertifikasi SICHMA ada di hampir setiap negara besar di dunia dan mencakup semua kategori industri makanan. Berikut tampilan halaman website Lembaga sertifikasi tersebut:

Gambar 5. 5 halaman utama website SICHMA

Untuk pembuatan kode *crawler* SICHMA berikut tahap-tahapnya:

Pada website SICHMA terdapat perbedaan hanya pada struktur *html* dibandingkan dengan website-website lainnya Untuk itu terdapat beberapa hal yang akan dilakukan agar data dari website ini dapat di ambil:

1. Menulis kode *crawler*.

i. Penulisan tautan.

Penulisan tautan untuk SICHMA sama halnya dengan penulisan kode *crawler* untuk institusi lainnya. Untuk SICHMA sendiri penulisan kode untuk tautannya berupa:

```
1.  class crawljakim(scrapy.Spider):
2.      name = 'aus'
3.      # allowed_domains = ['http://www.sichma.com.au']
4.      start_urls = ['http://www.sichma.com.au/certified-clients-ndash-uae.html']
```

Kode 5. 17 penulisan kode untuk tautan SICHMA

Penulisan tautan pada kode 5.17 terdapat pada baris ke empat.

ii. Membuat *selector*.

```
1.  def parse(self, response):
2.          list_of_row = response.xpath("//div[@id='284265197932887568']//tr")
3.          for row in list_of_row:
4.              item = AusItem()
5.
6.              product_name          = row.xpath(".//td[5]/text()").extract_first()
7.              product_brand         = row.xpath(".//td[1]/text()").extract_first()
8.              product_expire        = row.xpath(".//td[4]/text()").extract_first()
9.              product_certificate   = row.xpath(".//td[3]/text()").extract_first()
10.
11.
12.             item ['product_name'] = product_name
13.             item ['product_brand'] = product_brand
14.             item ['product_expire'] = product_expire
15.             item ['product_certificate'] = product_certificate
16.
17.
18.             yield item
```

Kode 5. 18 penuliisan kode untuk setiap item yang datanya akan diambil

Kode 5.18 diatas dilakukan *xpath* untuk mendapatkan data yang dibutuhkan (baris 6-9). Data produk yang akan di ambil disini terdapat empat item, masing-masing berupa nama produk, brand produk, *expire_date*, dan sertifikat. Ke empat atribut tersebut harus di inisiasi per item agar ketika di ekspor ke csv datanya menjadi rapi (baris 11-14). Dalam *scrapy* ketika menulis program *item* seperti diatas maka

harus membuat *field item* sesuai jumlah yang di inisiasi pada kode *crawler* yang telah ditulis. Berikut penulisan *field* pada file *item*:

```
1.  import scrapy
2.
3.  class AusItem(scrapy.Item):
4.      # define the fields for your item here like:
5.      # name = scrapy.Field()
6.      product_name         = scrapy.Field()
7.      product_brand        = scrapy.Field()
8.      product_expire       = scrapy.Field()
9.      product_certificate  = scrapy.Field()
10.     #pass
```

Kode 5. 19 penulisan field item pada file item

Penulisan *field item* pada kode 5.19 berada pada baris 6 hingga baris 9.

iii. Ekstraksi menjadi csv.

Penulisan kode program untuk ekstraksi data ke csv dilakukan pada *pipelines* berikut kode program yang di tulis:

```
1.  import csv
2.
3.  class AusPipeline(object):
4.      def open_spider(self, spider):
5.          self.file = open('outputData.csv', 'w')
6.          self.writer = csv.writer(self.file)
7.
8.      def close_spider(self, spider):
9.          self.file.close()
10.
11.     def process_item(self, item, spider):
12.         self.writer.writerow(
13.             [
14.                 item['product_name'],
15.                 item['product_brand'],
16.                 item['product_expire'],
17.                 item['product_certificate']
18.             ]
19.         )
20.         return item
```

Kode 5. 20 penulisan kode program untuk ekstraksi data SICHMA ke csv

Kode 5.20 menjelaskan bahwa penulisan fungsi pertama (baris 4-6) bertujuan untuk membuka spider dan membuat csv, selanjutnya fungsi kedua (baris 8-9) merupakan fungsi untuk mengakhiri atau menutup spider, dan fungsi yang terakhir (baris 11-13) bertujuan untuk penulisan dalam csv. Dalam fungsi ini penulisan item yang diiginkan harus sesuai dengan spider utama yang telah ditulis sebelumnya.

Dan jangan lupa untuk mengaktifkan item pipeline pada file setting.py:

```
1.  ITEM_PIPELINES = {
2.      'usa.pipelines.AusPipeline': 300,
3.  }
```

Kode 5. 21 menghilangkan komen item pipelines

Pengaktifan *item pipelines* pada kode 5.21 terdapat pada bais 1 hingga 3.

2. Melakukan Crawling Data.

Melakukan crawling data berarti menjalankan kode program yang telah ditulis sebelumnya. Pada tugas akhir ini kode program dijalankan secara terpisah karena struktur, elemen *html* untuk setiap website berbeda-beda. Setelah kode program untuk SICHMA dijalankan terdapat 40 data yang berhasil di *crawling.* Seluruh data tersebut nantinya akan dikumpulkan menjadi satu setelah seluruh kode program selesai dijalankan.

e. THIDA

Taiwan Halal Integrity Development Association (THIDA) merupakan lembaga yang melayani segala hal mengenai sertifikasi produk halal. THIDA didirikan karena lima alasan yaitu:

1. untuk memastikan 'Halalness' untuk semua konsumen Muslim di manapun di dunia;
2. untuk melindungi integritas halal dari lembaga sertifikasi lokal dari kesalahan atau kesalahan

penanganan oleh beberapa orang yang tidak bertanggung jawab

3. untuk menghindari konsekuensi dari kesalahan seperti itu yang dapat memengaruhi kepentingan perusahaan lain yang taat hukum
4. untuk memungkinkan pembagian sumber daya yang terbatas dari masing-masing Masjid di bidang Syariah dan teknis.

Berikut tampilan data produk yang terdapat pada halaman *website* THIDA:

No.	Product	Company Name Brand Name	Certification Body	Country	Expiry Date
1	**Milk Powder & Ghee**	Shyam Dairy Products *Shyam*	JUIHHT	India	20 Oct 2019
2	**Pasta**	Seville products LLC(Br)-Pasta. *Alfapasta, Allegro*	JUIHHT	United Arab Emirates	23 Dec 2019
3	**Wheat Flour,Samolina**	Emirates Grain Products Co. L.L.C. *Al-Awal,Hayat,Albaker*	JUIHHT	United Arab Emirates	19 Oct 2019
4	**NBDW HIGH OLEIC SUNFLOWER OIL**	CARGILL PALM PRODUCTS SDN BHD	JAKIM	Malaysia	15 Aug 2019
5	**NBDW SUNFLOWER OIL**	CARGILL PALM PRODUCTS SDN BHD	JAKIM	Malaysia	15 Aug 2019

Gambar 5. 6 halaman website THIDA

Pada gambar 5.6 Merupakan gambar halaman website THIDA serta halaman dimana daftar produk halal tersedia.

Pada website THIDA terdapat perbedaan pada struktur *html* dan *pagination* dibandingkan dengan website-website lainnya Untuk itu terdapat beberapa hal yang akan dilakukan agar data dari website ini dapat di ambil:

1. Menulis kode *crawler*.

i. Penulisan tautan.

Penulisan tautan untuk THIDA sama halnya dengan penulisan kode *crawler* untuk institusi lainnya. Untuk THIDA sendiri penulisan kode untuk tautan berupa:

```
1.  import scrapy
2.  from ..items import ThidaItem
3.
4.  class crawlthida(scrapy.Spider):
5.      name = 'thida'
6.      page_number = 2
7.      #allowed_domains = ['ttps://www.halalverified.com']
8.      start_urls = ['https://www.halalverified.com/Product/Check?Query=product&First-
    Load=&CategoryID=&SubCategoryID=&Page=1&MatchingOptionID=0&SearchFieldID=2']
```

Kode 5. 22 penulisan tautan pada kode crawler THIDA

Penulisan tautan pada kode 5.22 terdapat pada baris ke enam.

ii. Membuat *selector*.

```
1.  def parse(self, response):
2.          #self.log ("i visited: "+ response.url)
3.          # item = ThidaItem()
4.
5.          list_of_row = response.xpath("//tbody[@id='tbodyLoadedData']//tr")
6.          #list_of_row = response.css("#tbodyLoadedData tr")
7.          for row in list_of_row:
8.              # print(row.extract())
9.              item = ThidaItem()
10.
11.             # product_name = row.xpath("//td[2]//a/text()").extract_first()
12.             product_name = row.xpath(".//td[2]//a/@title").extract_first()
13.             product_brand = row.xpath(".//td[3]//span/@title").extract_first()
14.             product_expire = row.xpath(".//td[6]/text()").extract_first()
15.             product_certificationbodies = row.xpath(".//td[4]/text()").ex-
    tract_first()
16.             product_country = row.xpath(".//td[5]/text()").extract_first()
17.
18.
19.             item['product_name'] = product_name
20.             item['product_brand'] = product_brand
21.             item['product_expire'] = product_expire
22.             item['product_certificationbodies'] = product_certificationbodies
23.             item['product_country'] = product_country
```

Kode 5. 23 penulisan kode untuk setiap item pada kode crawler THIDA

Kode 5.23 diatas dilakukan *xpath* untuk mendapatkan data yang dibutuhkan (baris 11-16). Data produk yang akan di ambil ada sekitar enam item. Keenam atribut tersebut harus di inisiasi setiap *item* agar ketika di ekspor ke csv datanya menjadi rapi (baris 11-23). Dalam *scrapy* ketika menulis *item* seperti diatas maka harus membuat *field item* sesuai jumlah yang di inisiasi pada kode *crawler* yang telah ditulis. Berikut penulisan *field* pada file *item*:

```
1.  import scrapy
2.
3.  class ThidaItem(scrapy.Item):
4.      # define the fields for your item here like:
5.      # name = scrapy.Field()
6.      product_name                = scrapy.Field()
7.      product_brand               = scrapy.Field()
8.      product_expire              = scrapy.Field()
9.      product_certificationbodies = scrapy.Field()
10.     product_country             = scrapy.Field()
11.     #pass
```

Kode 5. 24 penulisan field item pada file item THIDA

Penulisan *field item* pada kode 5.24 berada pada baris 6 hingga baris 10.

iii. Ekstraksi menjadi csv.

Penulisan kode program untuk ekstraksi data ke csv dilakukan pada *pipeline*s berikut kode program yang di tulis:

```
1.  import csv
2.
3.  class ThidaPipeline(object):
4.      def open_spider(self, spider):
5.          self.file = open('outputData.csv', 'w')
6.          self.writer = csv.writer(self.file)
7.
8.      def close_spider(self, spider):
9.          self.file.close()
10.
11.     def process_item(self, item, spider):
12.         self.writer.writerow(
13.             [
14.                 item['product_name'],
15.                 item['product_brand'],
16.                 item['product_expire'],
17.                 item['product_certificationbodies'],
18.                 item['product_country']
19.             ]
20.         )
21.         return item
```

Kode 5. 25 penulisan kode program untuk ektraksi data THIDA ke csv

Kode 5.25 diatas menjelaskan bahwa penulisan fungsi pertama (baris 4-6) bertujuan untuk membuka spider dan membuat csv, selanjutnya fungsi kedua (baris 8-9) merupakan fungsi untuk mengakhiri atau menutup spider, dan fungsi yang terakhir (baris 14-18) bertujuan untuk

penulisan dalam csv. Dalam fungsi ini penulisan item yang diiginkan harus sesuai dengan spider utama yang telah ditulis sebelumnya.

Dan jangan lupa untuk mengaktifkan *item pipeline* pada file *setting.py*:

```
1. ITEM_PIPELINES = {
2.     'usa.pipelines.ThidaPipeline': 300,
3. }
```

Kode 5. 26 menghilangkan komen pada item pipelines

Pengaktifan *item pipelines* pada kode 5.26 terdapat pada bais 1 hingga 3.

2. Melakukan Crawling Data.

Melakukan crawling data berarti menjalankan kode program yang telah ditulis sebelumnya. Pada tugas akhir ini kode program dijalankan secara terpisah karena struktur, elemen *html* untuk setiap website berbeda-beda. Setelah kode program untuk THIDA dijalankan terdapat 100 data yang berhasil di *crawling.* Seluruh data tersebut nantinya akan dikumpulkan menjadi satu setelah seluruh kode program selesai dijalankan.

f. SANHA

South Afrika National Halaal Authority atau SANHA Pakistan merupakan organisasi sertifikasi halal yang bertujuan untuk menyediakan layanan profesional, kompeten, dan otoritatif untuk industri dan umum. SANHA memiliki misi yaitu untuk memberikan sertifikasi Halal yang kredibel dan layanan pemantauan sehingga hak-hak Muslim untuk memiliki akses dan mengkonsumsi makanan Halal dilindungi dan dipromosikan.

Berikut tampilan data produk yang terdapat pada halaman website SANHA:

Gambar 5. 7 halaman utama website SANHA

Gambar 5.7 Merupakan halaman utama dari website SANHA yang menjadi salah satu sasaran untuk pengambilan data pada tugas akhir ini.

Pada website SANHA terdapat perbedaan hanya pada struktur *html* dibandingkan dengan website-website lainnya Untuk itu terdapat beberapa hal yang akan dilakukan agar data dari website ini dapat di ambil:

1. Menulis kode *crawler.*

i. Penulisan tautan.

Penulisan tautan untuk SANHA sama halnya dengan penulisan tautan pada kode *crawler* untuk institusi lainnya. Untuk SANHA sendiri penulisan kode untuk tautan berupa:

```
1.  import scrapy
2.  from ..items import PakistanItem
3.
4.  class crawlpakistan(scrapy.Spider):
5.      name = 'pakistan'
6.      #allowed_domains = ['http://www.sanha.org.pk']
7.      start_urls = ['http://www.sanha.org.pk/certified-products/']
```

Kode 5. 27 penulisan tautan pada kode crawler SANHA

Penulisan tautan pada kode 5.27 terdapat pada baris ke tujuh.

ii. Membuat *selector*.

```
1.  def parse(self, response):
2.          list_of_row = response.xpath("//tbody//tr")
3.          for row in list_of_row:
4.              item = PakistanItem()
5.
6.              product_name        = row.xpath(".//td[2]/text()").extract_first()
7.              product_brand       = row.xpath(".//td[3]/text()").extract_first()
8.              product_manufacture = row.xpath(".//td[5]/text()").extract_first()
9.
10.             item ['product_name'] = product_name
11.             item ['product_brand'] = product_brand
12.             item ['product_manufacture'] = product_manufacture
13.
14.
15.             yield item
```

Kode 5. 28 penulisan item yang akan diambil datanya pada website SANHA

Dari kode 5.28 dilakukannya *xpath* untuk mendapatkan data yang dibutuhkan (baris 6-8). Data produk yang akan di ambil ada sekitar tiga *item*. Ketiga atribut tersebut harus di inisiasi per *item* agar ketika di ekspor ke csv datanya menjadi rapi (baris 10-12). Dalam *scrapy* ketika menulis *item* seperti diatas maka harus membuat *field item* sesuai jumlah yang di inisiasi pada kode crawler yang dibuat. Berikut penulisan *field* pada file *item*:

```
1.  import scrapy
2.
3.  class PakistanItem(scrapy.Item):
4.      # define the fields for your item here like:
5.      # name = scrapy.Field()
6.      product_name          = scrapy.Field()
7.      product_brand         = scrapy.Field()
8.      product_manufacture   = scrapy.Field()
9.      #pass
```

Kode 5. 29 penulisan field item pada file item

Penulisan *field item* pada kode 5.29 berada pada baris 6 hingga baris 8.

iii. Ekstraksi menjadi csv.

Penulisan kedalam csv dilakukan pada *pipelines* berikut kode program yang di tulis:

```
1.  import csv
2.
3.  class PakistanPipeline(object):
4.      def open_spider(self, spider):
5.          self.file = open('outputData.csv', 'w')
6.          self.writer = csv.writer(self.file)
7.
8.      def close_spider(self, spider):
9.          self.file.close()
10.
11.     def process_item(self, item, spider):
12.         self.writer.writerow(
13.             [
14.                 item['product_name'],
15.                 item['product_brand'],
16.                 item['product_manufacture']
17.             ]
18.         )
19.         return item
```

Kode 5. 30 penulisan kode untuk ekstraksi data SANHA ke csv

Kode 5.30 menjelaskan bahwa penulisan fungsi pertama (baris 4-6) bertujuan untuk membuka spider dan membuat csv, selanjutnya fungsi kedua (baris 8-9) merupakan fungsi untuk mengakhiri atau menutup spider, dan fungsi yang terakhir (baris 11-13) bertujuan untuk penulisan dalam csv. Dalam fungsi ini penulisan item yang diiginkan harus sesuai dengan spider utama yang telah ditulis sebelumnya.

Dan jangan lupa untuk mengaktifkan item pipeline pada file setting.py:

```
1.  ITEM_PIPELINES = {
2.      'usa.pipelines.PakistanPipeline': 300,
3.  }
```

Kode 5. 31 penghapusan komen pada item pipelines SANHA

Pengaktifan *item pipelines* pada kode 5.31 terdapat pada bais 1 hingga 3.

2. Melakukan Crawling Data.

Melakukan crawling data berarti menjalankan kode program yang telah ditulis sebelumnya. Pada tugas akhir ini kode program dijalankan secara terpisah karena struktur, elemen *html* untuk setiap website berbeda-beda. Setelah kode program untuk SANHA dijalankan terdapat 2.085 data

yang berhasil di *crawling.* Seluruh data tersebut nantinya akan dikumpulkan menjadi satu setelah seluruh kode program selesai dijalankan.

g. IFANCC

The Islamic Food and Nutrition Council of Canada (IFANCC). Adalah organisasi sertifikasi Halal terkemuka di Kanada. Organisasi ini menawarkan layanan pengawasan dan sertifikasi Halal di Kanada. Kanada memiliki populasi umat Muslim 1 juta. Dengan demikian, ada peluang ekonomi yang luar biasa bagi perusahaan Kanada untuk memenuhi kebutuhan umat Islam akan produk makanan halal. IFANCC merupakan otoritas jasa pengklaim halal pada suatu produk sehingga consumer lebih yakin dalam mengkonsumsi produk-produk yang telah tersertifikasi. IFANCC telah mengembangkan prosedur terdokumentasi untuk memproduksi produk halal. Prosedur ini konsisten dengan berbagai standar jaminan kualitas dan mudah diimplementasikan. Berikut tampilan halaman website IFANCC:

Gambar 5. 8 Halaman utama website IFANCC

Gambar 5.8 Merupakan halaman utama dari website IFANCC yang merupakan lembaga terakhir yang datanya akan diambil.

Pada website IFANCC terdapat perbedaan pada struktur *html* dan *pagination* dibandingkan dengan website-website lainnya

Untuk itu terdapat beberapa hal yang akan dilakukan agar data dari website ini dapat di ambil:

1. Menulis kode *crawler*.

i. Penulisan tautan.

Penulisan tautan untuk IFANCC sama halnya dengan penulisan tautan pada kode *crawler* untuk institusi lainnya. Akan tetapi karena IFANCC memiliki data produk yang berada pada halaman yang berbeda-beda sesuai dengan kategori untuk itu caranya hanya di tambah seluruh tautan yang dibutuhkan seperti pada gambar dibawah ini:

```
1.  import scrapy
2.  from ..items import IfanccItem
3.  from scrapy.spiders import CrawlSpider, Rule
4.  from scrapy.linkextractors import LinkExtractor
5.  from lxml import html
6.
7.  class crawlifancc(scrapy.Spider):
8.      name = 'ifancc'
9.      allowed_domains = ['http://ifancc.org']
10.     start_urls = ['http://ifancc.org/directories-categories/bakery-items/',
11.     # 'http//ifancc.org/directories-categories/beverages-beverage-concentrated/',
12.     'http://ifancc.org/directories-categories/cooking-oil/',
13.     'http://ifancc.org/directories-categories/dairy-products/',
14.     'http://ifancc.org/directories-categories/food-products/',
15.     'http://ifancc.org/directories-categories/meat-products/',
16.     'http://ifancc.org/directories-categories/nutritional-supplements/',]
```

Kode 5. 32 penulisan tautan pada kode crawler IFANCC

Penulisan tautan pada kode 5.32 terdapat pada baris ke 9 hingga baris ke 16.

ii. Membuat *selector*.

```
1.  def parse(self, response):
2.          list_of_row = response.xpath("//tbody//tr")
3.          for row in list_of_row:
4.              item = IfanccItem()
5.
6.              product_name        = row.xpath(".//td[3]/text()").extract_first()
7.              product_brand       = row.xpath(".//td[2]/text()").extract_first()
8.              product_certificate = row.xpath(".//td[4]/text()").extract_first()
9.              product_country     = row.xpath(".//td[5]/text()").extract_first()
10.
11.             item ['product_name'] = product_name
12.             item ['product_brand'] = product_brand
13.             item ['product_certificate'] = product_certificate
14.             item ['product_country'] = product_country
15.
16.             yield item
```

Kode 5. 33 penulisan item data yang akan diambil pada website IFANCC

Dari kode 5.33 diatas dilakukannya *xpath* untuk mendapatkan data yang dibutuhkan (baris 6-9). Data produk yang akan di ambil ada sekitar empat item. Ke empat atribut tersebut harus di inisiasi per *item* agar ketika di ekspor ke csv datanya menjadi rapi (baris 11-14). Dalam *scrapy* ketika menulis *item* seperti diatas maka harus membuat *field item* sesuai jumlah yang di inisiasi pada kode *crawler* yang telah ditulis. Berikut penulisan *field* pada file *item*:

```
1.  import scrapy
2.
3.  class IfanccItem(scrapy.Item):
4.      # define the fields for your item here like:
5.      # name = scrapy.Field()
6.      product_name        = scrapy.Field()
7.      product_brand       = scrapy.Field()
8.      product_certificate = scrapy.Field()
9.      product_country     = scrapy.Field()
10.     #pass
```

Kode 5. 34 penulisan field item pada file item IFANCC

Penulisan *field item* pada kode 5.34 berada pada baris 6 hingga baris 9.

iii. Ekstraksi menjadi csv.

Penulisan kode program untuk ekstraksi data ke csv dilakukan pada *pipelines* berikut kode program yang di tulis:

```
1.  import csv
2.
3.  class IfanccPipeline(object):
4.      def open_spider(self, spider):
5.          self.file = open('outputData.csv', 'w')
6.          self.writer = csv.writer(self.file)
7.
8.      def close_spider(self, spider):
9.          self.file.close()
10.
11.     def process_item(self, item, spider):
12.         self.writer.writerow(
13.             [
14.                 item['product_name'],
15.                 item['product_brand'],
16.                 item['product_certificate'],
17.                 item['product_country']
18.             ]
19.         )
20.         return item
```

Kode 5. 35 penulisan kode untuk ektraksi data IFANCC ke csv

Kode 5.35 diatas menjelaskan bahwa penulisan fungsi pertama (baris 4-6) bertujuan untuk membuka spider dan membuat csv, selanjutnya fungsi kedua (baris 8-9) merupakan fungsi untuk mengakhiri atau menutup spider, dan fungsi yang terakhir (baris 11-13) bertujuan untuk penulisan dalam csv. Dalam fungsi ini penulisan item yang diiginkan harus sesuai dengan spider utama yang telah ditulis sebelumnya.

Dan jangan lupa untuk mengaktifkan item pipeline pada file setting.py:

```
1.  ITEM_PIPELINES = {
2.      'usa.pipelines.IfanccPipeline': 300,
3.  }
```

Kode 5. 36 penghapusan komen pada item pipelines IFANCC

Pengaktifan *item pipelines* pada kode 5.36 terdapat pada bais 1 hingga 3.

2. **Melakukan Crawling Data.**
   Melakukan crawling data berarti menjalankan kode program yang telah ditulis sebelumnya. Pada tugas akhir ini kode program dijalankan secara terpisah karena struktur, elemen *html* untuk setiap website berbeda-beda. Setelah kode program untuk IFANCC dijalankan terdapat 183 data yang berhasil di *crawling.* Seluruh data tersebut nantinya akan dikumpulkan menjadi satu setelah seluruh kode program selesai dijalankan.

5.1.3 Mengumpulkan Seluruh Data Menjadi Satu

Setelah seluruh kode *crawler* dijalankan selanjutnya seluruh hasil dari implementasi kode *crawler* merupakan file csv. File csv tersebut berisi seluruh data produk yang ada pada website lembaga sertifikasi.

Setelah menggabungkan seluruh data dari tujuh lembaga sertifikasi yang telah dijelaskan diatas maka, terdapat 10.153 data yang berhasil dikumpulkan dan data tersebut selanjutnya akan memasuki tahap pembersihan yang dijelaskan berikutnya.

5.1.4 Tantangan yang dihadapi Peneliti

Tantangan yang didapat pada sub bab Akuisisi data yaitu setiap website lembaga sertifikasi memiliki struktur website yang berbeda-beda, tautan yang dimiliki website terkadang harus di pelajari lebih lanjut karena tidak bisa dibaca oleh kode program. Dua hal tersebut membuat pengerjaan pembuatan kode *crawl* membutuhkan waktu yang lama.

5.2 Data Cleansing

Data *cleansing* adalah pengolahan sejumlah data dengan berbagai cara yang dibutuhkan agar data dapat dimanfaatkan secara maksimal. Berikut ini beberapa tahap *data cleansing*

5.2.1 Penghapusan karakter yang tidak penting

Karakter tidak penting adalah karakter dimana sesuatu yang bukan bagian dari karakter utama sebuah data dan apabila dihapus karakter tersebut tidak menghilangkan makna dari sebuah kata.

Data yang berhasil dikumpulkan dari tujuh Lembaga sertifikasi halal sebanyak 10.153 data produk. Dari seluruh data yang terkumpul ternyata data yang semula di prediksi akan rapi ternyata banyak data yang terkumpul dengan karakter-karakter tidak penting. Penghapusan karakter menjadi penting karena ketika melakukan pengurutan sesuai alfabet akan menghasilkan hasil yang buruk begitu pula ketika melakukan pengukuran kesamaan antar produk. Berikut beberapa tahap penghapusan karakter tidak penting:

A. Penghapusan kolom yang kosong pada produk melalui csv.

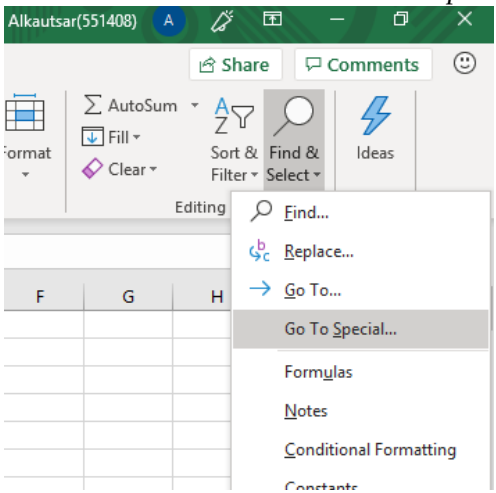
Berikut data yang telah di jadikan csv dimana terdapat kolom kosong setiap selang satu produk:

Product	Brand	expire
BIG VALUE O-CHOC- CREAM SANDWICH VANILLA FLAVOUR	FOODIE4U SDN BHD	30/06/2019
COOKIES - PISTACHIO BISCOTTI	PROFOUND PASTRY SDN. BHD.	15/06/2021
MINCE INDIAN BUFFALO	PRIMA FINE FOODS SDN BHD	31/12/2019
MYOTEIN PLUS - WHEY PROTEIN ISOLATE/HYDROLYSATE	BIOSCENERGY INTERNATIONAL SDN BHD	31/12/2020
PEANUT MUFFIN	HAZEL CAKE HOUSE	31/05/2019
(EOP) ANGUS PREMIUM RIBEYE MBS3	PALACE BUTCHER RESOURCES SDN. BHD.	15/10/2020
(EOP) ANGUS RIBEYE	PALACE BUTCHER RESOURCES SDN. BHD.	15/10/2020
(EOP) ANGUS STRIPLOIN	PALACE BUTCHER RESOURCES SDN. BHD.	15/10/2020

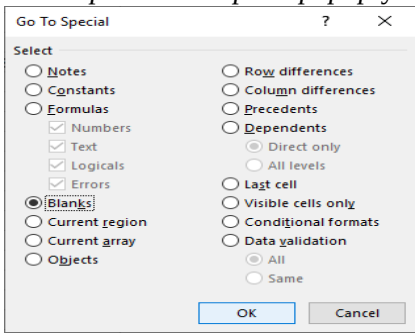
Gambar 5. 9 data yang memiliki kolom kosong

Dari gambar 5.9 diatas terdapat kolom kosong setiap satu produk. Oleh karena itu hal tersebut harus dihapus karena nantinya akan terbaca *NAN* oleh python dan menyebabkan kecepatan python berkurang ketika melakukan preprosessing ataupun proses-proses lainnya. Untuk menghapus kolom kosong yang jumlahnya sangat banyak tersebut dapat dilakukan dengan mudah langsung melalui excel.

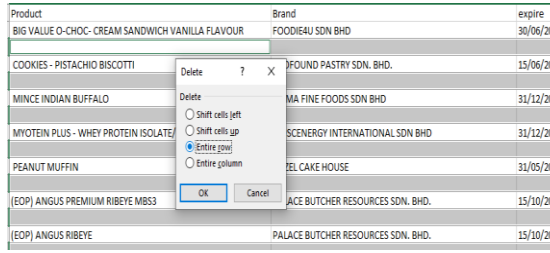
i) Melakukan *Find & Select* → *Go To Special*



ii) Pilih *option Blank pada popup* yang muncul



iii) Klik kanan lalu *delete Entire row*



iv) Selesai

Maka akan menampilkan data seperti dibawah:

Product	Brand	expire
BIG VALUE O-CHOC- CREAM SANDWICH VANILLA FLAVOUR	FOODIE4U SDN BHD	30/06/2019
COOKIES - PISTACHIO BISCOTTI	PROFOUND PASTRY SDN. BHD.	15/06/2021
MINCE INDIAN BUFFALO	PRIMA FINE FOODS SDN BHD	31/12/2019
MYOTEIN PLUS - WHEY PROTEIN ISOLATE/HYDROLYSATE	BIOSCENERGY INTERNATIONAL SDN BHD	31/12/2020
PEANUT MUFFIN	HAZEL CAKE HOUSE	31/05/2019
(EOP) ANGUS PREMIUM RIBEYE MBS3	PALACE BUTCHER RESOURCES SDN. BHD.	15/10/2020
(EOP) ANGUS RIBEYE	PALACE BUTCHER RESOURCES SDN. BHD.	15/10/2020
(EOP) ANGUS STRIPLOIN	PALACE BUTCHER RESOURCES SDN. BHD.	15/10/2020
(EOP) GRAIN FED RIBEYE	PALACE BUTCHER RESOURCES SDN. BHD.	15/10/2020
(EOP) GRAIN FED STRIPLOIN	PALACE BUTCHER RESOURCES SDN. BHD.	15/10/2020
(EOP) GRASS FED RIBEYE	PALACE BUTCHER RESOURCES SDN. BHD.	15/10/2020

B. Menghapus karakter “\r\r\n”.

Ketika data yang telah berhasil dikumpulkan ternyata masih banyak data yang terkumpul dengan karakter “\r\r\n”. oleh karena itu dibutuhkan penghapusan karakter tersebut. Dalam penelitian tugas akhir ini untuk menghapus karakter tersebut dilakukan *jupyter notebook* yang memanfaatkan bahasa pemograman *python*. Berikut kode program *python* yang ditulis untuk melakukan penghapusan karakter “\r\r\n”:

```
1. dcrawl['product_name']=dcrawl['product_name'].replace(to_re-
   place=r'\r\r\n', value='', regex=True)
2. dcrawl['product_brand']=dcrawl['product_brand'].replace(to_re-
   place=r'\r\r\n', value='', regex=True)
```

Kode 5. 37 kode program untuk menghapus karakter \r\r\n

Dari gambar 5.37 diatas diketahui bahwa data yang dihapus karakter tidak penting hanya pada kolom “product_name” baris pertama dan “product_brand”

baris kedua. Karena kedua kolom tersebut akan dilakukan *sorting* dan pengukuran *similarity*.
Berikut hasil dari implementasi kode 5.37:

1. Data sebelum dihapus menggunakan *python:*

	product_name	product_brand	expire_date	product_code
64	\r\r\n Soups\r\r\n	\r\r\n McDonalds\r\r\n	\r\r\n 29-08-16\r\r\n	NaN
59	\r\r\n BREAD, PASTRY, CONFECTIONERY MANUFACTUR...	\r\r\n KHALEEJ\r\r\n	\r\r\n 27-06-16\r\r\n	NaN
61	\r\r\n Baby food and hazelnut spread\r\r\n	\r\r\n Lino\r\r\n	\r\r\n 17-08-16\r\r\n	NaN
60	\r\r\n Beef Products\r\r\n	\r\r\n Podravka\r\r\n	\r\r\n 17-08-16\r\r\n	NaN
44	\r\r\n CANNED JUICE DRINK\r\r\n	\r\r\n RANI\r\r\n	\r\r\n 18-10-15\r\r\n	NaN
41	\r\r\n CHICKEN EGGS GRADE (A & AA)\r\r\n	\r\r\n GOLDEN EGGS\r\r\n	\r\r\n 2.9.2015\r\r\n	NaN
43	\r\r\n CHICKEN EGGS GRADE (A & AA)\r\r\n	\r\r\n GOLDEN EGGS\r\r\n	\r\r\n 4.10.2015\r\r\n	NaN

Gambar 5. 10 data sebelum penghapusan karakter tiadk penting

2. Data setelah implementasi kode program yang telah dijelaskan diatas.

	product_name	product_brand
64	Soups	McDonalds
256	BIG VALUE O-CHOC- CREAM SANDWICH VANILLA FLAVOUR	FOODIE4U SDN BHD
59	BREAD, PASTRY, CONFECTIONERY MANUFACTURING	KHALEEJ
61	Baby food and hazelnut spread	Lino
60	Beef Products	Podravka
44	CANNED JUICE DRINK	RANI
41	CHICKEN EGGS GRADE (A & AA)	GOLDEN EGGS
50	CHICKEN EGGS GRADE (A & AA)	SAHA

Gambar 5. 11 data setelah penghapusan karakter tidak penting

C. Menghapus spasi di awal dan di akhir.

Hal ini dilakukan karena sangat mempengaruhi ketika melakukan pengurutan sesuai abjad. Untuk penghapusan spasi ini juga dilakukan menggunakan *jupyter notebook.* Berikut kode program yang ditulis:

```
1. dcrawl.replace('^\s+', '', regex=True, inplace=True)
2. dcrawl.replace('\s+$', '', regex=True, inplace=True)
```

Kode 5. 38 kode program penghapusan spasi di awal dan di akhir

Dari kode 5.38 “dcrawl” adalah data frame yang merupakan data yang telah berhasil diambil sebelumnya. Lalu menggunakan fungsi *replace* untuk menghapus spasi yang ada. Kode pada baris pertama merupakan kode untuk menghapus spasi di awal string dan kode

pada baris kedua merupakan kode untuk untuk menghapus spasi di akhir string. Berikut hasil dari implementasi kode program tersebut:

1. Data sebelum dihapus spasi ketika di urutkan menjadi:

	product_name	product_brand	expire_date	product_code
64	\r\n Soups\r\n	\r\n McDonalds\r\n	\r\n 29-08-16\r\n	NaN
59	\r\n BREAD, PASTRY, CONFECTIONERY MANUFACTUR...	\r\n KHALEEJ\r\n	\r\n 27-06-16\r\n	NaN
61	\r\n Baby food and hazelnut spread\r\n	\r\n Linol\r\n	\r\n 17-08-16\r\n	NaN
60	\r\n Beef Products\r\n	\r\n Podravka\r\n	\r\n 17-08-16\r\n	NaN
44	\r\n CANNED JUICE DRINK\r\n	\r\n RANI\r\n	\r\n 18-10-15\r\n	NaN
41	\r\n CHICKEN EGGS GRADE (A & AA)\r\n	\r\n GOLDEN EGGS\r\n	\r\n 2.9.2015\r\n	NaN
43	\r\n CHICKEN EGGS GRADE (A & AA)\r\n	\r\n GOLDEN EGGS\r\n	\r\n 4.10.2015\r\n	NaN
50	\r\n CHICKEN EGGS GRADE (A & AA)\r\n	\r\n SAHA\r\n	\r\n 17-11-15\r\n	NaN
47	\r\n CHOCOLATE, MILK POWDER\r\n	\r\n KITKAT, NESTLE MINIS, MACKINTOSH QUALIT...	\r\n 26-10-15\r\n	NaN

Gambar 5. 12 data hasil sorting sebelum penghapusan spasi pada awal huruf

Dari gambar 5.12 diketahui bahwa hasil sorting terlihat sangat buruk bukan karena kode program melainkan pengaruh dari spasi yang terletak pada awal dan akhir string.

2. Data setelah di implementasi kode program yang telah dijelaskan diatas ketika di urutkan menjadi:

product_name	product_brand
ACAR SAUCE	SAUCE EMPIRE MANUFACTURING SDN BHD
ACAR/JELATAH(WAREHOUSE/FUNCTION)	TARBUSH CATERING & FOOD INDUSTRY SDN. BHD.
ACETIC ACID	KONG LONG HUAT CHEMICALS SDN BHD
ACHAR GOSHT MASALA	Convenience Food Industries (Private) Limited
ACHAR GOSHT PASTE	Convenience Food Industries (Private) Limited
ADDAS SOUP-WAREHOUSE	TARBUSH CATERING & FOOD INDUSTRY SDN. BHD.
ADEKA COOKING OIL FRY TOP	FOCAL MARKETING SDN. BHD.
AFZA 9623	SYNAROME MANUFACTURING COMPANY (PVT) LTD
AFZA BLOOM	SYNAROME MANUFACTURING COMPANY (PVT) LTD
AFZA BM	BELL MORE AROMATICS

Gambar 5. 13 data setelah penghapusan spasi di awal dan di akhir data

Dari gambar 5.13 dapat dilihat hasil *sortng* sudah benar.

5.2.2 Mengubah seluruh karakter kedalam *lower case.*

Pada bab sebelumnya telah dijelaskan bahwa mengubah seluruh karakter kedalam huruf kecil merupakan sesuatu yang penting karena sangat mempengaruhi proses pengecekan kesamaan karakter antar produk. Sehingga perlu dilakukan proses ini. Proses ini juga dilakukan menggunakan *jupyter notebook.* Berikut kode program yang ditulis untuk merubah seluruh karakter menjadi huruf kecil*:*

```
1. dcrawl['product_name']=dcrawl['product_name'].str.lower()
2. dcrawl['product_brand']=dcrawl['product_brand'].str.lower()
```

Kode 5. 39 kode program mengubah seluruh string menjadi huruf kecil

Pada kode 5.39 diketahui bahwa perubahan string kedalam *lower case* hanyalah pada kolom “product_name” baris pertama dan “product_brand” baris kedua. Oleh karena itu kode program dipisah seperti gambar diatas. Berikut hasil dari implementasi kode program tersebut:

1. Data sebelum dilakukan proses perubahan kedalam huruf kecil*:*

product_name	product_brand	expire_date	product_code
Angus Beef Patties	Al-Safa Halal	NaN	24967 00801
Chicken Hot dogs 12.6 Oz	Al-Safa Halal	NaN	24967 43660
Bulk Pack - Breaded Burgers 1.4Kg	Al-Safa Halal	NaN	24967 01603
Chicken Patties 24 Oz	Al-Safa Halal	NaN	24967 42500
Bulk Pack - Nuggets 1.4Kg	Al-Safa Halal	NaN	24967 01601
Chicken Burgers 680 g	Al-Safa Halal	NaN	24967 42400
Chicken Frankfurters 12 Oz	Al-Safa Halal	NaN	24967 50810
Popcorn Chicken 340 g	Al-Safa Halal	NaN	24967 60006

Gambar 5. 14 data sebelum dilakukan lower casw

Data dari gambar 5.14 terdapat banyak string yang tidak dalam huruf kecil.

2. Data setelah dilakukan proses perubahan kedalam huruf kecil*:*

product_name	product_brand	expire_date	product_code
angus beef patties	al-safa halal	NaN	24967 00801
chicken hot dogs 12.6 oz	al-safa halal	NaN	24967 43660
bulk pack - breaded burgers 1.4kg	al-safa halal	NaN	24967 01603
chicken patties 24 oz	al-safa halal	NaN	24967 42500
bulk pack - nuggets 1.4kg	al-safa halal	NaN	24967 01601
chicken burgers 680 g	al-safa halal	NaN	24967 42400
chicken frankfurters 12 oz	al-safa halal	NaN	24967 50810
popcorn chicken 340 g	al-safa halal	NaN	24967 60006

Gambar 5. 15 data setelah dilakukan lower case

Seluruh string pada *dataframe* telah mengalami *lower case*.

5.2.3 Pengurutan pada kolom nama produk.

Pengurutan hanya dilakukan pada kolom nama produk karena jika dilakukan pada brand produk maka nama produk dan brandnya menjadi tidak teratur. Dalam hal ini melakukan proses pengurutan juga dilakukan menggunakan *jupyter notebook*. Berikut kode program yang ditulis:

```
1. dcrawl.sort_values(by=['product_name'], inplace=True)
```

Kode 5. 40 kode program untuk pengurutan pada nama produk data hasil crawling

Kode 5.40 merupakan kode untuk mengurutkan data hasil *crawling* baris pertama. Data dari Halal Nutrition Food sendiri juga harus di lakukan *sorting* agar nantinya proses *similarity* menjadi lebih cepat. Berikut kode program untuk data Halal Nutrition Food:

```
1. halal.sort_values(by=['ProductHalal'], inplace= True)
```

Kode 5. 41 kode program untuk pengurutan pada nama produk data halal nutrition food

Dengan kode 5.41 baris pertama diatas otomatis dataframe akan tersimpan secara terurut pada dan siap dilakukan pengukuran kesamaan. Berikut hasil implementasi kedua kode diatas:

1. Data yang dikumpulkan dan data Halal Nutrition Food sebelum dilakukan pengurutan.

product_name	product_brand	expire_date	product_code
angus beef patties	al-safa halal	NaN	24967 00801
chicken hot dogs 12.6 oz	al-safa halal	NaN	24967 43660
bulk pack - breaded burgers 1.4kg	al-safa halal	NaN	24967 01603
chicken patties 24 oz	al-safa halal	NaN	24967 42500
bulk pack - nuggets 1.4kg	al-safa halal	NaN	24967 01601
chicken burgers 680 g	al-safa halal	NaN	24967 42400
chicken frankfurters 12 oz	al-safa halal	NaN	24967 50810

Gambar 5. 16 data hasil scapping sebelum dilakukan sorting

ProductHalal	ManufakturHalal
dehydrated ginger(powder)	shandong xingda foodstuffs group co., ltd.
dehydrated green chilli (coarse fraction & pow...	jain farm fresh foods ltd.
dehydrated green/ white leek	huisong pharmaceuticals
dehydrated horseradish (powder , flake)	shandong xingda foodstuffs group co., ltd.
dehydrated jamun seed powder	jain farm fresh foods ltd.
dehydrated leek	jain farm fresh foods ltd.
dehydrated mango	qingdao matsumoto foods co., ltd.
dehydrated mushroom	qingdao matsumoto foods co., ltd.
dehydrated onion	laiwu manhing vegetables fruits corporation
dehydrated onion (powder , granulated)	shandong xingda foodstuffs group co., ltd.

Gambar 5. 17 data Halal Nutrition Food sebelum dilakukan sorting

2. Data yang dikumpulkan dan data Halal Nutrition Food setelah dilakukan pengututan.

product_name	product_brand
aaloo bhujia	freshmate co.
aaloo pratha	dawn frozen foods
aaloo samosa	dawn frozen foods
aaloo samosa (economy)	dawn frozen foods
aaloo samosa (family)	dawn frozen foods
abok-abok	rahmat venture sdn bhd
acar limau	ainons cakes & pastries

Gambar 5. 18 data hasil scrapping setelah di scrapping

ProductHalal	ManufakturHalal
albert heijn,zwarte peper molen	NaN
albert hein,biologische 100% appel stroop	NaN
albert heinj,burger au potiron et au thym	NaN
albert lhuissier,rillettes de la sarthe	NaN
albert lhuissier,souchon d'auvergne,saucisson ...	NaN
albert mened,raifort albert menes	NaN
albert menes,14 biscottes aux raisins toastees	NaN

Gambar 5. 19 data Halal Nutrition Food setelah di sorting

5.2.4 Mengukur String untuk Mempercepat Proses Similarity

Dalam pengukuran string dilakukan ketika melakukan perulangan pada proses pengukuran kesamaan dengan alasan karena hal tersebut membuat perkerjaan lebih efektif selain itu juga tidak berpengaruh terhadap kecepatan proses pengukuran kesamaan. Untuk melakukan hal tersebut seperti halnya melakukan *preprocessing* juga dilakukan menggunakan *jupyter notebook*. Kode program yang ditulis untuk pengecekan string sebagai berikut:

```
1.  for a1 in range(dcrawl.shape[0]):
2.      for a2 in range(halal.shape[0]):
3.          dcr=dcrawl.product_name[a1]
4.          dcr1=dcrawl.product_brand[a1]
5.          hll=halal.ProductHalal[a2]
6.          hll1=halal.ManufakturHalal[a2]
7.          tes2= True
8.          test = True
9.
10.         try:
11.
12.             if(dcr1[:5]==hll1[:5]):
13.                 test=True
14.             else :
15.                 test=False
16.         except:
17.             test=False
18.             tes2=False
19.
20.         if dcr[:5] == hll[:5] or test:
21.             valueproduct=textdistance.hamming.normalized_distance(dcr,hll)
22.             if tes2:
23.                 valuebrand=textdistance.hamming.normalized_distance(dcr1,hll1)
24.             else:
25.                 valuebrand=0
26.             if(valueproduct>0.7 or valuebrand>0.7 ):
27.                 print(dcr,"&",hll,valueproduct,valuebrand,"=","identik")
```

kode 5. 42 penulisan untuk membandingkan kesamaan string

Pada kode 5.42 penulisan program pada baris 12 dan 20 merupakan kode program untuk melakukan pengecekan string. Dalam hal ini dilakukan pengecekan lima string awal apabila sama maka data akan langsung memasuki proses pengukuran kesamaan.

5.2.5 Tantangan yang dihadapi

Tantangan yang dihadapi pada sub bab 5.2 yaitu banyaknya data yang memiliki karakter tidak penting yang bukan bagian dari produk tersebut. Karakter tidak penting yang menjadi masalah yaitu karakter tersebut berbeda-beda setiap hasil *crawling* lembaga sertifikasi sehingga, membuat peneliti harus memetakan data agar dapat diproses dengan baik.

5.3 Similarity Data

Pada bagian ini data yang akan di *similarity* telah melalui proses pembersihan. *Similarity* dilakukan menggunakan *library* yang dimiliki oleh *python*. Sehingga kode program yang di tulis untuk setiap metode *similarity* sama sesuai denga *flowchart* yang terdapat pada sub bab 4.3 pada gambar 4.8, yang membuat perbedaan hanya ketika memanggil fungsi jenis *similarity* yang dibutuhkan.

5.3.1 Pembuatan Kode Program Similarity

Berikut kode program yang di buat untuk melakukan *similarity:*

```
1. for a1 in range(dcrawl.shape[0]):
2.     for a2 in range(halal.shape[0]):
3.
4.         dcr=dcrawl.product_name[a1]
5.         dcr1=dcrawl.product_brand[a1]
6.         hll=halal.ProductHalal[a2]
7.         hll1=halal.ManufakturHalal[a2]
```

Kode 5. 43 penulisan perulangan pada data yang akan dibandingkan

Dari kode 5.43 pada baris 1 sampai baris ke 2 menjelaskan perulangan data untuk kedua data yang akan dibandingkan. Dimana *a1* merupakan variabel untuk data hasil *crwaling* dan *a2* untuk data Halal Nutrition Food. Lalu setelahnya pada baris 4-7 adalah penulisan variabel untuk kolom atau list data yang akan di bandingkan.

```
1. if str(dcr)[:5] == str(hll)[:5] or str(dcr1)[:5]==str(hll1)[:5]:
2.           if len (str(hll1))==0:
3.               continue
4.           else:
5.               valueproduct = textdistance.cosine.normalized_similar-
   ity(str(dcr),str(hll))
6.               valuebrand = textdistance.cosine.normalized_similar-
   ity(str(dcr1),str(hll1))
```

Kode 5. 44 penulisan similarity dilakukan dengan kondisi tertentu

Kode 5.44 pada baris pertama memiliki arti dimana *similarity* akan dilakukan apabila perbandingan nama produk memiliki lima string awalan yang sama atau perbandingan

brand produk memiliki lima string awalan yang sama. Lalu setelah itu terdapat kondisi selanjutnya pada baris ke dua dimana apabila terdapat brand produk yang kosong maka lanjut ke baris selanjutnya untuk di lakukan perbandingan *string* lagi. Ketika semua kondisi terpenuhi (baris 1-4) setelah itu *similarity* dilakukan (baris 5-6).

```
1.  if(valueproduct>0.8 or valuebrand>0.8 ):
2.                          writer.writerow({'tulis csv'})
3.
4.  csv_file.close()
```

Kode 5. 45 penulisan hasil kedalam csv

Pada kode 5.45 pada baris 2 hasil *similarity* akan di tulis kedalam csv apabila memenuhi syarat dimana nilai *similarity* nama produk atau brand produk diatas sama dengan 0.8 pada baris 1. Untuk hasil *similarity* akan dijelaskan pada bab 6.

5.3.2 Tantangan yang dihadapi

Data yang dikumpulkan sebanyak 10.153 produk akan dibandingkan dengan data Halal Nutrition Food sebanyak 350.000 lebih produk. Sehingga perulangan yang dilakukan mencapai 3.553.550.000 perulangan. Untuk setiap 40.000 kali perulangan peneliti menghitung waktu sebesar 10 detik. Jika terdapat 3 milyar lebih perulangan maka, membutuhkan waktu sebanyak 250 jam untuk menyelesaikan pengukuran *similarity*. Solusi yang digunakan disini yaitu pembacaan data dibagi untuk setiap 35 juta kali perulangan untuk setiap kernel dan dijalankan pada komputer yang berbeda-beda sehingga waktu yang digunakan untuk pengukuran *similarity* hanya 5 jam.

5.4 Itegrasi Data

Integrasi data dalam tugas akhir ini dimaksudkan adalah pembaharuan data pada *website* Halal Nutrition Food. Pembaharuan data dilakukan memanfaatkan data yang telah di ambil oleh peneliti dan telah dilakukan pengolahan dan *similarity*. Untuk melakukan hal tersebut data di impor ke dalam basis data Halal Nutrition Food menggunakan *tools MySQL* yang diolah memanfaatkan bahasa pemograman *python* . Dalam

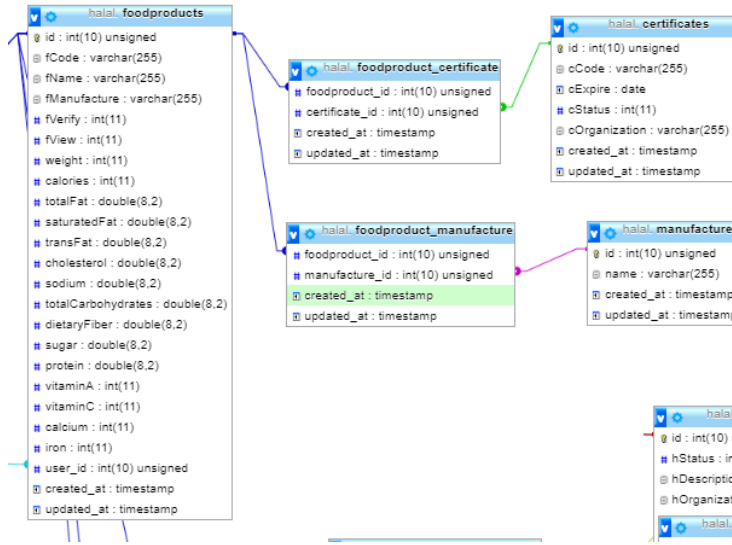
tugas akhir ini data yang akan digunakan merupakan hasil dari pengukuran *levenshtein* karena, metode tersebut merupakan yang terbaik diantara empat metode *similarity* yang telah dipraktikkan dan dianalisis.

5.3.1 Impor Data ke Basis Data Halal Nutrition Food

Pada bagian ini melakukan impor data memiliki dua tahap pembaharuan yaitu:

- Data produk yang belum ada pada basis data Halal Nutrition Food
- Data produk yang telah dilakukan similarity.

Dalam mengimpor data terlebih dahulu harus diketahui skema basis data yang dimiliki Halal Nutrition Food agar, ketika data dimasukkan sudah jelas akan dimasukkan tabel mana saja dan kedalam kolom mana saja. Berikut skema basis data Halal Nutrition Food:



Gambar 5. 20 Skema database Halal Nutrition Food

Gambar 5.20 merupakan potongan skema dari basis data Halal Nutrition Food. Dari skema diatas dapat diketahui atribut mana saja yang nantinya data dapat diperbaharui.

a. Pembaharuan Berdasarkan data produk yang belum ada pada Halal Nutrition Food

Pada bagian ini data produk yang sama telah dipisahkan melalui proses *similarity* sehingga, hanya tersisa data baru untuk Halal Nutrition Food. Mengimpor data ini dilakukan menggunakan *python* karena memudahkan peneliti dan juga lebih efektif dibandingkan mengimpor satu-satu ke setiap tabel.

Berikut penulisan kode program serta *query* untuk impor data baru ke dalam basis data Halal Nutrition Food:

```
1.  import MySQLdb
2.  import mysql.connector
3.  import pandas as pd
4.  import datetime
5.
6.  now = datetime.datetime.now()
7.
8.  db = MySQLdb.connect(host="localhost",
9.                       user="halal1",
10.                      passwd="123ewq",
11.                      db="halal")
12. df=pd.read_csv("datatest.csv")
13.
14. cur = db.cursor()
15.
16. for x in df.iterrows():
17.     cur.execute(("SELECT max(id) FROM foodproducts"))
18.     IDfood = cur.fetchall()[0][0]+1
19.     sqlInsertFood= "INSERT INTO `foodproducts`(`id`, `fCode`, `fName`, `fManufac-
    ture`, `fVerify`, `fView`, `weight`, `calories`, `totalFat`, `saturatedFat`, `trans-
    Fat`, `cholesterol`, `sodium`, `totalCarbohydrates`, `dietaryFiber`, `sugar`, `pro-
    tein`, `vitaminA`, `vitaminC`, `calcium`, `iron`, `user_id`, `created_at`, `up-
    dated_at`) VAL-
    UES (%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s)"
20.     valInsertFood=(ID-
    food,'',str(x[1][0]),str(x[1][1]),'','','','','','','','','','','','','','','','','','',''
    ,'')
21.     cur.execute(sqlInsertFood, valInsertFood)
22.     db.commit()
23.     cur.execute(("SELECT max(id) FROM manufactures"))
24.     IDman = cur.fetchall()[0][0]+1
25.     sqlInsertMan= "INSERT INTO `manufactures`(`id`, `name`, `created_at`, `up-
    dated_at`) VALUES (%s,%s,%s,%s)"
26.     valInsertMan=(IDman,str(x[1][1]),now,now)
27.     cur.execute(sqlInsertMan, valInsertMan)
28.     db.commit()
29.     cur.execute(("SELECT max(id) FROM certificates"))
30.     IDcer = cur.fetchall()[0][0]+1
31.     sqlInsertCer= "INSERT INTO `certificates`(`id`, `cCode`, `cExpire`, `cStatus`, `cOr-
    ganization`, `created_at`, `updated_at`) VALUES (%s,%s,%s,%s,%s,%s,%s)"
32.     valInsertCer=(IDcer,str(x[1][2]),str(x[1][5]),'',str(x[1][4]),now,now)
33.     cur.execute(sqlInsertCer, valInsertCer)
34.     db.commit()
35.     sqlInsertFM= "INSERT INTO `foodproduct_manufacture`(`foodproduct_id`, `manufac-
    ture_id`, `created_at`, `updated_at`) VALUES (%s,%s,%s,%s)"
36.     valInsertFM=(IDfood,IDman,now,now)
37.     cur.execute(sqlInsertFM, valInsertFM)
38.     db.commit()
39.     sqlInsertFC= "INSERT INTO `foodproduct_certificate`(`foodproduct_id`, `certifi-
    cate_id`, `created_at`, `updated_at`) VALUES (%s,%s,%s,%s)"
40.     valInsertFC=(IDfood,IDcer,now,now)
41.     cur.execute(sqlInsertFC, valInsertFC)
```

Kode 5. 46 penulisan kode dan query untuk insert data baru

Dari kode 5.46 kita ketahui bahwa terdapat lima tabel yang akan ditambahkan datanya yaitu: *foodproduct*, *manufacture*, *certificate*, *foodproduct_manufactures*, dan *foodproduct_certificates.* Untuk nilai-nilai yang dimasukkan kedalam tabel sesuai dengan kode program yang tertulis pada kode 5.54. untuk *query* data terdapat pada baris 31,35, dan 39 padakode 5.54.

b. Pembaharuan Berdasarkan Data produk yang telah dilakukan similarity.

Pada bagian ini data yang diperbaharui merupakan data dengan nama produk yang sama persis. Sehingga pembaharuan yang dilakukan pada atribut selain nama produk seperti manufaktur, certificate, dan lainnya. Karena dalam pembaharuan ini membutuhkan kondisi tertentu untuk menambahkan atribut baru pada produk maka seperti halnya sub bab 5.3.1 pembaharuan data dilakukan memanfaatkan *python.* Berikut penulisan kode untuk pembaharuan data:

```
import MySQLdb
import mysql.connector
import pandas as pd
import datetime

now = datetime.datetime.now()

db = MySQLdb.connect(host="localhost",
                     user="halal1",
                     passwd="123ewq",
                     db="halal")
df=pd.read_csv("datasamafix.csv")

cur = db.cursor()
cur.execute("SELECT id from `foodproducts` WHERE `fname`='bkkjkjk'")
test = cur.fetchall()

for x in df.iterrows():

    cur.execute(("SELECT id from `foodproducts` WHERE `fname`='")+str(x[1][0])+"'""'")
#     print(("SELECT id from `foodproducts` WHERE `fname`='")+str(x[1][0])+"'")

    results = cur.fetchall()
    if results != test :
        print(("SELECT id from `foodproducts` WHERE `fname`='")+str(x[1][0])+"'")
        cur.execute(("SELECT max(id) FROM manufactures"))
        manID = cur.fetchall()[0][0]+1
        cur.execute(("SELECT max(id) FROM certificates"))
        cerID = cur.fetchall()[0][0]+1
        sql=("INSERT INTO `manufactures`(`id`, `name`, `created_at`, `updated_at`) VALUES (%s,%s,%s,%s)")
        val= (manID,str(x[1][1]),now,now)
        cur.execute(sql,val)
        db.commit()
        sql=("INSERT INTO `foodproduct_manufacture`(`foodproduct_id`, `manufacture_id`, `created_at`, `updated_at`) VALUES (%s,%s,%s,%s)")
        val= (results,manID,now,now)
        cur.execute(sql,val)
        db.commit()
        sql=("INSERT INTO `certificates`(`id`, `cCode`, `cExpire`, `cStatus`, `cOrganization`, `created_at`, `updated_at`) VALUES (%s,%s,%s,%s,%s,%s,%s)")
        val= (cerID,str(x[1][2]),str(x[1][5]),"",str(x[1][6]),now,now)
        cur.execute(sql,val)
        db.commit()
        sql= "INSERT INTO `foodproduct_certificate`(`foodproduct_id`, `certificate_id`, `created_at`, `updated_at`) VALUES (%s,%s,%s,%s)"
        val=(results,cerID,now,now)
        cur.execute(sql, val)
        db.commit()
```

Kode 5. 47 penulisan kode program dan query pembaharuan data Halal Nutrition Food

Kode 5.47 mirip dengan kode 5.46 yang membuat perbedaan adalah pada kode 5.47 menjelaskan apabila terdapat produk yang sama maka id nya akan diambil lalu atribut untuk data dengan id tersebut akan di perbaharui (baris 18-23).

5.3.2 Tantangan yang dihadapi

Tantangam pada sub bab Integrasi data yaitu pada bagian memperbaharui data pada database Halal Nutrition Food dimana data yang ada harus dipetakan sesuai dengan tabel yang telah tersedia pada Halal Nutrition Food. Tabel yang terdapat pada basis data Halal Nutrition Food sangat banyak sehingga butuh melakukan beberapa *query insert* dan *update* untuk memperbaharui data sehingga pengerjaan integrasi menjadi sangat lama apabila dilakukan satu-persatu. Untuk itu peneliti memiliki solusi dimana *query* tersebut dijalankan bersamaan dengan menggunakan sebuah *tools* sebagai *controller* dan disini peneliti menggunakan *tools jupyter notebook* sebagai *controller query* tersebut.

5.4 Pengujian

Pada sub bab ini akan dilakukan tiga pengujian untuk penelitian ini yaitu:

5.4.1 Pengujian Jumlah data yang dihasilkan setiap *similarity*.

Pada pengujian ini peneliti mengelompokkan jumlah data berdasarkan *treshold* yang diberikan. *Treshold* yang menjadi acuan peneliti adalah dengan nilai lebih besar dari 0.8 dan lebih besar dari 0.9. berikut tabel serta jumlah produk setiap metode similaritas:

nama	Bbt(0.6 dan 0.4)	Bbt(0.4 dan 0.6)	Bbt(0.7 dan 0.3)	Bbt(0.3 dan 0.7)	Bbt(0.8 dan 0.2)	Bbt(0.2 dan 0.8)	Nama produk
Cosine	330	329	669	1242	1120	3381	7368
Jaccard	20	8	218	103	553	180	1572

Jaro	862	235	5349	1368	26135	5001	173326,
Leven	0	1	23	1	543	181	1282

Tabel 5. 2 jumlah produk dengan treshold lebih dari 0.8

nama	Bbt(0.6 dan 0.4)	Bbt(0.4 dan 0.6)	Bbt(0.7 dan 0.3)	Bbt(0.3 dan 0.7)	Bbt(0.8 dan 0.2)	Bbt(0.2 dan 0.8)	Nama produk
Cosine	0	0	23	8	239	111	1511
Jaccard	0	0	0	0	20	8	1337
Jaro	2	1	3	1	297	211	6597
Leven	0	0	0	0	0	1	1282

Tabel 5. 3 jumlah produk dengan treshold lebih dari 0.9

Dari kedua tabel yaitu tabel 5.2 dan tabel 5.3 diketahui pengukuran similaritas *jaccard* dan *levenshtein* lebih sensitif dibanding metode similaritas yang lain. Dikarenakan jumlah data yang dihasilkan untuk kedua *treshold* sedikit sehingga, *jaccard* ataupun *levenshtein* dapat dipertimbangkan untuk menjadi kandidat metode similaritas yang digunakan untuk integrasi data.

Ditetapkan *treshold* 0.8 dan 0.9 karena nilai tersebut dinilai sudah mirip antara satu data dengan data yang dibandingkan sehingga peneliti memutuskan untuk menguji pada kedua *treshold* tersebut.

5.4.2 Penghitungan *precision* dan *recall.*

Perhitungan p*precission, recall, accuracy,* dan *f-mesure* bagian ini hanya berdasarkan nilai similaritas pada nama produk. Karena peneliti telah melakukan pengecekan pada data bahwa tidak ada data yang *match* berdasarkan produk dan brand sehingga, pembobotan yang dilakukan tidak dihitung

precission, recall, accuracy, dan *f-mesure.* Berikut hasil dari perhitungan setiap metode *similarity*:

Nama Similarity	Precision	Recall	Accuracy	F-mesure
Cosine	0.909091	0.454545	0.87	0.606061
Jaccard	0.985714	0.896104	0.91	0.938776
Jaro	0.5	0.5	0.96	0.5
Levenshtein	1	0.897436	0.92	0.945946

Tabel 5. 4 Tabel hasil perhitungan precission, recall, accuracy, dan f-mesure pada setiap similarity

Tabel 5.4 merupakan hasil *precission, recall, accuracy,* dan *f-mesure* berdasarkan sampel dari 100 data yang diambil secara acak. Pada bagian ini metode yang memiliki nilai *f-mesure* terbaik akan digunakan untuk integrasi data. Dalam kasus tugas akhir ini *f-mesure* menjadi fokusan karena merupakan kombinasi dari *precission* dan *recall*. Karena pada tugas akhir ini *precission* dan *recall* merupakan hal yang perlu dipertimbangkan maka nilai *f-mesure* menjadi indikator penting untuk memilih metode pengukuran *similarity* terbaik.

Pada kolom yang di *highlight* pada tabel 5.4 nilai *f-mesure* tertinggi dan dimiliki oleh metode *levenshtein.* Sehingga dari tabel 5.3 dapat diambil kesimpulan bahwa metode *levenshtein* merupakan metode yang terbaik diantara seluruh metode similaritas yang digunakan. Karena cara menghitung nilai kesamaan *levenshtein* lebih sensitif dan lebih ketat. *Levenshtein* menghitung kemiripan dua objek dengan cara menghitung jarak antar objek tersebut dimana yang dihitung merupakan (insertion, deletion, dan subtituton) menggunakan sebuah matrik.

Dari tabel 5.3 di ketahui metode *cosine* memiliki *precission* dan *accuracy* yang baik tapi *recall* dan *f-mesure* rendah. Hal ini disebabkan karena *cosine* menghitung vektor antar dua string maka nilai yang diberikan lebih besar dari metode yang lain.

Metode *jaccard* memiliki *precission, recall, accuracy,* dan *f-mesure* yang cukup baik. Hal ini disebabkan karena cara *jaccard* menghitung dua buah *string* yaitu *intersection over union* dari kedua objek yang dibandingkan.

Pada metode *jaro* karena cara menghitung kemiripannya memiliki ciri khas sendiri yaitu dengan menambah nilai untuk panjang string pertama yang sama maka, *jaro* memberikan nilai yang sangat tinggi dalam pengukuran kesamaan sehingga ketika dilakukan pengujian *precission, recall, accuracy,* dan *f-mesure* akan menghasilkan pengujian yang tidak bagus.

5.4.3 Pengujian pengecekan *data base* Halal Nutrition Food.

Pengecekan pembaharuan basis data pada Halal Nutrition Food dapat dilakukan dengan dua cara yaitu:

c. Mengecek data *Mysql* pada tabel yang telah diperbaharui. Berikut pengecekan yang dilakukan:

id	fCode	fName	fManufacture	fVerify	fView	weight	calories	totalFat
392916	3.56E+12	tomme de chevre les croises 31%mg		1	0	0	0	0.00
392917	9.00E+12	nudeln,lasagne,blatter 500g packung piacelli		1	0	0	0	0.00
392918	75398	courgette spaghetti		1	0	0	0	0.00
392919	3.76E+12	courgettes grillees au basilic mijotees bio 460.		1	0	0	0	0.00
392920	3.76E+12	courgettes - cat. i		1	0	0	0	0.00
392921	8.06E+12	fleur de courgette		1	0	0	0	0.00
392922	3.56E+12	interdis,groupe carrefour,ratatouille		1	0	0	0	0.00
392923	41331028738	zucchini flower		1	0	0	0	0.00
392924	3.76E+12	fleurs de courgettes farcies a la mozarella et anc...		1	0	0	0	0.00
392925	3.08E+12	la courgette en rondelles		1	0	0	0	0.00
392926	3.76E+12	moorea,moorea commerce fruits,mini concombre		1	0	0	0	0.00

(a)

id	fCode	fName	fManufacture	fVerify	fView	weight	calories	totalFat
392926	3.76E+12	moorea,moorea commerce fruits,mini concombre		1	0	0	0	0.00
392927		100% Juice Black Cherry	365	0	0	0	0	0.00
392928		100% Juice Concorde Grape From Concentrate	365	0	0	0	0	0.00
392929		100% Juice Tart Cherry	365	0	0	0	0	0.00
392930		Almond Milk Original	365	0	0	0	0	0.00
392931		Applesauce Unsweetened	365	0	0	0	0	0.00
392932		Assortment Crackers	365	0	0	0	0	0.00
392933		Bran Flakes Cereal	365	0	0	0	0	0.00
392934		Brown Rice Crisp Cereal	365	0	0	0	0	0.00
392935		Chocolate Ice Cream	365	0	0	0	0	0.00

(b)

Gambar 5. 21 gambar (a) merupakan gambar tabel sebelum data di perbaharui, gambar (b) merupakan gambar tabel setelah data di perbaharui

Dari gambar 5.21 diketahui bahwa data telah di tambah dimana pada gambar (a) tabel memiliki id sebanyak 88416 kemudian, setelah itu pada gambar (b) tabel memiliki id yang bertambah mulai dari 88416 hingga jumlah data yang ditambah.

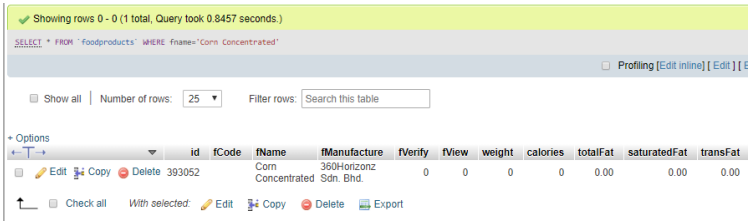
d. Pengecekan menggunakan *query*.

Cara ini dapat dilakukan dengan menampilkan data pada kondisi tertentu menggunakan *query*. Kondisi yang dimaksudkan disini adalah kondisi data yang di tambah. Misalnya data yang ditambah tidak memiliki manufaktur "x" sebelumnya kemudian data di perbaharui dengan menambahkan manufaktur "x" sehingga, data memiliki manufaktur "x". Adapun cara memastikan data tersebut memiliki manufaktur "x" maka dapat dilakukan pengecekan dengan menggunakan *query*. Berikut pengecekan yang dilakukan pada penelitian ini:

```
SELECT * FROM `foodproducts` WHERE fname='Corn Concentrated'
```

kode 5. 48 query pengecekan data

kode 5.48 merupakan contoh melakukan *query select* pada data yang telah diperbaharui. Hasil query pada kode 5.56 menghasilkan gambar sebagai berikut:



Gambar 5. 22 hasil query melihat jumlah produk

Dari gambar 5.22 diketahui bahwa jumlah data yang bertambah semuanya memiliki total sebanyak 141 data.

5.4.4 Tantangan yang dihadapi

Tantangan yang dihadapi pada sub bab pengujian yaitu data yang dihasilkan cukup banyak sehinhgga, untuk melakukan pengujian yang sifatnya aktual sangat sulit dilakukan. Solusi yang digunakan peneliti yaitu dalam pengujian seperti itu digunakan data sampel untuk melakukan pengujian.

BAB VI
HASIL DAN PEMBAHASAN

6.1 Hasil

Pada bab ini akan dijelaskan hasil itegrasi antara data yang diperoleh dari tujuh Lembaga sertifikasi data Halal Nutrition Food.

6.1.1 Jumlah Data yang dihasilkan berdasarkan nilai lebih dari 0.8

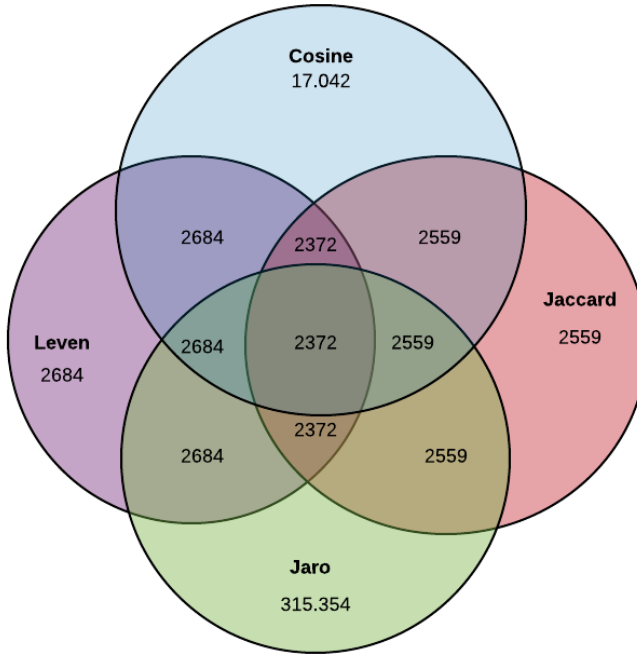
Gambar 6. 1 Jumlah seluruh hasil pengukuran diatas 0.8 setiap similarity

- Cosine = 17042
- Jaccard = 2559
- Jaro = 315354
- Levenshtein = 2685

6.1.2 Produk yang muncul pada setiap similartiy

Terdapat beberapa produk yang sama muncul di setiap *similarity*. Sehingga perlu dilakukan pengecekan produk yang seperti apa yang dapat muncul di seluruh *similarity*. Berikut terdapat dua pengelompokkan untuk mencari produk yang sama yang keluar di setiap *similarity.*

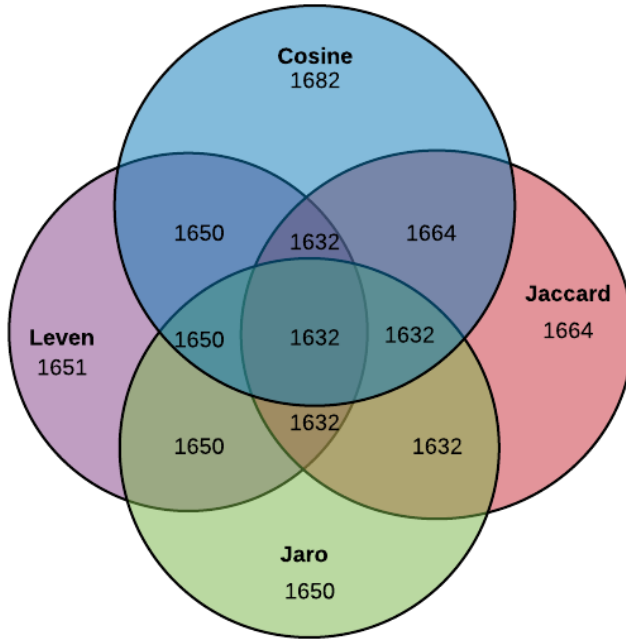
6.1.2.1 Seluruh produk yang muncul pada setiap silimilarity



Gambar 6. 2 Diagram jumlah produk setiap similarity

Pada Gambar 6.2 Lingkaran dengan warna yang berbeda mewakili setiap metode similaritas. Lingkaran dengan warna merah mewakili *Jaccard Similarity*, lingkaran hijau mewakili *Jaro Similarity*, lingkaran Ungu mewakili *Levenshtein Similarity*, dan yang terakhir llingkaran dengan warna biru mewakili *Cosine Similarity*.

6.1.2.2 Seluruh produk yang muncul pada setiap *similarity* dimana hanya yang memiliki nilai kesamaan sama dengan satu pada nama produk



Gambar 6. 3 Jumlah produk bernilai satu

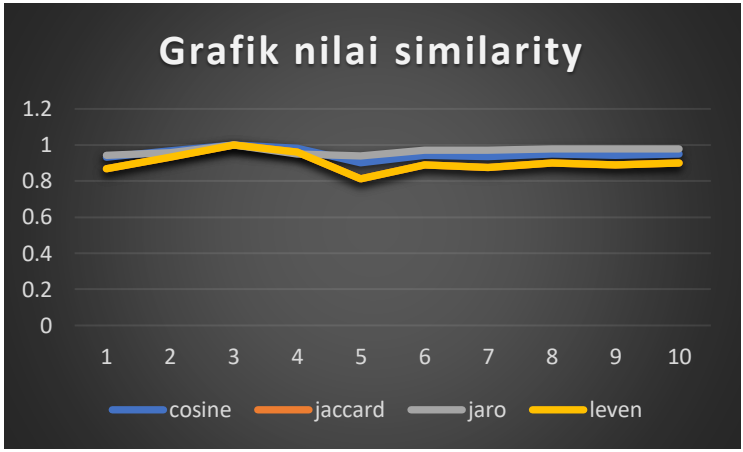
Sama halnya dengan gambar 6.2 sebelumnya, pada Gambar 6.3 hanya berbeda pada jumlah produk yang berada dalam diagram. Pada gambar 6.7 jumlah produk yang berada dalam diagram merupakan produk yang hanya memiliki nilai 1 (satu) pada atribut nama produk.

6.1.3 Hasil pengukuran akurasi untuk 10 produk acak

Tabel 6. 1 Nilai Similarity pada 10 produk secara acak

No	Nama Produk		cosine	jaccard	jaro	leven
1	1% low fat milk	1% lowfat mlk	0.9309	0.8667	0.9426	0.8667
2	1% low fat milk	1% lowfat milk	0.9661	0.9333	0.9581	0.9333
3	1% low fat milk	1% low fat milk	1	1	1	1
4	1% milk fat cottage cheese	1% milkfat cottage cheese	0.9806	0.9615	0.9483	0.9615
5	yellow corn tortilla chips	yellow corn round tortilla chips	0.9014	0.8125	0.9394	0.8125
6	yellow corn tortilla chips	yellow round tortilla chips	0.9436	0.8889	0.9695	0.8889
7	hydrolyzed vegetable protein(hvp powder)	hydrolyzed vegetable protein powder	0.9354	0.875	0.9695	0.875
8	ice cream	ice creame	0.9487	0.9	0.98	0.9
9	chocolate	chocolat	0.9428	0.8889	0.9778	0.8889
10	mayonnaise	mayonaise	0.9487	0.9	0.98	0.9

Tabel 6.1 diatas menampilkan 10 produk yang terpilih secara acak untuk setiap jenis similaritas. 10 produk tersebut digunakan untuk membandingkan hasil similaritas. Pada sub bab ini 10 produk tersebut disamakan untuk setiap *similarity* agar dapat di amati nilainya.



Gambar 6. 4 nilai similarity untuk 10 produk terpilih

Gambar 6.4 merepresentasikan grafik nilai *similarity* untuk 10 produk yang terdapat pada tabel 6.1. warna grafik yang berbeda-beda menunjukkan metode *similarity* yang berbeda

6.1.4 Perbandingan berdasarkan pembobotan dan similarity nama produk

Pada sub bab ini data dikartagorikan menjadia dua bagian sebagai berikut:

- Data untuk pembobotan
- Data untuk similariy nama produk

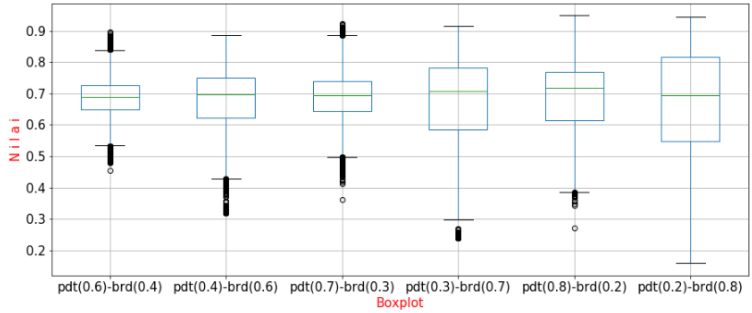
pembobotan dilakukan apabila brand produk tersedia pada data Halal Nutrition Food apabila tidak maka akan dilakukan perhitungan hanya pada nilai similaritas nama produk. Pembobotan dilakukan sebanyak enam kali yaitu:

- (nilai Nama produk*0.6) + (nilai Brand produk*0.4)
- (nilai Nama produk*0.4) + (nilai Brand produk*0.6)
- (nilai Nama produk*0.7) + (nilai Brand produk*0.3)
- (nilai Nama produk*0.3) + (nilai Brand produk*0.7)

- (nilai Nama produk*0.8) + (nilai Brand produk*0.2)
- (nilai Nama produk*0.2) + (nilai Brand produk*0.8)

Pada bagian ini dilakukan perbandingan untuk seluruh atribut pembobotan dan juga atribut *similarity* nama produk pada seluruh jenis similarias yaitu: *cosine*, *jaccard*, *jaro*, dan *levenshein*

6.1.4.1 Cosine

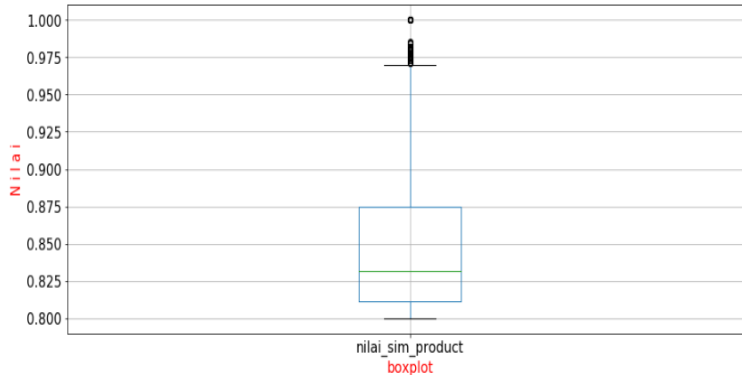


Gambar 6. 5 Box Plot Pembobotan pada produk

Pada gambar 6.5 terdapat enam *boxplot* yang berbeda-beda. Penjelasan setiap *boxplot* dapat dilihat pada tabel berikut:

Tabel 6. 2 Tabel penjelasan boxplot berdasarkan pembobotan cosine

	pdt0.6-brd0.4	pdt0.4-brd0.6	pdt0.7-brd0.3	pdt0.3-brd0.7	pdt0.8-brd0.2	pdt0.2-brd0.8
mean	0.686231	0.679781	0.689456	0.676557	0.692681	0.673332
std	0.062418	0.089571	0.080532	0.121406	0.110348	0.156709
min	0.453666	0.320131	0.362610	0.240098	0.271554	0.160065
25%	0.648084	0.621476	0.642901	0.584669	0.614986	0.547400
50%	0.690086	0.696821	0.695137	0.708404	0.718512	0.693832
75%	0.724644	0.750308	0.740048	0.781790	0.767565	0.817565
max	0.895840	0.886217	0.921880	0.914663	0.947920	0.943108



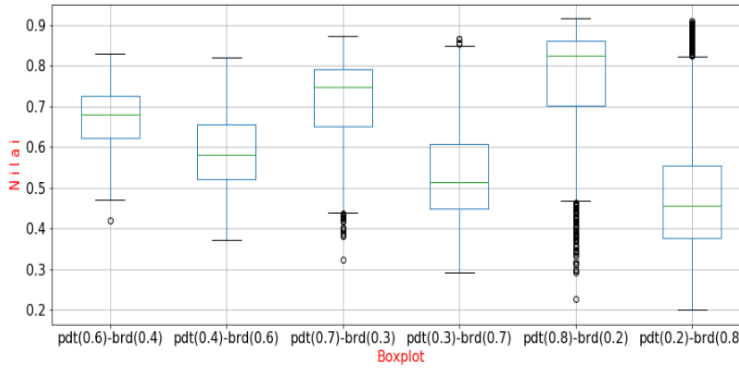
Gambar 6. 6 Box Plot Nilai similarity nama Produk

Pada gambar 6.6 hanya terdapat satu *boxplot.* Nilai yang dihitung hanya berdasarkan nilai *similarity* dari nama produk. Hal ini dikarenakan jika dari salah satu data tidak memiliki brand produk maka tidak efektif dilakukan pembobotan maka dari itu data harus dibagi dua. Berikut penjelasan dari gambar 6.6:

Tabel 6. 3 tabel penjelasan boxplot berdasarkan nilai similarity nama produk cosine

Pengukuran	Nilai
mean	0.859835
std	0.067910
min	0.800017
25%	0.811107
50%	0.831522
75%	0.875000
max	1.000000

6.1.4.2 Jaccard

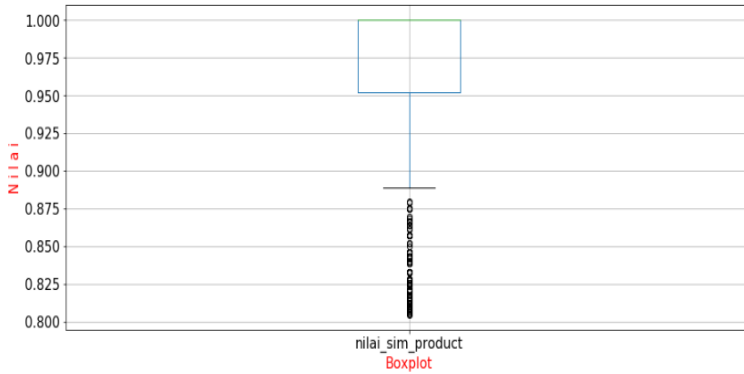


Gambar 6. 7 Boxplot pebobotan produk

Pada gambar 6.7 terdapat enam *boxplot* yang berbeda-beda. Penjelasan setiap *boxplot* dapat dilihat pada tabel berikut:

Tabel 6. 4 Tabel penjelasan boxplot berdasarkan pembobotan Jaccard

	pdt0.6-brd0.4	**pdt0.4-brd0.6**	**pdt0.7-brd0.3**	**pdt0.3-brd0.7**	**pdt0.8-brd0.2**	**pdt0.2-brd0.8**
mean	0.673212	0.590989	0.714323	0.549878	0.755434	0.508766
std	0.070044	0.096051	0.104538	0.139883	0.149682	0.188052
min	0.42069	0.371429	0.324138	0.290476	0.227586	0.2
25%	0.622048	0.52	0.65	0.447368	0.702934	0.376554
50%	0.68	0.58	0.74748	0.514444	0.825806	0.454545
75%	0.726316	0.656158	0.791667	0.608824	0.86087	0.555418
max	0.830769	0.82069	0.873077	0.865517	0.915385	0.910345



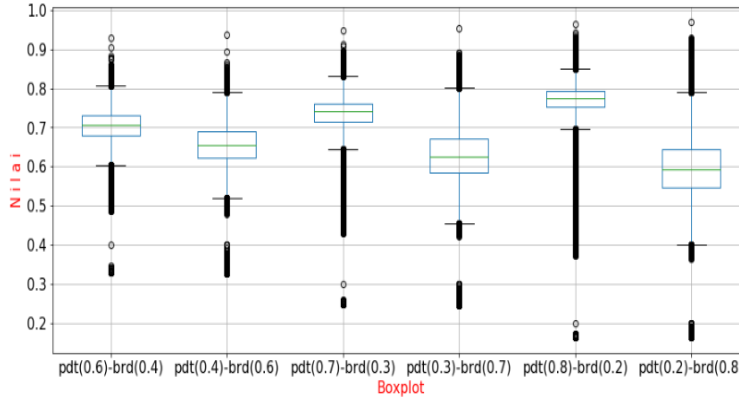
Gambar 6. 8 Boxplot nilai Similarity nama produk

Pada gambar 6.8 hanya terdapat satu *boxplot*. Nilai yang dihitung hanya berdasarkan nilai *similarity* dari nama produk. Hal ini dikarenakan jika dari salah satu data tidak memiliki brand produk maka tidak efektif dilakukan pembobotan maka dari itu dilakukan penghitungan hanya berdasarkan *similarity* nama produk. Berikut penjelasan dari gambar 6.8:

Tabel 6. 5 tabel penjelasan boxplot berdasarkan nilai similarity nama produk Jaccard

Pengukuran	**Nilai**
mean	0.966591
std	0.062118
min	0.804878
25%	0.952381
50%	1
75%	1
max	1

6.1.4.3 Jaro Winkler

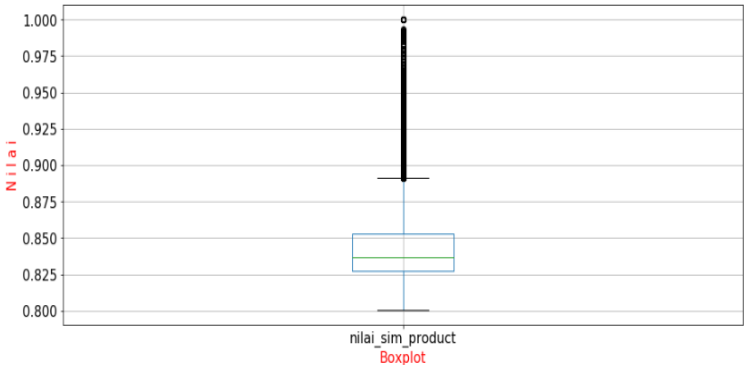


Gambar 6. 9 Boxplot pembobotan produk

Pada gambar 6.9 terdapat enam *boxplot* yang berbeda-beda. Penjelasan setiap *boxplot* dapat dilihat pada tabel berikut:

Tabel 6. 6 Tabel penjelasan boxplot berdasarkan pembobotan Jaro

	pdt0.6-brd0.4	**pdt0.4-brd0.6**	**pdt0.7-brd0.3**	**pdt0.3-brd0.7**	**pdt0.8-brd0.2**	**pdt0.2-brd0.8**
mean	0.702336	0.653207	0.726901	0.628643	0.751465	0.604078
std	0.046757	0.058219	0.060055	0.077543	0.079843	0.099922
min	0.329006	0.328	0.246754	0.246	0.164503	0.164
25%	0.680807	0.621354	0.714833	0.583811	0.753742	0.546201
50%	0.70768	0.655935	0.740861	0.624443	0.774123	0.592204
75%	0.731677	0.689328	0.761945	0.670444	0.792926	0.643976
max	0.929891	0.937333	0.947418	0.953	0.964946	0.968667



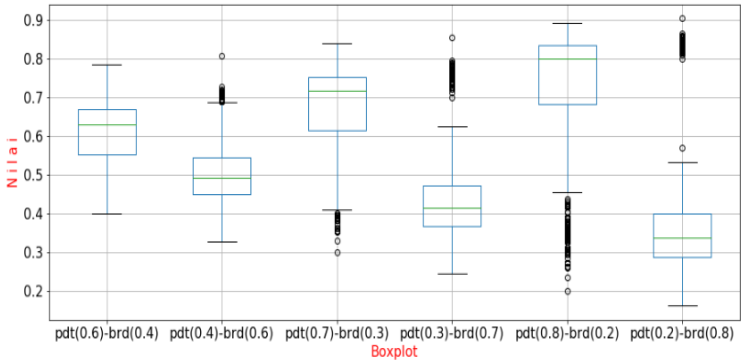
Gambar 6. 10 Boxplot nilai similarity nama produk

Pada gambar 6.10 hanya terdapat satu *boxplot.* Nilai yang dihitung hanya berdasarkan nilai *similarity* dari nama produk. Hal ini dikarenakan jika dari salah satu data tidak memiliki brand produk maka tidak efektif dilakukan pembobotan maka dari itu dilakukan penghitungan hanya berdasarkan *similarity* nama produk. Berikut penjelasan dari gambar 6.10:

Tabel 6. 7 tabel penjelasan boxplot berdasarkan nilai similarity nama produk Jaro

Pengukuran	**Nilai**
mean	0.844312
std	0.02559
min	0.800733
25%	0.827273
50%	0.836777
75%	0.852834
max	1

6.1.4.4 Levenshtein

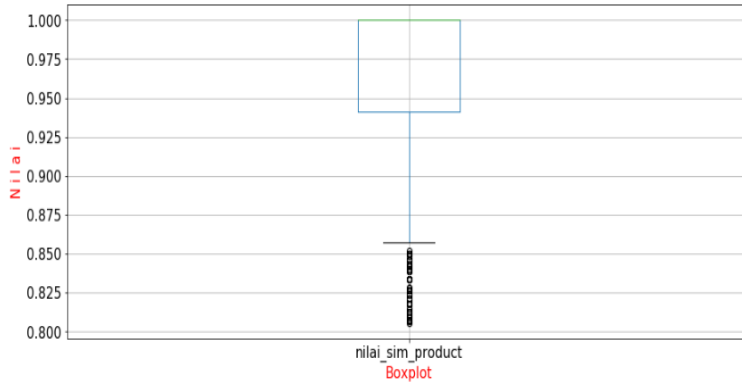


Gambar 6. 11 boxplot pembobotan produk

Pada gambar 6.11 terdapat enam *boxplot* yang berbeda-beda. Penjelasan setiap *boxplot* dapat dilihat pada tabel berikut:

Tabel 6. 8 Tabel penjelasan boxplot berdasarkan pembobotan Leven

	pdt0.6-brd0.4	**pdt0.4-brd0.6**	**pdt0.7-brd0.3**	**pdt0.3-brd0.7**	**pdt0.8-brd0.2**	**pdt0.2-brd0.8**
mean	0.612769	0.5128	0.662754	0.462815	0.712739	0.41283
std	0.070215	0.089369	0.11972	0.142991	0.176134	0.200483
min	0.4	0.327273	0.3	0.245455	0.2	0.163636
25%	0.552174	0.45	0.615584	0.368984	0.683208	0.289177
50%	0.62963	0.492308	0.717647	0.415686	0.8	0.33913
75%	0.670588	0.545455	0.752174	0.4725	0.834783	0.40069
max	0.785714	0.808	0.839286	0.856	0.892857	0.904



Gambar 6. 12 Boxplot nilai similarity nama produk

Pada gambar 6.12 hanya terdapat satu *boxplot.* Nilai yang dihitung hanya berdasarkan nilai *similarity* dari nama produk. Hal ini dikarenakan jika dari salah satu data tidak memiliki brand produk maka tidak efektif dilakukan pembobotan maka dari itu dilakukan penghitungan hanya berdasarkan *similarity* nama produk. Berikut penjelasan dari gambar 6.12:

Tabel 6. 9 tabel penjelasan boxplot berdasarkan nilai similarity nama produk Levenshtein

Pengukuran	**Nilai**
mean	0.958943
std	0.068304
min	0.805556
25%	0.941176
50%	1
75%	1
max	1

6.2 Pembahasan

Pada sub bab ini akan dijelaskan pembahasan mengenai seluruh gambar, tabel, dan grafik hasil output yang ada pada sub bab 6.1.

6.2.1 Jumlah Data yang dihasilkan setelah Pengukuran kesamaan

Jumlah data yang ditampilkan gambar 6.1 merupakan seluruh data yang telah berhasil dilakukan *similarity* dan telah di filter dengan nilai kesamaan lebih dari 0.8. Dari gambar 6.1 diketahui bahwa hasil dari similaritas Jaro jauh lebih besar dibandingkan dengan ketiga metode similaritas yang lain. Similaritas *Jaro* memiliki hasil produk sebanyak 315.354 data, hal ini disebabkan karena Jaro Winkler mengukur kesamaan dengan membandingkan panjang *string* sehingga, nilai yang dihasilkan tinggi. Sebuah produk memiliki nama yang panjang maka kemiripannya akan dihitung sepanjang string yang sama dan untuk kata setelahnya diabaikan. Sehingga Jaro winkler lebih cocok digunakan pada pengukuran nama produk yang singkat dengan satu atau dua kata saja.

Cosine Similarity juga memiliki hasil dengan nilai similaritas lebih besar dari 0.8. Dengan nilai similaritas tersebut menghasilkan data lebih banyak dibanding dengan metode *jacccard* dan *levenshtein*. Hal tersebut disebabkan karena *Cosine* menghitung atau membandingkan huruf yang muncul pada satu kalimat dengan kalimat yang satunya sehingga jika kalimat tersebut tidak muncul huruf yang sama maka dianggap "nol”.

Berbeda dengan Jaccard dimana seluruh karakter yang muncul pada dua buah string akan di lakukan *intersection* dan *union* lalu hasil *intersection* akan dibagi dengan hasil *union.*

Dan yang terakir yaitu Levenshtein, dimana jenis similaritas ini paling efektif karena metode ini membandingkan kedua string lalu menghitung jumlah *deletion,insertion,* dan *substitution* antara dua string tersebut.

6.2.2 Produk yang muncul pada setiap similartiy

Bagian ini menjelaskan mengenai produk yang muncul pada setiap perhitungan *similarity*.

6.2.2.1 Seluruh produk yang muncul pada setiap silimilarity

Setiap *similarity* menghasilkan nilai similaritas yang berbeda-beda pada setiap produk sehingga, ketika data produk dilakukan *filtering* dengan nilai *similarity* lebih besar dari 0.8 pada setiap *similarity* menghasikan jumlah produk yang berbeda-beda pula. Berikut jumlah produk yang muncul pada setiap *similarity* berdasarkan diagram:

e. Antar dua *similarity*
   - Cosine dan Jaccard = 2.559
   - Jaccard dan Jaro = 2.559
   - Jaro dan Leven = 2.684
   - Leven dan Cosine = 2.684
   - Jaro dan Cosine = 16.707
   - Leven dan Jaccard = 2.372

f. Antar tiga *similarity*
   - Cosine + Jaccard + Jaro = 2.559
   - Jaccard + Jaro + Leven = 2.372
   - Jaro + Leven + Cosine = 2.372

g. Seluruh *similarity*
   - Cosine + Jaccard + Jaro + Leven = 2.372

Jadi dari data diatas terdapat 2.372 produk yang sama muncul di setiap *similarity.*

6.2.2.2 Seluruh produk yang muncul pada setiap silimilarity yang bernilai satu

Sama halnya dengan pembahasan pada 6.2.2.1 yang membedakannya adalah nilai *similarity* nama produk. Pada bagian ini produk yang muncul hanya yang memiliki nilai

kesamaan satu (1) pada nama produk. Berikut jumlah produk yang sama dan muncul pada tiap *similarity* dengan nilai kesamaan satu pada nama produk:

a. Antar dua *similarity*
- Cosine dan Jaccard = 1.664
- Jaccard dan Jaro = 1.632
- Jaro dan Leven = 1.650
- Leven dan Cosine = 1.650
- Jaro dan Cosine = 1.650
- Leven dan Jaccard = 1.632

b. Antar tiga *similarity*
- Cosine + Jaccard + Jaro = 1.632
- Jaccard + Jaro + Leven = 1.632
- Jaro + Leven + Cosine = 1.650

c. Seluruh *similarity*
- Cosine + Jaccard + Jaro + Leven = 1.632

6.2.3 Perbandingan pada nilai similaritas untuk 10 produk acak

Pada tabel 6.1 dapat dilihat dari 10 data produk yang diambil secara acak terdapat satu data yang persis sama yaitu dengan nama produk "1% low fat milk". Untuk nama produk *1% low fat milk*, di setiap *similarity* memiliki nilai yang sama yaitu satu (1). Sehingga untuk akurasi ketepatan semuanya bagus dan benar yang membuat perbedaan hanya pada cara menghitung atau membandingkan kedua *string* tersebut. Sehingga pada beberapa produk memiliki nilai *similarity* berbeda-beda.

Grafik pada gambar 6.4 menjelaskan bahwa jaro winkler merupakan metode *similarity* yang nilainya paling besar sehingga, ada keterkaitan mengapa pada gambar 6.1 metode *jaro winkler* menghasilkan produk yang paling banyak dikarenakan, algoritma *jaro* menilai suatu produk yang mirip

dengan nilai yang tinggi sehingga menghasilkan banyak produk dibandingkan dengan metode lain. Lain halnya dengan *cosine* menilai produk yang memiliki kemiripan karakter dengan mengurutkan jumlah huruf yang sama pada satu data set sehingga, menghasilkan nilai yang juga tinggi. Untuk similaritas *jaccard* dan *levenshtein* dari gambar 6.4 memiliki nilai yang sama sehingga garis grafik yang muncul hanya satu untuk mewakili kedua similaritas tersebut.

6.2.4 Perbandingan berdasarkan pembobotan dan nilai similarity nama produk

Seperti penjelasan pada bagian 6.1.4 dimana data yang digunakan pada bagian ini dibagi dua maka pengamatan yang dilakukan untuk setiap metode *similarity* memiliki dua *boxplot.* Berikut penjelasan setiap metode *similarity*

6.2.4.1 Cosine Similarity

Pada metode *Cosine similarity* data pembobotan terdapat sebanyak 9.404 data produk sedangkan data yang tidak dilakukan pembobotan atau data yang hanya dihitung dari nilai similaritas nama produk terdapat sebanyak 7.368. Berikut beberapa pembahasan yaitu:

a. Bobot pdt(0.6)-brd(0.4) pada gambar 6.5

Pada Pembobotan ini terdapat

- Outlier.

  Pada pembobotan ini terdapat beberapa *outlier,* baik pada *wishker* nilai tertinggi pada batas atas, maupun *wishker* nilai terendah pada batas bawah. Pada *wishker* batas atas terdapat “95” produk yang berada diatas wishker tersebut sedangkan untuk wishker bawah terdapat “99” produk. Hal tersebut terjadi karena nilai produk-produk tersebut berada dibawah atau diatas nilai *wishker* yang ada pada boxplot. Total outlier yang ada yaitu sebanyak 194 jika dibandingkan dengan total data yaitu 9.404 maka hanya 2% data yang menjadi *outlier*. Meskipun begitu hal tersebut kurang baik karena dapat membuat data tidak berdistribusi normal

- Standar Deviasi, Median.

  Dari gambar 6.5 diketahui varian data yang ada sangat kecil dikarenakan standar deviasi yang rendah yaitu 0.08. Selain itu data sedikit miring atas karena *median* berada lebih dekat dengan quartil atas (Q3) di antara quartil bawah (Q1) dan quartil atas (Q3). Sehingga distribusi data cenderung kekiri (negative skewness). Hal ini juga disebabkan data *outlier* yang berada dibawah *boxplot* lebih banyak dibandingkan dengan yang atas.

b. *Bobot pdt(0.4)-brd(0.6) pada gambar 6.5.*

- Outlier.

  Pada pembobotan ini tidak terdapat *outlier* pada nilai maksimum akan tetapi terdapat *outlier* pada nilai minimum sebanyak 95 produk. Hal ini terjadi dikarenakan produk-produk tersebut bernilai dibawah batas nilai minimum dari *boxplot.* Jika dibandingkan dengan total data dari pembobotan yaitu 9.404 maka hanya sekitar 1% yang menjadi *outlier*. Meskipun begitu data belum dapat disebut berdistribusi normal karena data yang menjadi *outlier* semuanya berasal dari nilai minimum sebuah *boxplot* sehingga membuat data terdistribusi agak miring kekiri.

- Standar Deviasi, Median.

  Dari apa yang dilihat pada gambar 6.5 ***boxplot ke 2*** varian data lumayan besar dibandingkan dengan *boxplot* yang pertama pada gambar 6.5 tidak hanya itu nilai standar deviasi juga cukup besar yaitu 0.138. Median pada *boxplot* pada pembobotan ini miring keatas mendekati kuartil atas (Q3). Sehingga data dapat dikatakan berdistribusi kearah kiri (negative skewness). Hal ini juga disebabkan oleh banyaknya produk yang berada dibawah nilai minimum.

c. Bobot pdt(0.7)-brd(0.3) pada gambar 6.5.

- Outlier

  Pada pembobotan ini terdapat *outlier* pada nilai minimum dan nilai maksimum. Untuk *outlier* dibawah

nilai minimum sebanyak 95 data dan untuk *outlier* yang berada di atas nilai minimum sebanyak 95 data. Hal tersebut terjadi dikarenakan nilai-nilai dari pembobotan produk-produk tersebut berada diatas ataupun dibawah nilai maksimum dan minimum pada sebuah *boxplot.* dengan kata lain nilai-nilai tersebut berada diluar *wishker*. Jika dibandingkan dengan total data 9.404 produk maka, *outlier* yang berada diluar *wishker* totalnya sebesar 2% yang berada diluar *wishker*. oleh karena itu data ini juga belum dapat disebut terdistribusi normal.

- Standar Deviasi, Median.

  Pada pembobotan ini standar deviasi yang dimiliki lebih kecil dibandingkan dengan *boxplot* pada gambar 6.6 yaitu 0.076 dari sini diketahui bahwa data memiliki varian yang kecil. Dari gambar 6.5 *boxplot* ke 3 diketahui memiliki *median* yang berada tepat di tengah-tengah antara kuartil bawah (Q1) dan kuartil atas (Q3) sehingga data terdistribusi normal. Meskipun begitu dikarenakan *outlier* yang berada di luar *wishker* lebih dari 1% maka hal tersebut dapat mengganggu distribusi data

d. Bobot pdt(0.3)-brd(0.7) pada gambar 6.5.

- Outlier

  Outlier yang berada di luar *wishker* yaitu sebanyak 41 produk dimana seluruh produk tersebut berada dibawah nilai *wishker* bawah. Sehingga data tidak terdistribusi secara normal. Jika dibandingkan dengan total seluruh data memang berada dibawah 1% akan tetapi karena data *outlier* banyak yang berada dibawah maka, data terdistribusi tidak normal.

- Standar Deviasi, Median.

  Standar deviasi pada pembobotan ini termasuk besar sehingga varian data juga besar. Median pada pembobotan ini cenderung dekat ke kuartil atas (Q3) sehingga, data terdistribusi kearah kiri (negative skewness).

e. Bobot pdt(0.8)-brd(0.2) pada gambar 6.5.

- Outlier

  Outlier pada pembobotan ini hanya terdapat diluar *wishker* bawah yaitu sebanyak 28 produk atau sekitar 0.2% dari jumlah total data. Hal ini baik karena sangat kecil angka produk yang berada diluar *wishker*.
- Standar Deviasi, Median, Q1, dan Q3.

  Dilihat dari gambar 6.5 *boxplot* ke 5 bentuk kotaknya sedikit lebih kecil dibandingkan *botplox* ke empat pada gambar 6.5 akan tetapi, dengan standar deviasi yang lumayan besar yaitu 0.094 sehingga data memiliki varian yang juga lumayan besar. Median pada bagian ini berada mendekati kuartil atas (Q3) sehingga data yang ada pada pembobotan ini terdistribusi ke arah kiri (negative skewness).

f. Bobot pdt(0.2)-brd(0.8) pada gambar 6.5.

- Outlier

  Dari apa yang di amati dari gambar 6.5 *bot plox* ke enami tidak adanya *outlier* ataupun data yang berada diluar *wishker*. Meskipun hal ini bagus akan tetapi hal lain juga harus dipertimbangkan agar data terdistribusi normal seperti standar deviasi, Median, dan lainnya.
- Standar Deviasi, Median.

  Melihat standar deviasi pada bagian ini terlalu besar dibandingkan dengan pembobotan lainnya yaitu dengan nilai 0.217. oleh sebab itu data pada pembobotan ini memiliki varian yang besar. Untuk median berada sedikit dekat dengan kuartil atas (Q3) maka dapat dikatakan data terdistribusi kea rah kiri (negative skewness).

g. Nilai Similarity Nama Produk pada gambar 6.6.

- Outlier

  Pada gambar 6.6 data sedikit berbeda dimana bukan nilai pembobotan yang dihitung melainkan langsung nilai similaritas nama produk. *Outlier* yang ada pada data ini hanya berada di luar *wishker* atas yaitu

sebanyak 1.206 produk. Jumlah tersebut jika dibandingkan dengan total data yaitu 7.368 produk maka *outlier* yang diluar *wishker* sebesar 16% dan hal tersebut dapat mengganggu distribusi.

- Standar Deviasi, Median.
  Standar deviasi yang dimiliki bagian ini bernilai kecil yaitu 0.06 sehingga vairan data yang dimiliki juga kecil. Median yang dimiliki berada lebih dekat dengan kuartil bawah (Q1) hal ini juga berlawanan dengan gmambar 6.5 yang telah dijelaskan diatas. Oleh karena itu data berdistribusi ke arah kanan (positive skewness).

6.2.4.2 Jaccard Similarity

Pada Jaccard total data untuk pembobotan sebanyak 987 data produk dan untuk data yang tidak dilakukan pembobotan atau data yang hanya dihitung similaritas nama produk sebanyak 1.572 data produk. pada setiap similarity terdapat beberapa pembahasan yaitu:

a. Bobot pdt(0.6)-brd(0.4) pada gambar 6.7.
Pada Pembobotan ini terdapat:

- Outlier.
  Dibagian ini hanya terdapat satu *outlier* dimana produk tersebut berada diluar *wishker* bawah. Oleh karena itu dapat dipastikan *outlier* yang ada sudah pasti dibawah 1% dan hal ini tidak menganggu distribusi data.
- Standar Deviasi, Median.
  Untuk Standar deviasi pada bagian ini bernilai normal yaitu dengan nilai 0.089. Sehingga varian yang dimiliki data ini juga terbilang normal. Pada gambar 6.7 *boklot* pertama median berada sedikit dekat dengan kuartil atas (Q3) sehingga distribusi data kearah kiri (negative skewness).

b. Bobot pdt(0.4)-brd(0.6) pada gambar 6.7.

- Outlier.
  Pada bagian ini tidak ditemukannya *outlier* yang berarti semua produk berada dalam area *wishker*.
- Standar Deviasi, Median, Q1, dan Q3.

Standar deviasi yang dimiliki sedikit lebih besar dibandingkan dengan *boxplot* pertama pada gambar 6.7 yaitu sebesar 0.097 oleh karena itu varian data juga tidak besar. Median pada bagian ini condong ke arah kuartil bawah (Q1) sehingga dapat distribusi data kearah kanan (positive skewness).

c. Bobot pdt(0.7)-brd(0.3) pada gambar 6.7.

- Outlier

  Pada bagian ini *outlier* hanya ditemukan pada daerah luar *wishker* bawah sebanyak 12 produk. Jika dibandingkan dengan total data sebanyak 987 maka hanya sekitar 1,2% data yang berada di luar *wishker* walaupun sedikit tapi hal ini sudah dapa mengganggu distribusi data

- Standar Deviasi.

  Standar deviasi yang dimiliki yaitu sebesar 0.104 sedikit mirip dengan *boxplot* kedua pada gambar 6.7 sebelumnya akan tetapi sedikit lebih tinggi sehingga variannya juga masi terbilang tidak besar. Median pada data ini lebih dekat dengan kuartil atas (Q3) sehingga distribusi data kearah kiri (negative skewness).

d. Bobot pdt(0.3)-brd(0.7) pada gambar 6.7.

- Outlier

  Outlier yang didapat disini hanya ada dua produk dan kedua produk tersebut berada diluar *wishker* atas. Karena *outlier* dibawah 1% maka hal ini tidak terlalu mengganggu distribusi data.

- Standar Deviasi.

  Standar deviasi pada data ini sebesar 0.14 meskipun normal jika dibandingkan dengan *boxplot* ketiga pada gambar 6.7 sedikit lebih besar dan varian data pun sedikit lebih besar. Untuk median berada lebih dekat dengan kuartil bawah (Q1) maka data terdistribusi arah kanan (positive skewness).

e. Bobot pdt(0.8)-brd(0.2) pada gambar 6.7.

- Outlier

  Outlier yang ditemukan pada *boxplot* kelima pada gambar 6.7 berjumlah 75 produk. Semua *outlier* ditemukan di luar *wishker* bawah. Jika dilakukan prosentase maka terdapat 7.5% *outlier* pada data ini. Dengan angka sebesar itu sangat dapat mengganggu distribusi data. dan juga membuat distribusi berat sebelah.

- Standar Deviasi.

  Standar deviasi yang dimiliki sebesar 0.15 sedikit lebih besar dari *boxplot* keempat pada gambar 6.7 sehingga variannya juga terbilang sedikit lebih besar besar. Median yang dimiliki berada dekat dengan kuartil atas (Q3) sehingga data terdistribusi arah kiri (negative skewness). hal ini juga disebabkan banyak *outlier* yang berada dibawah *wishker* bawah.

f. Bobot pdt(0.2)-brd(0.8) pada gambar 6.7.

- Outlier

  Pada pagian ini ditemukan *outlier* yang terbillang cukup besar yaitu sebanyak 154 data. Seluruh *outlier* tersebut ditemukan di atas *wishker* atas. Hal ini dapat menganggu arah distribusi. Jika di presentasikan maka terdapat kurang lebih 15% *outlier* dan hal ini membuat distribusi data tidak normal karena jauh diatas 1% *outlier* yang terdapat pada data.

- Standar Deviasi.

  Standar deviasi yang dimiliki sebesar 0.19 dan angka tersebut merupakan yang paling besar diantara seluruh *boxplot* pada gambar 6.7 sehingga, variannya juga lebih besar dari yang lain. Median yang dimiliki berada didekat kuartil bawah (Q1) yang berarti distribusi data arah kanan (positive skewness).

g. Nilai Similarity Nama Produk pada gambar 6.8.

- Outlier

  Pada bagian ini juga ditemukan sangat banyak *outlier* yaitu sebanyak 222 produk. Seluruh produk tersebut

berada di luar *wishker* bawah. Jika dibandingkan dengan total data sebanyak 1.572 maka terdapat setidaknya 14% data yang menjadi *outlier*. Dengan angka sebesar itu sangat menganggu arah distribusi data.

- Standar Deviasi.
  Standar Deviasi yang dimiliki yaitu 0.06 dimana kita tahu bahwa angka tersebut kecil sehingga, data bagian ini terdapat varian yang kecil. Median pada data bagian ini sama dengan kuartil atas sehingga sudah pasti data terdistribusi ke arah kiri (negative skewness).

6.2.4.3 Jaro Winkler Similarity

Pada penghitungan similaritas *jaro* untuk bagian pembobotan terdapat 142.028 produk. Sedangkan untuk bagian similaritas nama produk terdapat 172.326 produk. Jumlah produk-produk tersebut merupakan yang terbesar diantara seluruh similaritas yang dilakukan.

a. Bobot pdt(0.6)-brd(0.4) pada gambar 6.9.

- Outlier.
  Outlier yang ditemukan disini lebih banyak dari seluruh *boxplot* yang telah dijelaskan dikarenakan produk hasil dari similaritas *jaro* juga sangat banyak jumlahnya. *Outlier* yang terdapat pada bagian ini sebanyak total 6.049 produk dimana 5.566 ditemukan diluar *wishker* bawah dan 483 didapat dari luar *wishker* atas. Sehingga total *oulier* yang ada yaitu 4.25% dari seluruh data, dan tentu saja hal ini berdampak kepada distribusi data yang tidak merata.
- Standar Deviasi.
  Pada bagian data ini terdapat standar deviasi yang sangat kecil yaitu 0.06 sehingga varian data yang ada kecil. Median pada *boxplot* pertama pada gambar 6.9 dekat dengan kuartil atas (Q3) sehingga distribusi data kearah kiri (negative skewness).

b. *Bobot pdt(0.4)-brd(0.6) pada gambar 6.9.*
   - Outlier.

     Outlier yang ada pada bagian ini sebanyak 2.308 produk dimana 1.897 produk berada di luar *wishker* bawah dan 411 produk berada di luar *wishker* atas. Total seluruh *outlier* yaitu 1.6% dari keseluruhan total produk yang terdapat pada data bagian ini.
   - Standar Deviasi.

     Standar deviasi yang dimiliki pun kecil yaitu 0.07 sehingga memiliki varian data yang juga kecil. Median yang dimiliki pada bagian ini juga terlihat dekat dengan kuartil atas sehingga distribusi data ke arah kiri (negative skewness).

c. *Bobot pdt(0.7)-brd(0.3) pada gambar 6.9.*
   - Outlier

     Total *outlier* ditemukan yaitu sebanyakb 15.159 produk. 14.700 produk berada di luar *wishker* bawah dan 459 produk berada di luar *wishker* atas. Sehingga total *outlier* yang ada sebesar 10% dari total produk keseluruhan. Hal ini mengganggu distribusi data tidak normal
   - Standar Deviasi.

     Sandar deviasi yang dimiliki yaitu sebesar 0.04 dimana nilai tersebut sangat kecil sehingga varian data juga kecil. Median yang terdapat pada *boxplot* ini juga agak sedikit lebih dekat dengan kuartil atas (Q3) sehingga distribusi data ke arah kiri (negative skewness).

d. *Bobot pdt(0.3)-brd(0.7) pada gambar 6.9.*
   - Outlier.

     Terdapat 1.480 *outlier* diluar *wishker* bawah dan 599 *outlier* diluar *wishker* atas. Total yaitu 2.079 produk yang berada diluar *wishker* atau sebesar 1.4% dari keseluruhan total produk. Meskipun hanya sedikit tetapi jika telah melebihi 1% *outlier* yang ada, hal tersebut dapat mengganggu distribusi data.
   - Standar Deviasi.

Standar deviasi yang dimiliki yaitu 0.07 nilai ini mirip dengan beberapa *boxplot* sebelumnya dimana memiliki varian data tidak besar. median juga sedikit lebih dekat dengan kuartil atas (Q3) sehingga data terdistribusi arah kiri (negative skewness).

e. Bobot pdt(0.8)-brd(0.2) pada gambar 6.9.

- Outlier

  Disini *outlier* ditemukan sebanyak 22.846 dimana 22.014 berada diluar *wishker* bawah dan 832 berada diluar *wishker* sehingga total dari seluruh produk yang ada yaitu sebesar 16%. Hal itu menjadikan distribusi data tidak bagus.

- Standar Deviasi.

  Untuk standar deviasi yaitu sebesar 0.07 sehingga varian juga tidak besar. Letak median dekat dengan kuartil atas (Q3) sehingga distribusi data ke arah kiri (negative skewness).

f. Bobot pdt(0.2)-brd(0.8) pada gambar 6.9.

- Outlier

  Outlier pada bagian ini berjumlah 7892 produk yang berada diluar *wishker* masing-masing 1583 berada diluar *wishker* bawah dan 6.303 berada di luar *wishker* atas. Sehingga besar *oulier* dari keseluruhan produk yaitu 5.5%. *outlier* sebesar itu tidak bagus dalam sebuah data.

- Standar Deviasi.

  Standar deviasi yang dimiliki sebesar 0.1 dmana termasuk kategori tidak besar sehingga vairan data pun tidak terlalu besar. Median pada data ini berada dekat dengan kuartil bawah (Q1) sehingga distribusi data kearah kanan (positive skewness).

g. Nilai Similarity Nama Produk pada gambar 6.10.

- Outlier

  Pada bagian ini ditemukannya *outlier* sebanyak 10.788 produk dan semuanya berasal dari luar *wishker* atas. Besar *outlier* yaitu 7% dari keseluruhan total data. Sehingga hal ini tidak bagus dalam sebuah data

- Standar Deviasi.

  Standar deviasi yang dimiliki cukup kecil yaitu sebesar 0.02 sehingga varian data pun sangat kecil. Median pada data ini terletak dengan kuartil bawah (Q1) yang berarti distribusi data arah kanan (positive skewness).

6.2.4.4 Levenshtein Similarity

Pada perhitungan similaritas *Levenshtein* data produk untuk pembobotan terdapat sebanyak 1.061 produk sedangkan yang untuk hanya similaritas nama produk sebanyak 1.624 produk.

a. Bobot pdt(0.6)-brd(0.4) pada gambar 6.11.

- Outlier

  Pada bagian data ini tidak terlihat satupun *outlier* pada *boxplot* baik pada *wishker* atas maupun bawah
- Standar Deviasi.

  Standar deviasi yang dimiliki yaitu 0.07 dan nilai tersebut termasuk kecil sehingga varian data juga kecil. Median terletak lebih dekat dengan kuartil atas (Q3) dibandingkan dengan kuartil bawah (Q1) sehingga distribusi data arah kiri (negative skewness).

b. Bobot pdt(0.4)-brd(0.6) pada gambar 6.11.

- Outlier.

  Pada bagian ini *outlier* hanya ditemukan diluar *wishker* atas yaitu sebanyak 48 produk. Jika dibandingkan dengan total produk maka sekitar 4% produk yang berada diluar *wishker* dan hal itu menganggu distribusi produk.
- Standar Deviasi.

  Pada bagian data ini standar deviasi bernilai 0.089 nilai tersebut sedikit besar dibandingkan dengan *boxplot* sebelumnya. Meskipun begitu varian yang ada masih terbilang kecil. Median pada data ini terletak dekat dengan kuartil bawah (Q1) sehingga distribusi data arah kanan (positive skewness).

c. Bobot pdt(0.7)-brd(0.3) pada gambar 6.11.

- Outlier

Outlier pada data ini hanya ditemukan pada *wishker* bawah yaitu sebanyak 48 data produk. Dimana sekitar 4% dari total kesuluruhan produk.

- Standar Deviasi.

  Standar deviasinya yaitu 0,12 masi dalam kategori kecil oleh sebab itu pula varian data yang dimiliki juga kecil. Median pada data ini terletak dekat dengan kuartil atas (Q3) sehingga data terdistribusi arah kiri (negative skewness).

d. Bobot pdt(0.3)-brd(0.7) pada gambar 6.11.

- Outlier.

  Outlier hanya ditemukan diluar *wishker* atas yaitu sebanyak 181 produk. Dimana sekitar 17% dari total jumlah produk. Yang berarti hal ini tidak baik dalam data.

- Standar Deviasi.

  Standar deviasi yang dimiliki yaitu sebesar 0.14 sehingga varian data yang dimiliki kecil. Letak median pada data ini mendekati kuartil bawah (Q1) yang berarti distribusi data arah kanan (positive skewness).

e. Bobot pdt(0.8)-brd(0.2) pada gambar 6.11.

- Outlier

  Pada bagian ini *outlier* ditemukan hanya pada *wishker* bawah yaitu sebanyak 179 produk. Sebesar kurang lebih 16% dari data yang ada. Hal tersebut dapat mempengaruhi distribusi data.

- Standar Deviasi.

  Standar deviasi yang dimiliki yaitu sebesar 0.18 masi dalam kategori kecil sehingga varian data juga kecil. Letak median berada dekat dengan kuartil atas (Q3) yang berarti distribusi data arah kiri (negative skewness).

f. Bobot pdt(0.2)-brd(0.8) pada gambar 6.11.

- Outlier

  Outlier ditemukan sebanyak 184 produk dimana seluruhnya berada diluar *wishker* atas. Besar *outlier*

yaitu 17% dari total keseluruhan data. Hal ini juga membuat distribusi data tidak normal.

- Standar Deviasi.

  Standar deviasi yang dimiliki yaitu sebesar 0.2 dimana angka ini terbesar diantara seluruh data pembobotan meskipun begitu masih terbilang kecil sehingga varian data juga kecil. Letak median lebih dekat dengan kuartil bawah sehingga data terdistribusi arah kanan (positive skewness).

g. Nilai Similarity Nama Produk pada gambar 6.12.

- Outlier

  Outlier ditemukan sebanyak 383 produk dimana seluruhnya berada diluar *wishker* bawah. Jika dibandingkan dengan total produk nilai similaritas nama produk maka terdapat 23% *outlier* dan hal tersebut mempengaruhi distribusi data.

- Standar Deviasi.

  Standar deviasi yang dimiliki data yaitu sebesar 0.29 dan ini sudah sedikit lebih besar dibandingkan dengan yang lainnya sehingga varian data juga sedikit besar. Letak median pada data ini sama dengan kuartil atas (Q3) yang berarti data terdistribusi arah kiri (negative skewness).

Halaman sengaja dikosongkan

BAB VII
KESIMPULAN DAN SARAN

Bab ini menjelaskan kesimpulan yang didapat dari seluruh proses pengerjaan tugas akhir dan saran untuk penelitian kedepannya untuk dapat dikembangkan dari tugas akhir ini.

7.1 Kesimpulan

Berikut adalah kesimpulan dari keseluruhan penelitian yang telah dilakukan oleh peneliti:

1. Total data yang diperoleh dari hasil *crawling* dari berbagai lembaga sertifikasi yaitu sebanyak 10.153 data produk.
2. Seluruh data yang telah dikumpulkan dari tujuh lembaga sertifikasi sebanyak 11,3% memiliki kesamaan dengan data yang sudah ada pada basis data Halal Nutrition Food.
3. Metode *similarity* yang terbaik menurut analisis yang telah dilakukan dengan mempertimbangkan nilai similaritas dan jumlah produk yang dihasilkan adalah metode *levenshtein similarity* dengan jumlah produk sebanyak 2.685 produk dari total 10.153 data. Metode *levenshtein* merupakan metode yang paling sensitif dalam pengukuran kesamaan produk.

7.2 Saran

Penelitian ini hanya dilakukan untuk integrasi data Halal Food Nutrition dengan menggunakan empat metode similartitas. Penelitian ini dapat dikembangkan dengan melakukan pengelompokkan data menggunakan *machine learning*.

Halaman sengaja dikosongkan

DAFTAR PUSTAKA

[1] A. Indrawan, “Inilah 10 Negara dengan Populasi Muslim Terbesar di Dunia _ Republika Online.” 2015.

[2] N. Tashandra, “Media Sosial, Penyebaran "Hoax", dan Budaya Berbagi... - Kompas.com,” 2017. [Online]. Available: https://nasional.kompas.com/read/2017/02/14/09055481/media.sosial.penyebaran.hoax.dan.budaya.berbagi. [Accessed: 12-Sep-2018].

[3] Kementerian Komunikasi dan Informatika RI, “Kementerian Komunikasi dan Informatika,” *Government Web*, 2013. [Online]. Available: https://kominfo.go.id/index.php/content/detail/8904/melawan-hoax/0/sorotan_media. [Accessed: 12-Sep-2018].

[4] A. M. Fajriya, “Rancang Bangun Sistem Pencarian Produk Halal Dalam Aplikasi Halal Nutrition Food Menggunakan Algoritma Okapi BM25F,” Jul. 2017.

[5] K. Mccormack and M. Smyth, “A Mathematical Solution to String Matching for Big Data Linking,” vol. 5, pp. 39–55, 2017.

[6] A. A. Firmansyah, “Pengembangan Pencarian Produk Terkait Menggunakan Euclidean Distance Dan Cosine Similarity Pada Aplikasi Halal Nutrition Food,” 2018.

[7] y. fajar, “integrasi data dan visualisasi graf pada aplikasi halal nutrition food data integration and graph visualization in halal nutrition food application,” 2018.

[8] A. Halim *et al.*, “Perancangan Aplikasi Web Crawler Untuk Menghasilkan Dokumen Teks Pada Domain,” vol. 1, no. 2, pp. 2–5, 2017.

[9] D. Gunawan, Amalia, and A. Najwan, “Improving Data Collection on Article Clustering by Using Distributed Focused Crawler,” *J. Comput. Appl. Informatics*, vol. 1, no. 1, pp. 39–50, 2017.

[10] V. Shkapenyuk and T. Suel, “Design and implementation of a high-performance distributed web crawler,” *Proc. - Int. Conf. Data Eng.*, pp. 357–368, 2002.

[11] R. Darwis, R. Resnawaty, M. Irfan, and A. Risman, “Institusi Lokal Dalam Kegiatan Pengembangan Masyarakat: Kasus Punggawa Ratu Pasundan Dalam Program Desa Wisata Di Desa …,” *Share Soc. Work J.*, vol. 0042, 2016.

[12] f. t. informasi, rancang bangun aplikasi android halal nutrition food menggunakan kombinasi query-independent dan query-dependent ranking building android halal nutrition food appli- cation using combined query-independent and query-dependent ranking. 2017.

[13] “Scrapy 1.6 documentation — Scrapy 1.6.0 documentation.” [Online]. Available: https://docs.scrapy.org/en/latest/. [Accessed: 14-Feb-2019].

[14] O. Nurdiana, J. Jumadi, and D. Nursantika, “Perbandingan Metode Cosine Similarity Dengan Metode Jaccard Similarity Pada Aplikasi Pencarian Terjemah Al-Qur’an Dalam Bahasa Indonesia,” *J. Online Inform.*, vol. 1, no. 1, p. 59, 2016.

[15] F. Rahutomo, T. Kitasuka, and M. Aritsugi, “Semantic Cosine Similarity,” *Semant. Sch.*, vol. 2, no. 4, pp. 4–5, 2012.

[16] C. Luo, J. Zhan, X. Xue, L. Wang, R. Ren, and Q. Yang, “Cosine normalization: Using cosine similarity instead of dot product in neural networks,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 11139 LNCS, pp. 382–391, 2018.

[17] A. Prasetyo, W. M. Baihaqi, and I. S. Had, “Algoritma

Jaro-Winkler Distance: Fitur Autocorrect dan Spelling Suggestion pada Penulisan Naskah Bahasa Indonesia di BMS TV," *J. Teknol. Inf. dan Ilmu Komput.*, vol. 5, no. 4, p. 435, 2018.

[18] F. Okta'mal, R. Saptono, and M. E. Sulistyo, "Jaro-Winkler Distance Dan Stemming Untuk Deteksi," *Semin. Nas. Sist. Inf. Indones.*, no. March, pp. 305–312, 2015.

[19] H. Chen, "String Metrics and Word Similarity applied to Information Retrieval," 2012.

[20] A. Fait and A. R. Fernie, "Data integration," *Plant Metab. Networks*, no. June, pp. 151–171, 2009.

Halaman sengaja dikosongkan

BIODATA PENULIS

Alkautsar, lahir di Pidie pada 18 februari 1997. Penulis lulus dari SMAN Modal Bangsa Aceh pada tahun 2015 dan melanjutkan studi di Departemen Sistem Informasi Institut Teknologi sepuluh Nopember. Selama kuliah penulis aktif di organisasi Pelajar Mahasiswa Kekeluargaan Tanah Rencong dan sempat menjabat sebagai salah satu ketua departemen keanggotaan, penulis juga aktif di Kajian Islam Sisten Informasi (KISI) dan sempat menjabat sebagai ketua departemen mentoring. Penulis memiliki ketertarikan di bidang pemrograman dan data science. Dalam rangka menyelesaikan program sarjana penulis memilih bidang minat Akuisisi Data dan Diseminasi Informasi (ADDI) dan topik penelitian seputar *Data Integration* untuk meningkatkan pengetahuan di bidang *data science*. Penulis dapat dihubungi melalui email alkautsr15@mhs.is.its.ac.id.