

**RANCANG BANGUN SISTEM *DATA CLEANING* UNTUK
MASTER DATA KONSUMEN DI PT XYZ DENGAN
MENERAPKAN METODE *SORTED NEIGHBOURHOOD* DAN
METODE *N-GRAM***

TUGAS AKHIR



**RAHMA MUALIFA
1112001011**

**PROGRAM STUDI INFORMATIKA
FAKULTAS TEKNIK DAN ILMU KOMPUTER
UNIVERSITAS BAKRIE
JAKARTA
2016**

**RANCANG BANGUN SISTEM *DATA CLEANING* UNTUK
MASTER DATA KONSUMEN DI PT XYZ DENGAN
MENERAPKAN METODE *SORTED NEIGHBOURHOOD* DAN
METODE *N-GRAM***

TUGAS AKHIR

**Diajukan sebagai salah satu syarat untuk memperoleh gelar Sarjana
Komputer**



**RAHMA MUALIFA
1112001011**

**PROGRAM STUDI INFORMATIKA
FAKULTAS TEKNIK DAN ILMU KOMPUTER
UNIVERSITAS BAKRIE
JAKARTA
2016**

HALAMAN PERNYATAAN ORISINALITAS

**Tugas akhir ini adalah hasil karya saya sendiri,
dan semua sumber baik yang dikutip maupun dirujuk
telah saya nyatakan dengan benar.**

Nama : Rahma Mualifa

NIM : 1112001011

Tanda Tangan :

Tanggal 21 Juni 2016

HALAMAN PENGESAHAN

Tugas Akhir ini diajukan oleh:

Nama : Rahma Mualifa
NIM : 1112001011
Program Studi : Informatika
Fakultas : Teknik dan Ilmu Komputer
Judul Skripsi : Rancang Bangun Sistem *Data Cleaning* Untuk
Master Data Konsumen di PT XYZ Dengan
Menerapkan Metode *Sorted Neighbourhood* dan
N-Gram

Telah berhasil dipertahankan di hadapan Dewan Penguji dan diterima sebagai bagian persyaratan yang diperlukan untuk memperoleh gelar Sarjana Komputer pada Program Informatika Fakultas Teknik dan Ilmu Komputer, Universitas Bakrie

DEWAN PENGUJI

Pembimbing : Yusuf Lestanto, S.T., M.Sc. (.....)

Penguji 1 : Guson Kuntarto, (.....)

Penguji 2 : (.....)

Ditetapkan di : Jakarta

Tanggal : Juni 2016

UNGKAPAN TERIMA KASIH

Puji dan syukur kehadirat Allah SWT karena atas rahmat-Nya dan karunia-Nya sehingga penulis dapat menyelesaikan Tugas Akhir ini dengan baik. Tugas Akhir dengan judul “Rancang Bangun Sistem *Data Cleaning* Untuk Master Data Konsumen di PT XYZ Dengan Menerapkan Metode *Sorted Neighbourhood* dan Metode *N-Gram*” ini ditulis untuk memenuhi salah satu syarat dalam menyelesaikan perkuliahan pendidikan strata satu (S1) pada Program Studi Informatika, Universitas Bakrie.

Banyak pihak yang telah membantu penulis dalam penelitian dan penulisan Tugas Akhir ini, baik itu berupa bimbingan, saran, maupun dukungan secara moril dan materil. Oleh karena itu, pada kesempatan ini penulis ingin menyampaikan rasa terima kasih dan penghargaan yang setinggi-tingginya kepada:

1. Hoga Saragih, S.T., M.T., selaku Kepala Program Studi Informatika, yang senantiasa memberikan masukan dan motivasi kepada penulis;
2. Yusuf Lestanto, S.T., M.Sc., selaku dosen pembimbing, yang telah meluangkan waktunya serta memberikan bimbingan, saran, dan perbaikan dalam menyelesaikan penelitian ini;
3. Guson P. Kuntarto, S.T., M.Sc., selaku dosen pembahas dan penguji yang memberikan saran dan perbaikan terhadap penelitian ini;
4. Seluruh Bapak/Ibu dosen Program Studi Informatika UB, yang telah memberikan banyak ilmu, pengetahuan, wawasan kepada penulis selama perkuliahan;
5. Keluarga tercinta, kedua Orang tua penulis (Ariyanto dan Siti Rohmatun) dan saudara kandung penulis (Rahma Maulida dan Rahman Abid) yang telah memberikan dukungan dan doa yang sangat berarti bagi penulis.;

6. Tim Magang (Mario Joel, Aulia Syarifuddin, Mega Silviana, Celina Maya Cantika, M. Fadil). Terima kasih telah memberikan semangat, motivasi, dukungan, suka cita dan kebersamaan selama ini;
7. Teman VMG (Destalia Dianing Putri, Septy Dwi Aryani, Eka Juniar, Andining Tyas, Diti Puspa Permata). Terima kasih telah memberikan semangat, motivasi, dukungan, suka cita dan kebersamaan selama ini;
8. Tim Seperjuangan (Sawitri Sadanti, Stefanny Uliarta, Sarah Putri, Rien Pratama, Rahmad Pratama Dita, Fazz Faidurrahman, Evi Margaretha, Rizky Akbarie, Chandra Setiawan). Terima kasih telah menjadi teman yang selalu memberikan semangat, motivasi, dukungan dan suka cita selama lebih dari 4 tahun.
9. Teman-teman TIF 2011 senasib seperjuangan. Terima kasih sudah menemani dan bekerja sama selama lebih dari 4 tahun masa studi di UB;
10. Seluruh pihak yang terlibat dalam penyusunan Tugas Akhir ini yang tidak dapat penulis sebutkan satu persatu;

Dengan segala keterbatasan yang ada, penulis menyadari bahwa penyusunan tugas akhir ini masih jauh dari kesempurnaan. Untuk itu, saran dan kritik akan selalu diterima agar penulis dapat memperbaiki setiap kekurangan untuk kesempurnaan dimasa mendatang.

Akhirnya, penulis menyampaikan ucapan terima kasih dan semoga Allah SWT membalas segala kebaikan serta melimpahkan berkat dan rahmat-Nya kepada semua pihak yang telah membantu selama ini. Penulis berharap semoga Tugas Akhir ini berguna dan bermanfaat bagi kita semua.

Jakarta, 20 Juni 2016

Rahma Mualifa

HALAMAN PERNYATAAN PERSETUJUAN PUBLIKASI

Sebagai sivitas akademik Universitas Bakrie, saya yang bertanda tangan di bawah ini:

Nama : Rahma Mualifa
NIM : 1112001011
Program Studi : Informatika
Fakultas : Teknik dan Ilmu Komputer
Jenis Tugas Akhir : Rancang Bangun

Demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Universitas Bakrie **Hak Bebas Royalti Noneksklusif** (*Non-exclusive Royalty-Free Right*) atas karya ilmiah saya yang berjudul:

Rancang Bangun Sistem *Data Cleaning* Untuk Master Data Konsumen di PT XYZ Dengan Menerapkan Metode *Sorted Neighbourhood* dan *N-Gram*

beserta perangkat yang ada (jika diperlukan). Dengan Hak Bebas Royalti Noneksklusif ini Universitas Bakrie berhak menyimpan, mengalih media/formatkan, mengelola dalam bentuk pangkalan data (*database*), merawat, dan mempublikasikan tugas akhir saya selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai pemilik Hak Cipta untuk kepentingan akademis.

Demikian pernyataan ini saya buat dengan sebenarnya.

Dibuat di : Jakarta

Pada tanggal : 25 Juli 2016

Yang menyatakan

Rahma Mualifa

Rancang Bangun Sistem *Data Cleaning* Untuk Master Data Konsumen di PT XYZ Dengan Menerapkan Metode *Sorted Neighbourhood* dan *N-Gram*

Rahma Mualifa

ABSTRAK

Penelitian ini membahas tentang rancang bangun sistem *data cleaning* untuk dapat mendeteksi duplikasi data yang ada pada master data konsumen Divisi *Consumer Care* PT XYZ. Metode yang digunakan dalam penelitian ini untuk mendeteksi duplikasi data adalah dengan menerapkan pendekatan metode *Sorted Neighbourhood* (SNM) dan *N-Gram*. Sistem *data cleaning* ini bertujuan membantu *user* untuk dapat mempermudah menemukan duplikasi data. Selain itu, sistem ini juga dapat membantu *user* untuk dapat merapikan format penulisan telepon dan fax yang ada pada master data konsumen Divisi *Consumer Care* PT XYZ. Sistem yang akan dibangun adalah sistem *web based* dengan menggunakan bahasa pemrograman C#. Hasil dari sistem *data cleaning* yang dibangun kemudian akan diuji coba kepada *user* dan dinilai seberapa efektif metode SNM dan *N-Gram* dalam mendeteksi duplikasi data dengan menghitung nilai *recall* dan *precision* terhadap hasil proses deteksi duplikasi data.

Kata kunci: *Data cleaning*, Deteksi Duplikasi Data, *Sorted Neighbourhood*, *N-gram*

Rancang Bangun Sistem *Data Cleaning* Untuk Master Data Konsumen di PT XYZ Dengan Menerapkan Metode *Sorted Neighbourhood* dan *N-Gram*

Rahma Mualifa

ABSTRACT

This research discusses data cleaning system to detect data duplication are exists in customer master data at Consumer Care Division PT XYZ. Method will be used in this research to detect data duplication is by implementing Sorted Neighbourhood Method (SNM) and N-Gram. This system aims to assist user to find duplicate data easier. Moreover, it also can assist user to fix the format number both phone and fax number within customer master data at Consumer Care Division PT XYZ. System will be developed as web-based system by using C# programming language. The result of this system development will be tested by user and rated its effectiveness through implementation of SNM and N-Gram. To compute effectiveness will be obtained recall and precision value to determine how effective this system in detecting duplication data.

Kata kunci: *Data cleaning, Duplicate Detection, Sorted Neighbourhood, N-gram*

DAFTAR ISI

HALAMAN PERNYATAAN ORISINALITAS.....	i
HALAMAN PENGESAHAN.....	ii
UNGKAPAN TERIMA KASIH.....	iii
ABSTRAK	vi
ABSTRACT	vii
DAFTAR ISI.....	viii
DAFTAR GAMBAR	xi
DAFTAR TABEL.....	xiii
DAFTAR RUMUS	xv
DAFTAR LAMPIRAN	xvi
DAFTAR SINGKATAN	xvii
1 Pendahuluan	1
1.1 Latar Belakang Masalah	1
1.2 Perumusan Masalah.....	3
1.3 Tujuan Penelitian.....	3
1.4 Manfaat Penelitian.....	3
1.5 Batasan Masalah.....	4
1.6 Sistematika Penulisan	4
2 Tinjauan Pustaka	6
2.1 Penelitian Terkait.....	6
2.2 <i>Data Cleaning</i>	8
2.3 Metode <i>Data Cleaning</i>	9
2.3.1 Algoritma Deteksi Duplikasi Data	10
2.3.2 Metode <i>Sorted Neighbourhood</i> Sebagai Metode Untuk Deteksi Duplikasi Data	11
2.3.3 Algoritma Perhitungan Kemiripan Antar String	12
2.3.4 Algoritma Pendekatan <i>N-Gram</i> Sebagai Algoritma Perhitungan Kemiripan Antar <i>String</i>	13
2.4 Pemrograman Berorientasi Objek	15

2.5	<i>Unified Modelling Language (UML)</i>	16
3	Metodologi Penelitian	20
3.1	Tahap Identifikasi Masalah	20
3.1.1	Prosedur Yang Sedang Berjalan.....	21
3.1.2	Master Data Konsumen Divisi <i>Consumer Care</i> di PT XYZ..	22
3.1.3	Struktur Organisasi.....	24
3.1.4	Bisnis Proses.....	26
3.1.5	Sistem <i>Data Cleaning</i> Yang Diajukan	27
3.2	Tahap Analisa Kebutuhan Sistem.....	28
3.2.1	Kebutuhan Non Fungsional Sistem.....	28
3.2.2	Analisa Kebutuhan Fungsional Sistem.....	28
3.3	Perancangan Sistem.....	28
3.4	Tahap Implementasi	29
3.5	Tahap Pengujian	29
4	Implementasi dan Pembahasan	30
4.1	Perancangan Sistem.....	30
4.1.1	Perancangan Alur Algoritma Deteksi Duplikasi Data	30
4.1.2	Perancangan <i>Database</i>	40
4.1.3	UML (<i>Unified Modelling Language</i>).....	41
4.1.3.1	<i>Use Case Diagram</i>	41
4.1.3.2	<i>Activity Diagram</i>	47
4.1.3.3	<i>Class Diagram</i>	51
4.2	Implementasi	52
4.2.3	Implementasi Sistem	52
4.2.4	Implementasi GUI (<i>Graphical User Interface</i>).....	53
4.2.5	Implementasi Algoritma.....	58
4.2.5.1	Implementasi Algoritma SNM.....	58
4.2.5.2	Implementasi Algoritma <i>N-Gram</i>	60
4.3	Pengujian	62
4.3.1	Pengujian <i>White Box</i>	62
4.3.1.1	Pengujian Algoritma SNM	63

4.3.1.2	Pengujian Algoritma N-Gram.....	67
4.3.2	Pengujian <i>Black Box</i>	72
4.3.2.1	Pengujian <i>Functionality</i>	72
4.3.2.2	Pengujian <i>Usability</i>	74
4.3.2.3	Pengujian <i>Compatibility</i>	75
4.3.2.4	Pengujian <i>Performance</i>	76
4.3.3	Evaluasi Data.....	77
5	Simpulan dan Saran.....	83
5.1	Simpulan.....	83
5.2	Saran	84
	Daftar Pustaka	86
	Lampiran 1 – Wawancara	90
	Lampiran 2 – <i>Requirement Elicitation</i>	94
	Lampiran 3 – <i>Software Requirement Specification</i>	98
	Lampiran 4 – <i>Template Import Data</i>	129
	Lampiran 5 – Hasil Pengujian Sistem.....	130

DAFTAR GAMBAR

Gambar 3.1 Struktur Organisasi Divisi <i>Consumer Care</i> PT XYZ.....	24
Gambar 3.2 Gambar Bisnis Proses Deteksi Data Kembar Pada Master Data Konsumen Divisi <i>Consumer Care</i> PT XYZ.....	27
Gambar 4.1 <i>Flowchart</i> Algoritma Deteksi Duplikasi Data	31
Gambar 4.2 <i>Flowchart</i> Tahap Pra-cleaning.....	34
Gambar 4.3 <i>Flowchart</i> Tahap Tokenisasi.....	36
Gambar 4.4 <i>Flowchart</i> Tahap Pemecahan Kata Berdasarkan Nilai N-Gram	38
Gambar 4.5 <i>Flowchart</i> Perhitungan Nilai Kemiripan Antar <i>Record</i>	39
Gambar 4.6 Rancangan <i>Database</i> Sistem <i>Data Cleaning</i> Pada <i>Database</i> Master Data Konsumen PT XYZ.....	41
Gambar 4.7 <i>Use Case</i> Sistem <i>Data Cleaning</i>	42
Gambar 4.8 <i>Activity Diagram</i> Login.....	47
Gambar 4.9 <i>Activity Diagram</i> <i>Detection Duplication</i>	48
Gambar 4.10 <i>Activity Diagram</i> <i>Import Data</i>	48
Gambar 4.11 <i>Activity Diagram</i> <i>View Duplicate Detection Result</i>	49
Gambar 4.12 <i>Activity Diagram</i> <i>Fixing Contact Number</i>	49
Gambar 4.13 <i>Activity Diagram</i> <i>Save Fix Result</i>	50
Gambar 4.14 <i>Activity Diagram</i> <i>Export Data</i>	50
Gambar 4.15 <i>Class Diagram</i>	51
Gambar 4.16 Tampilan <i>Login</i>	54
Gambar 4.17 Halaman Utama Sistem <i>Data Cleaning</i>	54
Gambar 4.18 Tampilan <i>View Clean Data</i>	55
Gambar 4.19 Tampilan <i>Drop Down List View Data</i> dan <i>Area</i>	55
Gambar 4.20 Halaman <i>Fix Contact Number</i>	56
Gambar 4.21 Tampilan Hasil <i>Fix Contact Number</i>	56
Gambar 4.22 Tampilan Halaman <i>Import Data</i>	57
Gambar 4.23 Tampilan Pengisian Halaman <i>Import Data</i>	57
Gambar 4.24 Gambar Grafik Hasil Pengujian <i>Usability</i> Pada <i>User Sistem Data Cleaning</i> PT XYZ.....	75

Gambar 4.25 Gambar Grafik Hasil Pengujian $D_{small} = 2500$ Data.....	80
Gambar 4.26 Gambar Grafik Hasil Pengujian $D_{large} = 25.000$ Data.....	82

DAFTAR TABEL

Tabel 2.1 Perbandingan Metode dari Beberapa Penelitian Terkait	7
Tabel 2.2 Tabel Simbol-Simbol Pada <i>Use Case Diagram</i>	16
Tabel 2.3 Tabel Simbol-Simbol Pada <i>Activity Diagram</i>	17
Tabel 2.4 Tabel Simbol-Simbol Pada <i>Class Diagram</i>	19
Tabel 3.1 Kamus Data Konsumen Divisi <i>Consumer Care</i> PT XYZ	22
Tabel 3.2 Tabel <i>Role</i> dan Deskripsi Kerja Divisi <i>Consumer Care</i> PT XYZ	24
Tabel 4.1 Tabel Rincian Karakter Yang Akan Dihilangkan Pada Proses Deteksi Duplikasi Data.....	32
Tabel 4.2 Tabel Sebelum Dilakukan Proses <i>Pra-cleaning</i>	33
Tabel 4.3 Tabel Contoh Setelah Melewati Tahap <i>Pra-Cleaning</i>	33
Tabel 4.4 Tabel Contoh Setelah Proses Tokenisasi	35
Tabel 4.5 Tabel Setelah Token Diurutkan dan Digabungkan	35
Tabel 4.6 Tabel <i>Clean</i>	40
Tabel 4.7 Deskripsi Aksi Aktor dan Respon Sistem Untuk <i>Use Case Login</i>	42
Tabel 4.8 Deskripsi Aksi Aktor dan Respon Sistem Untuk <i>Use Case Duplicate Detection</i>	43
Tabel 4.9 Deskripsi Aksi Aktor dan Respon Sistem Untuk <i>Use Case View Duplicate Detection Result</i>	44
Tabel 4.10 Deskripsi Aksi Aktor dan Respon Sistem Untuk <i>Use Case Import Data</i>	44
Tabel 4.11 Deskripsi Aksi Aktor dan Respon Sistem Untuk <i>Use Case Fixing Contact Number</i>	45
Tabel 4.12 Deskripsi Aksi Aktor dan Respon Sistem Untuk <i>Use Case Save Fix Result</i>	46
Tabel 4.13 Deskripsi Aksi Aktor dan Respon Sistem Untuk <i>Use Case Edit Export Data</i>	46
Tabel 4.14 Tabel Hasil Pengujian Algoritma SNM – Fungsi <i>cleanData()</i>	63
Tabel 4.15 Tabel Hasil Pengujian Algoritma SNM – Fungsi <i>ChopTheWords()</i> ..	64
Tabel 4.16 Tabel Hasil Pengujian Algoritma SNM – Fungsi <i>gramming()</i>	67

Tabel 4.17 Tabel Hasil Pengujian <i>Functionality</i>	73
Tabel 4.18 Tabel Hasil Pengujian <i>Usability</i>	75
Tabel 4.19 Tabel Hasil Pengujian <i>Compatibility</i>	75
Tabel 4.20 Tabel Hasil Pengujian <i>Performance</i>	76
Tabel 4.21 Tabel Hasil Percobaan $D_{small} = 2500$ Data Dengan Menguji Menggunakan Nilai <i>Threshold</i> , Panjang Pemotongan Token, dan N- Gram.....	79
Tabel 4.22 Tabel Hasil Pengujian $D_{large} = 25.000$ Data Dengan Menguji Menggunakan Nilai <i>Threshold</i> , Panjang Pemotongan Token, dan N- Gram.....	80

DAFTAR RUMUS

Rumus 2.1 Rumus <i>N-Gram</i> Untuk Menghitung Kemiripan Antar <i>String</i>	14
Rumus 4.1 Rumus <i>precision</i> , <i>recall</i> , dan <i>f-measure</i>	78

DAFTAR LAMPIRAN

Lampiran 1 – Wawancara	90
Lampiran 2 – <i>Requirement Elicitation</i>	94
Lampiran 3 – <i>Software Requirement Specification</i>	98
Lampiran 4 – <i>Template Import Data</i>	129
Lampiran 5 – Hasil Pengujian Sistem	130

DAFTAR SINGKATAN

IDE	Integrated Development Environment
IGASIS	Intra-Governmental Access To Shared Information System
KDD	Knowledge Discovery in Databases
OOP	Object Oriented Programming
SNM	Sorted Neighbourhood Method
UML	Unified Modelling Language

Bab I

Pendahuluan

1.1 Latar Belakang Masalah

Data merupakan hal yang sangat penting untuk dapat menghasilkan sebuah informasi. Data yang memiliki jumlah besar biasanya dimiliki oleh beberapa organisasi/perusahaan dengan proses bisnis yang sangat kompleks. Data yang besar ini berasal dari serangkaian proses bisnis untuk digunakan sebagai proses pengambilan keputusan. Tepat atau tidaknya sebuah organisasi/perusahaan untuk mengambil sebuah keputusan bergantung pada kualitas data yang organisasi/perusahaan miliki. Namun, data yang berasal dari sumber eksternal organisasi/perusahaan biasanya memiliki data kotor. Hal ini biasanya terjadi karena kualitas data yang rendah seperti adanya duplikasi data, penulisan ejaan yang salah, atau data yang tidak lengkap. Data kotor inilah yang menyebabkan hasil laporan sebuah organisasi/perusahaan menjadi tidak akurat sehingga akan menyebabkan suatu kesalahan keputusan dalam sebuah organisasi/perusahaan (Guo, dkk., 2012).

Data kotor yang muncul dapat terjadi karena beberapa alasan (meskipun sebuah organisasi/perusahaan hanya memiliki satu atau *single database*). Kesalahan ejaan pada saat proses memasukkan data dan penulisan yang tidak memiliki standar format merupakan penyebab munculnya data kotor. Selain itu, data baru yang ternyata sudah ada di dalam *database* dan kemudian dimasukkan kembali ke dalam *database* menyebabkan *database* menjadi memiliki duplikasi data. Terlebih lagi, jika suatu organisasi/perusahaan memiliki beberapa sumber data yang heterogen sehingga adanya perbedaan model data antara sumber data yang satu dengan yang lain (Couto, 2012).

PT XYZ merupakan salah satu perusahaan farmasi yang memiliki puluhan ribu data dalam *database* konsumen yang dimilikinya. Data konsumen yang dibahas dalam penelitian ini adalah data konsumen yang berada pada divisi *Consumer Care* yang ada pada PT XYZ. Data konsumen yang ada pada divisi

tersebut masih memiliki kualitas data yang rendah karena terdiri atas data yang berduplikasi dan terdapat beberapa data yang belum mempunyai standar format penulisan, khususnya penulisan nomor telepon dan fax.

Berdasarkan wawancara yang telah penulis lakukan terhadap staf *sales admin* yang mengelola data konsumen di Divisi *Consumer Care* PT XYZ, telah diketahui bahwa proses pembersihan data, yaitu berupa deteksi duplikasi data dan merapikan format telepon dan fax masih dilakukan secara manual dan membutuhkan waktu yang lama. Hal ini karena jumlah data konsumen berjumlah ± 25.000 *record*. Sedangkan, jumlah data yang berduplikasi di dalam data tersebut berkisar ± 1.300 *raw* data atau terdiri atas 5.2% data yang berduplikasi. Oleh karena itu, dibutuhkan tingkat akurasi yang stabil untuk mendeteksi duplikasi data yang ada di dalam master data konsumen PT XYZ.

Berdasarkan permasalahan yang ada di Divisi *Consumer Care* PT XYZ, penulis bermaksud menerapkan suatu sistem *data cleaning* untuk dapat mendeteksi duplikasi data yang ada pada data konsumen PT XYZ secara otomatis. Namun, untuk dapat menerapkan sistem tersebut dibutuhkan suatu metode deteksi duplikasi data untuk dapat membantu permasalahan Divisi *Consumer Care*. Metode pendekatan yang digunakan untuk membangun sistem *data cleaning* ini adalah dengan menerapkan Algoritma *Sorted Neighbourhood* dan menggunakan Algoritma *N-gram*.

Algoritma *Sorted Neighbourhood* merupakan algoritma untuk mendeteksi duplikasi data dengan membentuk token khusus lalu kemudian menggabungkan dan menghapus dua buah atau lebih data yang kembar (Hernandez dan Stolfo, 1995). Sedangkan, Metode *N-gram* merupakan salah satu metode untuk menghitung kemiripan antar *string* yang menjadi penentu apakah antar *record* merupakan *record* yang kembar atau tidak (Recchia dan Max, 2013). Kedua metode ini akan diterapkan dalam penelitian ini untuk dapat menemukan duplikasi data pada master data konsumen Divisi *Consumer Care* PT XYZ.

Berdasarkan uraian di atas, maka dalam penelitian ini penulis bermaksud merancang suatu sistem *data cleaning* untuk master data konsumen pada PT XYZ dengan memanfaatkan Algoritma SNM dan Metode *N-Gram* dengan judul

penelitian “Rancang Bangun Sistem *Data Cleaning* Untuk Master Data Konsumen di PT XYZ Dengan Menerapkan Metode *Sorted Neighbourhood* dan Metode *N-Gram*”.

1.2 Perumusan Masalah

Berdasarkan latar belakang masalah, disusun perumusan masalah sebagai berikut.

1. Bagaimana mengembangkan suatu sistem *data cleaning* untuk master data konsumen Divisi *Consumer Care* PT XYZ agar dapat menerapkan algoritma SNM dan N-Gram.
2. Langkah yang digunakan untuk mengukur nilai efektivitas dari hasil deteksi duplikasi data pada sistem *data cleaning* yang dibangun.

1.3 Tujuan Penelitian

Berdasarkan perumusan masalah, disusun tujuan penelitian sebagai berikut:

1. Mengembangkan sistem *data cleaning* untuk dapat menerapkan Algoritma *Sorted Neighbourhood* dan Algoritma *N-Gram* agar dapat mendeteksi duplikasi data yang ada pada master data konsumen Divisi *Consumer Care* PT XYZ.
2. Mengukur nilai efektivitas hasil deteksi duplikasi data terhadap sistem *data cleaning* yang dibangun untuk master data konsumen Divisi *Consumer Care* PT XYZ.

1.4 Manfaat Penelitian

Manfaat yang dapat diperoleh dari penelitian ini adalah:

- a. Bagi PT XYZ:
 1. Sistem *data cleaning* yang dibuat akan digunakan sebagai *tools* untuk menyeleksi duplikasi data dan merapikan format penulisan telepon dan fax yang ada pada data konsumen Divisi *Consumer Care* PT XYZ.

2. Waktu yang digunakan oleh PT XYZ dalam proses mendeteksi duplikasi data menjadi lebih cepat dibanding dengan metode konvensional yang biasa dilakukan oleh staf *Sales Admin* Divisi *Consumer Care* PT XYZ.
- b. Bagi Universitas Bakrie:
Hasil penelitian dapat dijadikan sebagai dokumen akademik yang dapat dijadikan sebagai bahan literatur bagi sivitas akademika Universitas Bakrie.

1.5 Batasan Masalah

Batasan ruang lingkup dari penelitian yang dibahas dalam penelitian ini adalah sebagai berikut:

- a. Masalah yang diteliti adalah tentang sistem *data cleaning* yang terbatas untuk master data konsumen yang ada pada Divisi *Consumer Care* PT XYZ.
- b. Implementasi algoritma *Sorted Neighbourhood Method* dan *N-Gram* diterapkan dalam sistem untuk mendeteksi duplikasi data pada data konsumen Divisi *Consumer Care* PT XYZ.
- c. Sistem *data cleaning* yang akan dibangun hanya terbatas untuk mendeteksi duplikasi data dan merapikan format penulisan telepon dan *fax*.
- d. Sistem yang akan dibangun merupakan sistem berbasis *web*.

1.6 Sistematika Penulisan

Sistematika penulisan pada penelitian ini adalah sebagai berikut:

- a. Bab I Pendahuluan

Pada bab ini dijelaskan mengenai latar belakang penelitian, perumusan masalah, tujuan penelitian, manfaat penelitian, batasan penelitian, dan sistematika penulisan.

- b. Bab II Landasan Teori

Pada bab ini dibahas mengenai dasar-dasar teori, rujukan dan metode yang digunakan sebagai dasar dan alat untuk menyelesaikan permasalahan.

c. Bab III Analisis dan Perancangan Sistem

Pada bab ini dijelaskan tentang analisis serta perancangan sistem *data cleaning* untuk master data konsumen Divisi *Consumer Care* PT XYZ.

d. Bab IV Implementasi Program dan Pengujian

Pada bab ini berisi penerapan program dan pengujian sistem *data cleaning* yang dibangun telah sesuai dengan kebutuhan *Sales Admin* Divisi *Consumer Care* PT XYZ atau tidak.

e. Bab V Simpulan dan Saran

Pada bab ini berisi tentang simpulan dari hasil pembuatan sistem *data cleaning* dan saran-saran yang ditujukan kepada semua pihak yang bersangkutan.

Bab II

Tinjauan Pustaka

2.1 Penelitian Terkait

Pada tahun 1995, Hernandez dan Stolfo melakukan penelitian terhadap metode terkait *data cleaning* dengan penelitiannya yang berjudul *The Merge/Purge Problem for Large Databases*. Dalam penelitian ini, dijelaskan tentang Metode *Sorted Neighbourhood Method* (SNM) dan Metode SNM dengan *Clustering* data terlebih dahulu sebagai metode untuk menyelesaikan masalah penggabungan atau penghapusan dua buah atau lebih data yang kembar. Kemudian, Hernandez dan Stolfo juga membandingkan dua buah metode tersebut dan ternyata Metode SNM yang menerapkan tahap *clustering* data terlebih dahulu lebih memiliki performa yang lebih baik dibandingkan dengan Metode SNM.

Pada tahun 1999, Lee, dkk. melakukan penelitian terhadap metode terkait *data cleaning* dengan penelitiannya yang berjudul *Cleansing Data for Mining and Warehousing*. Dalam penelitian ini, metode *Sorted Neighbourhood Method* (SNM) dengan sedikit pengembangan digunakan untuk mendeteksi adanya *record* yang kembar. Dalam penelitian ini, dilakukan beberapa langkah untuk melakukan proses *data cleaning*, yaitu (1) pembersihan *field* dari data kotor, (2) melakukan proses tokenisasi dan mengurutkan token yang ada pada *field*, (3) mengurutkan *record*, (4) membandingkan *record*, dan (5) menggabungkan dua atau lebih *record* yang sama menjadi satu *record*.

Selanjutnya, penelitian berikutnya adalah penelitian yang dilakukan oleh Tian, dkk. pada tahun 2001. Penelitian tersebut membahas tentang metode algoritma *n-gram* untuk proses *data cleaning*. Dengan menggunakan pendekatan *n-gram*, setiap *record* dihitung jumlah nilai *n-gram* berdasarkan total dari nilai *n-gram* dari setiap *string* yang ada pada *record*. Jumlah angka tiap *record* kemudian dikelompokkan. *Record* yang berada dalam kelompok yang sama adalah kelompok yang terdeteksi memiliki data yang duplikat.

Kemudian, penelitian pada tahun 2006 dilakukan oleh Azma mengenai penerapan *data cleaning* dalam *data warehouse* sistem IGASIS (*Intra-Governmental Access To Shared Information System*). Proses pembersihan data yang dilakukan menerapkan metode pendekatan *schema matching* untuk membantu *user* atau *domain expert* dalam proses membersihkan dan memetakan data untuk dimasukkan ke *data warehouse*. Kemudian, menerapkan metode *linguistic-matching* dengan menggunakan metode *n-gram* untuk mengukur nilai kesamaan dari dua buah *string*.

Secara ringkas, penulis merangkum penelitian terkait pada tabel 2.1.1 di bawah ini.

Tabel 2.1 Perbandingan Metode dari Beberapa Penelitian Terkait

Judul	Pengarang	Tahun	Metode	Penelitian yang dilakukan
<i>The Merge/Purge Problem for Large Databases</i>	Hernandez, Mauricio A dan Stolfo, Savatore J.	1995	Metode SNM standar dan Metode SNM dengan teknik <i>Clustering</i>	Menerapkan metode SNM sebagai metode untuk melakukan penggabungan atau penghapusan dua buah data kembar. Kemudian, mengajukan teknik <i>Clustering</i> sebagai metode yang lebih baik dibandingkan metode SNM.
<i>Cleansing Data for Mining and Warehousing</i>	Lee, Mong Li; Lu, Hongjun; Ling, Tok Wang; Ko, Yee Teng	1999	Metode <i>Sorted Neighbourhood Method</i> (SNM)	Menerapkan metode SNM untuk mendeteksi duplikasi <i>record</i> dan mengajukan beberapa metode pra-pemrosesan agar <i>record</i> yang sama berada di posisi yang berdekatan.
<i>An n-gram-based approach for</i>	Tian, Zengping; Lu, Hongjun; Ji,	2001	Metode pendekatan <i>n-gram</i>	Mengajukan suatu metode pendekatan berbasis <i>n-gram</i>

<i>detecting approximately duplicate Database records</i>	Wenyun; Zhou, Aoying; Tian, Zhong			untuk mendeteksi duplikasi pada <i>record</i> .
Pembuatan Alat Bantu Dalam Proses <i>Data Cleaning</i> Pada <i>Intra-Governmental Access to Shared Information System</i> (IGASIS)	Azma, Syarifatul	2006	Metode pendekatan <i>schema matching</i>	Membangun suatu <i>tools</i> untuk memproses <i>data cleaning</i> untuk sistem IGASIS dengan menerapkan metode <i>schema-based</i> dalam memetakan data yang akan dimasukkan ke dalam <i>data warehouse</i> dan <i>instance-based</i> yang merupakan gabungan dari <i>linguistic matching</i> , <i>structural matching</i> , dan inputan <i>matches-mismatches</i> untuk mendeteksi duplikasi data.

2.2 Data Cleaning

Data Cleaning merupakan sebuah proses yang digunakan untuk menentukan data yang tidak akurat, tidak lengkap, atau data yang tidak jelas yang kemudian diperbaiki agar memiliki data yang berkualitas. Proses tersebut dapat terdiri atas pengecekan format, pengecekan kelengkapan, menghilangkan duplikasi atau kesalahan lain yang ada pada data (Chapman, 2005).

Menurut Maimon & Rokach (2006) dalam bukunya, *Data Mining and Knowledge Discovery Handbook*, *data cleaning* erat kaitannya dengan proses akuisisi dan definisi data untuk meningkatkan kualitas data yang ada pada sistem. *Data cleaning* merupakan bagian dari salah satu tahap awal proses *data mining*.

Data cleaning juga biasa dikenal dengan sebutan *data scrubbing*, *data cleansing*, *error checking*, *error correction*, atau *error detection*.

Dalam proses *data mining*, *data cleaning* merupakan suatu tahap awal atau pra pemrosesan sejumlah data sebelum data diproses menjadi sebuah informasi atau pengetahuan. Sedangkan, dalam *data warehouse*, *data cleaning* didefinisikan sebagai bagian dari fase proses ETL (*extract*, *transform*, dan *load*) yang berfokus pada proses deteksi dan koreksi terhadap data yang *error* yang bertujuan untuk meningkatkan kualitas data (Huda, 2010).

Salah satu tugas krusial dalam *data cleaning* atau *data scrubbing* adalah proses mendeteksi duplikasi data. Pada *database*, biasanya secara normal akan menghadapi permasalahan data seperti: (1) kesalahan atau kurang lengkapnya penulisan akibat *human error* ketika proses memasukkan data, (2) nilai yang dimasukkan tidak konsisten karena adanya perbedaan format ketika memasukkan data, (3) tidak lengkapnya informasi, (4) klien pindah dari satu tempat ke tempat lainnya tanpa ada pemberitahuan, dan (5) adanya kesalahan klien ketika memasukkan nama dan alamatnya (Lee, dkk., 1999).

Menurut Maletic dan Marcus (2000), bagaimanapun tidak ada definisi dan perspektif yang tepat yang diberikan terhadap proses *data cleaning*. Berbagai KDD (*Knowledge Discovery in Databases*) dan sistem *data mining* mengimplementasikan proses *data cleaning* dengan berbagai cara berdasarkan permasalahan dari kumpulan data yang ada.

2.3 Metode Data Cleaning

Secara umum, terdapat tiga langkah utama dalam proses *data cleaning*, yaitu:

1. Meng-audit data untuk mengidentifikasi jenis kesalahan yang mengurangi kualitas data,
2. Memilih metode yang cocok untuk mengotomatisasi pendeteksian dan penghilangan kesalahan, dan
3. Menerapkan metode tersebut pada *record* di dalam *dataset*.

Langkah (1) dan (2) dapat dilihat sebagai tahap spesifikasi dan tahap eksekusi dari alur kerja *data cleaning*. Sebagai tambahan, terdapat langkah selanjutnya, yaitu tahap *post-processing* atau kontrol dimana *user* dapat menguji hasil dan melakukan penanganan kesalahan terhadap hasil dari proses *data cleaning* (Maimon dan Rokach, 2005).

2.3.1 Algoritma Deteksi Duplikasi Data

Hernandez dan Stolfo (1995) dalam penelitiannya membahas tentang proses penggabungan atau penghapusan data dimana metode tersebut merupakan metode untuk menggabungkan duplikasi data dari dua atau lebih data yang kembar. Metode untuk mendeteksi duplikasi data tersebut diterapkan oleh penulis sebagai metode untuk membangun sistem *data cleaning* dalam penelitian ini. Berikut ini metode untuk mendeteksi duplikasi data menurut Hernandez dan Stolfo (1995).

1. Metode *Sorted Neighbourhood Method* (SNM)

Merupakan suatu metode pendekatan untuk membawa *record* yang berduplikasi berada pada posisi yang berdekatan. Lalu, proses deteksi duplikasi data dilakukan dalam ukuran *windows* tertentu untuk membatasi proses perbandingan satu data ke data lainnya. Memori yang dibutuhkan untuk memproses metode ini adalah $O(N \log N)$. Dimana N merupakan jumlah *record* di dalam database.

2. Metode SNM dengan teknik *Clustering*

Metode ini hampir serupa dengan metode SNM. Bedanya, data terlebih dahulu dibagi ke dalam beberapa *cluster* atau kelompok. Pembagian *cluster* atau kelompok dapat dilakukan secara independen sesuai dengan karakteristik data yang akan dibersihkan. Setelah data dikelompokkan, kemudian proses deteksi duplikasi data dilakukan pada tiap *cluster*. Memori yang dibutuhkan untuk memproses metode ini adalah $O(N \log \frac{N}{C})$. Dimana N merupakan jumlah *record* di dalam database dan C adalah ukuran *cluster*.

Berdasarkan pemaparan di atas, dapat disimpulkan bahwa Metode SNM dengan teknik *clustering* membutuhkan memori yang lebih sedikit dibandingkan dengan Metode SNM standar dimana $O(N \log \frac{N}{c}) < O(N \log N)$. Sehingga, pada penelitian ini akan diterapkan metode SNM dengan teknik *clustering* sebagai metode untuk mendeteksi duplikasi data.

2.3.2 Metode *Sorted Neighbourhood* Sebagai Metode Untuk Deteksi Duplikasi Data

Metode *Sorted Neighbourhood* yang akan digunakan pada penelitian ini adalah metode *Sorted Neighbourhood* dengan teknik *clustering* yang terdiri atas langkah berikut ini (Hernandez, 1995).

1. *Cluster data*. *Cluster data* dilakukan dengan menerapkan *constant partitioning* di mana data dikelompokkan ke dalam beberapa kelompok berdasarkan nilai yang ada pada atribut. Untuk mengelompokkan data dapat dilakukan dengan mengidentifikasi data yang ada. Dalam penelitian ini, data dapat dipisahkan berdasarkan atribut Area. Dengan demikian, proses perbandingan data dan proses pendeteksian data hanya dilakukan di tiap kelompok atau *cluster Area*.
2. Membentuk *key* atau token: Sebelum membentuk token, dilakukan tahap *pra-cleaning* terlebih dahulu seperti yang dilakukan oleh (Lee, dkk. 1999), yaitu berupa penghapusan kata, titel, tanda baca atau karakter tertentu. Untuk mendapatkan daftar karakter yang akan dihapus harus dilakukan observasi terlebih dahulu terhadap data yang akan dibersihkan. Setelah itu, baru dilakukan pembentukan token dengan mengambil satu atau beberapa huruf atau angka pada tiap kata yang ada pada tiap *field*. Misalnya, diambil dari tiga huruf pertama dari tiap *string* atau kata. Proses ini disebut dengan proses tokenisasi *record*.

3. Mengurutkan data: Mengurutkan *string* atau kata di dalam *field* dengan menggunakan *key* yang telah ditentukan pada tahap 1.
4. Menggabungkan data: sebuah *window* yang berukuran w bergerak melalui setiap *record* untuk membatasi proses perbandingan terhadap *record* yang berpotensi memiliki kesamaan data. Dimana nilai w adalah nilai jumlah pembagian tiap *cluster*. Setiap *record* baru yang masuk ke *window* dibandingkan dengan $w - 1$ *record* untuk menemukan *record* yang memiliki kesamaan.

2.3.3 Algoritma Perhitungan Kemiripan Antar String

Ketika antar *record* dibandingkan satu sama lain, dibutuhkan suatu metode untuk menghitung nilai kemiripan antar *string* yang ada di dalam *record* tersebut. Hal ini dilakukan untuk mengetahui apakah *record* tersebut sebenarnya termasuk *record* yang duplikat atau tidak. Berikut ini terdapat beberapa metode untuk menghitung kemiripan antar string menurut Recchia dan Max (2013):

1. Edit Distance

Algoritma ini mengukur banyaknya perbedaan antar *string* dalam hal jumlah adanya penyisipan, penghapusan, substitusi, dan/atau transposisi yang diperlukan untuk menghasilkan *string* pertama dari *string* kedua. Standar *Levenshtein Distance* menentukan nilai 1 untuk setiap penyisipan, penghapusan, dan substitusi. Kemudian, operasi tersebut dapat diubah menjadi nilai kemiripan antar *string* dengan membagi nilai aktual *Levenshtein Distance* dan nilai panjang *string* yang lebih panjang, dan mengurangi hasilnya dari nilai 1. Metode ini memiliki berbagai variasi yang mana variasi dari algoritma ini menentukan perbedaan bobot edit berdasarkan tipe operasinya dan pertimbangan lainnya (Navarro, 2001).

2. Longest Common Substring (LCS)

Metode LCS digunakan pada penelitian Friedman dan Sideli (1992) untuk mendeteksi dan membenarkan data pasien yang memiliki kesalahan ejaan

dengan menentukan nilai LCS antar *string*. Nilai LCS didapat dengan menghitung pembagian antara jumlah *string* dengan posisi yang bersamaan dengan jumlah *string* terpendek atau jumlah *string* terpanjang ataupun rata-rata dari panjang kedua *string*.

3. *Smith-Waterman Distance*

Seperti *Edit Distance*, algoritma *Smith-Waterman* menentukan serangkaian operasi yang dibutuhkan untuk mentransformasikan dari satu *string* ke *string* lainnya. Nilai kemiripan *string* dapat diperoleh dengan membagi antara hasil penghitungan *Edit Distance* dan panjang dari *string* terpanjang, *string* terpendek, atau rata-rata kedua *string*.

4. *N-Gram*

Metode ini mengukur banyaknya jumlah *n-gram* yang sama di antara dua *string* yang dibandingkan. Nilai kemiripan antar *string* didapat baik dengan membagi jumlah *n-gram* yang sama dengan jumlah *n-gram* di *string* yang pendek atau dengan jumlah *n-gram* di *string* yang panjang atau dengan rata-rata dari jumlah kedua *string* (Navarro, 2001).

Dari beberapa algoritma di atas, menurut Recchia dan Max (2013) dalam penelitiannya mengungkapkan bahwa algoritma *N-Gram* memiliki performa yang baik dari segi jumlah penemuan data (*recall*) dan ketepatannya (*precision*) dibandingkan dengan algoritma *Edit Distance*, *LCS*, dan *Smith-Waterman*. Oleh karena itu, penulis bermaksud untuk menerapkan metode *N-Gram* pada penelitian ini sebagai metode untuk menghitung nilai kemiripan antar string.

2.3.4 Algoritma Pendekatan *N-Gram* Sebagai Algoritma Perhitungan Kemiripan Antar *String*

Untuk membandingkan dua *record*, maka diterapkan algoritma pendekatan *n-gram* untuk mengukur nilai kesamaan dari dua *string* atau kata yang berbeda.

Maksud dari *n-gram* adalah *n* huruf yang berturut-turut dari sebuah kata. Nilai *n* yang digunakan adalah 2, 3, dan 4. Jika *n* = 2, maka disebut digram atau bigram. Jika *n* = 3 disebut dengan trigram, dan seterusnya (Tian, dkk., 2001).

Berikut ini rumus *n-gram* untuk menghitung kemiripan antara *string* A dan *string* B:

$$Sim_{AB} = \frac{(2 (|ngram(A) \cap ngram(B)|))}{ngram(A) + ngram(B)}$$

Rumus 2.1 Rumus *N-Gram* Untuk Menghitung Kemiripan Antar *String*

Pengukuran nilai kemiripan antara dua *string* yaitu antara nilai 0 sampai 1. Jadi, semakin mirip *string* maka nilainya semakin mendekati 1. Sebaliknya, semakin tidak mirip nilainya akan mendekati 0. Berikut ini contoh penggunaan *n-gram* untuk menghitung nilai kemiripan antar *string*, dimana *n* = 2:

String A = “APOTIK”, mempunyai 2-gram:

AP, PO, OT, TI, IK

String B = APOTEK, mempunyai 2-gram:

AP, PO, OT, TE, EK

Dua contoh *string* di atas memiliki 3 buah bigram yang sama, yaitu AP, PO, dan OT sehingga nilai kemiripannya adalah:

$$Sim_{AB} = \frac{2 \times 3}{5+5} = 0,6$$

Untuk menentukan apakah dua *string* merupakan data yang kembar atau bukan maka akan dibuat ketentuan, yaitu dengan menentukan nilai ambang batas (*threshold*). Misalnya, untuk kasus ini, ditentukan nilai ambang batas = 0,6. Jadi, jika nilai kesamaan antara 2 *string* $\geq 0,6$, maka dapat dinyatakan bahwa 2 *string*

tersebut memiliki kemiripan. Sebaliknya, jika nilai kesamaan antara 2 *string* $\leq 0,6$, maka dapat dinyatakan bahwa 2 *string* tersebut berbeda (Azma, 2006).

2.4 Pemrograman Berorientasi Objek

Pemrograman berorientasi objek adalah suatu pendekatan dalam pengembangan perangkat lunak di mana struktur perangkat lunaknya disusun berdasarkan objek-objek yang berinteraksi satu sama lain untuk menyelesaikan suatu tugas (Dan, 2011). Berikut ini adalah karakteristik dasar dari pemrograman berorientasi objek.

1. *Encapsulation* (Pemodulan/Pengkapsulan)

Metode untuk menggabungkan data dengan fungsi. Dalam konsep ini data dan fungsi digabung menjadi satu kesatuan yaitu kelas.

2. *Inheritance* (Pewarisan)

Dari konsep penurunan ini, suatu kelas bisa diturunkan menjadi kelas baru yang masih mewarisi sifat-sifat orang tuanya. Pewarisan dapat dilakukan jika:

- Ada beberapa atribut dan *method* yang sama yang digunakan oleh beberapa kelas berbeda (reduksi penulisan kode).
- Ada satu atau beberapa kelas yang sudah pernah dibuat yang dibutuhkan oleh aplikasi (*reusability*).
- Ada perubahan kebutuhan fungsional atau *feature* aplikasi dimana sebagian atau seluruh perubahan tersebut tercakup di satu atau beberapa kelas yang sudah ada (*extend*).

3. *Polymorphism* (Polimorfisme)

Polimorfisme berarti kelas-kelas yang berbeda tetapi berasal dari satu orang tua, dapat mempunyai metode yang sama tetapi cara pelaksanaannya berbeda. Atau dengan kata lain, suatu fungsi akan memiliki perilaku berbeda jika dilewatkan ke kelas yang berbeda-beda.

Jika dibandingkan dengan pemrograman secara prosedural, pemrograman berorientasi objek lebih memiliki keunggulan sebagai berikut (Ningsih, 2009).

1. Data dan fungsi dibungkus dalam kelas-kelas atau objek-objek sehingga dapat memudahkan pengembang aplikasi dalam memahami program.
2. Efektif digunakan untuk menyelesaikan masalah besar karena pemrograman berorientasi objek terdiri dari kelas-kelas yang memisahkan setiap *code* program menjadi kelompok-kelompok kecil sesuai dengan fungsinya.
3. Objek dan kelas dapat digunakan berkali-kali sehingga dapat menghemat *space* memori.


2.5 Unified Modelling Language (UML)

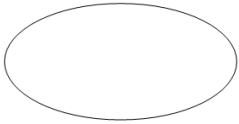

UML adalah sebuah bahasa yang telah menjadi standar dalam industri untuk visualisasi, merancang, dan mendokumentasikan sistem perangkat lunak (Sulistiyorini, 2009). UML menyediakan beberapa jenis diagram, diantaranya adalah sebagai berikut.

1. Use Case Diagram

Use case diagram adalah *diagram* yang menggambarkan interaksi antara sistem dan pengguna sistem. *Diagram* ini menjelaskan siapa yang akan menggunakan sistem dan memberikan sebuah narasi bagaimana *user* tersebut dapat berinteraksi dengan sistem. *Use case diagram* memiliki beberapa simbol antara lain:

Tabel 2.2 Tabel Simbol-Simbol Pada *Use Case Diagram*




Nama Simbol	Notasi	Keterangan
<i>Actor</i>		Merepresentasikan pengguna sistem, tidak hanya manusia tetapi semua yang akan berinteraksi dengan sistem.

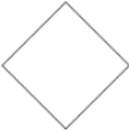


Nama Simbol	Notasi	Keterangan
<i>Use Case</i>		Sebuah skenario (kegiatan) yang akan dilakukan untuk menyelesaikan suatu pekerjaan.
<i>Relationship</i>		Garis yang menghubungkan dua simbol pada <i>use case diagram</i> . Terdapat beberapa tipe <i>relationship</i> antar simbol yaitu: <i>association</i> , <i>extends</i> , <i>uses</i> , <i>depends on</i> , dan <i>inheritance</i> .

2. Activity Diagram

Activity diagram memodelkan alur sebuah proses bisnis, tahapan *use case*, atau perilaku sebuah objek (*method*). Diagram ini hampir sama dengan *flowchart* yang menggambarkan urutan kerja dari sebuah *use case*.

Tabel 2.3 Tabel Simbol-Simbol Pada *Activity Diagram*

Nama Simbol	Notasi	Keterangan
<i>Initial Node</i>		Merepresentasikan awal dari sebuah proses.
<i>Action</i>		Menggambarkan sebuah tahapan/aksi.
<i>Flow</i>		Menggambarkan alur kerja.

Nama Simbol	Notasi	Keterangan
<i>Decision</i>		Menggambarkan sebuah kondisi.
<i>Fork</i>		Menggambarkan aksi yang terjadi secara bersamaan.
<i>Activity Final</i>		Merepresentasikan akhir dari sebuah proses.


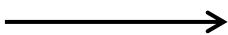

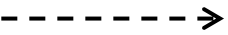

3. *Class Diagram*

Class diagram adalah model statis yang menggambarkan struktur dan deskripsi kelas serta hubungannya antara kelas. Kelas terdiri dari nama kelas, atribut dan operasi/metode. Atribut dan operasi (metode) dapat memiliki salah satu sifat berikut:

- Private*, hanya bisa dipanggil dari dalam kelas itu sendiri. Penulisan metode/atribut diawali dengan tanda “-”.
- Protected*, hanya dapat dipanggil oleh kelas yang bersangkutan dan kelas turunannya. Penulisan metode diawali dengan tanda “#”.
- Public*, dapat dipanggil dari semua objek. Penulisan metode/atribut diawali dengan tanda “+”.

Tabel berikut ini menjelaskan tentang simbol hubungan antar kelas yang digunakan pada *class diagram*.

Tabel 2.4 Tabel Simbol-Simbol Pada *Class Diagram*

Nama Simbol	Notasi	Keterangan
Asosiasi / <i>Association</i>		Relasi antar kelas dengan makna umum, asosiasi biasanya juga disertai dengan <i>multiplicity</i> .
Asosiasi Berarah / <i>Directed Association</i>		Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi ini biasanya juga disertai dengan <i>multiplicity</i> .
Generalisasi		Relasi antar kelas dengan makna generalisasi – spesialisasi (umum – khusus) atau untuk menyatakan hubungan <i>inheritance</i> .
Kebergantungan / <i>Dependency</i>		Relasi antar kelas dengan makna kebergantungan antar kelas.
Agregasi / <i>Aggregation</i>		Relasi antar kelas dengan makna semua-bagian (whole-part).

Bab III

Metodologi Penelitian

Dalam melaksanakan penelitian ini, Penulis membuat kerangka penelitian sebagai panduan dalam melakukan kegiatan secara berurutan mulai dari awal penelitian ini dijalankan hingga akhir hasil penelitian. Kerangka penelitian dibuat berdasarkan pengembangan metode air terjun (*waterfall*). Alasan penulis memilih metode ini adalah karena metode ini merupakan metode pengembangan tradisional yang umum digunakan dalam pembangunan perangkat lunak. Namun, metode ini tetap membuat kualitas perangkat lunak tetap terjaga karena pengembangannya yang terstruktur dan terawasi. Di sisi lain, model ini merupakan jenis model yang bersifat dokumen lengkap sehingga proses pemeliharaan dapat dilakukan dengan mudah (Binanto, 2014).

Metode *waterfall* yang digunakan dalam penelitian ini adalah metode *waterfall* berdasarkan Sommerville (2011) yang terdiri atas tahap identifikasi masalah, tahap analisis kebutuhan (*requirement analysis*), tahap desain sistem (*system design*), tahap implementasi (*implementation*), tahap pengujian (*testing*), dan tahap pemeliharaan (*maintenance*). Pada bab ini akan dibahas tentang tahap analisis kebutuhan dan desain atau perancangan sistem. Sedangkan, tahap implementasi, pengujian dan pemeliharaan akan dibahas di bab berikutnya.

3.1 Tahap Identifikasi Masalah

Dalam mengidentifikasi permasalahan yang ada, penulis melakukan observasi terlebih dahulu terhadap permasalahan yang ada. Setelah itu, penulis melakukan wawancara kepada narasumber untuk mengetahui permasalahan data yang ada di PT XYZ (terlampir naskah wawancara antara penulis dengan narasumber). Pada tahap ini diperoleh beberapa hal berikut ini.

3.1.1 Prosedur Yang Sedang Berjalan

Saat ini, proses *data cleaning* data konsumen yang dilakukan di Divisi *Consumer Care* PT XYZ masih menggunakan cara manual, yang terdiri dari tiga aktivitas utama berikut.

1. Mengecek *record* yang memiliki kemiripan.

Aktivitas ini merupakan aktivitas utama dari proses *cleaning* yang dilakukan. Langkah ini dilakukan dengan cara mengurutkan dan mengelompokkan data menggunakan beberapa atribut dengan memakai fitur *pivot* yang ada di Ms. Excel. Atribut yang digunakan untuk melakukan pembersihan data adalah *Area*, *Outlet Type*, *Name*, dan *Address*. Kemudian, data yang telah terurut dan terbagi menjadi beberapa kelompok dibaca dengan *read-scanning* untuk dapat menemukan data yang memiliki kemiripan. Jika menemukan data yang mirip, maka tidak akan langsung digabung menjadi satu *record*. Tetapi, akan dibentuk satu buah ID baru yang disebut dengan *Clean Code*.

2. Mengecek kolom yang kosong.

Memberikan tanda pada kolom yang kosong karena tidak lengkap ketika proses *input*. Terutama, terhadap atribut yang wajib untuk diisi. Kemudian, data yang kosong ini akan ditanyakan kepada pihak yang bekerja di lapangan (*field force*) untuk membantu melengkapi data yang kosong tersebut.

3. Memformat beberapa atribut yang belum sesuai dengan standar penulisan.

Merapikan struktur penulisan data yang masih tidak rapi dan tidak sesuai dengan standar penulisan. Format penulisan yang dirapikan adalah format penulisan yang ada pada kolom *Phone* dan *Fax*.

3.1.2 Master Data Konsumen Divisi *Consumer Care* di PT XYZ

Master data konsumen yang dibahas dalam studi kasus ini merupakan master data konsumen yang terdapat pada divisi *Consumer Care* yang ada pada PT XYZ. Master data yang terdapat pada divisi tersebut terdiri dari data yang besar, yaitu sekitar 25.000 data yang masih menggunakan *Ms. Excel 2010* dalam menjalankan proses operasionalnya.

Data konsumen yang dimiliki oleh divisi *Consumer Care* berasal dari data yang dikumpulkan oleh pihak distributor. Sementara, data yang berasal dari pihak distributor tersebut memiliki beberapa kesalahan yang harus dilakukan pengecekan data oleh pihak PT XYZ. Diantaranya adalah adanya *record* yang kembar.

Saat ini, pembuatan *database* untuk mengatur master data konsumen PT XYZ masih dalam proses pengembangan. Terdapat beberapa atribut yang ada pada master data konsumen tersebut. Berikut ini rincian dekripsi dari tiap atribut yang dimiliki oleh master data konsumen divisi *Consumer Care* PT XYZ di mana penulisan kamus data di bawah ini mengikuti acuan yang ada pada Schacherer (2012).

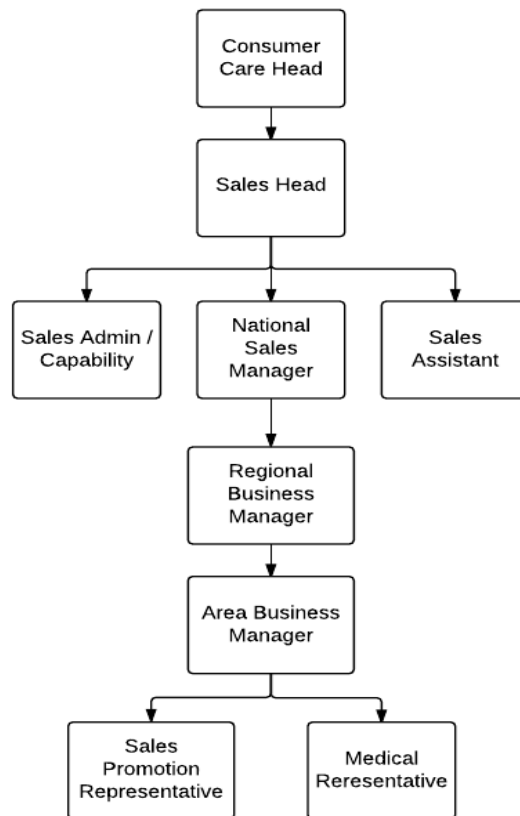
Tabel 3.1 Kamus Data Konsumen Divisi *Consumer Care* PT XYZ

Variabel	Deskripsi	Tipe Data	Sifat Pengisian	Nilai yang diharapkan
ID_Organization	Nomor ID yang dimiliki oleh toko.	<i>String</i>	Wajib	Terdiri atas angka yang unik
Name	Nama took / <i>outlet</i>	<i>String</i>	Wajib	Penulisan nama toko yang lengkap
Address	Alamat took	<i>String</i>	Wajib	Penulisan alamat toko yang lengkap
Type	Klasifikasi yang lebih khusus dari tipe	<i>String</i>	Wajib	Pilihan kategori tipe toko yang terdiri atas <i>Pharmacy</i> , <i>drug</i>

	channel took			<i>store, minimarket, supermarket, dan lain-lain.</i>
Region	Pembagian wilayah regional terhadap wilayah pemasaran PT XYZ.	<i>String</i>	Wajib	Terdiri dari Sumatera, Jabotaponsa (Jakarta, Bogor, Tangerang, Pontianak, Samarinda), <i>Central</i> , dan <i>East</i>
Area	Area pembagian area pemasaran produk yang ada di PT XYZ	<i>String</i>	Wajib	Terdapat beberapa pilihan pembagian area, seperti: Bandung, Jakarta, Tangerang, dan lain-lain.
Province	Provinsi dimana lokasi toko berada	<i>String</i>	Wajib	Provinsi tempat toko berada. Terdapat beberapa pilihan provinsi yang ada di seluruh Indonesia.
City	Kota dimana lokasi toko berada	<i>String</i>	Wajib	Kota tempat toko berada. Terdapat beberapa pilihan kota yang ada di seluruh Indonesia.
Zipcode	Kode pos dimana lokasi toko berada	<i>String</i>	Wajib	Kodepos tempat toko berada
Class	Tipe kelas dari toko.	<i>String</i>	Wajib	Tipe kelas yang terdiri dari kelas A, B, dan C.
Phone	Nomor <i>handphone</i> yang dapat dihubungi	<i>String</i>	<i>Optional</i>	08xx-xxxx-xxxx
Fax	Nomor <i>fax</i> yang dapat dihubungi	<i>String</i>	<i>Optional</i>	xxxx-xxxx
Email	Email aktif yang dimiliki	<i>String</i>	<i>Optional</i>	email@provider.domain
Website	<i>Website</i> yang dimiliki took	<i>String</i>	<i>Optional</i>	www.websitename.domain
Status	Status toko apakah aktif atau tidak aktif	<i>String</i>	Wajib	<i>Active/Deactive</i>

3.1.3 Struktur Organisasi

Berikut ini merupakan struktur organisasi dari Divisi *Consumer Care* di PT XYZ beserta peran dan deskripsi kerja yang dijelaskan pada tabel berikut ini.



Gambar 3.1 Struktur Organisasi Divisi *Consumer Care* PT XYZ

Tabel 3.2 Tabel *Role* dan Deskripsi Kerja Divisi *Consumer Care* PT XYZ

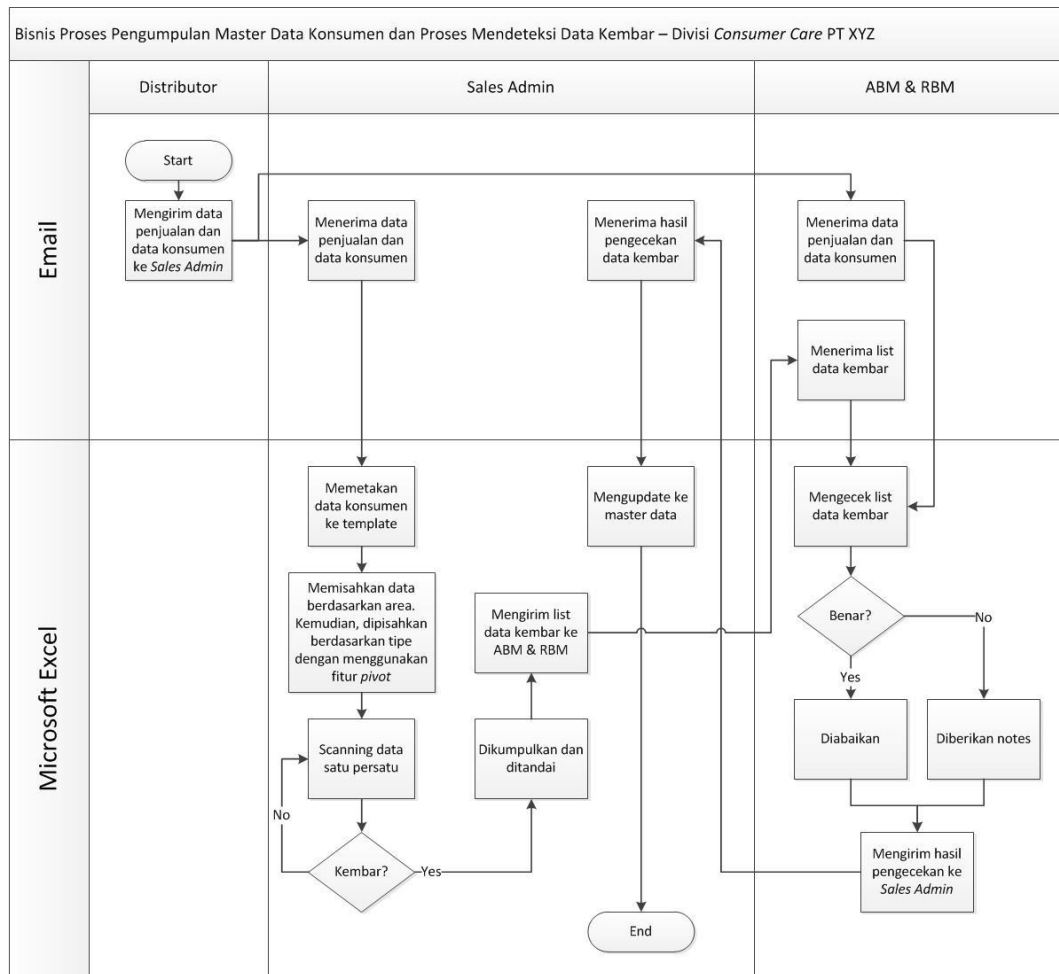
No.	<i>Role</i>	Deskripsi Kerja
1	<i>Consumer Care Manager</i>	<ul style="list-style-type: none"> - Mengatur dan mengkoordinasi <i>resource medical</i> dan <i>non-medical</i>, fasilitas, dan layanan. - Meningkatkan dan memperluas pasar produk <i>consumer care</i>. - Memastikan layanan yang ada telah memenuhi standar nasional dan internasional. - Mengatur registrasi/notifikasi produk untuk memastikan peningkatan bisnis di Indonesia. - Mengatur <i>budget</i>.
2	<i>Sales Manager</i>	<ul style="list-style-type: none"> - Mengatur penjualan produk dan layanan <i>consumer care</i>. - Memastikan nilai yang konsisten dalam

		<p>peningkatan pendapatan penjualan.</p> <ul style="list-style-type: none"> - Menempatkan dan mengatur personil-personil <i>sales</i>. - Mengidentifikasi tujuan, strategi, misi untuk meningkatkan <i>short</i> dan <i>long-term</i> penjualan dan pendapatan.
3	<i>Sales Admin / Capability</i>	<ul style="list-style-type: none"> - Merencanakan kebutuhan <i>sales training</i>, dan menyusun rencana <i>training</i> tahunan. - Menyusun persiapan <i>training</i> untuk perusahaan dan <i>sales force</i> distributor dalam membahas strategi dan proses menjual, manajemen penjualan, manajemen distribusi. - Mengkoordinasi penjualan dengan manajer <i>regional</i> dan <i>area</i>. - Melaksanakan kerja lapangan dan secara teratur melakukan audit untuk memastikan pelaksanaan dan implementasi di lapangan. - Menindaklanjuti dan bertanggung jawab terhadap hasil penjualan dan presentase <i>sales achievement</i>.
4	<i>National Sales Manager</i>	<ul style="list-style-type: none"> - Memastikan tujuan <i>sales</i> telah sesuai, tidak hanya pada region tertentu, tetapi seluruh <i>region</i>. - Mengidentifikasi kelemahan pada rencana <i>marketing</i> dan membuat sebuah aturan jika diperlukan. - Memperkirakan penjualan dalam seminggu, sebulan, atau quarter. - Menganalisa data penjualan untuk mengidentifikasi kekuatan dan kelemahan kegiatan promosi tertentu. - Mengawasi <i>budget sales force</i> perusahaan. - Menjaring dengan konsumen yang potensial dan partner bisnis untuk mempromosikan produk tertentu. - Menyetujui kontrak besar.
5	<i>Sales Assistant</i>	<ul style="list-style-type: none"> - Membantu <i>sales manager</i> dalam mengelola program khusus dan kebutuhan operasional bisnis. - Memberikan dukungan yang cepat kepada tim <i>sales</i>. - Mengatur dan mengurus administrasi yang dibutuhkan oleh tim <i>sales</i>. - Membantu dalam upaya merekrut untuk semua posisi tim <i>sales</i> yang dibutuhkan. - Mengambil peran aktif dalam melaksanakan pelatihan dan pengembangan tim.
6	<i>Regional Business Manager</i>	<ul style="list-style-type: none"> - Bertanggung jawab dan memantau aktivitas harian bisnis regional. - Mengatur persiapan <i>viable bids</i>, proposal, dan strategi baru pada region yang dipimpin. - Melaporkan ke bagian komisaris, bekerja dengan rekan agen/distributor dan manajer Area. - Berkontribusi dalam melakukan penelitian pasar, penerapan model layanan, pengembangan model layanan, dan keberhasilan dalam penawaran produk di wilayah regional tertentu.
7	<i>Area Business Manager</i>	<ul style="list-style-type: none"> - Bertanggung jawab dan memantau aktivitas harian bisnis area.

		<ul style="list-style-type: none"> - Melaporkan ke bagian regional manajer, bekerja dengan rekan agen/distributor dan <i>sales promotion representative</i>. - Berkontribusi dalam melakukan penelitian pasar, penerapan model layanan, pengembangan model layanan, dan keberhasilan dalam penawaran produk di wilayah area tertentu. - Memantau distributor dan <i>sales force</i> produk distributor. - Meningkatkan target dan meningkatkan profitabilitas distributor. - Mendata data seluruh konsumen yang berada di area manajemennya.
8	<i>Sales Promotion Representative</i>	<ul style="list-style-type: none"> - Membangun dan menjaga hubungan bisnis dengan konsumen pada wilayah tertentu. - Melakukan kunjungan dan presentasi ke konsumen. - Menangani masalah dan complain dari konsumen. - Menganalisa potensi pasar dan menentukan nilai dan prospektif konsumen terhadap organisasi. - Mengidentifikasi kelebihan dan membandingkan produk/layanan yang diberikan. - Mengkoordinasi <i>sales effort</i> dengan tim <i>marketing, sales management</i>, dan <i>accounting</i>.
9	<i>Medical Representative</i>	<ul style="list-style-type: none"> - Mengatur pertemuan dengan dokter, apoteker, dan tim medis rumah sakit. - Mengadakan presentasi ke dokter, staf atau perawat di rumah sakit dan/atau dokter dan apoteker di sektor retail. - Membangun dan mengatur hubungan positif dengan staf <i>medical</i> dan administrasi. - Mendata data seluruh dokter, apoteker, dan tim medis. - Memantau informasi tentang kegiatan pelayanan kesehatan pada area tertentu.

3.1.4 Bisnis Proses

Proses bisnis yang akan dipaparkan di sini adalah proses bisnis dalam mengumpulkan data konsumen dari pihak *Area Business Manager* kepada tim *Sales Admin / Capability* dan proses pendeteksian duplikasi data yang dilakukan secara manual. Berikut ini gambaran bisnis proses terkait proses tersebut.



Gambar 3.2 Gambar Bisnis Proses Deteksi Data Kembar Pada Master Data Konsumen Divisi *Consumer Care* PT XYZ

3.1.5 Sistem *Data Cleaning* Yang Diajukan

Berdasarkan pembahasan di atas, sistem *data cleaning* yang akan diajukan dalam penelitian ini adalah aktivitas nomor 1 dan 3 karena menurut narasumber untuk aktivitas nomor 2 tidak membutuhkan sistem khusus untuk memecahkan masalahnya. Hal-hal yang akan dicapai dalam pembuatan sistem deteksi duplikasi data pada sistem *data cleaning* dalam penelitian ini adalah:

- Proses *pra-cleaning*, yaitu proses pembersihan data dari kata, titel, tanda baca atau karakter tertentu sebelum memasuki tahap pendeteksian duplikasi data.

- b. Proses *cleaning*, yaitu proses utama yang terdiri atas pendeteksian duplikasi data.
- c. *Result*, yaitu hasil data yang telah bersih atau laporan atas duplikasi data yang telah ditemukan dengan memungkinkan *user* mengekspor hasil proses deteksi duplikasi data.

3.2 Tahap Analisa Kebutuhan Sistem

Analisis kebutuhan sistem dalam penelitian ini terdiri atas analisa kebutuhan non fungsional dan fungsional seperti yang akan dipaparkan pada sub-bab berikut.

3.2.1 Kebutuhan Non Fungsional Sistem

Kebutuhan non fungsional adalah tipe kebutuhan yang yang berisi properti perilaku yang dimiliki oleh sistem, seperti deskripsi dari fitur-fitur, karakteristik, dan batasan-batasan yang lain yang mendefinisikan sistem yang memuaskan (Al Fatta, 2007). Adapun kebutuhan non fungsional yang dipertimbangkan dalam pembuatan sistem *data cleaning* dapat dilihat pada lampiran 2.

3.2.2 Analisa Kebutuhan Fungsional Sistem

Kebutuhan fungsional adalah jenis kebutuhan yang berisi proses-proses apa saja yang nantinya dilakukan oleh sistem. Analisa kebutuhan fungsional sistem dilakukan untuk menganalisis apa saja kebutuhan yang diajukan untuk sistem *data cleaning* pada penelitian ini. Adapun kebutuhan fungsional mencakup deskripsi dari aktivitas-aktivitas dan layanan-layanan yang harus disediakan oleh sistem (Al Fatta, 2007). Kebutuhan fungsional yang dibutuhkan dalam penelitian ini dapat dilihat pada lampiran 2.

3.3 Perancangan Sistem

Setelah tahap analisis kebutuhan, tahap selanjutnya adalah proses perancangan sistem. Tahap perancangan sistem merupakan proses penting dalam proses perancangan aplikasi untuk menentukan hasil akhir dari rencana program

yang akan dibuat. Perancangan sebuah sistem mempengaruhi hasil akhir dari pembangunan aplikasi sehingga perlu diperhatikan proses pembuatannya. Dibutuhkan hasil analisis yang benar agar hasil dapat diimplementasikan dan sesuai dengan kebutuhan sistem. Penerapan sebuah algoritma pada salah satu fungsi dalam sistem yang akan dibangun menjadi sebuah alur penting dalam menyelesaikan kasus permasalahan yang terdapat dalam penelitian ini. Perancangan sistem terdiri atas perancangan alur algoritma, perancangan *database*, dan UML yang akan dijelaskan pada bab berikutnya.

3.4 Tahap Implementasi

Setelah melakukan perancangan sistem, tahap selanjutnya adalah implementasi. Implementasi adalah proses merubah desain menjadi bahasa pemrograman yang secara teknis biasanya dikerjakan oleh *programmer*. Pada penelitian ini, implementasi dikerjakan sendiri oleh Penulis dengan bahasa pemrograman C# dan berorientasi objek (OOP). Hasil dari tahapan ini adalah sistem *data cleaning* untuk master data konsumen Divisi *Consumer Care* PT XYZ dengan fungsi seluruh sistem yang sudah berjalan dengan baik.

3.5 Tahap Pengujian

Tahap ini merupakan tahap pengujian atas implementasi yang telah dilakukan pada tahap sebelumnya. Tahap pengujian dilakukan untuk melihat kebenaran dari logika yang dijalankan sistem dan menilai apakah implementasi yang dilakukan telah sesuai dengan yang diinginkan dalam mencapai tujuan pembuatan sistem (Pressman, 2010). Pengujian akan dilakukan dengan menggunakan metode *white box* dan *black box* serta evaluasi metode yang telah diterapkan pada sistem *data cleaning* dengan menggunakan dua sampel data (*Dlarge* data dan *Dsmall* data) berdasarkan metode yang dilakukan oleh Weis, dkk. (2008).

Bab IV

Implementasi dan Pembahasan

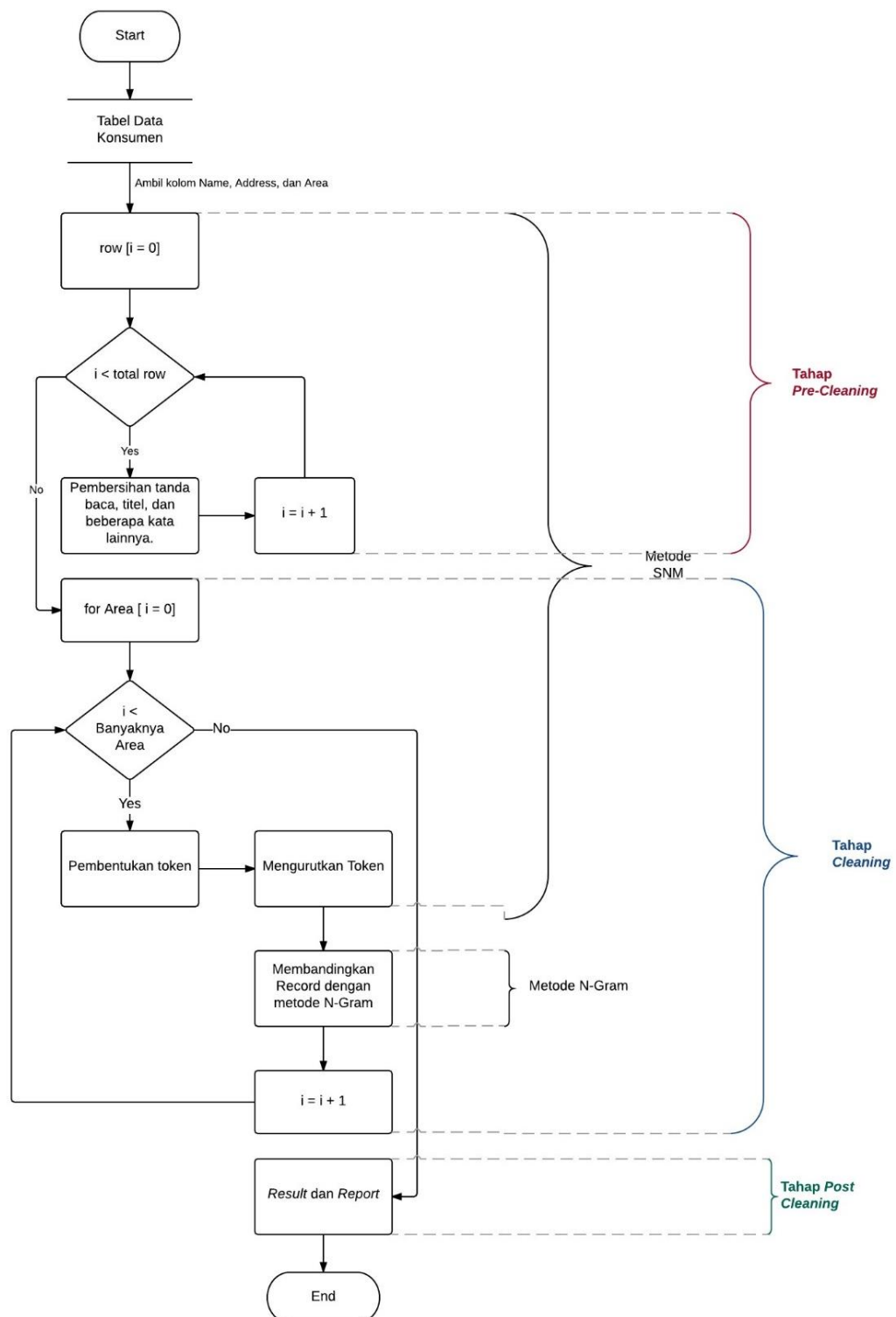
Bab ini menjelaskan pembahasan tentang perancangan dan pembangunan aplikasi manajemen aset kelas dengan penerapan algoritma *Sorted Neighbourhood* dan N-Gram. Pengembangan aplikasi dilakukan dengan menggunakan metode *Waterfall*.

4.1 Perancangan Sistem

Sebelum masuk ke tahap implementasi sistem, perancangan sistem sangat dibutuhkan dalam sebuah pembangunan sistem atau aplikasi. Ini akan menjadi acuan untuk dapat mengimplementasi ke dalam sebuah *code* atau pemrograman. Perancangan sistem terdiri atas perancangan alur algoritma, perancangan *database*, dan UML yang akan dijelaskan pada sub-bab berikut ini.

4.1.1 Perancangan Alur Algoritma Deteksi Duplikasi Data

Setelah penulis melakukan tinjauan pustaka yang terdapat pada Bab II dan melakukan observasi data serta menganalisa kebutuhan sistem, maka dihasilkan alur algoritma untuk deteksi duplikasi data seperti berikut ini.



Gambar 4.1 *Flowchart* Algoritma Deteksi Duplikasi Data

Proses pendeteksian duplikasi data dibagi menjadi beberapa tahap, yaitu tahap *pra-cleaning*, tahap *cleaning*, dan tahap *post-cleaning*. Berikut ini tahap-tahap deteksi duplikasi data yang dilakukan pada master data konsumen PT XYZ.

I. Proses Pra-cleaning

Dalam studi kasus ini, tabel yang akan dilakukan pembersihan adalah tabel master data konsumen divisi *Consumer Care* PT XYZ yang disebut dengan tabel *Organization*. Ketika tabel masuk ke dalam sistem, khusus atribut *Name*, *Address*, dan *Area* tidak boleh kosong karena atribut tersebut akan digunakan dalam mendeteksi duplikasi data.

Sebagai langkah pertama dalam proses *pra-cleaning* adalah menghilangkan karakter yang berada pada atribut *Name* dan *Address* terlebih dahulu untuk memudahkan sistem mendeteksi duplikasi data. Berikut ini rincian karakter yang akan dihilangkan terlebih dahulu.

Tabel 4.1 Tabel Rincian Karakter Yang Akan Dihilangkan Pada Proses Deteksi Duplikasi Data

No.	Atribut	Tindakan
1	<i>Name</i>	<ul style="list-style-type: none"> - Tanda baca seperti: titik, koma, tanda kurung, garis miring, dan tanda baca lainnya dihilangkan. - <i>Title</i> toko seperti PT, CV, APT, MM, SPM, TKLO, PBF, TOB, RS, TKOS, COS, TK dan <i>title</i> lainnya dihilangkan.
2	<i>Address</i>	<ul style="list-style-type: none"> - Tanda baca seperti: titik, koma, tanda kurung, titik dua, tanda hubung, garis miring, dan tanda baca lainnya dihilangkan. - Singkatan seperti Jl, Jln, Jalan, No dan <i>title</i> lainnya dihilangkan.

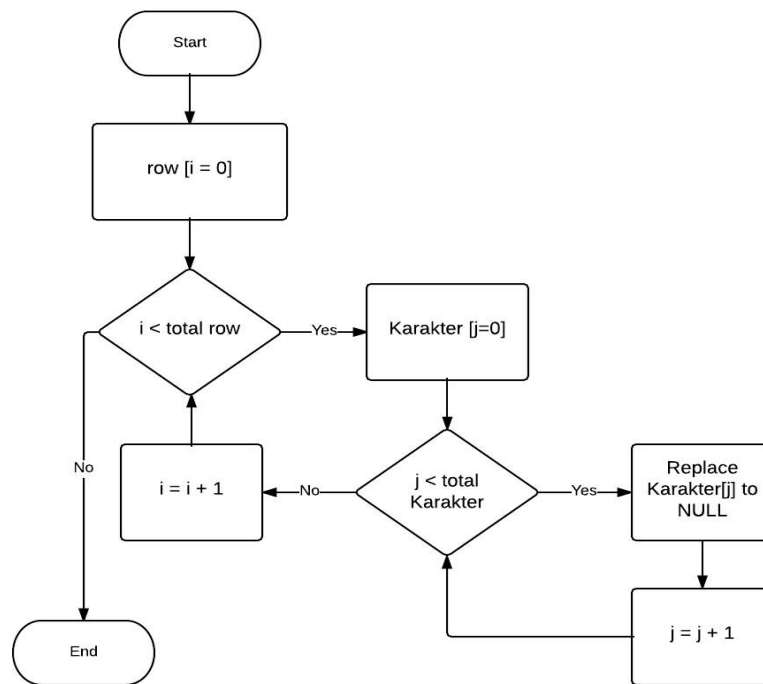
Tabel 4.2 Tabel Sebelum Dilakukan Proses Pra-*cleaning*

ID	Name	Address
109770	GUARD MEDAN SUN PLAZA, SPM	K.H.ZAINUL ARIFIN NO.7, JL
123627	GUARDIAN SUN PLAZA, APT	K.H.ZAINUL ARIFIN NO.7, JL

Tabel 4.3 Tabel Contoh Setelah Melewati Tahap Pra-*Cleaning*

ID	Name	Address
109770	GUARD MEDAN SUN PLAZA	KHZAINUL ARIFIN 7
123627	GUARDIAN SUN PLAZA	KHZAINUL ARIFIN 7

Dari tabel 3.3.2 dan tabel 3.3.3 di atas, dapat terlihat bahwa pada kolom *Name* tanda baca koma dan titel ‘SPM’ dan ‘APT’ telah dihapus. Selain itu, pada kolom *Address*, tanda baca titik, koma, dan kata ‘JL’ telah dihapus. Dengan demikian, setelah data dilakukan proses pra-*cleaning*, dihasilkan data yang bersih dari tanda baca dan titel seperti yang ada pada tabel 3.3.3. Berikut ini merupakan *flowchart* dari tahap pra-*cleaning*.



Gambar 4.2 *Flowchart Tahap Pra-cleaning*

II. Proses *Cleaning*

Setelah nilai dari setiap atribut dirapikan, proses *data cleaning* selanjutnya adalah tahap pendeteksian duplikasi dan tahap penghitungan nilai kemiripan antar *string*. Di mana kedua tahap ini merupakan dua tahap utama atau metode utama dalam sistem *data cleaning*.

a. Pendeteksian Duplikasi Data Dengan Metode SNM

Tahap pendeteksian duplikasi dilakukan dengan menerapkan Metode *Sorted Neighbourhood* sebagai berikut.

1. *Clustering Data*

Clustering data dilakukan dengan menerapkan *constant partitioning* di mana data dibagi berdasarkan area sehingga proses membandingkan *record* dilakukan pada setiap area. Misalnya, area Bandar Lampung terdiri atas 1.000 *record* dan area Bandung terdiri atas 800 *record*. Oleh karena itu, proses deteksi duplikat untuk Bandar Lampung hanya pada 1.000 *record* saja dan area Bandung

hanya pada 800 *record* saja. Dengan demikian, satu *record* yang ada dalam *database* konsumen tersebut tidak perlu dilakukan komparasi terhadap seluruh data yang ada. Tetapi, hanya pada data yang satu area saja.

2. Pembentukan Token

Dalam menentukan token dilakukan dengan menggunakan n huruf pertama pada tiap kata yang ada pada *string* yang ada di dalam *field*. Di mana nilai n akan di-input oleh pengguna ketika akan memulai proses deteksi duplikasi data. Seperti contoh tabel di bawah ini menggunakan $n = 3$.

Tabel 4.4 Tabel Contoh Setelah Proses Tokenisasi

ID	Name	Address
109770	GUA MED SUN PLA	KHZ ARI 7
123627	GUA SUN PLA	KHZ ARI 7

Berdasarkan tabel 4.4, terlihat proses pemotongan kata sebanyak 3 huruf dari setiap kata. Jika dibandingkan dengan tabel 4.3, terlihat token yang dihasilkan dengan menggunakan tiga huruf awal pada tiap kata dari tiap *string* yang ada di dalam *field*.

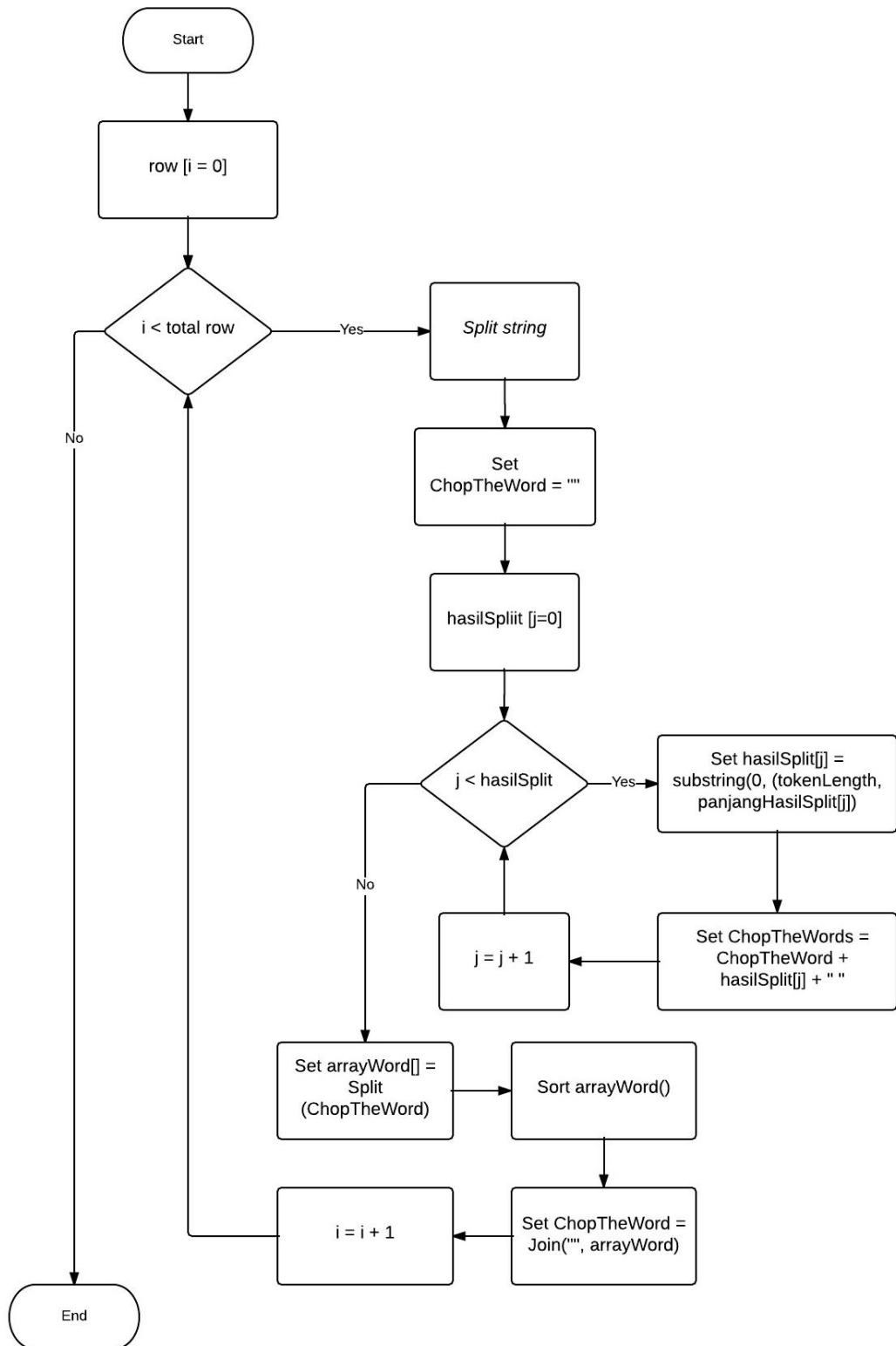
3. Mengurutkan *token*

Token yang berada pada tiap *field* kemudian diurutkan dan digabungkan seperti pada tabel contoh di bawah ini.

Tabel 4.5 Tabel Setelah Token Diurutkan dan Digabungkan

ID	Name	Address
109770	GUAMEDPLASUN	7ARIKHZ
123627	GUAPLASUN	7ARIKHZ

Berikut ini adalah *flowchart* dari proses pembentukan dan pengurutan token.



Gambar 4.3 *Flowchart* Tahap Tokenisasi

4. Menggabungkan *record*

Untuk menggabungkan *record* dilakukan dengan mengasumsikan sebuah *window* yang berukuran w bergerak melalui setiap *record* untuk membatasi proses perbandingan terhadap *record* yang berpotensi memiliki kemiripan data. Dimana nilai w yang digunakan adalah jumlah *record* dalam setiap area. Dalam kasus data di PT XYZ ini, data akan digabungkan dengan memunculkan kode baru, yaitu *Clean Code* yang sama pada dua atau lebih data yang kembar. Penomoran *Clean Code* akan dibahas setelah tahap penghitungan kemiripan antar *string* berikut ini.

b. Membandingkan Nilai Kemiripan *Record* Dengan Metode *N-Gram*

Untuk membandingkan *record* digunakan metode pendekatan *N-Gram*. Untuk menghitung nilai kemiripan digunakan rumus 2.1. Sebagai contoh, proses penghitungan kemiripan antar *string* dengan nilai $n = 2$ adalah sebagai berikut.

String 1 = GU UA AM ME ED DP PL LA AS SU UN = 11

String 2 = GU UA AP PL LA AS SU UN = 8

Jumlah gram yang sama = 7

Nilai kemiripan untuk *field Name* adalah $(2 \times 7) / (11+8) = 0.7$

String 1 = 7A AR RI IK KH HZ = 6

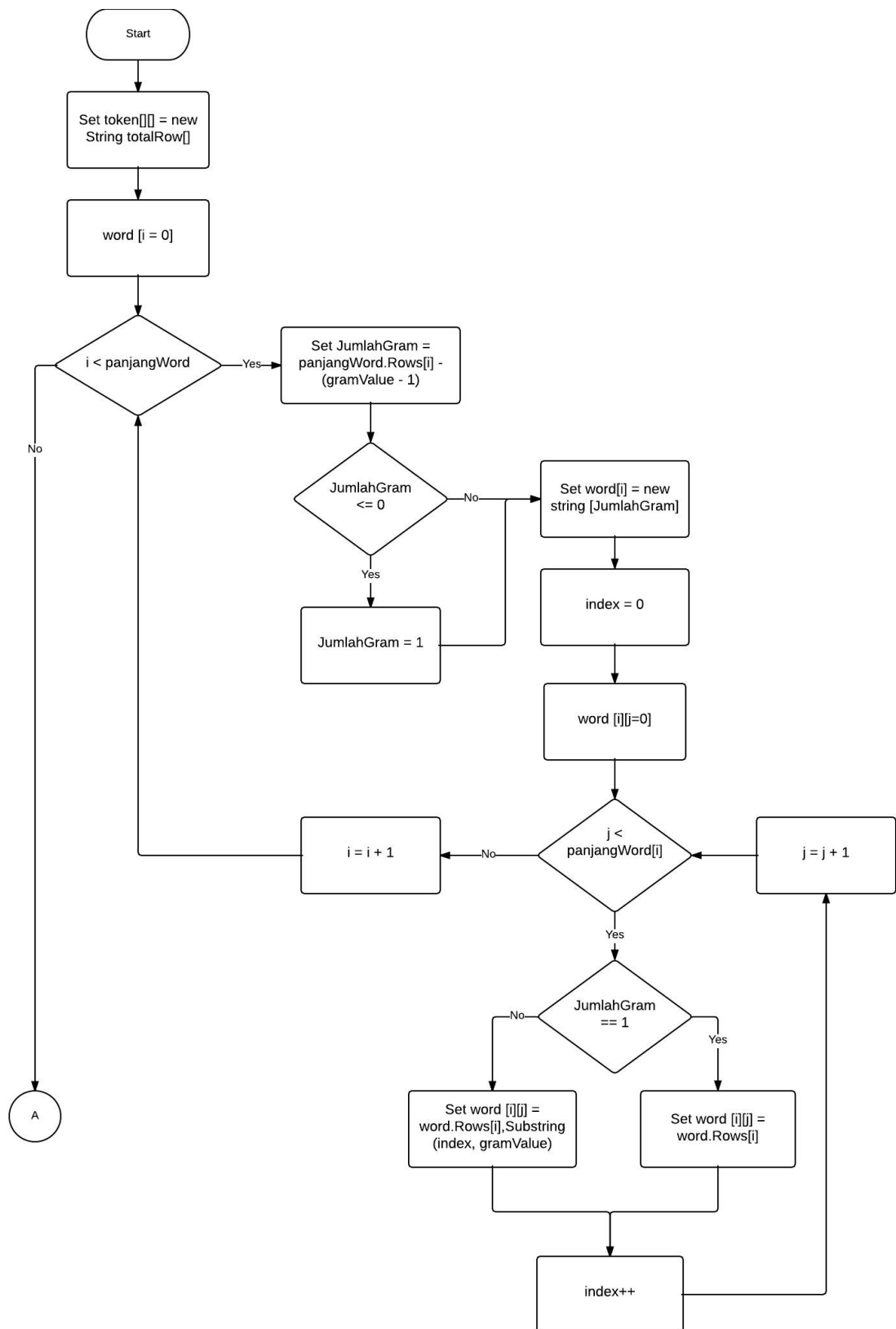
String 2 = 7A AR RI IK KH HZ = 6

Jumlah gram yang sama = 6

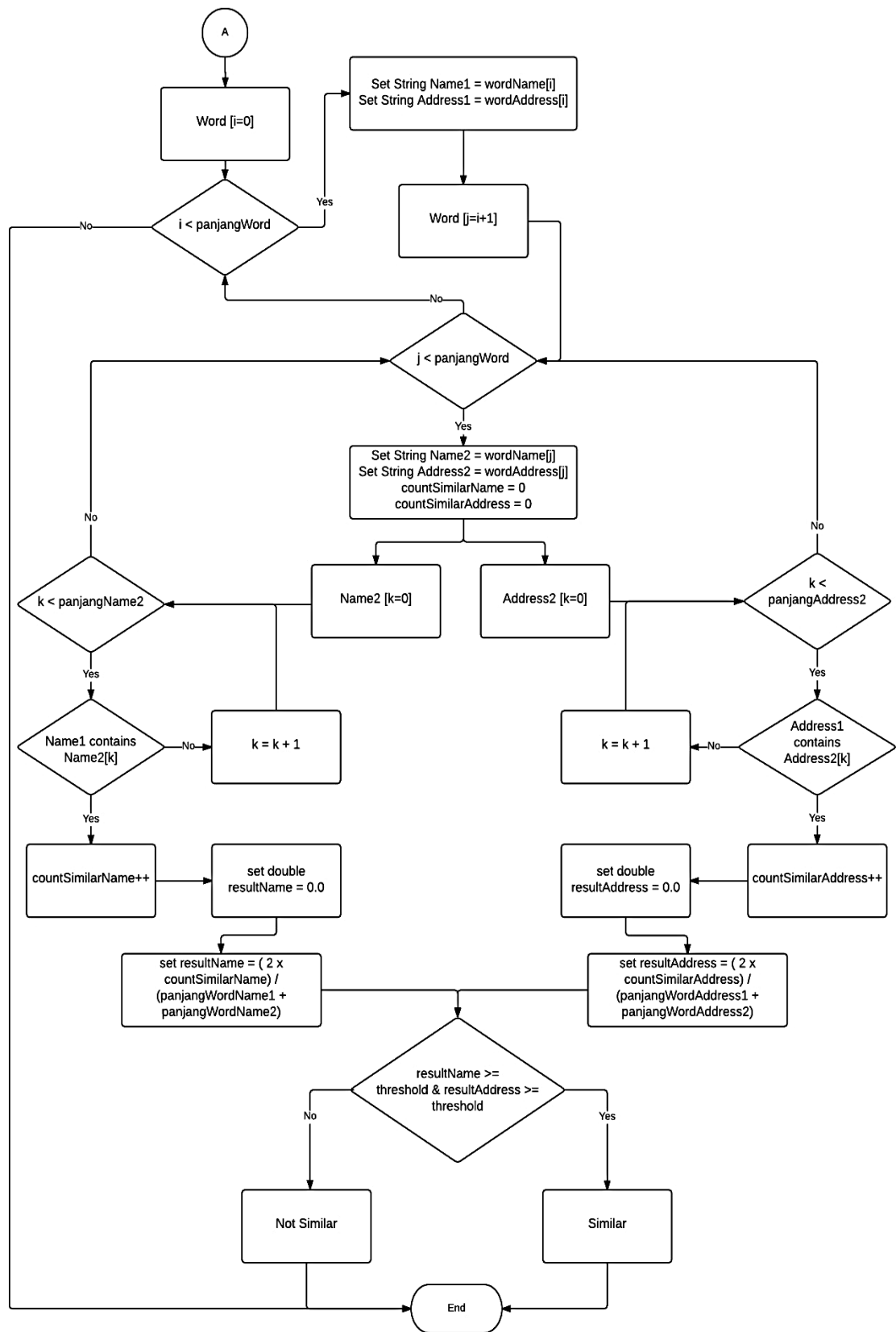
Nilai kesamaan untuk *field Address* adalah $(2 \times 6) / (6+6) = 1$

Diasumsikan nilai ambang batas (*threshold*) dari *field Name* dan *Address* adalah 0,6. Maka, *record* yang ada pada contoh di atas dinyatakan kembar karena nilai kemiripan *field Name* dan *Address*, keduanya melebihi nilai *threshold*.

Berikut ini adalah *flowchart* dari proses penghitungan kemiripan antar *string*.



Gambar 4.4 *Flowchart* Tahap Pemecahan Kata Berdasarkan Nilai N-Gram



Gambar 4.5 Flowchart Perhitungan Nilai Kemiripan Antar Record

III. *Result dan Report*

Hasil dari proses *data cleaning* dalam hal pendeteksian duplikasi data yaitu munculnya ID baru yang disebut dengan *Clean Code*. Tujuan pembuatan *Clean Code* adalah sebagai kode yang akan menyatukan data yang terdeteksi sebagai data kembar. Sehingga, dalam proses pembersihan data konsumen pada PT XYZ tidak ada proses penggabungan atau penghapusan (*merge/purge*) seperti pada sistem *data cleaning* pada umumnya. Hal ini dilakukan untuk meminimalisasi adanya kesalahan pemilihan data ketika proses penggabungan atau penghapusan. Berikut ini adalah contoh tabel setelah data telah dibersihkan.

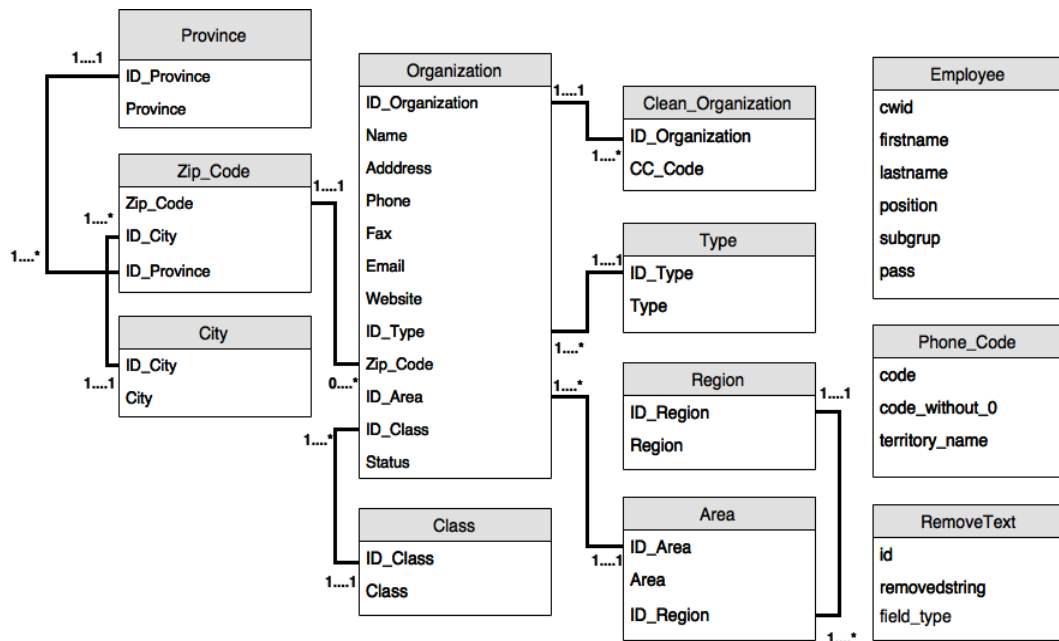
Tabel 4.6 Tabel *Clean*

Clean Code	Area	Organization ID	Name	Address
B101-1	Medan	109770	GUARD MEDAN SUN PLAZA, SPM	K.H.ZAINUL ARIFIN NO.7, JL
B101-1	Medan	123627	GUARDIAN SUN PLAZA, APT	K.H.ZAINUL ARIFIN NO.7, JL

Dari Tabel 4.6 di atas terlihat contoh tabel *clean*. Pada tabel di atas, *record* dengan *Clean Code* = B101-1 memiliki dua buah *Organization_ID* yang berbeda namun memiliki data yang duplikat. *Clean Code* inilah yang akan digunakan untuk menyatukan data yang terdeteksi sebagai data duplikat.

4.1.2 Perancangan *Database*

Rancangan *database* digunakan untuk menjelaskan hubungan antar data dalam basis data berdasarkan objek-objek dasar data yang mempunyai hubungan antar relasi. Berikut ini rancangan *database* sistem *data cleaning* master data konsumen PT XYZ.



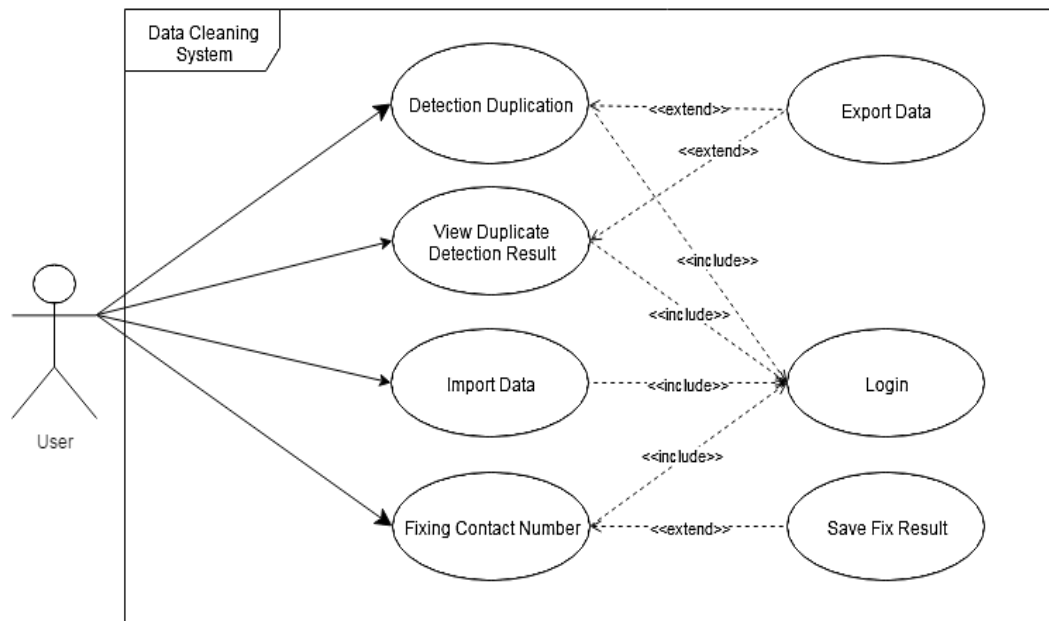
Gambar 4.6 Rancangan *Database* Sistem *Data Cleaning* Pada *Database* Master Data Konsumen PT XYZ

4.1.3 UML (*Unified Modelling Language*)

UML merupakan suatu bahasa yang digunakan untuk menentukan, memvisualisasikan, membangun, dan mendokumentasikan suatu sistem informasi dengan pemodelan program berorientasi objek (OOP) (Kroenke, dkk., 2003). Berikut ini merupakan diagram-diagram UML yang mempresentasikan perilaku sistem.

4.1.3.1 *Use Case Diagram*

Use case diagram merupakan diagram UML yang digunakan untuk memodelkan dan menyatakan unit fungsi atau layanan yang disediakan oleh sistem. *Use case* berfungsi untuk menggambarkan interaksi antara sistem dan aktor (*user*), termasuk pertukaran pesan dan tindakan yang dilakukan oleh sistem. Berikut ini diagram *use case* untuk sistem *data cleaning* dalam penelitian ini.



Gambar 4.7 Use Case Sistem Data Cleaning

Nama Use Case : **Login**

Aktor : User

Pre-condition : Aktor masuk ke dalam sistem

Post-condition : Aktor dapat login ke dalam sistem data cleaning

Deskripsi : Aktor melakukan proses login ke dalam sistem

Tabel 4.7 Deskripsi Aksi Aktor dan Respon Sistem Untuk Use Case Login

Aktor	Sistem
	2. Sistem menampilkan permintaan Username dan Password.
3. Aktor memasukkan Username dan Password.	4. Sistem melakukan proses validasi data yang di-input oleh aktor.
	5. Jika validasi benar, Aktor dapat masuk ke dalam sistem. Jika salah, Aktor menerima pesan bahwa username dan password yang dimasukkan oleh Aktor salah dan

	dapat memasukkan <i>username</i> dan <i>password</i> kembali.
--	---

Nama *Use Case* : ***Detection Duplication***
 Aktor : *User*
Pre-condition : Aktor telah masuk ke halaman *Duplicate Detection* dan telah melakukan *import data*.
Post-condition : Sistem mendeteksi duplikasi data konsumen dan menampilkan hasilnya.
 Deskripsi : Aktor melakukan proses deteksi duplikasi data dengan menggunakan Algoritma *Sorted Neighbourhood* dan *N-gram*.

Tabel 4.8 Deskripsi Aksi Aktor dan Respon Sistem Untuk *Use Case Duplicate Detection*

Aktor	Sistem
	1. Sistem memunculkan pilihan agar aktor dapat menginput teks atau karakter, nilai <i>threshold</i> , <i>token length</i> , dan <i>gram</i> .
2. Aktor menginput teks atau karkater kedalam <i>list removed text</i> , memilih nilai <i>threshold</i> , <i>token length</i> , dan <i>gram</i> .	
3. Aktor menekan tombol <i>start cleaning</i> .	
	4. Sistem melakukan proses <i>cleaning</i> .
	5. Sistem menampilkan hasil proses pendeteksian duplikasi data.

Nama Use Case : ***View Duplicate Detection Result***
 Aktor : *User*
 Pre-condition : Aktor telah masuk ke halaman *Duplicate Detection*.
 Post-condition : Sistem menampilkan data yang telah dideteksi duplikasi datanya.
 Deskripsi : Aktor melihat hasil data yang telah dideteksi.

Tabel 4.9 Deskripsi Aksi Aktor dan Respon Sistem Untuk *Use Case View Duplicate Detection Result*

Aktor	Sistem
1. Aktor menekan tombol <i>View Clean Data</i> .	
	2. Sistem menampilkan data yang telah dideteksi.

Nama Use Case : ***Import Data***
 Aktor : *User*
 Pre-condition : Aktor telah masuk ke halaman *Import Data*.
 Post-condition : Aktor memasukkan master data konsumen ke dalam *database*.
 Deskripsi : Aktor melakukan proses *import* data ke dalam *database*.

Tabel 4.10 Deskripsi Aksi Aktor dan Respon Sistem Untuk *Use Case Import Data*

Aktor	Sistem
	1. Sistem menyediakan kolom <i>upload file</i> .
2. Aktor memilih <i>file excel</i> yang akan di- <i>upload</i> dengan <i>template</i> yang telah	

ditentukan.	
3. Aktor menekan tombol <i>Save File</i> .	4. Data akan tersimpan dan di kolom <i>Choose Sheet</i> akan muncul pilihan <i>sheet</i> .
5. Aktor memilih <i>sheet</i> dimana data konsumen berada.	
6. Aktor memasukkan nilai <i>Start Row</i> dan <i>Start Column</i> dimana data yang ada di dalam <i>file</i> akan mulai dibaca oleh sistem. Kemudian menekan tombol <i>Import Data</i> .	7. Sistem melakukan validasi data. Jika benar, data akan berhasil masuk ke dalam sistem. Jika salah, sistem akan mengeluarkan notifikasi.

Nama Use Case : ***Fixing Contact Number***
 Aktor : *User*
 Pre-condition : Sistem masuk ke halaman *Fix Contact Number*.
 Post-condition : Sistem menampilkan format nomor telepon dan fax yang telah dirapikan.
 Deskripsi : Sistem merapikan format nomor telepon dan fax.

Tabel 4.11 Deskripsi Aksi Aktor dan Respon Sistem Untuk *Use Case Fixing Contact Number*

Aktor	Sistem
1. Aktor membuka menu <i>Fix Contact Number</i> lalu menekan tombol <i>Start Cleaning</i> .	
	2. Sistem melakukan proses merapikan format penulisan.
	3. Sistem menampilkan data lama dan data baru yang telah dibersihkan.

Nama Use Case : ***Save Fix Result***

Aktor : *User*

Pre-condition : Sistem telah menampilkan data *phone* dan *fax* yang telah dirapikan formatnya.

Post-condition : Data *phone* dan *fax* yang telah dirapikan disimpan ke dalam *database*.

Deskripsi : Aktor dapat menyimpan hasil format penulisan yang telah dirapikan.

Tabel 4.12 Deskripsi Aksi Aktor dan Respon Sistem Untuk *Use Case Save Fix Result*

Aktor	Sistem
	1. Sistem menampilkan data <i>phone</i> dan <i>fax</i> yang telah dirapikan formatnya.
2. Aktor menekan tombol <i>Save</i> .	
	3. Data telah berhasil disimpan ke dalam <i>database</i> .

Nama *Use Case* : ***Export Data***

Aktor : *User*

Pre-condition : Sistem telah menampilkan data yang telah dideteksi duplikasi datanya.

Post-condition : Data dapat diekspor ke dalam file *excel*.

Deskripsi : Aktor dapat melakukan ekspor data yang telah dideteksi duplikasi datanya.

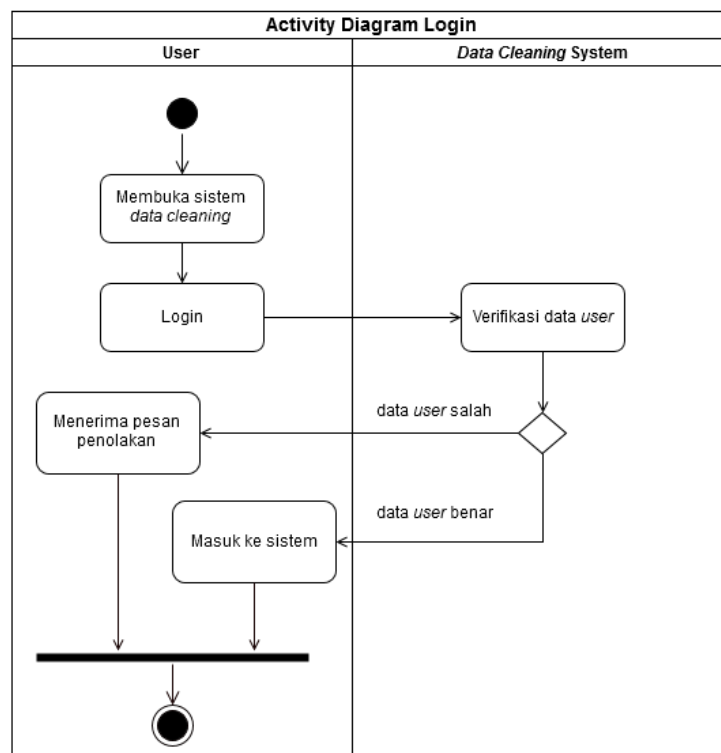
Tabel 4.13 Deskripsi Aksi Aktor dan Respon Sistem Untuk *Use Case Edit Export Data*

Aktor	Sistem
-------	--------

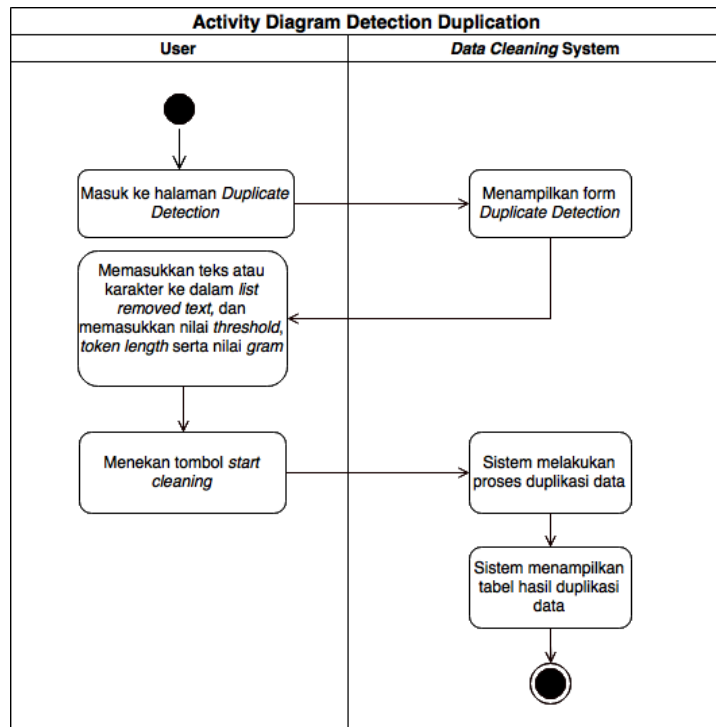
	1. Sistem menampilkan data yang telah dideteksi duplikasi datanya atau dengan menekan tombol <i>View Clean Data</i> .
2. Aktor menekan tombol <i>Export Data</i> .	3. Sistem memunculkan tampilan untuk mengunduh <i>file</i> dan menyimpan <i>file</i> .
4. Aktor mengunduh file dengan menekan tombol <i>OK</i> .	5. Sistem mengunduh <i>file excel</i> dan sistem berhasil diekspor ke dalam <i>file excel</i> .

4.1.3.2 Activity Diagram

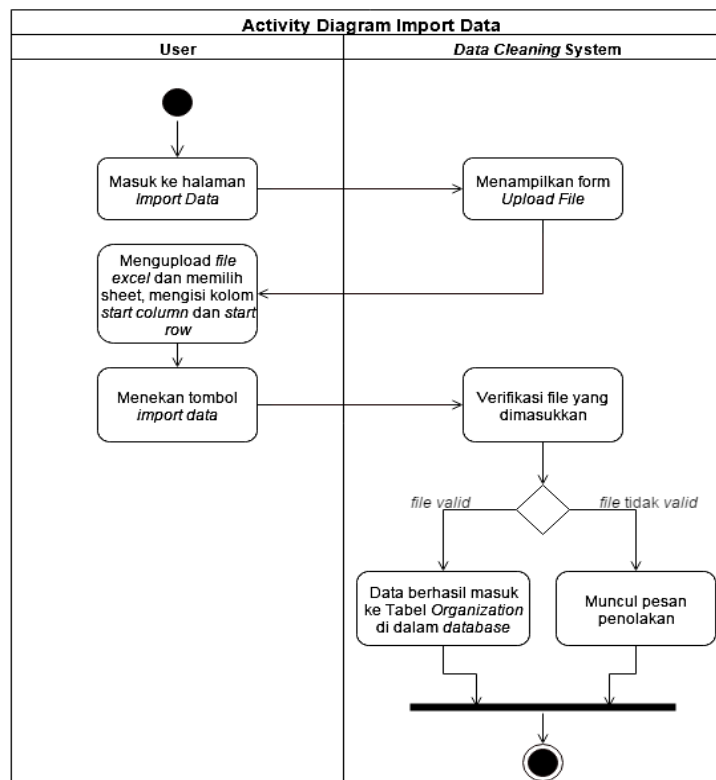
Diagram ini akan menjelaskan tentang seluruh tahapan alur kerja dengan menggambarkan berbagai alur aktivitas dalam sistem yang dirancang. Berikut ini *activity diagram* yang terdapat pada sistem *data cleaning* PT XYZ.



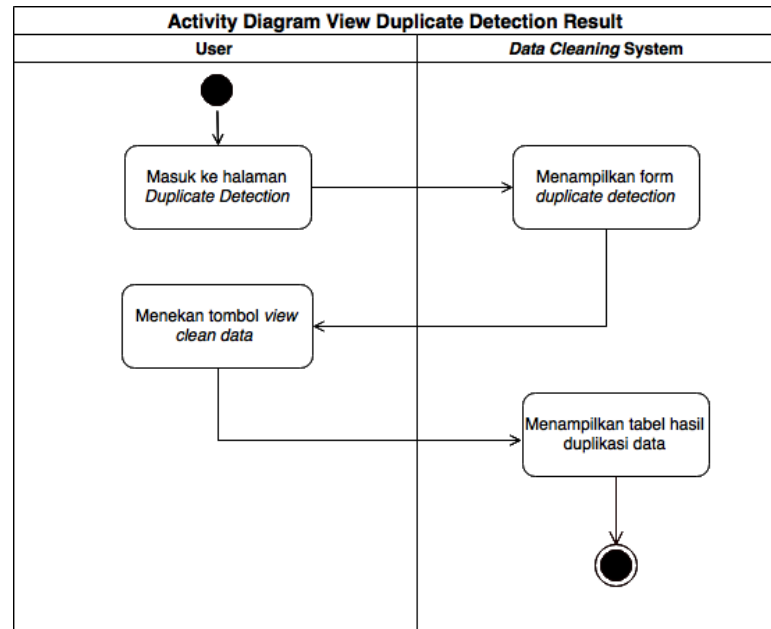
Gambar 4.8 Activity Diagram Login



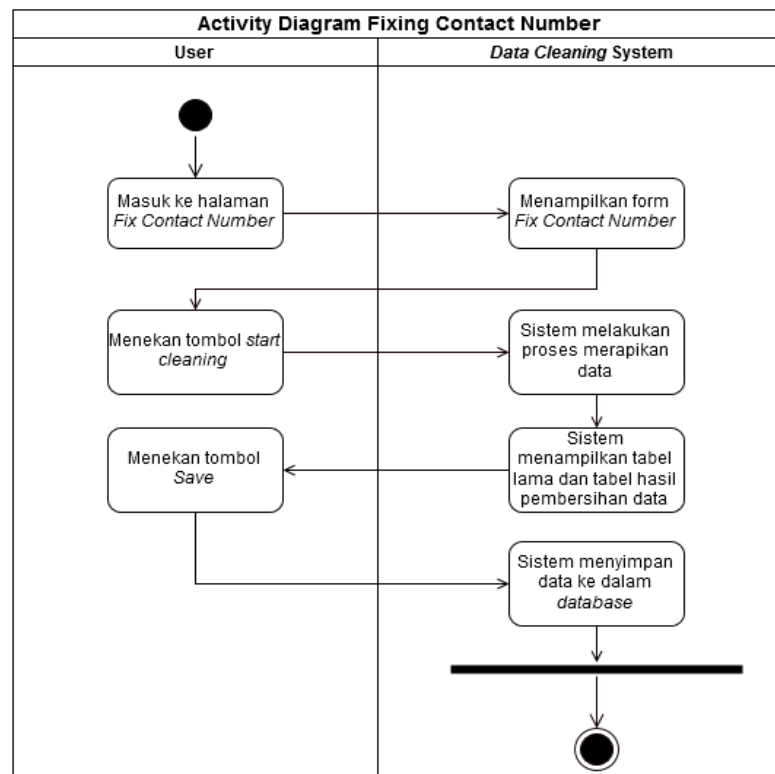
Gambar 4.9 Activity Diagram Detection Duplication



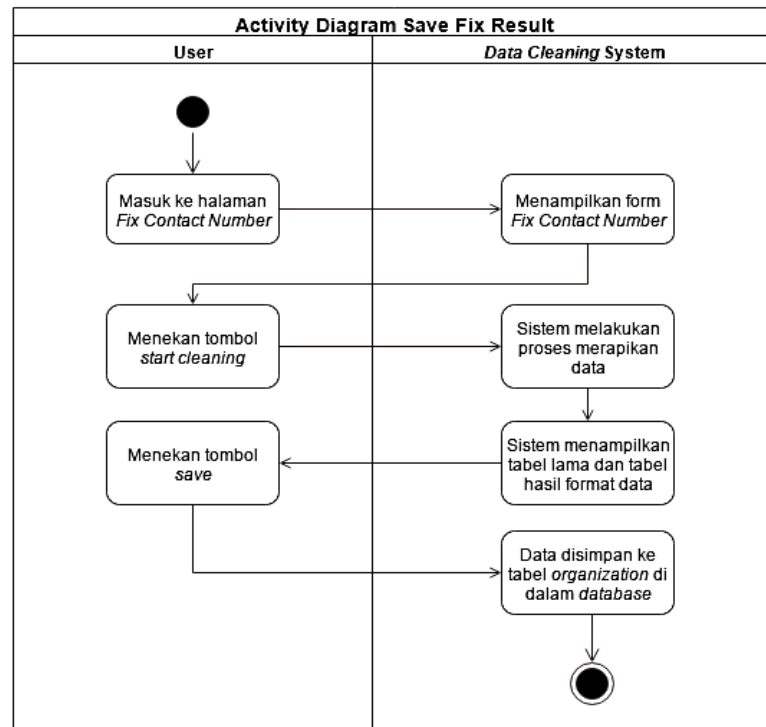
Gambar 4.10 Activity Diagram Import Data



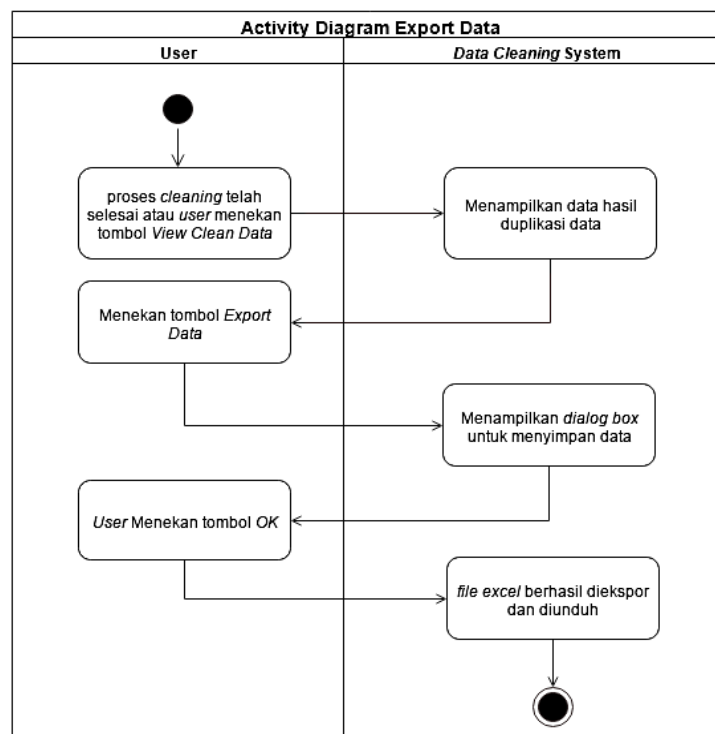
Gambar 4.11 Activity Diagram View Duplicate Detection Result



Gambar 4.12 Activity Diagram Fixing Contact Number



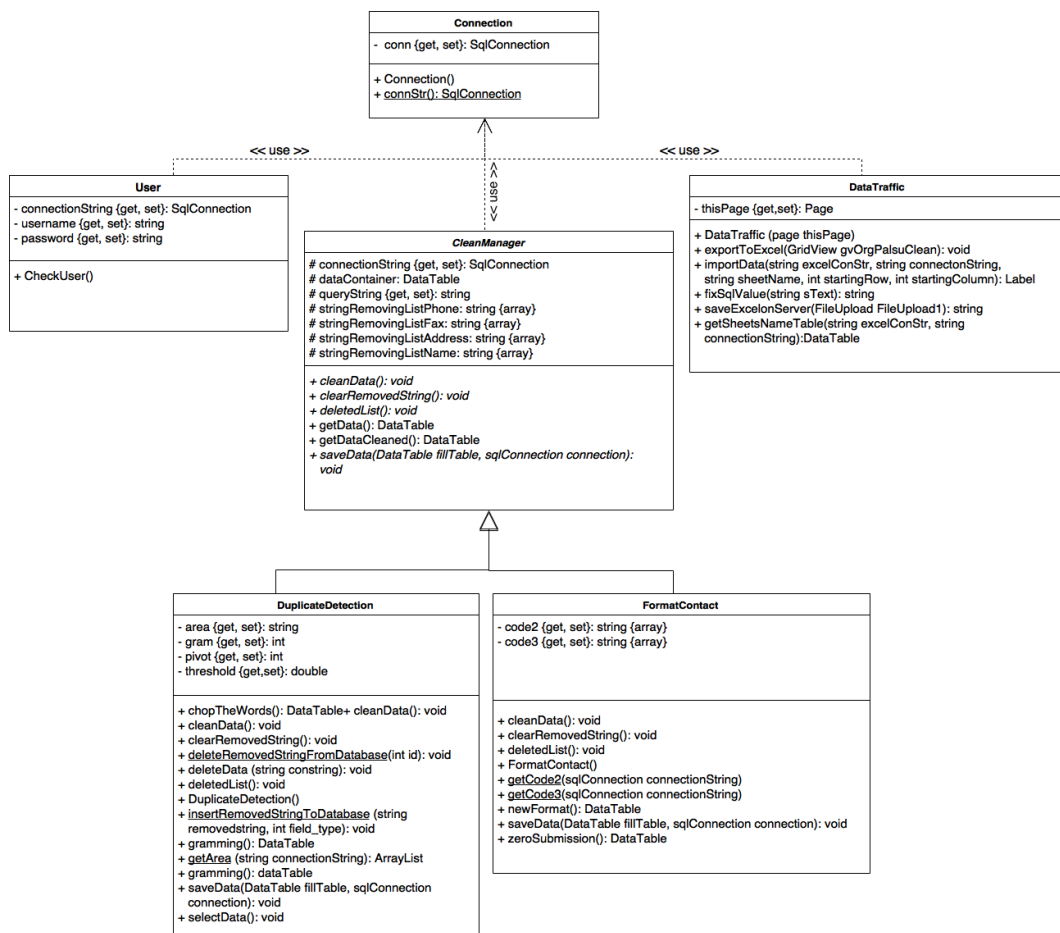
Gambar 4.13 Activity Diagram Save Fix Result



Gambar 4.14 Activity Diagram Export Data

4.1.3.3 Class Diagram

Class Diagram adalah diagram UML yang menggambarkan kelas-kelas dalam sebuah sistem dan hubungannya antara satu dengan yang lain, serta terdapat pula atribut dan operasi di dalamnya. Berikut ini *class diagram* sistem *data cleaning* pada penelitian ini.



Gambar 4.15 *Class Diagram*

Berdasarkan gambar 3.2.11 di atas terdapat delapan kelas yang digunakan dalam pembuatan sistem *data cleaning* pada penelitian ini. Berikut ini penjelasan terkait setiap kelas pada gambar di atas.

1. *Connection*, kelas yang digunakan untuk menghubungkan koneksi ke *database* master data konsumen.
2. *DataTraffic*, kelas yang digunakan untuk proses impor dan ekspor data.
3. *CleanManager*, kelas yang digunakan sebagai manajemen data untuk melakukan proses *cleaning*. Kelas ini merupakan super kelas dari kelas *DuplicateDetection* dan *FormatContact*.
4. *DuplicateDetection*, kelas yang digunakan untuk memproses deteksi duplikasi data.
5. *FormatContact*, kelas yang digunakan untuk memproses perubahan format telepon dan *fax*.
6. *User*, kelas yang digunakan sebagai pengelola data *user* dalam menggunakan sistem *data cleaning*.

4.2 Implementasi

Tahap implementasi sistem akan mewujudkan hasil perancangan yang telah dipaparkan pada Bab 3 ke dalam pemrograman yang merupakan kesatuan interaksi dari elemen-elemen yang memiliki tujuan yang diinginkan pada sistem *data cleaning* PT XYZ.

4.2.3 Implementasi Sistem

Implementasi sistem diharuskan memenuhi semua kebutuhan yang ada pada tahap analisa dan perancangan. Pada tahap ini akan dijelaskan spesifikasi perangkat keras dan perangkat lunak yang dibutuhkan untuk dapat mengimplementasi sistem *data cleaning* di PT XYZ.

a. Perangkat Keras / *Hardware*

Kebutuhan perangkat keras yang dibutuhkan untuk mengimplementasikan sistem *data cleaning* minimal memiliki spesifikasi komputer (PC) sebagai berikut:

- Prosesor Intel Core i-5 @ 2.4 GHz
- Memori dengan RAM 2 GB

- 32-bit *Windows Operating System*
- *Hard Disk 320 GB*

b. Perangkat Lunak / *Software*

Perangkat lunak yang dibutuhkan untuk mengimplementasikan sistem *data cleaning* adalah sebagai berikut:

- Sistem basis data *Microsoft SQL Server*
- IDE *Microsoft Visual Studio 2010*
- IIS *version 8.0* sebagai *web server*
- Sistem Operasi *Windows 7*
- *Web browser Internet Explorer* atau *Mozilla Firefox*

4.2.4 Implementasi GUI (*Graphical User Interface*)

GUI merupakan suatu antar muka pada sistem operasi atau komputer yang menggunakan menu grafis agar mempermudah penggunaanya untuk berinteraksi dengan komputer atau sistem operasi. Pada sub-bab ini terdapat GUI untuk *user* sistem *data cleaning* di PT XYZ. Berikut ini adalah tampilan GUI yang telah diimplementasikan.

a. Halaman *Login*

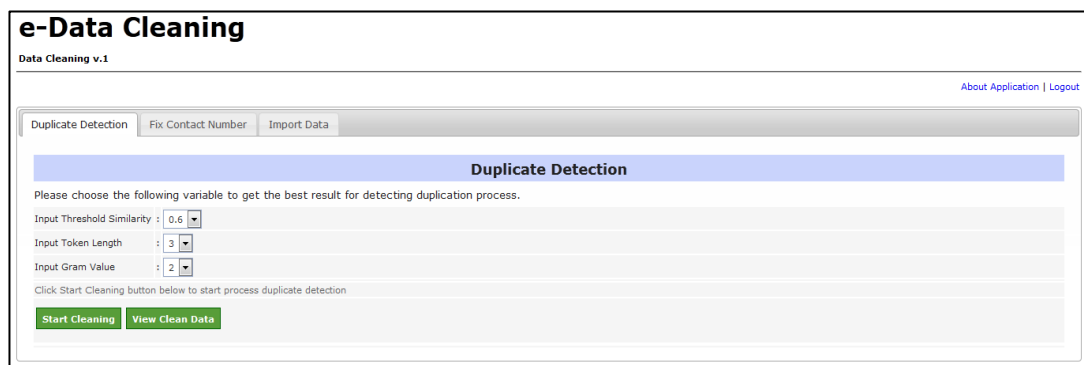
Tampilan *login* akan muncul ketika *user* pertama kali masuk ke dalam sistem. Berikut ini tampilan *login* pada sistem *data cleaning* PT XYZ.

The screenshot shows a web application window titled "e-Data Cleaning" with a subtitle "Data Cleaning v.1". In the top right corner, there is a link that says "About Application". The main area of the window contains a login form. The form has a title "Login" in a blue header. Below the title, there are two input fields: "Employee ID" and "Password", each preceded by a colon. Below the input fields is a green button labeled "Login".

Gambar 4.16 Tampilan *Login*

b. Halaman Utama Sistem *Data Cleaning* PT XYZ

Halaman pertama yang ditampilkan pada saat *user* berhasil membuka sistem *data cleaning* adalah sebagai berikut.



Gambar 4.17 Halaman Utama Sistem *Data Cleaning*

Pada halaman utama, terdapat tiga buah *tab*, yaitu *Duplicate Detection* dan *Fix Contact Number* dan *Import Data*. Untuk melakukan *cleaning data*, *user* dapat melakukan *import data* ke *database* terlebih dahulu. Jika memang data telah di-*import* sebelumnya, maka *user* dapat langsung melakukan *cleaning data* dengan memilih nilai *threshold*, *token length* dan *gram value*, kemudian menekan tombol *Start Cleaning*.

Untuk mengetahui data yang telah dibersihkan, *user* dapat menekan tombol *View Clean Data*. Kemudian, akan muncul tampilan halaman seperti berikut.

Duplicate Detection

Fix Contact Number

Import Data

View Data

All

Area

All

Total Data

: 40

Duplicate Detection Result

ID Organization	Name	Address	Area	Clean Code
39635	TIARA, APT	LETTU ROHANI NO.20 KEDATON, JL	Bandar Lampung	B107-1
131994	TIARA, APT	LETTU ROHANI NO.20 KEDATON, JL	Bandar Lampung	B107-1
489201	FAMILY JAYA, APT	UNTUNG SUROPATI NO 1 LABUHAN RATU,	Bandar Lampung	B107-2
39736	AL-HIJRAH, TOB	PEMUDA NO.2, JL	Bandar Lampung	B107-3
107966	PADUSSHI, TKLO	GAJAH MADA NO.67-A (SIMPANG Dr.HARU	Bandar Lampung	B107-4
149162	NOVA, APT	URIP SUMOHARJO NO.92 SUKARAME, JL	Bandar Lampung	B107-5
487977	NATAHARI, APT	SAM RATULANGI NO. 75 C GEDONG AIR,	Bandar Lampung	B107-6
375707	KONDANG SEHAT, APT	OLAH RAGA NO.01 KUNCUP, JL	Bandar Lampung	B107-7
366611	BERSAUDARA, TKOS	KESEHATAN NO.116 LINGKUNGAN III RT,	Bandar Lampung	B107-8
107696	RIDHO ASIH, APT	RAYA PEKALONGAN NO.9, JL	Bandar Lampung	B107-9
417664	TIFA FARMA, APT	DESA PEKALONGAN KEC. PEKALONGAN	Bandar Lampung	B107-10
355514	PUTRA ABADI, TKOS	KESEHATAN NO.1220 DEPAN APT. PRINGSE	Bandar Lampung	B107-11

Export Data

Gambar 4.18 Tampilan *View Clean Data*

Jika *user* ingin memfilter data yang ditampilkan, *user* dapat memilih *drop down list View Data* dan *Area*. Seperti pada gambar berikut.

View Data	: Duplicate	
Area	: All	
Total Data	: All	
	Bandar Lampung	
	Bandung 1	
	Bandung 2	
	Banjarmasin	
	Batam	
	Bogor	
	Cakung Medical	
	Cakung MO	
	Ciputat	
	Cirebon	
	Denpasar	
	Jakarta	
	Jambi	
	Jayapura	
	Jember	
	Kediri	
	Kupang	
	Makassar	
	Malang	
	SAUDARA TOR	
	PATTIMUF	

Gambar 4.19 Tampilan *Drop Down List View Data* dan *Area*

Kemudian, *user* juga dapat mengekspor data yang tampil ke dalam format *file excel* dengan menekan tombol *Export Data* yang terdapat di pojok kiri bawah halaman *View Clean Data*.

c. Halaman *Fix Contact Number*

Pada halaman ini, akan ditampilkan fitur untuk merapikan format *phone* dan *fax* yang ada pada master data konsumen PT XYZ divisi *Consumer Care*. Berikut ini tampilan ketika membuka *tab Fix Contact Number*.

e-Data Cleaning

Data Cleaning v.1 About Application | Logout

Fix Contact Number

Fix Contact Number

Click start fixing button to start process repairing phone and fax number format

Start Fixing

Gambar 4.20 Halaman *Fix Contact Number*

Ketika masuk ke halaman ini, *user* dapat langsung menekan tombol *Start Fixing* untuk merapikan data *phone* dan *fax* yang ada pada master data konsumen PT XYZ. Kemudian, akan muncul hasil proses merapikan data seperti berikut.

e-Data Cleaning

Data Cleaning v.1 About Application

Fix Contact Number

Fix Contact Number Result

Old Data

Organization ID	Phone	Fax
85515	(0361) 7151140	
62831	0283671163	
102210	02470780193	
486352	061-6842299	
31887	061-4527923	
363364	061-6640148	
250300	061-91382091	
84075	08126420889	
34040	085270115444	

New Data

Organization ID	Phone	Fax
85515	0361-7151140	
62831	0283-6711-63	
102210	0247-0780-193	
486352	0616-6422-99	
31887	0614-5279-23	
363364	0616-6401-48	
250300	0619-1382-091	
84075	0812-6420-889	
34040	0852-7011-5444	

Save

Gambar 4.21 Tampilan Hasil *Fix Contact Number*

Hasil dari proses merapikan data *phone* dan *fax* terdapat di sebelah kanan. Sedangkan, data sumber (data sebelum dirapikan) terdapat di sebelah kiri. Untuk menyimpan hasilnya ke dalam *database*, maka *user* dapat menekan tombol *Save*.

d. Halaman *Import Data*

Pada halaman ini, akan ditampilkan halaman untuk mengimpor data konsumen Divisi *Consumer Care* PT XYZ. Untuk mengimpor data, *user* harus menggunakan *template* khusus agar data akan masuk ke dalam *database* dengan benar. *Template* dapat dilihat pada lampiran 4. Berikut ini adalah tampilan halaman *Import Data*.

e-Data Cleaning

Data Cleaning v.1

[About Application](#) | [Logout](#)

Import Data

Import Outlet Customer Data

Please attach only file excel in the provided field below.

Document : No file selected.

Choose Sheet :

Start Column :

Start Row :

Gambar 4.22 Tampilan Halaman *Import Data*

Kemudian, untuk mengimpor data, *user* harus melampirkan *file* dan melengkapi beberapa kolom yang tersedia, seperti pada gambar di bawah ini.

e-Data Cleaning

Data Cleaning v.1

[About Application](#) | [Logout](#)

Import Data

Import Outlet Customer Data

Please attach only file excel in the provided field below.

Document : Data Organization untuk Data Cleaning update 1.xlsx

Choose Sheet :

Start Column :

Start Row :

Gambar 4.23 Tampilan Pengisian Halaman *Import Data*

Setelah kolom dilengkapi, selanjutnya *user* menekan tombol *import data*. Sistem *data cleaning* kemudian akan memroses dan jika berhasil, maka akan keluar notifikasi atau pesan bahwa data telah berhasil masuk ke dalam *database*.

4.2.5 Implementasi Algoritma

4.2.5.1 Implementasi Algoritma SNM

Dalam pembuatan sistem *data cleaning*, algoritma SNM digunakan untuk mendeteksi duplikasi data dengan membuat suatu token khusus pada kolom *Name* dan *Address*. Berikut ini implementasi metode SNM ke dalam bentuk *code* pemrograman yang terdiri atas beberapa tahap.

1. Pembersihan Data

Pembersihan data disebut sebagai tahap *pre-cleaning*. Data sumber yang masuk pertama kali akan melalui tahap *pre-cleaning* di mana tahap ini mencakup pembersihan data dari beberapa *titel* konsumen dan tanda baca. Berikut implementasi *code* dari tahap pembersihan data untuk tiap *field Name* dan *Address*.

```
protected void cleanData()
{
    for (int i = 0; i < dataContainer.Rows.Count; i++)
    {
        dataContainer.Rows[i]["name"] =
            dataContainer.Rows[i]["name"].ToString().ToUpper();
        dataContainer.Rows[i]["address"] =
            dataContainer.Rows[i]["address"].ToString().ToUpper();

        for (int j = 0; j < stringRemovingListName.Count; j++)
        {
            dataContainer.Rows[i]["name"] =
                dataContainer.Rows[i]["name"].ToString().Replace((string)stringR
                    emovingListName[j], "");
        }
        for (int j = 0; j < stringRemovingListAddress.Count; j++)
        {
            dataContainer.Rows[i]["address"] =
                dataContainer.Rows[i]["address"].ToString().Replace((string)stri
                    ngRemovingListAddress[j], "");
        }
    }
}

public override void deletedList()
{
    SqlConnection conn = Connection.connStr();
    String queryString = "SELECT removedstring WHERE field_type = '1'";
    String queryString2 = "SELECT removedstring WHERE field_type = '2'";

    SqlCommand command = new SqlCommand(queryString, conn);
    SqlCommand command2 = new SqlCommand(queryString2, conn);

    conn.Open();
    SqlDataReader reader = command.ExecuteReader();
    while (reader.HasRows)
    {
        stringRemovingListName.Add(reader.GetString(0));
    }
}
```

```

    }
    conn.Close();
    conn.Open();
    reader = command2.ExecuteReader();
    while (reader.HasRows)
    {
        stringRemovingListAddress.Add(reader.GetString(0));
    }
    conn.Close();
}
}

```

2. Tahap Pembentukan dan Pengurutan Token

Pembentukan token dilakukan pada *field Name* dan *Address*. Token dibentuk dengan memotong atau mencomot beberapa huruf dari tiap kata. Misalnya, jika *field Name* berisi “SUMBER UTAMA FARMA” dan panjang token (*pivot*) adalah 3, maka akan dibentuk token “SUM UTA FAR” seperti pada implementasi kode di bawah ini.

```

public DataTable chopTheWords()
{
    DataTable maintable = sourceTable;

    for (int i = 0; i < maintable.Rows.Count; i++)
    {
        string name = maintable.Rows[i]["name"].ToString();
        string address = maintable.Rows[i]["address"].ToString();

        char[] delimiter = { ' ' };
        string[] hasilSplitName = name.Split(delimiter);
        string[] hasilSplitAddress = address.Split(delimiter);
        name = "";
        address = "";
        for (int j = 0; j < hasilSplitName.Length; j++)
        {
            hasilSplitName[j] = hasilSplitName[j].Substring(0, Math.Min(pivot,
            hasilSplitName[j].Length));
            name = name + hasilSplitName[j] + " ";
        }
        for (int j = 0; j < hasilSplitAddress.Length; j++)
        {
            hasilSplitAddress[j] = hasilSplitAddress[j].Substring(0,
            Math.Min(pivot, hasilSplitAddress[j].Length));
            address = address + hasilSplitAddress[j] + " ";
        }
    }
}

```

Setelah itu, token akan diurutkan pada tiap *field*. Misalnya, “SUM UTA FAR” diurutkan menjadi “FAR SUM UTA”. Kemudian, token akan digabungkan menjadi “FARSUMUTA”. Token final inilah yang nantinya akan

dibandingkan dengan *record* lainnya untuk mengetahui nilai kemiripannya. Berikut ini implementasi ke dalam bentuk kode.

```

        string[] arrName = name.Split(null);
        string[] arrAddress = address.Split(null);
        Array.Sort(arrName);
        Array.Sort(arrAddress);
        name = string.Join("", arrName);
        address = string.Join("", arrAddress);

        maintable.Rows[i]["name"] = name;
        maintable.Rows[i]["address"] = address;
    }
    return maintable;
}

```

4.2.5.2 Implementasi Algoritma *N-Gram*

Algoritma *N-Gram* diterapkan pada sistem *data cleaning* dalam penelitian ini untuk menentukan nilai kemiripan antar *record*. Jika nilai kemiripan melebihi batas *threshold* yang telah ditentukan, maka *record* yang dibandingkan dapat dinyatakan sebagai *record* yang kembar atau duplikat. Berikut ini penerapan algoritma *N-Gram* ke dalam bentuk *code* pemrograman.

```

public DataTable gramming()
{
    DataTable maintable = sourceTable;

    if (!maintable.Columns.Contains("similarity"))
    {
        maintable.Columns.Add("similarity");
    }
    string[][] gramName = new string[maintable.Rows.Count][];
    string[][] gramAddress = new string[maintable.Rows.Count][];
    for (int i = 0; i < gramName.Length; i++)
    {
        Int64 stringLengthName = maintable.Rows[i]["name"].ToString().Length - (gram - 1);
        if (stringLengthName <= 0)
        {
            stringLengthName = 1;
        }
        gramName[i] = new string[stringLengthName];
        int index = 0;
        for (int j = 0; j < gramName[i].Length; j++)
        {
            if (stringLengthName == 1)
            {
                gramName[i][j] = maintable.Rows[i]["name"].ToString();
            }
            else
            {
                gramName[i][j] = maintable.Rows[i]["name"].ToString().Substring(index, gram);
            }
        }
    }
}

```

```

        }
        index++;
    }

    Int64 stringlengthAddress = maintable.Rows[i]["address"].ToString().Length -
    (gram - 1);
    if (stringlengthAddress <= 0)
    {
        stringlengthAddress = 1;
    }
    gramAddress[i] = new string[stringlengthAddress];
    index = 0;
    for (int j = 0; j < gramAddress[i].Length; j++)
    {
        if (stringlengthAddress == 1)
        {
            gramAddress[i][j] = maintable.Rows[i]["address"].ToString();
        }
        else
        {
            gramAddress[i][j] = maintable.Rows[i]["address"].ToString().
            Substring(index, gram);
        }
        index++;
    }
}

int simicount = 1;
for (int i = 0; i < gramName.Length; i++)
{
    string simila = "";
    string id = area + "-" + simicount;
    if (maintable.Rows[i]["similarity"] != DBNull.Value)
    {
        simila = maintable.Rows[i]["similarity"].ToString();
    }
    else
    {
        maintable.Rows[i]["similarity"] = id;
    }
    if (simila == "")
    {
        simicount++;
        string[] pihak1Name = gramName[i];
        string[] pihak1Address = gramAddress[i];

        for (int j = i + 1; j < gramName.Length; j++)
        {
            string[] pihak2Name = gramName[j];
            string[] pihak2Address = gramAddress[j];
            int countSimilarName = 0;
            int countSimilarAddress = 0;
            if (maintable.Rows[j]["similarity"] == DBNull.Value)
            {
                for (int k = 0; k < pihak2Name.Length; k++)
                {
                    if (pihak1Name.Contains(pihak2Name[k]))
                    {
                        countSimilarName++;
                    }
                }
            }
            double resultName = 0.0;
            try
            {
                resultName = ((double)2 * countSimilarName)/(pihak1Name.Length
                + pihak2Name.Length);
            }
        }
    }
}

```


phone dan *fax* akan diuji untuk membuktikan bahwa implementasi kode yang diterapkan pada program telah dapat berjalan sesuai fungsionalitas yang telah ditentukan. Berikut ini adalah pembahasannya yang akan dibagi ke dalam beberapa poin di bawah ini.

4.3.1.1 Pengujian Algoritma SNM

a. Fungsi *cleanData()*

Nama Fungsi	: <i>cleanData()</i>
Deskripsi	: Fungsi yang digunakan untuk membersihkan <i>field Name</i> dan <i>Address</i> dari beberapa beberapa titel toko, kata, karakter atau tanda baca.
Input	: <i>String</i> yang ada pada <i>field Name</i> dan <i>Address</i>
Output	: <i>String</i> yang ada pada <i>field Name</i> dan <i>Address</i> telah dibersihkan dari beberapa titel toko, kata, karakter atau tanda baca.

Tabel 4.14 Tabel Hasil Pengujian Algoritma SNM – Fungsi *cleanData()*

Input	Expected Output	Actual Output	Status
Row[i]	Row[i]	Row[i]	
Name: Area: B102 (Pekan Baru) [0] => MITRA FARMA, TOB [1] => GIANT EXPRESS NANGKA Area: B104 (Padang) [0] => RAHMAT ESA, APT [1] => RAHMAD ESA, APT Address: Area: B102 (Pekan	Name: Area: B102 (Pekan Baru) [0] => MITRA FARMA [1] => GIANT EXPRESS NANGKA Area: B104 (Padang) [0] => RAHMAT ESA [1] => RAHMAD ESA Address: Area: B102 (Pekan Baru) [0] => SUDIRMAN [1] => NANGKA TUANKU TAMBUSAI 27 Area: B104 (Padang)	Name: Area: B102 (Pekan Baru) [0] => MITRA FARMA [1] => GIANT EXPRESS NANGKA Area: B104 (Padang) [0] => RAHMAT ESA [1] => RAHMAD ESA Address: Area: B102 (Pekan Baru) [0] => SUDIRMAN [1] => NANGKA TUANKU TAMBUSAI 27 Area: B104 (Padang)	Berhasil

Input	Expected Output	Actual Output	Status
Baru) [0] => SUDIRMAN, JL [1] => NANGKA TUANKU TAMBUSAI NO. 27, JL Area: B104 (Padang) [0] => PASAR KOTO BARU ABAI SIAT, KOMP. [1] => PASAR KOTO BARU, DHARMASRAYA, KOMP	[0] => PASAR KOTO BARU ABAI SIAT KOMP [1] => PASAR KOTO BARU DHARMASRAYA KOMP	[0] => PASAR KOTO BARU ABAI SIAT KOMP [1] => PASAR KOTO BARU DHARMASRAYA KOMP	

b. Fungsi *chopTheWords()*

Nama Fungsi : *chopTheWords()*

Deskripsi : Fungsi yang digunakan untuk membentuk token pada setiap *field Name* dan *Address*. Dimana nilai panjang token yang diuji adalah 3, 4, dan 5.

Input : *String* hasil keluaran dari fungsi *cleanData()*

Output : *String* yang ada pada *field Name* dan *Address* telah terbentuk token sesuai dengan nilai panjang token yang diinput *user*.

Tabel 4.15 Tabel Hasil Pengujian Algoritma SNM – Fungsi *ChopTheWords()*

Input		Expected Output	Actual Output	Status
Token Length	Row[i]	Row[i]	Row[i]	
3	Name: Area: B102 (Pekan Baru) [0] => MITRA FARMA	Name: Area: B102 (Pekan Baru) [0] => FARMIT [1] => EXPGIANAN	Name: Area: B102 (Pekan Baru) [0] => FARMIT [1] => EXPGIANAN	Berhasil

	Input	Expected Output	Actual Output	Status
	<p>[1] => GIANT EXPRESS NANGKA</p> <p>Area: B104 (Padang)</p> <p>[0] => RAHMAT ESA</p> <p>[1] => RAHMAD ESA</p> <p>Address:</p> <p>Area: B102 (Pekan Baru)</p> <p>[0] => SUDIRMAN</p> <p>[1] => NANGKA TUANKU TAMBUSAI 27</p> <p>Area: B104 (Padang)</p> <p>[0] => PASAR KOTO BARU ABAI SIAT KOMP</p> <p>[1] => PASAR KOTO BARU DHARMASRAYA KOMP</p>	<p>Area: B104 (Padang)</p> <p>[0] => ESARAH</p> <p>[1] => ESARAH</p> <p>Address:</p> <p>Area: B102 (Pekan Baru)</p> <p>[0] => SUD</p> <p>[1] => 27NANTAMTUA</p> <p>Area: B104 (Padang)</p> <p>[0] => ABABARKOMKOTPAS SIA</p> <p>[1] => BARDHAKOMKOTPAS</p>	<p>Area: B104 (Padang)</p> <p>[0] => ESARAH</p> <p>[1] => ESARAH</p> <p>Address:</p> <p>Area: B102 (Pekan Baru)</p> <p>[0] => SUD</p> <p>[1] => 27NANTAMTUA</p> <p>Area: B104 (Padang)</p> <p>[0] => ABABARKOMKOTPAS SIA</p> <p>[1] => BARDHAKOMKOTPAS</p>	
4	<p>Name:</p> <p>Area: B102 (Pekan Baru)</p> <p>[0] => MITRA FARMA</p> <p>[1] => GIANT EXPRESS NANGKA</p> <p>Area: B104 (Padang)</p> <p>[0] => RAHMAT ESA</p> <p>[1] => RAHMAD ESA</p> <p>Address:</p>	<p>Name:</p> <p>Area: B102 (Pekan Baru)</p> <p>[0] => FARMMITR</p> <p>[1] => EXPRGIANNANG</p> <p>Area: B104 (Padang)</p> <p>[0] => ESARAHM</p> <p>[1] => ESARAHM</p> <p>Address:</p> <p>Area: B102 (Pekan Baru)</p> <p>[0] => SUDI</p> <p>[1] => 27NANGTAMBTUAN</p> <p>Area: B104 (Padang)</p>	<p>Name:</p> <p>Area: B102 (Pekan Baru)</p> <p>[0] => FARMMITR</p> <p>[1] => EXPRGIANNANG</p> <p>Area: B104 (Padang)</p> <p>[0] => ESARAHM</p> <p>[1] => ESARAHM</p> <p>Address:</p> <p>Area: B102 (Pekan Baru)</p> <p>[0] => SUDI</p> <p>[1] => 27NANGTAMBTUAN</p>	Berhasil

	Input	Expected Output	Actual Output	Status
	Area: B102 (Pekan Baru) [0] => SUDIRMAN [1] => NANGKA TUANKU TAMBUSAI 27 Area: B104 (Padang) [0] => PASAR KOTO BARU ABAI SIAT KOMP [1] => PASAR KOTO BARU DHARMASRAYA KOMP	[0] => ABAIBARUKOMPKOT OPASASIAT [1] => BARUDHARKOMPKOT OPASA	Area: B104 (Padang) [0] => ABAIBARUKOMPKOT OPASASIAT [1] => BARUDHARKOMPKOT OPASA	
5	Name: Area: B102 (Pekan Baru) [0] => MITRA FARMA [1] => GIANT EXPRESS NANGKA Area: B104 (Padang) [0] => RAHMAT ESA [1] => RAHMAD ESA Address: Area: B102 (Pekan Baru) [0] => SUDIRMAN [1] => NANGKA TUANKU TAMBUSAI 27 Area: B104 (Padang) [0] => PASAR KOTO BARU	Name: Area: B102 (Pekan Baru) [0] => FARMAMITRA [1] => EXPREGIANTNANGK Area: B104 (Padang) [0] => ESARAHMA [1] => ESARAHMA Address: Area: B102 (Pekan Baru) [0] => SUDIR [1] => 27NANGKTAMBU NK Area: B104 (Padang) [0] => ABAIBARUKOMPKOT OPASARSIAT [1] => BARUDHARMKOMP TOPASAR	Name: Area: B102 (Pekan Baru) [0] => FARMAMITRA [1] => EXPREGIANTNANGK Area: B104 (Padang) [0] => ESARAHMA [1] => ESARAHMA Address: Area: B102 (Pekan Baru) [0] => SUDIR [1] => 27NANGKTAMBU NK Area: B104 (Padang) [0] => ABAIBARUKOMPKOT OPASARSIAT [1] => BARUDHARMKOMP OTOPASAR	Berhasil

Input	Expected Output	Actual Output	Status
ABAI SIAT KOMP [1] => PASAR KOTO BARU DHARMASRAYA KOMP			

4.3.1.2 Pengujian Algoritma N-Gram

Nama Fungsi	: <i>gramming()</i>
Deskripsi	: Fungsi yang digunakan untuk membandingkan nilai kemiripan antar <i>record</i> . Dimana nilai panjang token = 3. Nilai N-Gram = 2, 3, 4. Sedangkan, nilai <i>threshold</i> = 0,6
Input	: <i>String</i> hasil keluaran dari fungsi <i>chopTheWords()</i>
Output	: Nilai kemiripan antar <i>record</i> . Jika nilai yang diperoleh sama dengan atau melebihi nilai <i>threshold</i> , yaitu 0,6. Maka, <i>record</i> yang dibandingkan dinyatakan kembar dan dibentuk <i>clean code</i> baru yang sama.

Tabel 4.16 Tabel Hasil Pengujian Algoritma SNM – Fungsi *gramming()*

Input		Expected Output	Actual Output	Status
Gram Value	Row[i]	Row[i]	Row[i]	
2	Name: Area: B102 (Pekan Baru) [0] => FARMIT [1] => EXPGIANAN Area: B104 (Padang) [0] => ESARAH [1] => ESARAH	String Gram Name: Area: B102 (Pekan Baru) [0] => FA AR RM MI IT [1] => EX XP PG GI IA AN NA AN Area: B104 (Padang) [0] => ES SA AR RA AH [1] => ES SA AR RA AH	String Gram Name: Area: B102 (Pekan Baru) [0] => FA AR RM MI IT [1] => EX XP PG GI IA AN NA AN Area: B104 (Padang) [0] => ES SA AR RA AH [1] => ES SA AR RA	Berhasil

Input	Expected Output	Actual Output	Status
Address: Area: B102 (Pekan Baru) [0] => SUD [1] => 27NANTAMTUA Area: B104 (Padang) [0] => ABABARKOMKOT PASSIA [1] => BARDHAKOMKOT PAS	Nilai Kemiripan Antar Name: Area: B102 (Pekan Baru) [0] & [1] => 0,0 Area: B104 (Padang) [0] & [1] => 1,0 String Gram Address: Area: B102 (Pekan Baru) [0] => SU UD [1] => 27 7N NA AN NT TA AM MT TU UA Area: B104 (Padang) [0] => AB BA AB BA AR RK KO OM MK KO OT TP PA AS SS SI IA [1] => BA AR RD DH HA AK KO OM MK KO OT TP PA AS Nilai Kemiripan Antar Address: Area: B102 (Pekan Baru) [0] & [1] => 0,0 Area: B104 (Padang) [0] & [1] => 0,6 Nilai Kemiripan Antar Record: Area: B102 (Pekan Baru) [0] & [1] => 0,0 Area: B104 (Padang) [0] & [1] => 0.8 Clean Code: Area: B102 (Pekan Baru) [0] => B102-1 [1] => B102-2 Area: B104 (Padang)	AH Nilai Kemiripan Antar Name: Area: B102 (Pekan Baru) [0] & [1] => 0,0 Area: B104 (Padang) [0] & [1] => 1,0 String Gram Address: Area: B102 (Pekan Baru) [0] => SU UD [1] => 27 7N NA AN NT TA AM MT TU UA Area: B104 (Padang) [0] => AB BA AB BA AR RK KO OM MK KO OT TP PA AS SS SI IA [1] => BA AR RD DH HA AK KO OM MK KO OT TP PA AS Nilai Kemiripan Antar Address: Area: B102 (Pekan Baru) [0] & [1] => 0,0 Area: B104 (Padang) [0] & [1] => 0,6 Nilai Kemiripan Antar Record: Area: B102 (Pekan Baru) [0] & [1] => 0,0 Area: B104 (Padang) [0] & [1] => 0.8 Clean Code: Area: B102 (Pekan Baru) [0] => B102-1 [1] => B102-2	

	Input	Expected Output	Actual Output	Status
		[0] => B104-1 [1] => B104-1	Area: B104 (Padang) [0] => B104-1 [1] => B104-1	
3	Name: Area: B102 (Pekan Baru) [0] => FARMIT [1] => EXPGIANAN Area: B104 (Padang) [0] => ESARAH [1] => ESARAH Address: Area: B102 (Pekan Baru) [0] => SUD [1] => 27NANTAMTUA Area: B104 (Padang) [0] => ABABARKOMKOT PASSIA [1] => BARDHAKOMKOT PAS	String Gram Name: Area: B102 (Pekan Baru) [0] => FAA ARM RMI MIT [1] => EXP XPG PGIGIA IAN ANA NAN Area: B104 (Padang) [0] => ESA SAR ARA RAH [1] => ESA SAR ARA RAH Nilai Kemiripan Antar Name: Area: B102 (Pekan Baru) [0] & [1] => 0,0 Area: B104 (Padang) [0] & [1] => 1,0 String Gram Address: Area: B102 (Pekan Baru) [0] => SUD [1] => 27N 7NA NAN ANT NTA TAM AMT MTU TUA Area: B104 (Padang) [0] => ABA BAB ABABAR ARK RKO KOMOMK MKO KOT OTP TPA PAS ASS SSI SIA [1] => BAR ARD RDHDHA HAK AKO KOMOMK MKO KOT OTP TPA PAS Nilai Kemiripan Antar	String Gram Name: Area: B102 (Pekan Baru) [0] => FAA ARM RMI MIT [1] => EXP XPG PGIGIA IAN ANA NAN Area: B104 (Padang) [0] => ESA SAR ARA RAH [1] => ESA SAR ARA RAH Nilai Kemiripan Antar Name: Area: B102 (Pekan Baru) [0] & [1] => 0,0 Area: B104 (Padang) [0] & [1] => 1,0 String Gram Address: Area: B102 (Pekan Baru) [0] => SUD [1] => 27N 7NA NAN ANT NTA TAM AMT MTU TUA Area: B104 (Padang) [0] => ABA BAB ABABAR ARK RKO KOMOMK MKO KOT OTP TPA PAS ASS SSI SIA [1] => BAR ARD RDHDHA HAK AKO KOMOMK MKO KOT OTP TPA PAS Nilai Kemiripan Antar	Berhasil

Input		Expected Output	Actual Output	Status
		Address: Area: B102 (Pekan Baru) [0] & [1] => 0,0 Area: B104 (Padang) [0] & [1] => 0,5 Nilai Kemiripan Antar <i>Record:</i> Area: B102 (Pekan Baru) [0] & [1] => 0,0 Area: B104 (Padang) [0] & [1] => 0,8 <i>Clean Code:</i> Area: B102 (Pekan Baru) [0] => B102-1 [1] => B102-2 Area: B104 (Padang) [0] => B104-1 [1] => B104-1	Address: Area: B102 (Pekan Baru) [0] & [1] => 0,0 Area: B104 (Padang) [0] & [1] => 0,5 Nilai Kemiripan Antar <i>Record:</i> Area: B102 (Pekan Baru) [0] & [1] => 0,0 Area: B104 (Padang) [0] & [1] => 0,8 <i>Clean Code:</i> Area: B102 (Pekan Baru) [0] => B102-1 [1] => B102-2 Area: B104 (Padang) [0] => B104-1 [1] => B104-1	
4	Name: Area: B102 (Pekan Baru) [0] => FARMIT [1] => EXPGIANAN Area: B104 (Padang) [0] => ESARAH [1] => ESARAH Address: Area: B102 (Pekan Baru) [0] => SUD [1] => 27NANTAMTUA Area: B104 (Padang) [0] => ABABARKOMKOT PASSIA	String Gram Name: Area: B102 (Pekan Baru) [0] => FAAA ARMI RMIT [1] => EXPG XPGI PGIA GIAN IANA ANAN Area: B104 (Padang) [0] => ESAR SARA ARAH [1] => ESAR SARA ARAH Nilai Kemiripan Antar Name: Area: B102 (Pekan Baru) [0] & [1] => 0,0 Area: B104 (Padang) [0] & [1] => 1,0	String Gram Name: Area: B102 (Pekan Baru) [0] => FAAA ARMI RMIT [1] => EXPG XPGI PGIA GIAN IANA ANAN Area: B104 (Padang) [0] => ESAR SARA ARAH [1] => ESAR SARA ARAH Nilai Kemiripan Antar Name: Area: B102 (Pekan Baru) [0] & [1] => 0,0 Area: B104 (Padang) [0] & [1] => 1,0	Berhasil

Input	Expected Output	Actual Output	Status
[1] => BARDHAKOMKOT PAS	String Gram Address: Area: B102 (Pekan Baru) [0] => SUD [1] => 27NA 7NAN NANT ANTA NTAM TAMT AMTU MTUA Area: B104 (Padang) [0] => ABAB BABA ABAR BARK ARKO RKOM KOMK OMKO MKOT KOTP OTPA TPAS PASS ASSI SSIA [1] => BARD ARDH RDHA DHAK HAKO AKOM KOMK OMKO MKOT KOTP OTPA TPAS Nilai Kemiripan Antar Address: Area: B102 (Pekan Baru) [0] & [1] => 0,0 Area: B104 (Padang) [0] & [1] => 0,4 Nilai Kemiripan Antar <i>Record:</i> Area: B102 (Pekan Baru) [0] & [1] => 0,0 Area: B104 (Padang) [0] & [1] => 0.7 <i>Clean Code:</i> Area: B102 (Pekan Baru) [0] => B102-1 [1] => B102-2 Area: B104 (Padang) [0] => B104-1 [1] => B104-1	String Gram Address: Area: B102 (Pekan Baru) [0] => SUD [1] => 27NA 7NAN NANT ANTA NTAM TAMT AMTU MTUA Area: B104 (Padang) [0] => ABAB BABA ABAR BARK ARKO RKOM KOMK OMKO MKOT KOTP OTPA TPAS PASS ASSI SSIA [1] => BARD ARDH RDHA DHAK HAKO AKOM KOMK OMKO MKOT KOTP OTPA TPAS Nilai Kemiripan Antar Address: Area: B102 (Pekan Baru) [0] & [1] => 0,0 Area: B104 (Padang) [0] & [1] => 0,4 Nilai Kemiripan Antar <i>Record:</i> Area: B102 (Pekan Baru) [0] & [1] => 0,0 Area: B104 (Padang) [0] & [1] => 0.7 <i>Clean Code:</i> Area: B102 (Pekan Baru) [0] => B102-1 [1] => B102-2 Area: B104 (Padang) [0] => B104-1 [1] => B104-1	

Berdasarkan pada seluruh tabel hasil pengujian di atas, terlihat bahwa hasil dari pengujian *whitebox* menghasilkan status ‘berhasil’ karena mengeluarkan hasil ‘*Actual Output*’ yang sesuai dengan hasil ‘*Expected Output*’. Dengan demikian, dapat disimpulkan bahwa algoritma SNM dan N-gram telah berhasil digunakan untuk menemukan duplikasi data pada master data konsumen Divisi *Consumer Care* PT XYZ.

4.3.2 Pengujian *Black Box*

Pengujian *black box* merupakan pengujian fungsional sistem yang berfokus pada *output* yang dihasilkan oleh sistem berdasarkan *input* dan kondisi tertentu. Pengujian *black box* tidak berhubungan dengan analisis kode dan struktur internal program seperti pada *white box*. Pengujian ini berfungsi untuk memastikan apakah seluruh sistem yang dibangun telah memenuhi kebutuhan *user* (Agarwal, dkk., 2010).

Pengujian *black box* yang akan dilakukan dalam penelitian ini akan menggunakan pengujian komprehensif berdasarkan pedoman pengujian aplikasi web yang ada pada (Vijay, 2015). Pengujian yang akan diuji adalah pengujian *functionality*, *usability*, *compatibility*, dan *performance* yang akan dibahas pada sub-poin berikut.

4.3.2.1 Pengujian *Functionality*

Pengujian ini dilakukan kepada *user* sistem *data cleaning*, yaitu *sales admin* Divisi *Consumer Care* PT XYZ. Kasus pengujian yang diajukan kepada *user* disesuaikan terhadap kebutuhan fungsional yang telah disepakati antara peneliti dan *user*. Hasil pengujian dapat dilihat pada Lampiran 5. Berikut ini adalah tabel ringkasan dari hasil pengujian *functionality* yang telah dilakukan kepada *user*.

Tabel 4.17 Tabel Hasil Pengujian *Functionality*

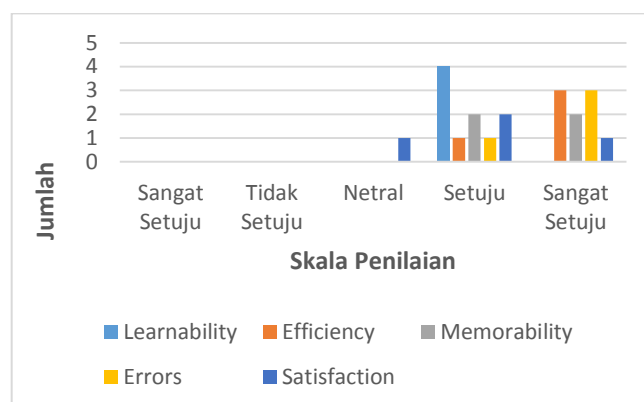
No	Skenario Pengujian	Test Case	Status
1.	Mengijinkan <i>user</i> dapat melakukan <i>login</i> ke dalam sistem.	<i>User input employee id dan password pada form login dan menekan tombol login.</i>	Berhasil
2.	Mengijinkan <i>user</i> dapat memasukan atau mengimpor data konsumen Divisi <i>Consumer Care</i> ke dalam <i>database</i> .	<i>User melampirkan file excel dan menekan tombol save file.</i>	Berhasil
		<i>User menekan tombol remove file.</i>	Berhasil
		<i>User memilih sheet yang ada di dalam file excel.</i>	Berhasil
		<i>User mengisi kolom “start column” dan “start row”.</i>	Berhasil
		<i>User menekan tombol import data.</i>	Berhasil
3.	Mengijinkan <i>user</i> untuk melakukan pendeteksian duplikasi data.	<i>User memasukkan teks atau karakter pada list removed text dan memilih nilai threshold, token length, serta gram value.</i>	Berhasil
		<i>User menekan tombol start cleaning.</i>	Berhasil
4.	Mengijinkan <i>user</i> untuk merapikan format penulisan <i>phone</i> dan <i>fax</i> .	<i>User menekan tombol start fixing.</i>	Berhasil
5.	Mengijinkan <i>user</i> untuk menampilkan hasil deteksi duplikasi data.	<i>User menekan tombol view clean data. Kemudian dapat otefmemfilter data dari kolom “view data” dan “area”.</i>	Berhasil
6.	Mengijinkan <i>User</i> untuk mengekspor atau mengunduh hasil pendeteksian duplikasi data ke dalam <i>file excel</i> .	Setelah <i>user</i> melakukan proses deteksi duplikasi data, <i>user</i> menekan tombol <i>Export Data</i> . Atau dengan menekan tombol <i>view clean data</i> lalu menekan tombol <i>Export Data</i> .	Berhasil
7.	Mengijinkan <i>User</i> untuk menyimpan hasil perubahan penulisan format <i>phone</i> dan <i>fax</i> ke dalam <i>database</i> .	Setelah <i>user</i> melakukan proses pemformatan nomor telepon dan <i>fax</i> , <i>user</i> menekan tombol <i>save</i> .	Berhasil

Berdasarkan tabel di atas, terlihat bahwa seluruh *test case* yang diujikan kepada *user* menghasilkan status ‘Berhasil’. Oleh karena itu, dapat disimpulkan bahwa semua fungsi dalam aplikasi telah berjalan sesuai dengan fungsionalitasnya.

4.3.2.2 Pengujian *Usability*

Pengujian *usability* dilakukan untuk mengidentifikasi kesalahan sistem dengan menilai *user* dalam menggunakan dan mengoperasikan sistem. Menurut Nielsen, dalam (Aydar, dkk., 2016), *usability* dapat dikelompokkan ke dalam 5 faktor, yaitu *efficiency*, *learnability*, *memorability*, *errors/safety*, dan *satisfaction*. Kelima faktor ini akan dijadikan acuan penulis untuk menilai *user* dalam menggunakan sistem *data cleaning* yang terdapat pada Lampiran 5.

Pada pengujian *usability*, respon *user* memiliki nilai angka yang akan digunakan sebagai pengukuran. Nilai 1 merepresentasikan Sangat Tidak Setuju (STS), nilai 2 merepresentasikan Tidak Setuju (TS), nilai 3 merepresentasikan Netral (N), nilai 4 merepresentasikan Setuju (S), dan nilai 5 merepresentasikan Sangat Setuju (SS). Berdasarkan pengujian atas 20 pertanyaan yang telah diajukan kepada *user* sistem *data cleaning* Divisi *Consumer Care* PT XYZ, telah diperoleh hasil penilaian seperti berikut.



Gambar 4.24 Gambar Grafik Hasil Pengujian *Usability* Pada *User Sistem Data Cleaning PT XYZ*

Tabel 4.18 Tabel Hasil Pengujian *Usability*

Aspek	Skala Penilaian					Poin Skala Penilaian					Total Nilai
	STS	TS	N	S	STS	1	2	3	4	5	
<i>Learnability</i>				4					16		16
<i>Efficiency</i>				1	3				4	15	19
<i>Memorability</i>				2	2				8	10	18
<i>Errors</i>				1	3				4	15	19
<i>Satisfaction</i>			1	2	1			3	8	5	16
Total			1	10	9			3	40	45	88

Berdasarkan tabel hasil pengujian *usability*, terlihat bahwa nilai *usability* yang diujikan kepada *user* diperoleh nilai 88 dimana nilai maksimal yang dapat diperoleh adalah $20 \text{ pertanyaan} \times 5 \text{ poin} = 100$. Sehingga, nilai *usability* yang diperoleh dari pengujian terhadap *user* dapat dikatakan mendapatkan nilai baik atau mendapat kecenderungan respon yang positif. Dengan demikian, dapat disimpulkan bahwa sistem telah memenuhi syarat *usability*.

4.3.2.3 Pengujian *Compatibility*

Pengujian *compatibility* dilakukan untuk menguji kompatibilitas sistem apakah masih berjalan dengan baik jika digunakan pada *browser*, *hardware*, dan sistem operasi yang berbeda. Berikut ini adalah hasil dari pengujian *compatibility* sistem *data cleaning* pada PT XYZ.

Tabel 4.19 Tabel Hasil Pengujian *Compatibility*

<i>Conduct in Different</i>	Spesifikasi	Hasil
<i>Browser</i>	Internet Explorer 11 Mozilla Firefox 44.0	<i>Compatible</i>
<i>Hardware</i>	Laptop Lenovo ThinkPad T400 PC Lenovo ThinkCentre M 73	<i>Compatible</i>

<i>Operation System</i>	Windows 7	<i>Compatible</i>
	Windows 8.1	

Berdasarkan tabel di atas, terlihat bahwa hasil pengujian terhadap *browser hardware*, dan *operation system* yang berbeda memperoleh hasil ‘*compatible*’. Dengan demikian, dapat disimpulkan bahwa sistem *data cleaning* yang dibangun telah memenuhi syarat *compatibility*.

4.3.2.4 Pengujian *Performance*

Pengujian *performance* merupakan istilah yang biasa digunakan untuk mengetes sistem agar sistem telah memenuhi persyaratan kinerjanya (Mansharamani, 2011). Pengujian ini dilakukan untuk menentukan performa sistem dalam mengukur kualitas sistem seperti responsivitas, kecepatan, skalabilitas, stabilitas dan sebagainya. Sistem dites dengan serangkaian *workload* tertentu dan diuji waktu yang dibutuhkan pada saat bekerja pada *workload* tersebut (Software Testing Class, 2013).

Pada penelitian ini, tambahan *workload* dengan menambahkan seperti prosesor, memori, kapasitas penyimpanan, lalu lintas jaringan dan beberapa kondisi lainnya tidak masuk dalam ruang lingkup penelitian sehingga performa hanya diukur berdasarkan *response time* karena hal ini erat hubungannya dengan kepuasan *user*. Pengujian dilakukan dengan menggunakan perangkat keras Notebook Dell Inspiron N4050 dengan prosesor Intel Core i5, RAM 4 GB, Tipe Memori DDR3 dan perangkat lunak Windows 7 32-bit serta Mozilla Firefox sebagai *browser*-nya. Berikut ini adalah tabel hasil pengujian performa menggunakan *test case* dengan *workload* tertentu.

Tabel 4.20 Tabel Hasil Pengujian *Performance*

No	<i>Test Case</i>	Skenario	<i>Response time</i>
1	Mengimpor data sebanyak 27.695 baris data.	Menyimpan 27.695 data ke dalam sistem dengan menekan tombol “Save”.	0,03 detik

		Mengimpor 27.695 data ke <i>database</i> dengan menekan tombol “Import Data”.	36,78 detik
2	Deteksi duplikasi data sebanyak 27.695 baris data.	Menjalankan deteksi duplikasi sebanyak 27.695 baris data dengan menekan tombol “Start cleaning”	1 menit 2 detik
3	Format penulisan <i>phone</i> dan <i>fax</i> .	Format penulisan <i>phone</i> dan <i>fax</i> dari 27.695 baris data dengan menekan tombol “Start fixing”	10,69 detik
4	Menampilkan data yang telah dilakukan proses deteksi duplikasi data.	Menampilkan seluruh data (duplikat data dan non-duplikat data)	9,96 detik
		Menampilkan data yang duplikat	4,86 detik

Berdasarkan tabel hasil pengujian performa, dapat diperoleh nilai *response time* ketika sistem diuji coba dengan jumlah baris data yang relatif besar. Jumlah data yang akan digunakan oleh *user* untuk mendeteksi duplikasi data adalah \pm 25.000 baris data. Sedangkan, dalam pengujian ini telah dilakukan 27.695 baris data. Dari hasil yang diperoleh nilai *response time* dapat dikatakan relatif cepat jika dibandingkan dengan jumlah baris data yang relatif besar. Terlebih, jika dibandingkan dengan proses pendeteksian duplikasi data dan pemformatan penulisan telepon dan fax secara manual. Dengan demikian, dapat disimpulkan bahwa sistem *data cleaning* telah memenuhi aspek *performance*.

4.3.3 Evaluasi Data

Tahap ini dilakukan untuk mengevaluasi metode yang telah diterapkan pada sistem *data cleaning* yang telah dibangun. Berdasarkan Weis, dkk. (2008), untuk mengevaluasi metode dapat dilakukan dengan dua sampel data berikut.

1. D_{small} yang terdiri atas sejumlah *sample* dari seluruh data yang dipilih secara acak, dalam penelitian ini digunakan 10% dari seluruh total data, yaitu 2.500 data.
2. D_{large} yang terdiri atas seluruh total data yang ada, yaitu 25.000 data.

Sebelum melakukan evaluasi, jumlah data yang duplikat harus sudah diketahui secara manual untuk menghitung nilai efektivitas pada sampel yang telah dipilih. Untuk menghitung nilai efektivitas diperoleh dengan menghitung nilai *recall*, *precision* dan *f-measure* berikut (Bilenko, 2002).

$$precision = \frac{\text{Total duplikasi data yang benar yang telah ditemukan}}{\text{Total data yang telah ditemukan}}$$

$$recall = \frac{\text{Total duplikasi data yang benar yang telah ditemukan}}{\text{Total data duplikat yang sebenarnya}}$$

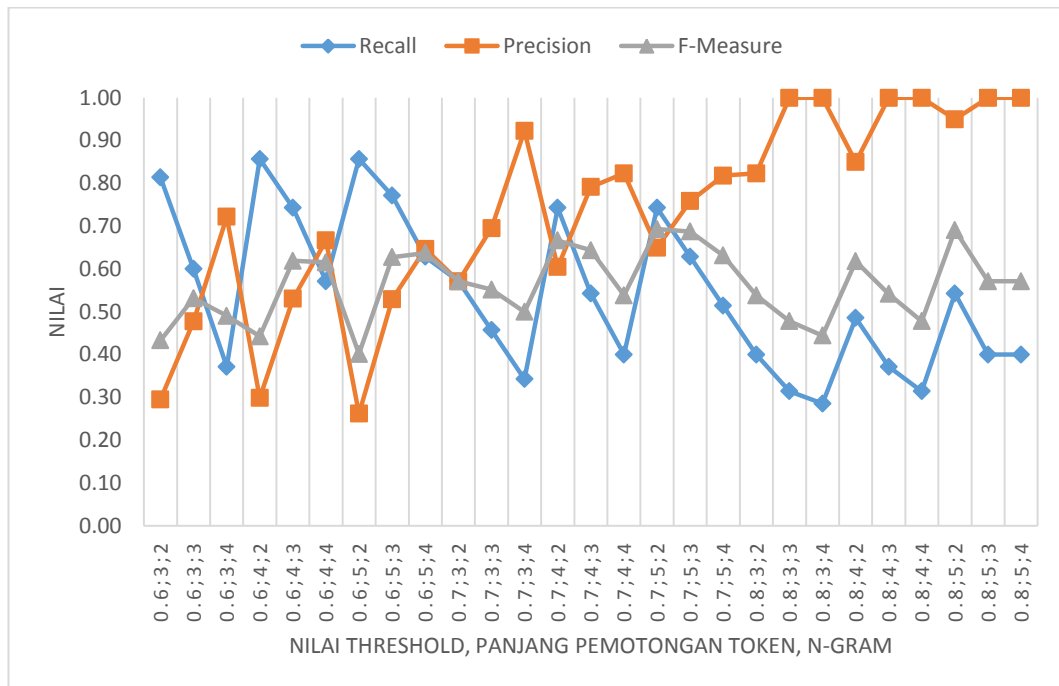
$$f - measure = \frac{2 \times precision \times recall}{precision + recall}$$

Rumus 4.1 Rumus *precision*, *recall*, dan *f-measure*

Percobaan pertama adalah dengan menggunakan D_{small} data, yaitu 2.500 data. Di mana di dalam D_{small} data yang diuji cobakan terdapat 70 baris data yang duplikat. Oleh karena pada saat proses deteksi duplikasi data, *user* harus memilih nilai *threshold*, panjang pemotongan token, dan *N-Gram*, maka percobaan D_{small} data diuji coba dengan menggunakan perpaduan-perpaduan antara 3 nilai yang disediakan. Hal ini berarti dapat membuktikan berapakah nilai perpaduan yang tepat untuk bisa mendapatkan hasil yang optimal atau yang memiliki nilai *f-measure* (nilai perpaduan antara *recall* dan *precision*) yang paling tinggi. Berikut ini adalah hasil pengujian dengan D_{small} data.

Tabel 4.21 Tabel Hasil Percobaan $D_{small} = 2500$ Data Dengan Menguji Menggunakan Nilai *Threshold*, Panjang Pemotongan Token, dan N-Gram

<i>Threshold</i>	Panjang Pemotongan Token	<i>Gram</i>	Total Duplikat Yang Ditemukan	Total Duplikasi Yang Benar	<i>Recall</i>	<i>Precision</i>	<i>F-Measure</i>
0.6	3	2	193	57	0.81	0.30	0.43
0.6	3	3	88	42	0.60	0.48	0.53
0.6	3	4	36	26	0.37	0.72	0.49
0.6	4	2	201	60	0.86	0.30	0.44
0.6	4	3	98	52	0.74	0.53	0.62
0.6	4	4	60	40	0.57	0.67	0.62
0.6	5	2	229	60	0.86	0.26	0.40
0.6	5	3	102	54	0.77	0.53	0.63
0.6	5	4	68	44	0.63	0.65	0.64
0.7	3	2	70	40	0.57	0.57	0.57
0.7	3	3	46	32	0.46	0.70	0.55
0.7	3	4	26	24	0.34	0.92	0.50
0.7	4	2	86	52	0.74	0.60	0.67
0.7	4	3	48	38	0.54	0.79	0.64
0.7	4	4	34	28	0.40	0.82	0.54
0.7	5	2	80	52	0.74	0.65	0.69
0.7	5	3	58	44	0.63	0.76	0.69
0.7	5	4	44	36	0.51	0.82	0.63
0.8	3	2	34	28	0.40	0.82	0.54
0.8	3	3	22	22	0.31	1.00	0.48
0.8	3	4	20	20	0.29	1.00	0.44
0.8	4	2	40	34	0.49	0.85	0.62
0.8	4	3	26	26	0.37	1.00	0.54
0.8	4	4	22	22	0.31	1.00	0.48
0.8	5	2	40	38	0.54	0.95	0.69
0.8	5	3	28	28	0.40	1.00	0.57
0.8	5	4	28	28	0.40	1.00	0.57



Gambar 4.25 Gambar Grafik Hasil Pengujian $D_{small} = 2500$ Data

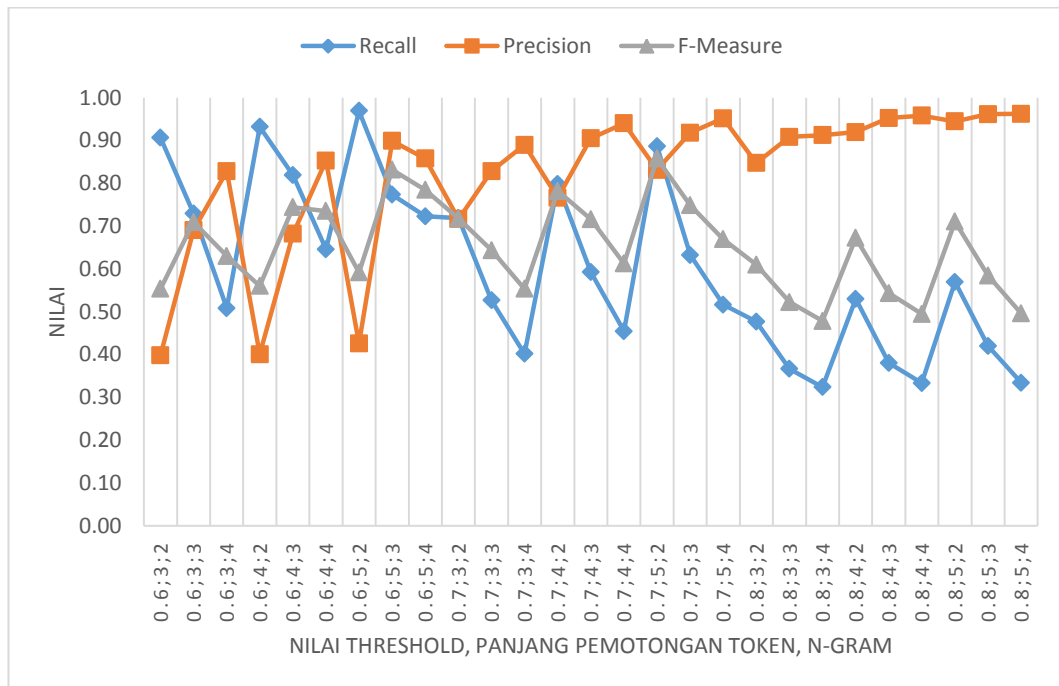
Berdasarkan hasil grafik di atas, terlihat bahwa nilai *f-measure* yang paling tinggi terdapat pada nilai perpaduan *threshold*, panjang pemotongan token, dan nilai *N-Gram* sebesar 0,7; 5; dan 2. Di mana nilai *f-measure* adalah sebesar 0,69 dengan nilai *recall* sebesar 0,74 dan *precision*-nya sebesar 0,65.

Selanjutnya adalah uji coba data dengan menggunakan D_{large} data, yaitu sebanyak 25.000 data. Dari 25.000 data yang akan diuji coba terdapat data duplikasi sebanyak 1.388 data. Berikut ini adalah hasil pengujian dengan menggunakan D_{large} data.

Tabel 4.22 Tabel Hasil Pengujian $D_{large} = 25.000$ Data Dengan Menguji Menggunakan Nilai *Threshold*, Panjang Pemotongan Token, dan N-Gram

<i>Threshold</i>	Panjang Pemotongan Token	<i>Gram</i>	Total Duplikat Yang Ditemukan	Total Duplikasi Yang Benar	<i>Recall</i>	<i>Precision</i>	<i>F-Measure</i>
0.6	3	2	3160	1259	0.91	0.40	0.55
0.6	3	3	1467	1013	0.73	0.69	0.71
0.6	3	4	852	706	0.51	0.83	0.63

<i>Threshold</i>	Panjang Pemotongan Token	<i>Gram</i>	Total Duplikat Yang Ditemukan	Total Duplikasi Yang Benar	<i>Recall</i>	<i>Precision</i>	<i>F-Measure</i>
0.6	4	2	3229	1294	0.93	0.40	0.56
0.6	4	3	1666	1137	0.82	0.68	0.74
0.6	4	4	1051	897	0.65	0.85	0.74
0.6	5	2	3160	1346	0.97	0.43	0.59
0.6	5	3	1194	1074	0.77	0.90	0.83
0.6	5	4	1170	1004	0.72	0.86	0.78
0.7	3	2	1392	998	0.72	0.72	0.72
0.7	3	3	882	731	0.53	0.83	0.64
0.7	3	4	627	558	0.40	0.89	0.55
0.7	4	2	1445	1108	0.80	0.77	0.78
0.7	4	3	909	823	0.59	0.91	0.72
0.7	4	4	671	631	0.45	0.94	0.61
0.7	5	2	1480	1231	0.89	0.83	0.86
0.7	5	3	956	878	0.63	0.92	0.75
0.7	5	4	753	717	0.52	0.95	0.67
0.8	3	2	781	662	0.48	0.85	0.61
0.8	3	3	560	509	0.37	0.91	0.52
0.8	3	4	493	450	0.32	0.91	0.48
0.8	4	2	800	736	0.53	0.92	0.67
0.8	4	3	554	528	0.38	0.95	0.54
0.8	4	4	483	463	0.33	0.96	0.49
0.8	5	2	837	791	0.57	0.95	0.71
0.8	5	3	606	583	0.42	0.96	0.58
0.8	5	4	482	464	0.33	0.96	0.50



Gambar 4.26 Gambar Grafik Hasil Pengujian $D_{large} = 25.000$ Data

Berdasarkan hasil grafik di atas, terlihat bahwa nilai *f-measure* yang paling tinggi terdapat pada nilai perpaduan *threshold*, panjang pemotongan token, dan nilai *N-Gram* sebesar 0,7; 5; dan 2. Di mana nilai *f-measure* adalah sebesar 0,86 dengan nilai *recall* sebesar 0,89 dan *precision*-nya sebesar 0,83.

Hasil yang diperoleh dari percobaan menggunakan D_{small} dan D_{large} data, telah didapatkan nilai *f-measure* yang lebih besar. Selain itu, dari kedua percobaan tersebut telah didapatkan nilai perpaduan *threshold*, panjang pemotongan token, dan nilai *N-Gram* sebesar 0,7; 5; dan 2 untuk mendapatkan hasil *f-measure* yang paling baik.

Bab V

Simpulan dan Saran

5.1 Simpulan

Berdasarkan penelitian yang telah dilakukan, penulis mengambil beberapa simpulan sebagai berikut:

1. Perancangan dan pembangunan sistem *data cleaning* untuk Divisi *Consumer Care* PT XYZ dengan menerapkan metode SNM dan N-Gram telah berhasil dibangun. Pada tahap perancangan, pertama kali dilakukan wawancara terhadap *user* sistem untuk mendapatkan data analisa kebutuhan sistem secara fungsional. Dari data analisa tersebut menghasilkan sebuah rancangan yang direpresentasikan dalam bentuk UML Diagram yang terdiri dari *use case diagram*, *class diagram*, dan *activity diagram*. Selain itu, juga menghasilkan rancangan *database* sebagai data model dalam pengembangan sistem *data cleaning* ini. Kemudian, dirancang pula alur algoritma SNM dan N-Gram untuk dapat diterapkan pada sistem serta prosedur untuk merapikan format penulisan telepon dan fax. Perancangan ini yang digunakan sebagai acuan untuk mengembangkan sistem *data cleaning* di PT XYZ. Pada tahap pengembangan aplikasi, implementasi sistem dari hasil rancangan aplikasi diterapkan menggunakan bahasa pemrograman C# dan visual studio sebagai IDE pengembang aplikasinya. Sedangkan, untuk implementasi *database*-nya digunakan SQL Server dari Microsoft. Sebagai *web server* aplikasi yang digunakan pada pengembangan menggunakan IIS 8.0. Kemudian, pada implementasi rancangan *user interface* disesuaikan dengan fungsi utama aplikasi yaitu deteksi duplikasi data dan merapikan format penulisan telepon dan fax. Selain itu, tampilan yang diimplementasi disesuaikan dengan *template* tamplan aplikasi web yang ada pada PT XYZ.
2. Implementasi algoritma SNM dan N-Gram dalam sistem *data cleaning* terdapat pada kelas *DuplicateDetection*. Algoritma SNM digunakan

sebagai algoritma awal untuk melakukan tahap *pre-cleaning* atau tahap pembersihan data dari karakter atau titel tertentu dan memproses pembentukan token serta membandingkan seluruh data. Proses perbandingan data dibagi berdasarkan area konsumen Divisi *Consumer Care* PT XYZ. Pada saat pembentukan token, *user* dapat menginput berapa nilai pemotongan token yang akan dijadikan parameter dalam proses pembentukan token yang ada pada fungsi *chopTheWords*. Untuk membandingkan nilai kesamaan antar *record*, diterapkan algoritma N-Gram yang ada pada fungsi *gramming*. Di mana *user* dapat memasukkan nilai *N-Gram* dan *threshold* atau ambang batas yang akan dijadikan parameter untuk menentukan nilai kesamaan antar *record*.

3. Untuk mengetahui apakah aplikasi berjalan sesuai dengan kebutuhan dilakukan dua pengujian, yaitu pengujian *white box* yang dilakukan oleh penulis sendiri dan pengujian *black box* yang dilakukan kepada *user*, yaitu *Sales Admin* Divisi *Consumer Care* PT XYZ. Dari hasil pengujian menunjukkan bahwa seluruh *codes* dan kebutuhan fungsional sistem berjalan dengan baik. Selain itu, dilakukan pula evaluasi data dari deteksi duplikasi data untuk mengetahui nilai efektivitas dari sistem yang telah dibangun dengan menghitung nilai *recall*, *precision*, dan *f-measure*. Kemudian, telah diperoleh nilai perpaduan *threshold*, panjang pemotongan token, dan nilai *N-Gram* sebesar 0,7, 5, dan 2 untuk mendapatkan hasil *f-measure* atau nilai efektivitas yang paling baik.

5.2 Saran

Untuk pengembangan penelitian lebih lanjut, penulis memberikan beberapa saran untuk ke depannya, sebagai berikut:

1. Sistem *data cleaning* yang dibangun masih menggunakan Microsoft Excel untuk memasukkan data sumber. Hal ini karena kebutuhan *user* dan penyimpanan data konsumen Divisi *Consumer Care* masih menggunakan

Microsoft Excel. Diharapkan setelah *database* konsumen telah selesai dibangun oleh pihak PT XYZ, algoritma yang ada di dalam sistem *data cleaning* dalam penelitian ini dapat diterapkan pada *database* konsumen Divisi *Consumer Care* PT XYZ sehingga *user* tidak perlu melakukan *import* data atau *export* data.

2. Sistem *data cleaning* yang dibangun masih bersifat statis terhadap data yang masuk ke dalam sistem. Hanya input dan output saja. Oleh karena itu, diharapkan sistem ke depannya dapat berjalan secara dinamis terhadap data yang masuk ke dalam sistem. Maksudnya adalah ketika terdapat data baru masuk ke dalam sistem, maka data baru tersebut hanya memperbarui data yang telah ada. Artinya, data lama tidak dicek kembali dan data baru secara otomatis mengecek dengan sesama data baru dan dicek pula ke data lama yang telah ada. Dengan demikian, nilai *clean code* data lama tidak berubah. Kemudian, jika terdapat data baru yang berduplikasi dengan data lama, maka *clean code* data baru akan sama dengan *clean code* data lama. Sebaliknya, data baru yang tidak memiliki duplikasi data dengan data lama, maka akan dibentuk *clean code* yang baru pula.
3. Sistem *data cleaning* yang dibangun pada saat ini masih menggunakan *localhost* dan diharapkan dapat diimplementasi pada *server* PT XYZ.

Daftar Pustaka

- Agarwal, B. B., Tayal, S. P., & M., G. (2010). *Software Engineering and Testing*. Sudbury: Jones and Bartlett.
- Al Fatta, H. (2007). *Analisis dan Perancangan Sistem Informasi Untuk Keunggulan Bersaing Perusahaan dan Organisasi Modern*. Yogyakarta: Amikom.
- Aydar, d. (2016). *Software Testing Tutorial*. [www.tutorialspoint.com: http://www.tutorialspoint.com/software_testing/index.htm](http://www.tutorialspoint.com/software_testing/index.htm)
- Azma, S. (2006). Pembuatan Alat Bantu Dalam Proses Data Cleaning Pada Intra-Governmental Access to Shared Information System (IGASIS) . Bandung, Jawa Barat, Indonesia.
- Bilenko, M. d. (2002). *Learning to Combine Trained Distance Metrics for Duplicate Detection in Databases*. Austin: Department of Computer Science University of Texas .
- Binanto, I. (2014). Analisa Metode Classic Life Cycle (Waterfall) Untuk Pengembangan Perangkat Lunak Multimedia. Yogyakarta: <http://www.researchgate.net/publication/264497046>.
- Chapman, A. D. (2005). Principles and Methods of Data Cleaning - Primary Species and Species-Occurrence Data, version 1.0 . (p. 1). Queensland, Australia: Global Biodiversity Information Facility.
- Couto, P. D. (2012, October). Support for User Interaction in a Data Cleaning Process (Dissertation). Germany.
- Dan, C. (2011). *Beginning C# Object-Oriented*. New York: Apress.
- Friedman, C., & Sideli, R. (1992). Tolerating Spelling Errors During Patient Validation. *Computers and Biomedical Research*, (pp. 486-509). New York.

- Guo, L., Wang, W., Chen, F., Tang, X., & Wang, W. (2012). A Similar Duplicate Data Detection Method Based on Fuzzy Clustering for Topology Information. In *PRZEGLĄD ELEKTROTECHNICZNY (Electrical Review)*, 01b (pp. 26-31).
- Hernandez, M. A. (1995). A Generalization of Band Joins and The Merge/Purge Problem. *Thesis Proposal, Department of Computer Science*, 16-17.
- Hernandez, M. A., & Stolfo, S. J. (1995). The Merge/Purge Problem for Large Database. (pp. 128-129). New York: NYS Science and Technology Foundation.
- Huda, N. M. (2010). Aplikasi Data Mining Untuk Menampilkan Informasi Tingkat Kelulusan Mahasiswa (Skripsi). Semarang, Indonesia.
- Kroenke, D. M., Saat, S., & Nugraha, D. (2003). *Database Processing Jilid 1 Edisi 9*. Jakarta: Erlangga.
- Lee, M. L., Lu, H., Ling, T. W., & Ko, Y. T. (1999). Cleansing Data for Mining and Warehousing. *10th International Conference on Database and Expert Systems Applications*. Italy: Cleansing Data for Mining and Warehousing.
- Lemos, O. A., Franchin, I. G., & Masiero, P. C. (2009). Integration testing of Object-Oriented and Aspect-Oriented programs. *Science of Computer Programming*, 74, 861-878.
- Lemos, O. A., Franchin, I. G., & Masiero, P. C. (2009). Integration Testing of Object-Oriented and Aspect-Oriented Programs. *Science of Computer Programming*.
- Liu, B. (2011). Web Data Mining: Exploring Hyperlinks, Contents, and Usage Data. *Data-Centric Systems and Applications* (p. 63). Berlin: Springer.
- Low, W. L., Lee, M. L., & Ling, T. W. (2001, May 20). A Knowledge-Based Approach for Duplicate Elimination in Data Cleaning. Singapore.
- Maimon, O., & Rokach, L. (2005). *The Data Mining and Knowledge Discovery Handbook*. Tel Aviv, Israel: Springer.
- Maletic, J. I., & Marcus, A. (2000). Data Cleansing: Beyond Integrity Analysis. *Conference on Information Quality*. Memphis: The Office of Naval Research.

- Mansharamani, R. (2011). *Performance Testing*.
<http://www.softwareperformanceengineering.com/uploads/1/2/6/6/12667295/performancetesting.pdf>
- MathWorks. (2015). *Machine-Learning*. mathworks.com:
<http://www.mathworks.com/solutions/machine-learning/index.html>
- Navarro, G. (2001). A Guided Tour to Approximate String. *ACM Computing Surveys*. Santiago: Dept. of Computer Science, University of Chile.
- Ningsih, V. M. (2009). *OOP vs Prosedural*. Telemetri:
<http://blog.neotelemetri.com/index.php/pemrograman/8-oop-vs-prosedural>
- Pressman, R. S. (2010). *Software Engineering*. New York: McGraw-Hill.
- Rahaman, G. M., Rahman, A., & Ripon, K. S. (2010, December 12). A Domain-Independent Data Cleaning Algorithm for Detecting Similar-Duplicates. *Journal of Computers, Vol. 5, No. 12*. Bangladesh: Academy Publisher.
- Recchia, G., & Max, L. (2013). A Comparison of String Similarity Measures for Toponym Matching. *ACM SIGSPATIAL COMP'13*. New York.
- Schacherer, C. W. (2012). SAS® Data Management Technique: Cleaning and Transforming Data for Delivery of Analytic Datasets.
- Software Testing Class*. (2013, October).
<http://www.softwaretestingclass.com/what-is-performance-testing/>
- Software Testing Help*. (n.d.). December 25, 2015,
<http://www.softwaretestinghelp.com/software-compatibility-testing/>
- Sommerville, I. (2011). *Software Engineering 9th Edition*. San Fransisco: Addison-Wesley.
- Sulardi. (2002). Pengujian Perangkat Lunak Dengan Teknik Pengujian Basis Patah. Undergraduate Thesis, FMIPA UNDIP. 10.
- Sulistyorini, P. (2009). Pemodelan Visual dengan Menggunakan. *Jurnal Teknologi Informasi DINAMIK*, 23-29.
- T. Sembok, T. M., & Abu Bakar, Z. (2011). Effectiveness of Stemming and N-grams String Similarity Matching on Malay Documents. *International Journal of Applied Mathematics and Informatics*, (pp. 208-215). Bangi, Malaysia.

- Tian, Z., Lu, H., Ji, W., Zhou, A., & Tian, Z. (2001). An n-gram-based Approach for Detecting Approximately Duplicate Database Records. *Springer Verlag*.
- Vijay. (2015, November 3). *Software Testing Help*.
<http://www.softwaretestinghelp.com/web-application-testing/>
- Weis, M., Naumann, F., Ulrich, J., Luftner, J., & Schuster, H. (2008). Industry-Scale Duplicate Detection. (pp. 1260-1263). New Zealand: Dept., ACM, Inc.
- Whitten, J. L., & Bentley, L. D. (2007). *System Analysis and Design Methods*. New York: McGraw-Hill Irwin.
- Yannakoudakis, E. J., & Angelidakis, G. (1988). An Insight into The Entropy and Redundancy of The English Dictionary. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (pp. 960-970).
<http://doi.ieeecomputersociety.org/10.1109/34.9119>.

Lampiran 1 – Wawancara

Berikut ini merupakan rangkuman beberapa wawancara yang penulis (selanjutnya ditulis dengan P) lakukan dengan calon *user*, yaitu Widi Rizky Ayudya (selanjutnya ditulis dengan WRA) selaku Staf Divisi *Consumer Care* di PT XYZ:

P : Apakah benar selama ini Mba Widi merasa sangat kewalahan dalam memvalidasi data Consumer Care?

WRA : Betul sekali. Karena data yang Saya *handle* sangat banyak dan terus bertambah setiap bulan.

P : Kalau boleh tahu berapa banyak data yang sekarang Mba Widi *handle*?

WRA : Sekitar 25.000 *row* data dan data akan terus bertambah tiap bulannya sekitar 1.000 data.

P : Dengan menggunakan *tools* atau *software* apa biasanya Mba Widi melakukan proses validasi data Consumer Care?

WRA : Proses validasi yang Saya lakukan menggunakan cara manual dan menggunakan *Ms. Excel* 2010 untuk melakukan validasi data.

P : Kalau boleh tahu, untuk data konsumen Consumer Care-nya sendiri itu disimpan di dalam sistem *database* atau disimpan dengan menggunakan *excel*?

WRA : Masih manual menggunakan *excel*.

P : Untuk sumber datanya sendiri didapat darimana ya, Mba?

WRA : Jadi, sumber data konsumen yang PT XYZ punya ini berasal dari distributor PT XYZ. Namun, jika ada data yang menurut Saya tidak

valid atau janggal maka Saya akan bertanya ke pihak ABM (*Area Business Manager*) PT XYZ untuk menanyakan kevalid-an data tersebut.

P : Pihak distributor mengirimkan data konsumen biasanya menggunakan *software* atau *tools* seperti apa dan *file* datanya itu berbentuk *excel* juga kah?

WRA : Biasanya dikirim melalui *email* dan benar sekali dikirimnya berupa *file .xls*.

P : Apa saja sih *problem* dari proses validasi data yang Mba Widi lakukan?

WRA : Problemnya itu ada data yang kembar. Seperti ini.. (memberikan contoh datanya). Kemudian, terkadang kolom *outlet type / category* tidak sesuai dengan deskripsi namanya. (memberikan contoh datanya) Nah, kalau kasus seperti itu, penyelesaiannya adalah ditanyakan ke pihak ABM. Selain itu, seperti kolom-kolom yang kosong. Kalau ada kolom yang wajib diisi kemudian kosong, maka Saya akan tanyakan lagi ke pihak ABM. Setelah itu, seperti penulisan-penulisan yang belum rapi. Contohnya, kolom *phone* dan *fax* ini. Ini Saya rapikan penulisannya. Karena *urgency*-nya tidak sepenting yang lain, jadi untuk kasus ini tidak Saya dahulukan dan memang sekali dua kali saja Saya kerjakan, jika sempat.

P : Lalu, bagaimana cara Mba Widi menemukan data kembar dari jumlah data yang sangat besar ini?

WRA : Caranya, pertama data ini Saya urutkan terlebih dahulu. Jadi, Saya menggunakan fitur *pivot* untuk mengurutkan datanya. Begini.. (memeragakan caranya). Pertama, Saya blok terlebih dahulu. Kemudian saya *insert – Pivot*. Setelah itu, Saya *drag Area* ke bagian *Row Labels*. Nah, ini data tiap Area akan secara otomatis berkelompok berdasarkan area masing-masing. Kemudian, Saya *drag Outlet Type* ke *Row Labels*. Kemudian, data yang sudah terbagi ke tiap area akan terbagi lagi berdasarkan tipe *outlet* masing-masing.

Setelah itu, Saya *drag Name* dan *Address* ke *Row Labels*. Lalu, *drag Name* ke *Values*. Saya *drag* ke *Values* agar Saya bisa tahu jumlah dari tiap *Row* yang sudah diurutkan. Karena kalau jumlahnya lebih dari satu, bisa jadi data itu adalah data kembar. Nah, Setelah semua terurut, Saya baca satu-satu. Saya *scanning*. Seperti ini... Nanti kalau ada yang datanya terlihat agak mirip. Maka, akan saya cek keseluruhan datanya apakah benar-benar mirip atau tidak. Karena bisa jadi, seperti toko-toko seperti Kimia Farma, *Carrefour*, pokoknya toko-toko yang tersebar dimana-mana. Itu biasanya, alamatnya sama karena mereka melakukan pembelanjaannya secara terpusat. Namun, sebenarnya *outlet* tersebut berbeda letaknya. Kalau memang menemukan kasus yang seperti itu, artinya data tersebut tidak kembar.

P : Bukankah untuk kolom *Outlet Type* datanya terkadang tidak benar seperti yang tadi Mba Widi katakan?

WRA : Nah, itu dia masalahnya. Coba bayangkan kalau misalnya Saya tidak membagi datanya berdasarkan *Outlet Type*-nya. Dalam satu area Saya harus mengecek sekitar 1.000 data secara bersamaan. Tidak mungkin bukan? Makanya, Saya bagi saja menjadi *Outlet Type* yang berbeda. Ya, walaupun tidak semua akurat. Tapi, tidak ada pilihan lain. Jadi, setidaknya kalau dengan membagi datanya ke tipe *outlet*-nya masing-masing akan sangat membantu Saya untuk menemukan data kembar.

P : Mengapa tidak menggunakan kolom *City* atau *Subcity* untuk memisahkan datanya Mba Widi?

WRA : Karena data yang ada di kolom *City* dan *Subcity* tidak sepenuhnya benar. Sedangkan, kalau kolom *Area* sudah pasti benar. Saya juga memisahkannya dengan *Area* kan tadi. Karena memang *area* itu merupakan pembagian *area marketing* dari tiap ABM PT XYZ, jadi, isinya pasti benar dan lengkap.

P : Bukankah bisa dikatakan apa yang Mba Widi lakukan ini tidak akan ada habisnya karena data akan terus bertambah bukan, Mba?

- WRA : Bisa dikatakan begitu. Tapi, setidaknya, Saya disini yang akan *maintain* data. Coba bayangkan kalau tidak ada yang *maintain* datanya.
- P : Kalau misalnya ada sistem yang melakukan apa yang Mba Widi kerjakan secara otomatis. Apakah Mba Widi setuju?**
- WRA : Wah.. setuju sekali. Karena jujur itu memudahkan pekerjaan Saya sekali. Kadang, untuk *maintain* data ini suka tidak bisa semua saya *handle*. Jadi, kalau misalkan ada sistem yang bisa secara otomatis membaca data kembar saja itu sudah memudahkan pekerjaan Saya.

Narasumber,



Widi Rizky Ayudya

Sales Admin Divisi Consumer Care

Lampiran 2 – Requirement Elicitation

Requirement Elicitation

Requirement Elicitation Sistem Data Cleaning Divisi Consumer Care PT XYZ

Requirement Elicitation Tahap 1

Fungsional	
No.	Analisa Kebutuhan
Saya ingin sistem dapat	
1.	Mengijinkan <i>user</i> dapat melakukan <i>login</i> ke dalam sistem.
2.	Mengijinkan <i>user</i> dapat memasukan atau mengimpor data konsumen Divisi <i>Consumer Care</i> ke dalam <i>database</i> .
3.	Mengijinkan <i>user</i> untuk melakukan pendeteksian duplikasi data.
4.	Mengijinkan <i>user</i> untuk merapikan format penulisan <i>phone</i> dan <i>fax</i> .
5.	Mengijinkan <i>user</i> untuk menampilkan hasil deteksi duplikasi data.
6.	Mengijinkan <i>user</i> untuk mengekspor atau mengunduh hasil pendeteksian duplikasi data ke dalam <i>file excel</i> .
7.	Mengijinkan <i>user</i> untuk menyimpan hasil perubahan penulisan format <i>phone</i> dan <i>fax</i> ke dalam <i>database</i> .

Non-fungsional	
No.	Analisa Kebutuhan
Saya ingin sistem dapat	
1.	Memiliki hasil pendeteksian duplikasi data yang cukup akurat.
2.	Mampu berjalan dengan berbasis web.
3.	Menampilkan tampilan <i>web</i> yang sesuai dengan <i>template</i> sistem yang ada pada PT XYZ.

Requirement Elicitation Tahap 2

Elisitasi Tahap II dibentuk berdasarkan Elisitasi Tahap I yang diklasifikasikan melalui metode MDI (*Mandatory, Desirable, Inessential*). Berikut penjelasan dari beberapa *requirement* yang mendapatkan opsi M, D, atau I.

Fungsional				
No.	Analisis Kebutuhan	M	D	I
Saya ingin sistem dapat				
1.	Mengijinkan <i>user</i> dapat melakukan <i>login</i> ke dalam sistem.	√		
2.	Mengijinkan <i>user</i> dapat memasukan atau mengimpor data konsumen Divisi <i>Consumer Care</i> ke dalam <i>database</i> .	√		
3.	Mengijinkan <i>user</i> untuk mereset atau menghapus data konsumen yang ada di dalam <i>database</i> .	√		
4.	Mengijinkan <i>user</i> untuk melakukan pendeteksian duplikasi data.	√		
5.	Mengijinkan <i>user</i> untuk merapikan format penulisan <i>phone</i> dan <i>fax</i> .		√	
6.	Mengijinkan <i>user</i> untuk menampilkan hasil deteksi duplikasi data dan perubahan penulisan format <i>phone</i> dan <i>fax</i> .	√		
7.	Mengijinkan <i>user</i> untuk mengekspor atau mengunduh hasil pendeteksian duplikasi data ke dalam <i>file excel</i> .	√		
8.	Mengijinkan <i>user</i> untuk menyimpan hasil perubahan penulisan format <i>phone</i> dan <i>fax</i> ke dalam <i>database</i> .		√	

Non Fungsional				
No.	Analisis Kebutuhan	M	D	I
Saya ingin sistem dapat				
1.	Memiliki hasil pendeteksian duplikasi data yang cukup akurat.	√		
2.	Mampu berjalan dengan berbasis web.	√		
3.	Menampilkan tampilan <i>web</i> yang sesuai dengan <i>template</i> sistem yang ada pada PT XYZ.	√		

Keterangan:

M = Mandatory (yang diinginkan),

D = Desirable (diperlukan),

I = Inessential (yang tidak diinginkan)

Requirement Elicitation Tahap 3

Berdasarkan Elisitasi Tahap II di atas, dibentuklah Elisitasi Tahap III yang diklasifikasikan kembali dengan menggunakan metode TOE (*Technical, Operational, Economic*) dengan opsi LMH (*Low, Medical, High*). Berikut adalah *requirement elicitation* yang ada pada tahap 3.

Fungsional										
Feasibility		T			O			E		
		L	M	H	L	M	H	L	M	H
No.	Analisis Kebutuhan									
Saya ingin sistem dapat										
1.	Mengijinkan <i>User</i> dapat melakukan <i>login</i> ke dalam sistem.		√				√	√		
2.	Mengijinkan <i>User</i> dapat memasukan atau mengimpor data konsumen Divisi <i>Consumer Care</i> ke dalam <i>database</i> .			√			√	√		
3.	Mengijinkan <i>User</i> untuk mereset atau menghapus data konsumen yang ada di dalam <i>database</i> .	√				√		√		
4.	Mengijinkan <i>User</i> untuk melakukan pendeteksian duplikasi data.			√			√		√	
5.	Mengijinkan <i>User</i> untuk merapikan format penulisan <i>phone</i> dan <i>fax</i> .			√		√			√	
6.	Mengijinkan <i>User</i> untuk menampilkan hasil deteksi duplikasi data dan perubahan penulisan format <i>phone</i> dan <i>fax</i> .	√					√	√		
7.	Mengijinkan <i>User</i> untuk mengekspor atau mengunduh hasil pendeteksian duplikasi data ke dalam <i>file excel</i> .			√			√		√	

8.	Mengijinkan <i>User</i> untuk menyimpan hasil perubahan penulisan format <i>phone</i> dan <i>fax</i> ke dalam <i>database</i> .		√			√		√		
----	---	--	---	--	--	---	--	---	--	--

Non Fungsional										
Feasibility		T			O			E		
		L	M	H	L	M	H	L	M	H
No.	Analisis Kebutuhan									
Saya ingin sistem dapat										
1.	Memiliki hasil pendeteksian duplikasi data yang cukup akurat.			√			√			√
2.	Mampu berjalan dengan berbasis web.		√				√	√		
3.	Menampilkan tampilan <i>web</i> yang sesuai dengan <i>template</i> sistem yang ada pada PT XYZ.		√				√	√		

Keterangan:

T = *Technical* O = *Operational* E = *Economic*

M = *Middle* L = *Low* H = *High*

Lampiran 3 – *Software Requirement Specification*

Software Requirement Specification

Sistem Data Cleaning

Versi 1.0

Rahma Mualifa

5 Juli 2015

Dipersiapkan untuk

Kelengkapan Tugas Akhir Informatika Universitas Bakrie

Dosen Pembimbing: Yusuf Lestanto

1. PENDAHULUAN

Penulisan dokumen SRS ini akan menggambarkan penjelasan seluruh kebutuhan pengembangan sistem *data cleaning* untuk Divisi *Consumer Care* PT XYZ sesuai dengan spesifikasi kebutuhan perangkat lunak. Dokumen ini dibuat berdasarkan standar penulisan SRS IEEE – 830.

1.1 Tujuan

Tujuan spesifikasi ini adalah menjelaskan secara mendetail tentang pengembangan sistem *data cleaning* di PT XYZ dengan menggunakan algoritma *Sorted Neighbourhood Method* dan *N-Gram* dalam mendeteksi duplikasi data. Dokumen menjelaskan tujuan dan fungsi, antarmuka, dan apa saja yang dapat dilakukan dalam aplikasi. Perancang dan pengembang dapat pula menggunakan dokumen ini sebagai pedoman untuk penerapan sistem di lapangan.

1.2 Ruang Lingkup

Sistem *data cleaning* ini dibuat berdasarkan kebutuhan staf *Sales Admin (user)* yang ada di Divisi *Consumer Care* PT XYZ. Sistem ini dibuat untuk membantu dalam membersihkan data konsumen berupa deteksi duplikasi data dan format penulisan telepon dan fax. Oleh karena Divisi *Consumer Care* PT XYZ masih menggunakan Microsoft Excel dalam melaksanakan kegiatan operasionalnya, sehingga untuk melakukan proses pembersihan data, *user* dapat mengimpor data dengan *file excel* ke dalam *database* sistem *data cleaning*.

Dalam mendeteksi duplikasi data akan diterapkan algoritma SNM dan *N-Gram* ke dalam fitur deteksi duplikasi data. Sementara, untuk memformat penulisan telepon dan fax, penulis tidak menggunakan algoritma khusus dan hanya mengeksekusi logika dari hasil observasi data yang dilakukan oleh peneliti dengan bantuan *user*.

1.3 Glosarium

<i>Term</i>	Definisi
Sistem <i>data cleaning</i>	Sebuah proses yang digunakan untuk menentukan data yang tidak akurat, tidak lengkap, atau data yang tidak jelas yang kemudian diperbaiki agar memiliki data yang berkualitas. Proses tersebut dapat terdiri atas pengecekan format, pengecekan kelengkapan, menghilangkan duplikasi atau kesalahan lain yang ada pada data (Chapman, 2005).
SNM	<i>Sorted Neighbourhood Method</i> , metode penggabungan atau penghapusan data yang digunakan pada pembuatan sistem <i>data cleaning</i> ini untuk menggabungkan duplikasi data dari dua atau lebih data yang kembar.
N-Gram	Metode yang digunakan untuk menghitung nilai kemiripan antar <i>string</i> . Maksud dari <i>n-gram</i> adalah <i>n</i> huruf yang berturut-turut dari sebuah kata. Nilai <i>n</i> yang digunakan adalah 2, 3, dan 4. Jika <i>n</i> = 2, maka disebut digram atau bigram. Jika <i>n</i> = 3 disebut dengan trigram, dan seterusnya (Tian, dkk., 2001).
<i>Database</i>	Kumpulan data yang disimpan secara sistematis dalam komputer.
IIS	<i>Internet Information Service</i> , sebuah HTTP <i>web server</i> yang digunakan dalam sistem operasi server Windows. Layanan ini berfungsi sebagai pendukung protokol TCP/IP yang berjalan dalam lapisan aplikasi (<i>application layer</i>). IIS juga menjadi fondasi dari platform Internet dan Intranet Microsoft.
<i>Web Server</i>	<i>Software</i> yang memberikan layanan berbasis data dan berfungsi menerima permintaan dari HTTP atau HTTPS pada klien yang dikenal dan biasanya kita kenal dengan nama web browser dan untuk mengirimkan kembali yang hasilnya dalam bentuk beberapa halaman web dan pada umumnya akan berbentuk dokumen HTML.
<i>Class Diagram</i>	Diagram UML yang menggambarkan kelas-kelas dalam sebuah sistem dan hubungannya antara satu dengan yang lain, serta terdapat pula atribut dan operasi di dalamnya.

<i>Use Case</i>	Diagram UML yang digunakan untuk memodelkan dan menyatakan unit fungsi atau layanan yang disediakan oleh sistem.
<i>Database Server</i>	Sebuah program komputer yang menyediakan layanan pengelolaan basis data dan melayani komputer atau program aplikasi basis data yang menggunakan model klien/server.
<i>Web browser</i>	Sebuah aplikasi pada komputer untuk menerima dan menampilkan informasi di internet.
<i>Web-based</i>	Suatu aplikasi yang dapat berjalan dengan menggunakan basis teknologi web atau browser.

1.4 Referensi

- IEEE. IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications. IEEE Computer Society, 1998.
- Chapman, A. D. (2005). Principles and Methods of Data Cleaning - Primary Species and Species-Occurrence Data, version 1.0 . (p. 1). Queensland, Australia: Global Biodiversity Information Facility.
- Tian, Z., Lu, H., Ji, W., Zhou, A., & Tian, Z. (2001). An n-gram-based Approach for Detecting Approximately Duplicate Database Records. *Springer Verlag*
- Dokumen pada Lampiran 1 - Wawancara.

1.5 Overview

Dokumen SRS ini terdiri dari tiga bab. Pada Bab 2 akan dijelaskan gambaran umum dari sistem *data cleaning* yang akan dibangun, yaitu tujuan, fungsi, batasan umum, dan asumsi-asumsi. Sementara, pada Bab 3 akan dijelaskan spesifikasi kebutuhan dalam pengembangan sistem *data cleaning*.

2. GAMBARAN UMUM

2.1 Perspektif Produk

Sistem *data cleaning* merupakan sistem *web-based* yang dibuat untuk mendeteksi duplikasi data dengan menerapkan algoritma SNM dan *N-Gram* dan memformat penulisan telepon dan fax yang ada pada master data konsumen Divisi *Consumer Care* PT XYZ.

2.2 Fungsi Produk

Sistem *data cleaning* merupakan sistem yang digunakan untuk *Sales Admin* Divisi *Consumer Care* pada PT XYZ yang memiliki fungsi sebagai berikut.

- Meningkatkan efektivitas dan efisiensi *user* dalam mendeteksi duplikasi data yang ada pada master data konsumen Divisi *Consumer Care* PT XYZ.
- Merapikan penulisan format telepon dan fax yang ada pada master data konsumen Divisi *Consumer Care* PT XYZ.
- Menyediakan fungsi impor data dan ekspor data berupa *file excel* dikarenakan manajemen *database* konsumen sampai saat ini masih menggunakan Microsoft Excel.

2.3 Karakteristik User

User pengguna sistem *data cleaning* adalah staf *Sales Admin* bagian *maintenance data* di Divisi *Consumer Care* PT XYZ. Kemampuan pengguna dalam menggunakan sistem ini dapat dikatakan sudah cukup memahami dalam mengaplikasikan sistem berbasis *web*. Selain itu, melihat usianya yang terbilang muda, yaitu 25 tahun. Akan tetapi, pengenalan atau *training* dalam menggunakan sistem dibutuhkan sebagai pembelajaran awal penggunaan.

2.4 Batasan Umum

Sistem ini hanya dirancang berdasarkan kebutuhan *user* sebagai pengelola data yang ada pada bagian *Sales Admin* Divisi *Consumer Care* PT XYZ. Berikut ini adalah batasan umum dari sistem *data cleaning* yang akan dibangun.

- Sistem *data cleaning* yang dibangun hanya terbatas sebagai deteksi duplikasi data dan merapikan format penulisan *phone* dan *fax* serta memungkinkan *user* untuk dapat mengimpor dan mengeksport data ke dan dari sistem.

- Sistem yang akan dibangun merupakan sistem berbasis *web*.
- Implementasi algoritma *Sorted Neighbourhood Method* dan *N-Gram* diterapkan dalam sistem untuk mendeteksi duplikasi data pada data konsumen Divisi *Consumer Care* PT XYZ.

2.5 Asumsi dan Ketergantungan

- *User* sistem ini minimal memiliki pengetahuan dalam menggunakan aplikasi atau sistem berbasis *web*.
- *User* menggunakan sistem *data cleaning* untuk mendeteksi duplikasi data dengan jumlah data sekitar 25.000 baris data secara berkala dalam sistem.

3. KEBUTUHAN SPESIFIKASI

3.1 User Interface

Untuk menampilkan antarmuka *user*, rancangan akan menggunakan *template* aplikasi *web* yang dimiliki oleh PT XYZ. Antarmuka dapat dibuka melalui *browser* Internet Explorer atau Mozilla Firefox yang mendukung sistem operasi Windows. Persyaratan tampilan antarmuka yang diharapkan oleh pengguna adalah tampilan yang *simple* yang sesuai dengan aturan *template* aplikasi *web* yang ada di PT XYZ dan mudah dimengerti dalam menggunakannya. Berikut ini adalah rancangan antarmuka *user* sistem *data cleaning*.

e-Data Cleaning
Version

[About Application](#)

Gambar 3.1 Rancangan Antarmuka Halaman *Login*

e-Data Cleaning
Version

[About Application | Logout](#)

Duplicate Detection
Fix Contact Numbe
Import Contac

Input Threshold Similari
0.6

Input Token Lengi
3

Input Gram Valu
2

Start Cleaning
View Clean Data

Gambar 3.2 Rancangan Antarmuka Halaman Utama atau Halaman *Duplicate Detection*

e-Data Cleaning
Version

[About Application](#) | [Logout](#)

Duplicate Detection | **Fix Contact Number** | **Import Contact**

Result

Old Data			New Data		
Organization ID	Phone	Fax	Organization ID	Phone	Fax

Save

Gambar 3.5 Rancangan Antarmuka Halaman *View* Hasil Format Penulisan Telepon dan Fax

e-Data Cleaning
Version

[About Application](#) | [Logout](#)

Duplicate Detection | **Fix Contact Number** | **Import Contact**

Input Threshold Similarity **Browse** **Save File**

Choose Sheet **Please save your file fir:**

Start Column

Start Row

Import Data

Gambar 3.6 Rancangan Antarmuka Halaman *Import Data*

3.2 Kebutuhan *Hardware*

Hardware yang dibutuhkan untuk mengembangkan produk yaitu:

1. Sebuah *server* untuk penyimpanan data sistem.

2. Sebuah komputer *laptop* atau PC yang digunakan untuk merancang, membangun dan menjalankan sistem, dengan minimal spesifikasi sebagai berikut.
 - Prosesor Intel Core i-5 @ 2.4 GHz
 - Memori dengan RAM 2 GB
 - 32-bit *Windows Operating System*
 - *Hard Disk* 320 GB

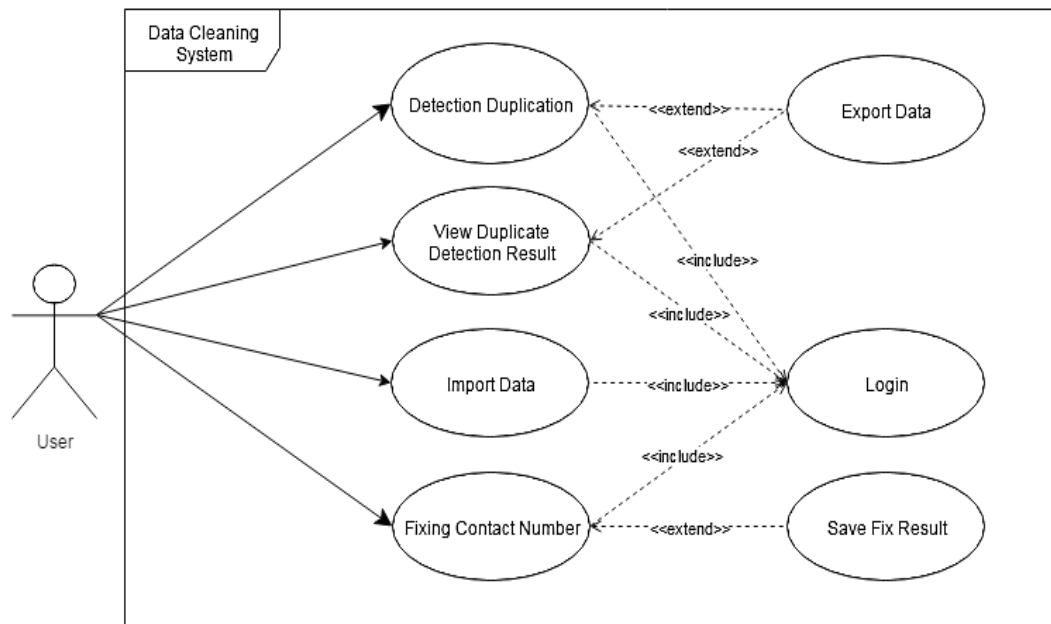
3.3 Kebutuhan Software

Software yang dibutuhkan untuk menjalankan sistem pelacakan kendaraan adalah sebagai berikut.

- Sistem basis data *Microsoft SQL Server*
- IDE *Microsoft Visual Studio 2010*
- IIS *version* 8.0 sebagai *web server*
- Sistem Operasi *Windows 7*
- *Web browser Internet Explorer* atau *Mozilla Firefox*

3.4 Kebutuhan Fungsional

Kebutuhan fungsional merujuk pada aktivitas apa saja yang dapat dilakukan melalui sistem pelacakan kendaraan. Kebutuhan fungsional ini digambarkan pada diagram *use case* berikut.



Gambar 3.7 Use Case Diagram Sistem Data Cleaning

Nama Use Case : **Login**

Aktor : *User*

Pre-condition : Aktor masuk ke dalam sistem

Post-condition : Aktor dapat login ke dalam sistem data cleaning

Deskripsi : Aktor melakukan proses login ke dalam sistem

Tabel 3.1 Deskripsi Aksi Aktor dan Respon Sistem Untuk Use Case Login

Aktor	Sistem
	1. Sistem menampilkan permintaan <i>Username</i> dan <i>Password</i> .
2. Aktor memasukkan <i>Username</i> dan <i>Password</i> .	3. Sistem melakukan proses validasi data yang di-input oleh aktor.
	4. Jika validasi benar, Aktor dapat masuk ke dalam sistem. Jika salah, Aktor menerima pesan bahwa <i>username</i> dan <i>password</i> yang dimasukkan oleh Aktor salah dan

	dapat memasukkan <i>username</i> dan <i>password</i> kembali.
--	---

- Nama *Use Case* : ***Detection Duplication***
- Aktor : *User*
- Pre-condition* : Aktor telah masuk ke halaman *Duplicate Detection* dan telah melakukan *import data*.
- Post-condition* : Sistem mendeteksi duplikasi data konsumen dan menampilkan hasilnya.
- Deskripsi : Aktor melakukan proses deteksi duplikasi data dengan menggunakan Algoritma *Sorted Neighbourhood* dan *N-gram*.

Tabel 3.2 Deskripsi Aksi Aktor dan Respon Sistem Untuk *Use Case Detection Duplication*

Aktor	Sistem
	1. Sistem memunculkan pilihan nilai <i>threshold</i> , <i>token length</i> , dan <i>gram</i> .
2. Aktor memasukkan karakter atau teks ke dalam <i>list removed text</i> dan memilih nilai <i>threshold</i> , <i>token length</i> , serta <i>gram</i> .	
3. Aktor menekan tombol <i>start cleaning</i> .	
	4. Sistem melakukan proses <i>cleaning</i> .
	5. Sistem menampilkan hasil proses pendeteksian duplikasi data.

Nama *Use Case* : ***View Duplicate Detection Result***
 Aktor : *User*
Pre-condition : Aktor telah masuk ke halaman *Duplicate Detection*.
Post-condition : Sistem menampilkan data yang telah dideteksi duplikasi datanya.
 Deskripsi : Aktor melihat hasil data yang telah dideteksi.

Tabel 3.3 Deskripsi Aksi Aktor dan Respon Sistem Untuk *Use Case View Duplicate Detection Result*

Aktor	Sistem
1. Aktor menekan tombol <i>View Clean Data</i> .	
	2. Sistem menampilkan data yang telah dideteksi.

Nama *Use Case* : ***Import Data***
 Aktor : *User*
Pre-condition : Aktor telah masuk ke halaman *Import Data*.
Post-condition : Aktor memasukkan master data konsumen ke dalam *database*.
 Deskripsi : Aktor melakukan proses *import* data ke dalam *database*.

Tabel 3.4 Deskripsi Aksi Aktor dan Respon Sistem Untuk *Use Case Import Data*

Aktor	Sistem
	1. Sistem menyediakan kolom <i>upload file</i> .
2. Aktor memilih <i>file excel</i> yang akan di- <i>upload</i> dengan <i>template</i> yang telah ditentukan.	

3. Aktor menekan tombol <i>Save File</i> .	4. Data akan tersimpan dan di kolom <i>Choose Sheet</i> akan muncul pilihan <i>sheet</i> .
5. Aktor memilih <i>sheet</i> dimana data konsumen berada.	
6. Aktor memasukkan nilai <i>Start Row</i> dan <i>Start Column</i> dimana data yang ada di dalam <i>file</i> akan mulai dibaca oleh sistem. Kemudian menekan tombol <i>Import Data</i> .	7. Sistem melakukan validasi data. Jika benar, data akan berhasil masuk ke dalam sistem. Jika salah, sistem akan mengeluarkan notifikasi.

Nama *Use Case* : ***Fixing Contact Number***
 Aktor : *User*
Pre-condition : Sistem masuk ke halaman *Fix Contact Number*.
Post-condition : Sistem menampilkan format nomor telepon dan fax yang telah dirapikan.
 Deskripsi : Sistem merapikan format nomor telepon dan fax.

Tabel 3.5 Deskripsi Aksi Aktor dan Respon Sistem Untuk *Use Case Fixing Contact Number*

Aktor	Sistem
1. Aktor membuka menu <i>Fix Contact Number</i> lalu menekan tombol <i>Start Cleaning</i> .	
	2. Sistem melakukan proses merapikan format penulisan.
	3. Sistem menampilkan data lama dan data baru yang telah dibersihkan.

Nama *Use Case* : ***Save Fix Result***
 Aktor : *User*

Pre-condition : Sistem telah menampilkan data *phone* dan *fax* yang telah dirapikan formatnya.

Post-condition : Data *phone* dan *fax* yang telah dirapikan disimpan ke dalam *database*.

Deskripsi : Aktor dapat menyimpan hasil format penulisan yang telah dirapikan.

Tabel 3.6 Deskripsi Aksi Aktor dan Respon Sistem Untuk *Use Case Save Fix Result*

Aktor	Sistem
	1. Sistem menampilkan data <i>phone</i> dan <i>fax</i> yang telah dirapikan formatnya.
2. Aktor menekan tombol <i>Save</i> .	
	3. Data telah berhasil disimpan ke dalam <i>database</i> .

Nama *Use Case* : ***Export Data***

Aktor : *User*

Pre-condition : Sistem telah menampilkan data yang telah dideteksi duplikasi datanya.

Post-condition : Data dapat diekspor ke dalam file *excel*.

Deskripsi : Aktor dapat melakukan ekspor data yang telah dideteksi duplikasi datanya.

Tabel 3.7 Deskripsi Aksi Aktor dan Respon Sistem Untuk *Use Case Export Data*

Aktor	Sistem
	1. Sistem menampilkan data yang telah dideteksi duplikasi datanya atau dengan menekan tombol <i>View Clean Data</i> .

2. Aktor menekan tombol <i>Export Data</i> .	3. Sistem memunculkan tampilan untuk mengunduh <i>file</i> dan menyimpan <i>file</i> .
4. Aktor mengunduh file dengan menekan tombol <i>OK</i> .	5. Sistem mengunduh <i>file excel</i> dan sistem berhasil diekspor ke dalam <i>file excel</i> .

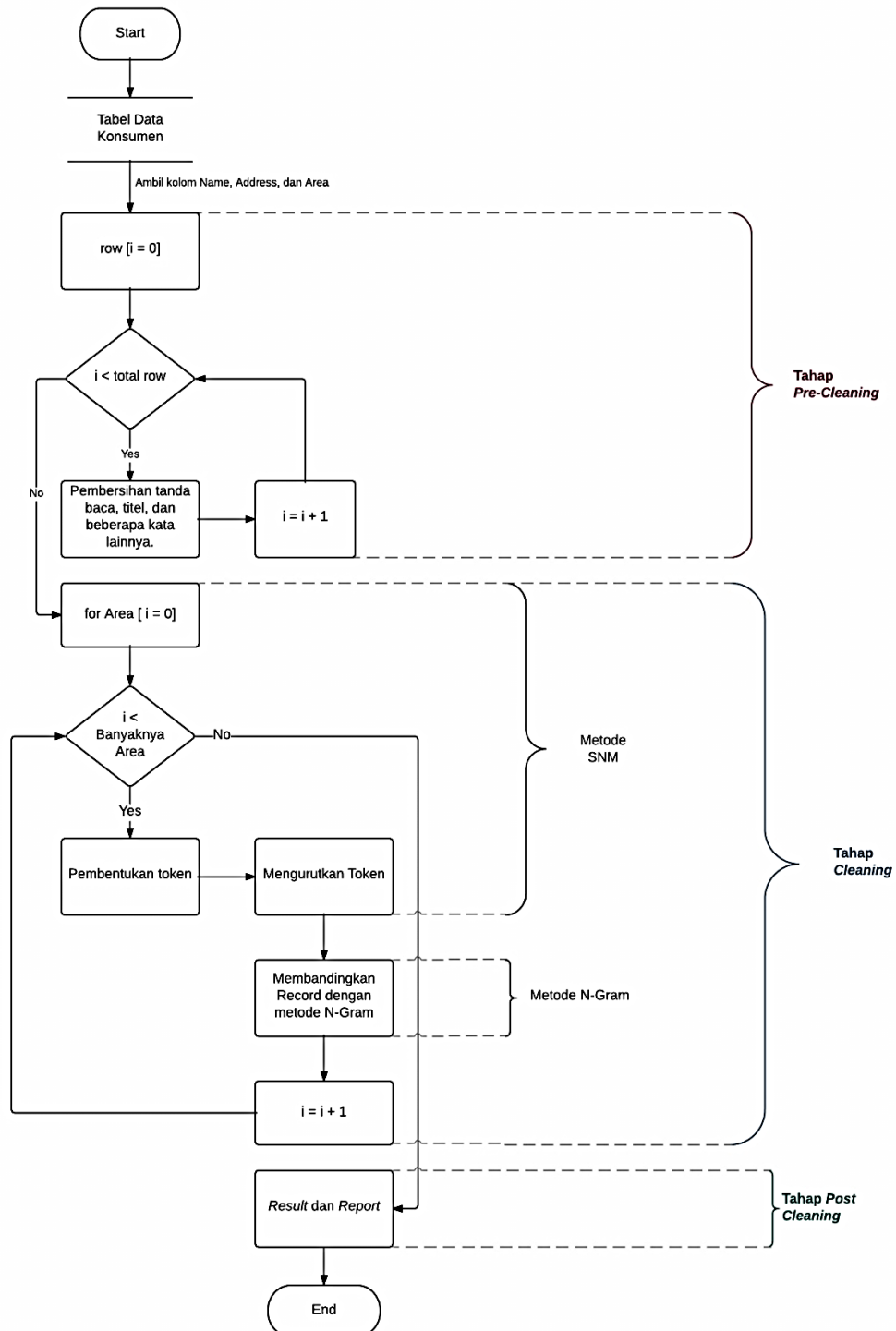
3.5 Kebutuhan Non-fungsional

Terdapat beberapa aspek non-fungsional yang dibutuhkan pada sistem *data cleaning* Divisi *Consumer Care* PT XYZ, yaitu:

- Aspek kemudahan, menampilkan *user interface* yang *simple* dan mudah dipahami.
- Aspek kontrol, mengizinkan *user* untuk melakukan pembatalan tindakan dalam aplikasi.
- Aspek *portability*, mampu berjalan pada *platform* atau sistem operasi *windows* dengan *browser* Mozilla Firefox atau Internet Explorer dan dapat pula berjalan melalui *mobile device*.
- Aspek efektivitas, seberapa efektif metode yang telah diterapkan pada sistem *data cleaning* yang telah dibangun.

3.6 Kebutuhan Lainnya

3.6.1 Perancangan Alur Algoritma Deteksi Duplikasi Data



Gambar 3.8 *Flowchart* Algoritma Deteksi Duplikasi Data

Proses pendeteksian duplikasi data dibagi menjadi beberapa tahap, yaitu tahap *pra-cleaning*, tahap *cleaning*, dan tahap *post-cleaning*.

I. Proses Pra-cleaning

Sebagai langkah pertama dalam proses *pra-cleaning* adalah menghilangkan karakter yang berada pada atribut *Name* dan *Address* terlebih dahulu untuk memudahkan sistem mendeteksi duplikasi data. Berikut ini rincian karakter yang akan dihilangkan terlebih dahulu.

Tabel 3.8 Rincian Karakter Yang Akan Dihilangkan Pada Proses Deteksi Duplikasi Data

No.	Atribut	Tindakan
1	<i>Name</i>	<ul style="list-style-type: none"> - Tanda baca seperti: titik, koma, tanda kurung, garis miring, dan tanda baca lainnya dihilangkan. - <i>Title</i> toko seperti PT, CV, APT, MM, SPM, TKLO, PBF, TOB, RS, TKOS, COS, TK dan <i>title</i> lainnya dihilangkan.
2	<i>Address</i>	<ul style="list-style-type: none"> - Tanda baca seperti: titik, koma, tanda kurung, titik dua, tanda hubung, garis miring, dan tanda baca lainnya dihilangkan. - Singkatan seperti Jl, Jln, Jalan, No dan <i>title</i> lainnya dihilangkan.

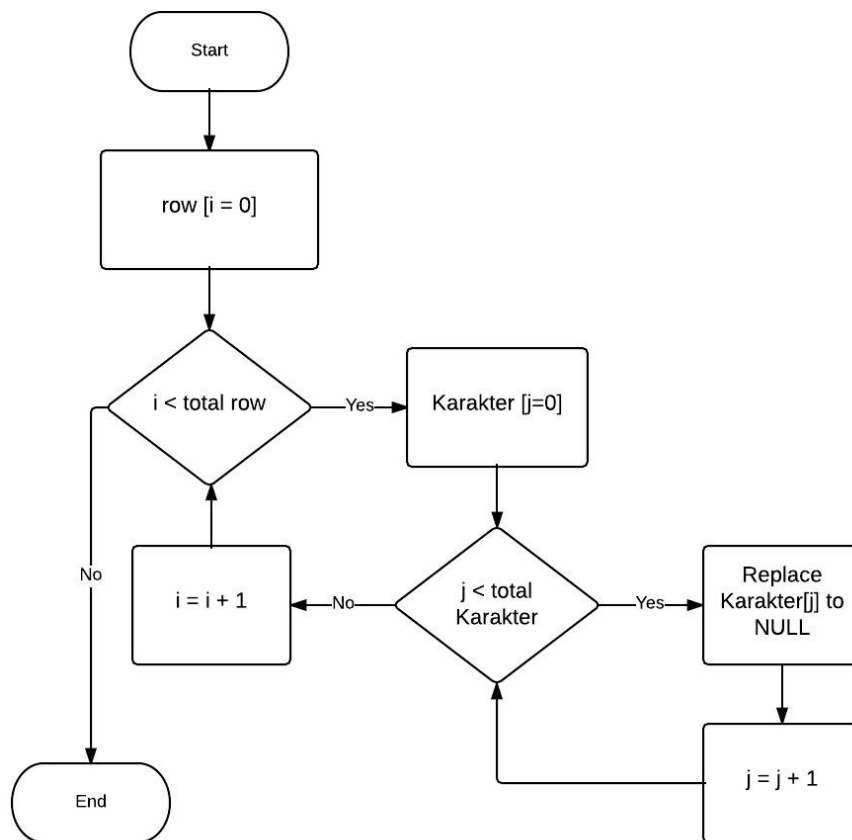
Tabel 3.9 Sebelum Dilakukan Proses Pra-cleaning

ID	Name	Address
109770	GUARD MEDAN SUN PLAZA, SPM	K.H.ZAINUL ARIFIN NO.7, JL
123627	GUARDIAN SUN PLAZA, APT	K.H.ZAINUL ARIFIN NO.7, JL

Tabel 3.10 Contoh Setelah Melewati Tahap Pra-Cleaning

ID	Name	Address
109770	GUARD MEDAN SUN PLAZA	KHZAINUL ARIFIN 7
123627	GUARDIAN SUN PLAZA	KHZAINUL ARIFIN 7

Berikut ini merupakan *flowchart* dari tahap pra-cleaning.



Gambar 3.9 *Flowchart* Tahap Pra-cleaning

II. Proses Cleaning

Proses *data cleaning* selanjutnya adalah tahap pendeteksian duplikasi dan tahap penghitungan nilai kemiripan antar *string*. Di mana kedua tahap ini merupakan dua tahap utama atau metode utama dalam sistem *data cleaning*.

a. Pendeteksian Duplikasi Data Dengan Metode SNM

Tahap pendeteksian duplikasi dilakukan dengan menerapkan Metode *Sorted Neighbourhood* sebagai berikut.

1. Pembagian Area

Data dibagi berdasarkan area sehingga proses membandingkan *record* dilakukan pada setiap area. Misalnya, area Bandar Lampung terdiri atas 1.000 *record* dan area Bandung terdiri atas 800 *record*. Oleh karena itu, proses deteksi duplikat untuk Bandar Lampung hanya pada 1.000 *record* saja dan area Bandung hanya pada 800 *record* saja. Dengan demikian, satu *record* yang ada dalam *database* konsumen tersebut tidak perlu dilakukan komparasi terhadap seluruh data yang ada. Tetapi, hanya pada data yang satu area saja.

2. Pembentukan Token

Dalam menentukan token dilakukan dengan menggunakan *n* huruf pertama pada tiap *string* yang ada di dalam *field*. Di mana nilai *n* akan di-input oleh pengguna ketika akan memulai proses deteksi duplikasi data. Seperti contoh tabel di bawah ini menggunakan $n = 3$.

Tabel 3.11 Contoh Setelah Proses Tokenisasi

ID	Name	Address
109770	GUA MED SUN PLA	KHZ ARI 7
123627	GUA SUN PLA	KHZ ARI 7

Berdasarkan tabel contoh di atas, terlihat proses perubahan data setelah melewati tahap *pre-cleaning*. Pada tabel di atas terdapat token yang dihasilkan dengan menggunakan tiga huruf awal pada tiap *string*.

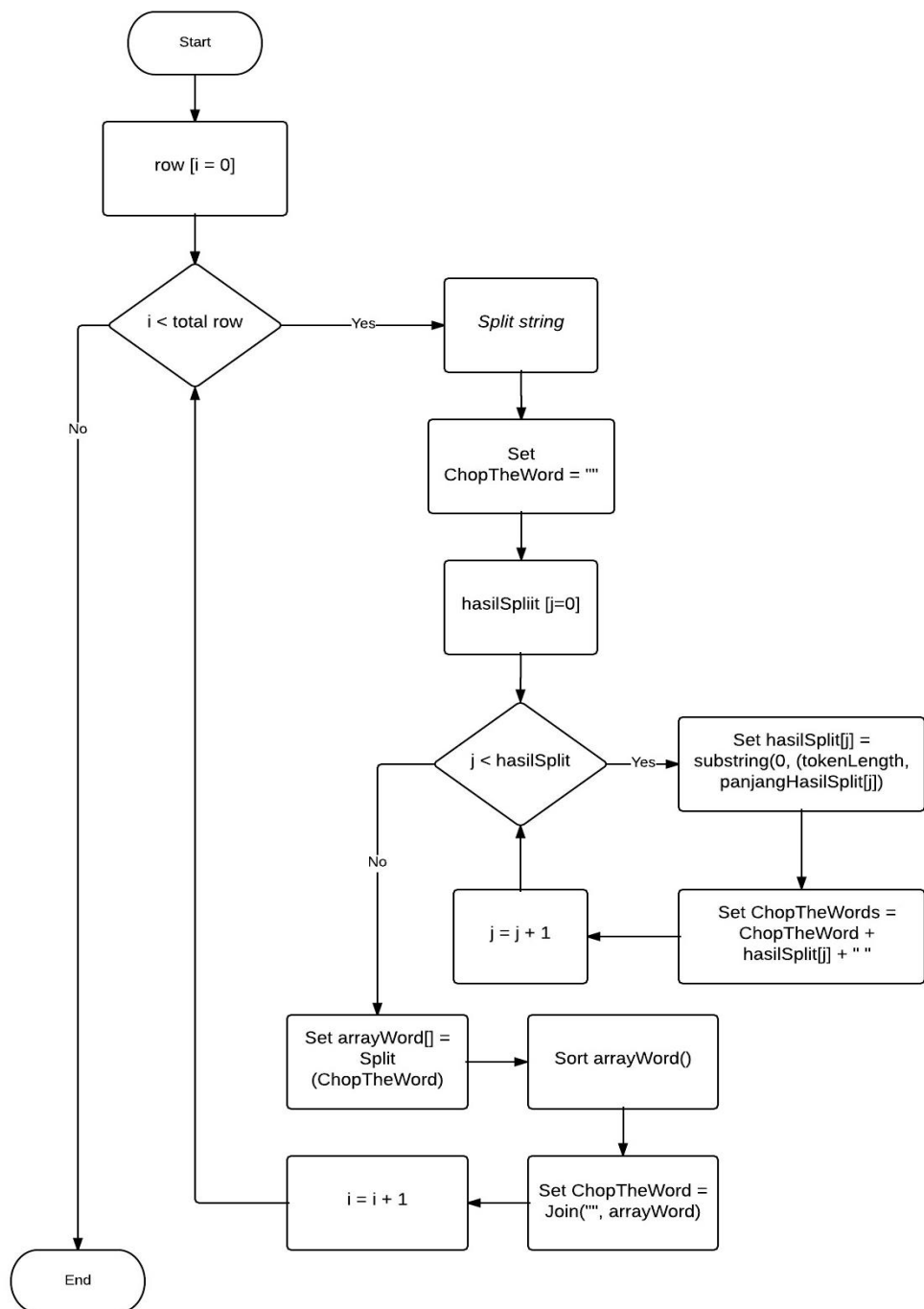
3. Mengurutkan *token*

Token yang berada pada tiap *field* kemudian diurutkan dan digabungkan seperti pada tabel contoh di bawah ini.

Tabel 3.12 Setelah Token Diurutkan dan Digabungkan

ID	Name	Address
109770	GUAMEDPLASUN	7ARIKHZ
123627	GUAPLASUN	7ARIKHZ

Berikut ini adalah *flowchart* dari proses pembentukan dan pengurutan token.



Gambar 3.10 *Flowchart* Tahap Tokenisasi

4. Menggabungkan *record*

Untuk menggabungkan *record* dilakukan dengan mengasumsikan sebuah *window* yang berukuran w bergerak melalui setiap *record* untuk membatasi proses perbandingan terhadap *record* yang berpotensi memiliki kemiripan data. Dimana

nilai w yang digunakan adalah jumlah *record* dalam setiap area. Dalam kasus data di PT XYZ ini, data akan digabungkan dengan memunculkan kode baru, yaitu *Clean Code* yang sama pada dua atau lebih data yang kembar. Penomoran *Clean Code* akan dibahas setelah tahap penghitungan kemiripan antar *string* berikut ini.

b. Membandingkan Nilai Kemiripan *Record* Dengan Metode *N-Gram*

Untuk membandingkan *record* digunakan metode pendekatan *N-Gram*. Sebagai contoh proses penghitungan kemiripan antar *string* dengan nilai $n = 2$ adalah sebagai berikut.

String 1 = GU UA AM ME ED DP PL LA AS SU UN = 11

String 2 = GU UA AP PL LA AS SU UN = 8

Jumlah gram yang sama = 7

Nilai kemiripan untuk *field Name* adalah $(2 \times 7) / (11+8) = 0.7$

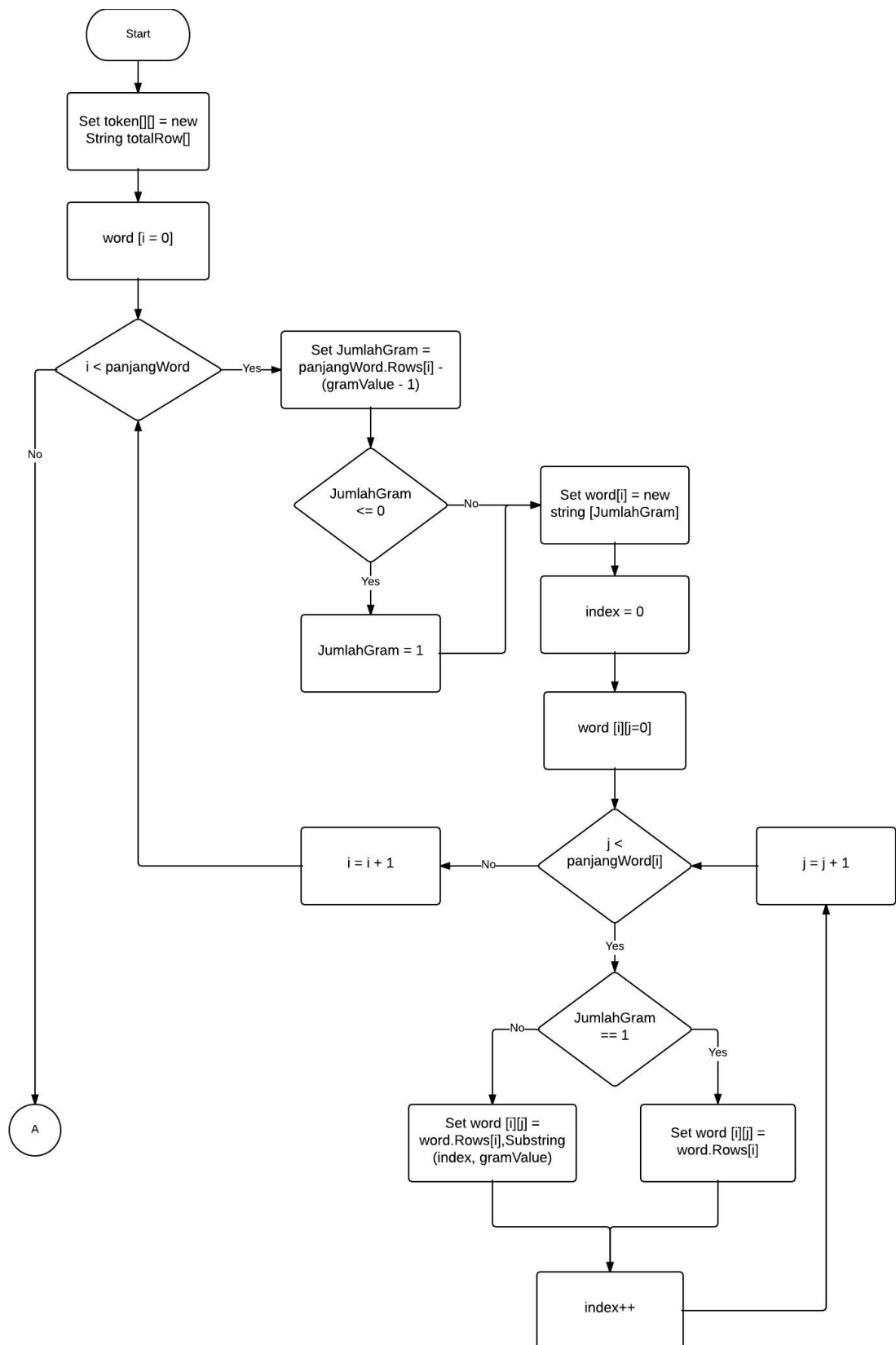
String 1 = 7A AR RI IK KH HZ = 6

String 2 = 7A AR RI IK KH HZ = 6

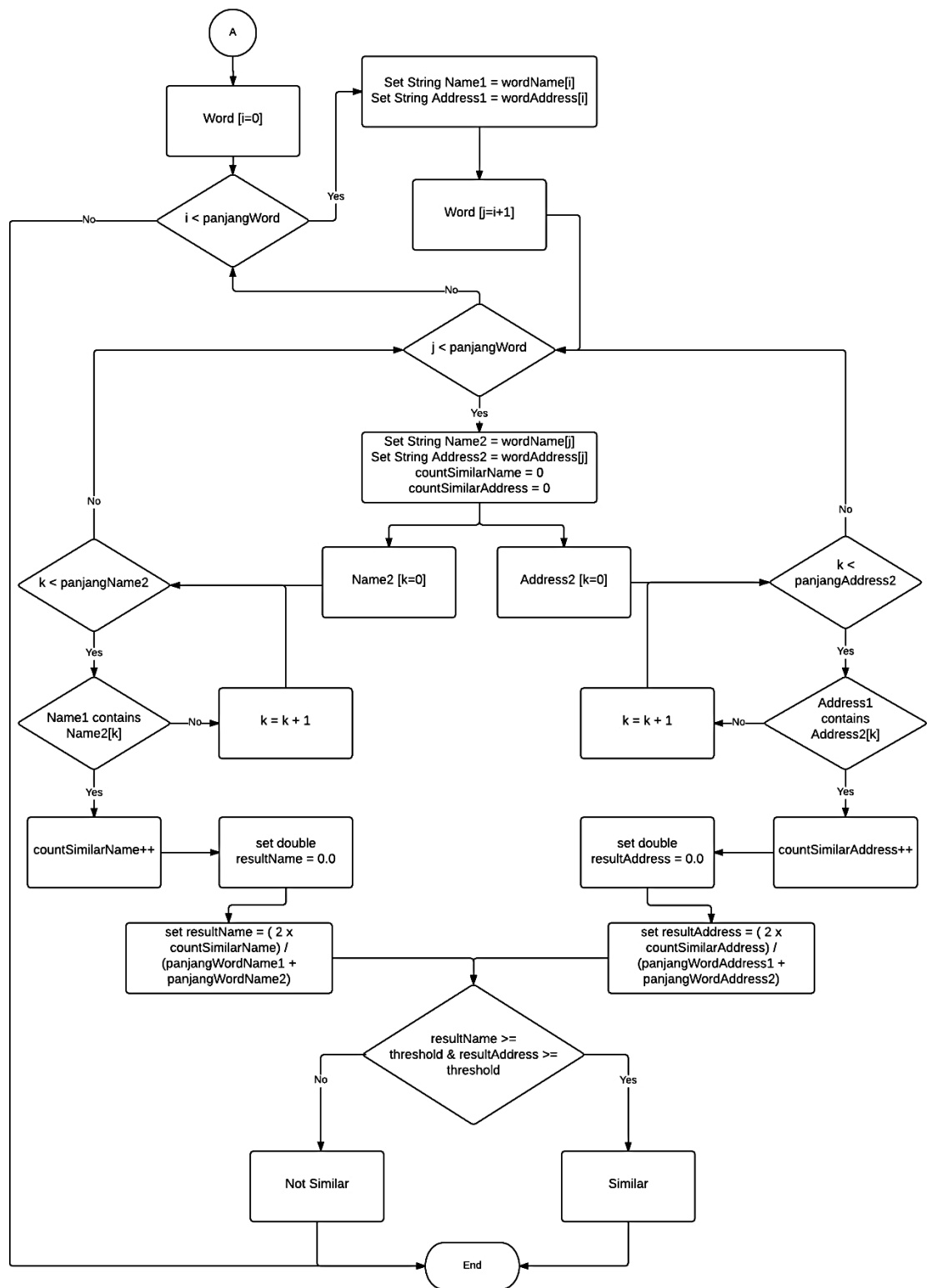
Jumlah gram yang sama = 6

Nilai kesamaan untuk *field Address* adalah $(2 \times 6) / (6+6) = 1$

Diasumsikan nilai ambang batas (*threshold*) dari *field Name* dan *Address* adalah 0,6. Maka, *record* yang ada pada contoh di atas dinyatakan kembar karena nilai kemiripan *field Name* dan *Address*, keduanya melebihi nilai *threshold*.



Gambar 3.11 *Flowchart* Tahap Pemecahan Kata Berdasarkan Nilai N-Gram



Gambar 3.12 Flowchart Perhitungan Nilai Kemiripan Antar Record

III. *Result dan Report*

Hasil dari proses *data cleaning* dalam hal pendeteksian duplikasi data yaitu munculnya ID baru yang disebut dengan *Clean Code*. Tujuan pembuatan *Clean Code* adalah sebagai kode yang akan menyatukan data yang terdeteksi sebagai data kembar. Sehingga, dalam proses pembersihan data konsumen pada PT XYZ tidak ada proses penggabungan atau penghapusan (*merge/purge*) seperti pada sistem *data cleaning* pada umumnya. Hal ini dilakukan untuk meminimalisasi adanya kesalahan pemilihan data ketika proses penggabungan atau penghapusan. Berikut ini adalah contoh tabel setelah data telah dibersihkan.

Tabel 3.13 Tabel *Clean*

Clean Code	Area	Organization ID	Name	Address
B101-1	Medan	109770	GUARD MEDAN SUN PLAZA, SPM	K.H.ZAINUL ARIFIN NO.7, JL
B101-1	Medan	123627	GUARDIAN SUN PLAZA, APT	K.H.ZAINUL ARIFIN NO.7, JL

Dari Tabel di atas terlihat contoh tabel *clean*. Pada tabel di atas, *record* dengan *Clean Code* = B101-1 memiliki dua buah *Organization_ID* yang berbeda namun memiliki data yang duplikat. *Clean Code* inilah yang akan digunakan untuk menyatukan data yang terdeteksi sebagai data duplikat.

3.6.2 Perancangan Prosedur Untuk Format Penulisan *Phone* dan *Fax*

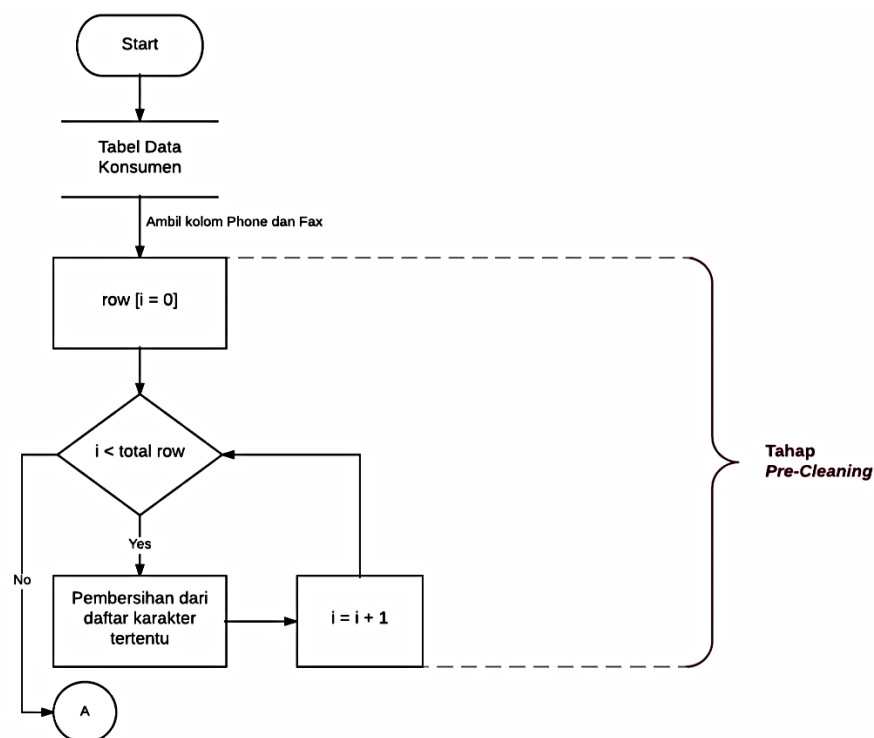
Proses merapikan format penulisan *phone* dan *fax* tidak menggunakan metode secara khusus seperti yang dilakukan pada proses deteksi duplikasi data. Prosedur yang digunakan untuk memformat penulisan *phone* dan *fax* serupa dengan proses *pra-cleaning* deteksi duplikasi data, yaitu menghilangkan beberapa daftar kata atau karakter. Kemudian, ditambahkan beberapa aturan baik untuk

phone dan *fax*. Berikut ini rincian prosedur untuk merapikan format *phone* dan *fax*.

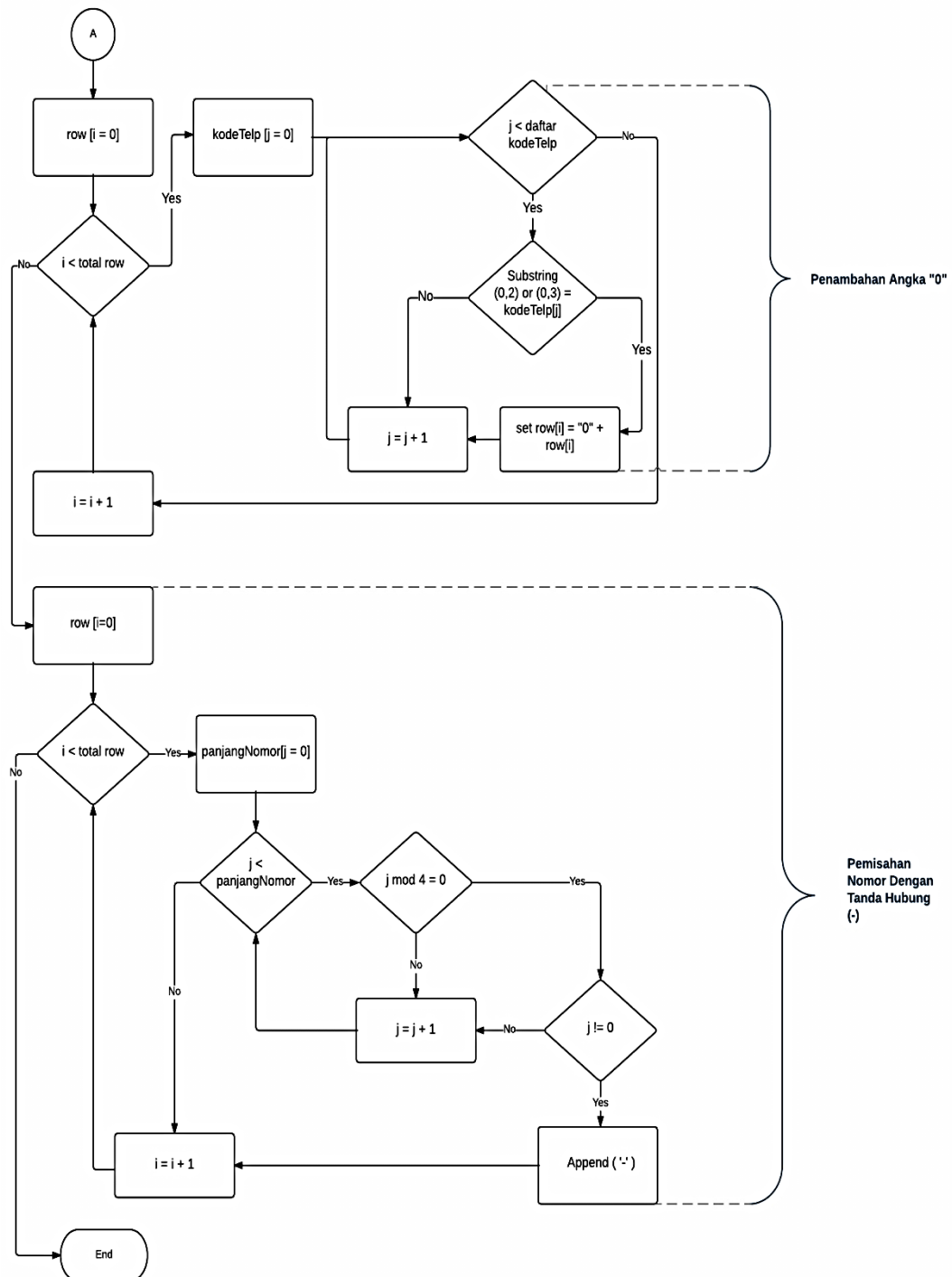
Tabel 3.14 Rincian Prosedur Dalam Proses Format Penulisan *Phone* dan *Fax*

Atribut	Prosedur
<i>Phone & Fax</i>	<ul style="list-style-type: none"> - Tanda baca seperti tanda hubung, tanda kurung, titik, koma, dan tanda baca lainnya dihilangkan terlebih dahulu. - Lalu, setiap karakter awal yang mengandung daftar kode telepon dan seluler Indonesia, maka akan ditambah angka 0 di bagian depannya. Misalnya, 218875241. Oleh karena 21 terbaca sebagai kode telepon daerah Jakarta (021), maka akan ditambah angka 0 di bagian depannya menjadi 0218875241. - Setelah itu, setiap 4 nomor telepon akan diberikan pemisah / tanda hubung (-).

Berdasarkan tabel di atas, dapat digambarkan *flowchart* prosedur untuk merapikan format *phone* dan *fax* seperti pada gambar berikut ini.



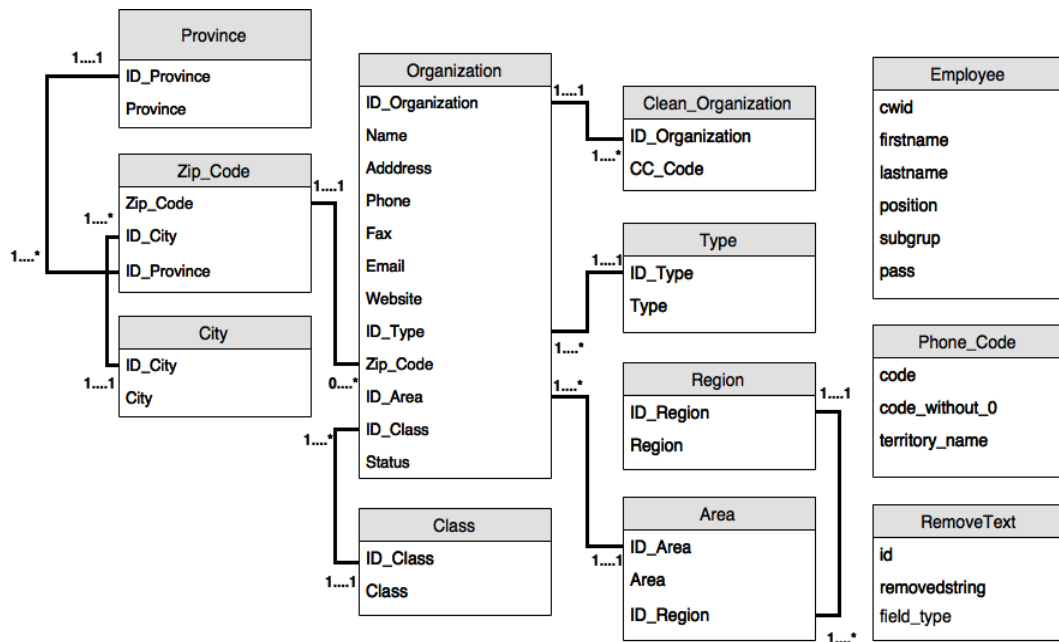
Gambar 3.13 *Flowchart* Prosedur Merapikan Format *Phone* dan *Fax* (1)



Gambar 3.14 *Flowchart* Prosedur Merapikan Format *Phone* dan *Fax* (2)

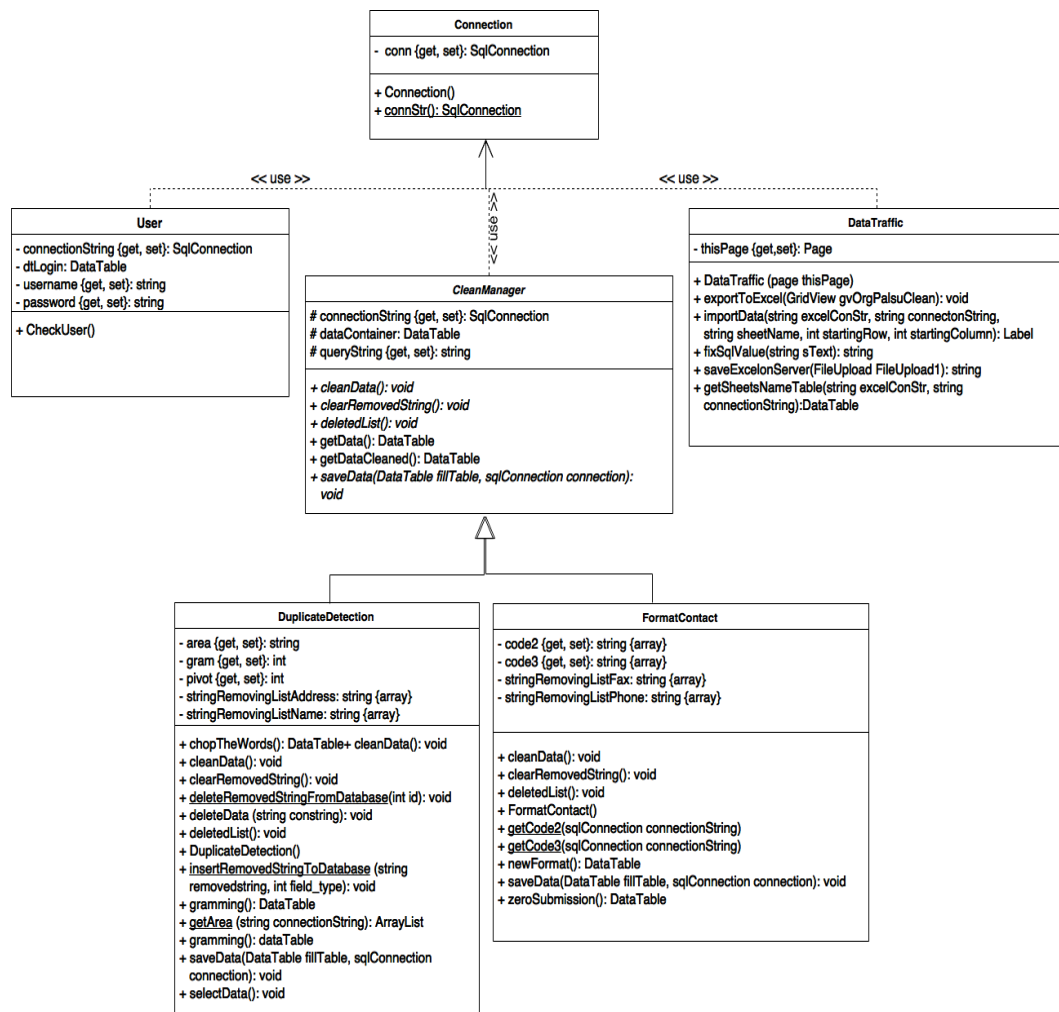
3.6.3 Rancangan Database

Rancangan *database* digunakan untuk menjelaskan hubungan antar data dalam basis data berdasarkan objek-objek dasar data yang mempunyai hubungan antar relasi. Berikut ini rancangan *database* sistem *data cleaning* master data konsumen PT XYZ.



Gambar 3.15 Rancangan Database Sistem Data Cleaning Pada Database Master Data Konsumen PT XYZ

3.6.4 Rancangan Class Diagram



Gambar 3.16 Class Diagram Sistem Data Cleaning Pada Database Master Data

Konsumen PT XYZ

Lampiran 4 – *Template Import Data*

Template Import Data

Data Konsumen Divisi *Consumer Care* PT XYZ

ID_Organization	Name	Address	ID_Type	ID_Area

Zip_Code	ID_Class	Phone	Fax	Email	Website	Status

Mengetahui,

Sales Admin PT XYZ



Widi Rizy Ayudya

Lampiran 5 – Hasil Pengujian Sistem

Pengujian Sistem *Data Cleaning*

Sebagai kebutuhan tugas akhir mahasiswa jurusan Teknik Informatika Universitas
Bakrie

Nama User : Widi Rizky Ayudya
Pekerjaan : Staf *Data Administrator*
Instansi : Divisi *Consumer Care* PT XYZ

Nama Penguji : Rahma Mualifa
Pekerjaan : Mahasiswa
Instansi : Universitas Bakrie

Jakarta, Desember 2015

User



(Widi Rizky Ayudya)

1. Aspek *Functionality*

Pengujian *functionality* mengacu pada kebutuhan fungsional sistem yang telah disepakati antara peneliti dan *user*. Pengujian ini dilakukan untuk memastikan bahwa sistem telah berjalan sesuai dengan fungsionalitas yang telah ditentukan.

No	Skenario Pengujian	Test Case	Output Yang Diharapkan	Output	Status
1.	Mengijinkan <i>user</i> dapat melakukan login ke dalam sistem.	<i>User input employee id dan password pada form login dan menekan tombol login.</i>	<i>User berhasil masuk ke dalam sistem. Jika employee id dan password tidak dimasukkan, maka akan muncul notifikasi untuk memasukkan employee id dan/atau password. Jika Jika employee id dan password yang dimasukkan salah, maka akan muncul notifikasi bahwa employee id dan password yang dimasukkan tidak benar dan user dapat memasukkannya kembali.</i>	Sesuai dg output Berhasil yg diharapkan	
2.	Mengijinkan <i>user</i> dapat memasukan atau mengimpor data konsumen Divisi <i>Consumer Care</i> ke dalam database.	<i>User melampirkan file excel dan menekan tombol save file.</i>	<i>User dapat melampirkan file excel. Jika user melampirkan file yang ekstensi-nya bukan xls atauxlsx, maka sistem akan mengeluarkan notifikasi untuk meminta user memasukkan kembali file excel.</i>	berhasil melampirkan file excel	Berhasil
		<i>User menekan tombol remove file.</i>	<i>File excel yang telah dilampirkan berhasil terhapus.</i>	dapat menghapus file excel yg dilampirkan	berhasil

No	Skenario Pengujian	Test Case	Output Yang Diharapkan	Output	Status
		User memilih <i>sheet</i> yang ada di dalam <i>file excel</i> .	User dapat memilih <i>sheet</i> berdasarkan <i>sheet</i> yang ada di <i>file excel</i> dari <i>drop down list</i> kolom "choose sheet".	sheet dapat dipilih	berhasil
		User mengisi kolom "start column" dan "start row".	User dapat mengisi "start column" dan "start row".	sesuai	berhasil
		User menekan tombol <i>import data</i> .	User berhasil mengimpor data ke dalam <i>database</i> .	data berhasil masuk	berhasil
3.	Mengijinkan user untuk melakukan pendeteksian duplikasi data.	User memasukkan teks atau karakter ke dalam <i>list removed text</i> dan memilih nilai <i>threshold</i> , <i>token length</i> , dan <i>gram value</i> .	User berhasil memasukkan teks atau karakter ke dalam <i>list removed text</i> dan memilih nilai <i>threshold</i> , <i>token length</i> , serta <i>gram value</i> melalui <i>drop down list</i> yang tersedia.	sesuai	berhasil
		User menekan tombol <i>start cleaning</i> .	Sistem memproses data yang telah di-impor untuk dideteksi duplikasi datanya kemudian memunculkan hasil deteksi duplikasi data.	sesuai	berhasil
4.	Mengijinkan user untuk merapikan format penulisan <i>phone</i> dan <i>fax</i> .	User menekan tombol <i>start fixing</i> .	Sistem memproses data yang telah di-impor untuk dirapikan format nomor telepon dan <i>fax</i> kemudian memunculkan hasilnya.	sesuai	berhasil
5.	Mengijinkan user untuk menampilkan hasil deteksi duplikasi data.	User menekan tombol <i>view clean data</i> . Kemudian dapat melakukan filter data dari kolom "view data" dan "area".	User dapat melihat hasil deteksi duplikasi data yang terakhir dilakukan dan dapat melakukan filter data dari kolom "view data" dan "area". Selain itu, total data yang ditampilkan juga sudah sesuai jumlahnya.	sesuai	berhasil

No	Skenario Pengujian	Test Case	Output Yang Diharapkan	Output	Status
6.	Mengijinkan User untuk mengekspor atau mengunduh hasil pendeteksian duplikasi data ke dalam file excel.	Setelah user melakukan proses deteksi duplikasi data, user menekan tombol <i>Export Data</i> . Atau dengan menekan tombol <i>view clean data</i> lalu menekan tombol <i>Export Data</i> .	User berhasil melakukan mengekspor data yang ditampilkan ke file excel.	Sesuai	Berhasil
7.	Mengijinkan User untuk menyimpan hasil perubahan penulisan format <i>phone</i> dan <i>fax</i> ke dalam database.	Setelah user melakukan proses pemformatan nomor telepon dan <i>fax</i> , user menekan tombol <i>save</i> .	User berhasil menyimpan hasil pemformatan nomor telepon dan <i>fax</i> ke dalam database.	Sesuai	Berhasil

2. Aspek Usability

Pengujian *usability* mengacu pada penilaian *user* dalam menggunakan dan mengoperasikan sistem. Pengujian ini akan dibedakan ke dalam 5 faktor, yaitu *learnability*, *efficiency*, *memorability*, *errors/safety*, dan *satisfaction*.

No	Pertanyaan	STS	TS	N	S	SS
Learnability						
1.	Penempatan menu pada sistem mudah dipahami dan dijangkau.				✓	
2.	Keterangan pada setiap halaman yang ada cukup mudah untuk dipahami.				✓	
3.	Kata-kata yang dipakai dalam memberikan informasi sudah jelas.				✓	
4.	Teks yang digunakan pada sistem mudah dan jelas.				✓	
Efficiency						
5.	Saat menu Anda klik, sistem dapat menampilkan dengan cepat.					✓
6.	Proses pendeteksian duplikasi data berjalan dengan baik dan membutuhkan waktu yang relatif cepat.				✓	
7.	Proses <i>import data</i> berjalan dengan baik dan cepat.					✓
8.	Proses merapikan format penulisan <i>phone</i> dan <i>fax</i> dapat berjalan dengan benar dan cepat.					✓
Memorability						
9.	Letak posisi menu atau tombol mudah diingat.				✓	
10.	Pemilihan kata dalam aplikasi mudah diingat.				✓	
11.	Suasana dalam aplikasi menggambarkan web yang dimiliki oleh PT XYZ dalam penelitian ini.					✓
12.	Nama aplikasi 'e-Data Cleaning' mudah diingat dan sesuai dengan pengaplikasiannya.					✓
Errors						
13.	Tidak ada informasi yang tumpang tindih dalam sistem					✓
14.	Terdapat pesan yang jelas ketika terdapat <i>error</i> input.				✓	
15.	Tidak terdapat link yang tidak sesuai atau belum jadi dalam aplikasi ketika menu atau <i>link</i> di klik.					✓

No	Pertanyaan	STS	TS	N	S	SS
16.	Tidak terdapat klik menu yang tidak memberikan respon atau sejenisnya.					✓
Satisfaction						
17.	Sistem memberikan layanan dan informasi yang mudah dipahami dan nyaman digunakan.			✓		
18.	Secara keseluruhan, Anda puas dengan informasi yang diberikan pada sistem.				✓	
19.	Secara keseluruhan, Anda puas dengan fitur, dan kemudahan yang diberikan pada sistem.				✓	
20.	Apabila telah diimplementasi, Anda ingin untuk menggunakan sistem ini sebagai sistem yang membantu Anda dalam mengatasi duplikasi data dan format penulisan <i>phone</i> dan <i>fax</i> pada master data konsumen Divisi <i>Consumer Care</i> di PT XYZ.					✓

Keterangan:

STS = Sangat Tidak Setuju

TS = Tidak Setuju

N = Netral

SS = Setuju

SS = Sangat Setuju