TUGAS AKHIR – KS 141501

PEMBUATAN APLIKASI UNTUK QUERY ELEMEN TABEL HTML DENGAN MENGGUNAKAN MEKANISME *INCREMENTAL ADAPTIVE TIME TO LIVE* (STUDI KASUS: WEBSITE NILAI TUKAR MATA UANG, BMKG, DAN PENGUMUMAN)

DEVELOP APPLICATION FOR QUERY TABLE ELEMENT HTML WITH INCREMENTAL ADAPTIVE TIME TO LIVE MEKANISM (CASE STUDY: WEBSITE CURRENCY, BMKG, AND ANNOUNCEMENT)

AHSANUL KHULUQ SAIFULLAH
NRP 05211 4400 00106

Dosen Pembimbing :
Radityo Prasetianto Wibowo, S.Kom., M.Kom.

DEPARTEMEN SISTEM INFORMASI
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya 2018

TUGAS AKHIR – KS 141501

PEMBUATAN APLIKASI UNTUK QUERY ELEMEN TABEL HTML DENGAN MENGGUNAKAN MEKANISME *INCREMENTAL ADAPTIVE TIME TO LIVE* (STUDI KASUS: WEBSITE NILAI TUKAR MATA UANG, BMKG, DAN PENGUMUMAN)

AHSANUL KHULUQ SAIFULLAH
NRP 05211 4400 00106

Dosen Pembimbing :
Radityo Prasetianto Wibowo, S.Kom., M.Kom.

DEPARTEMEN SISTEM INFORMASI
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya 2018

FINAL PROJECT – KS 141501

DEVELOP APPLICATION FOR QUERY TABLE ELEMENT HTML WITH INCREMENTAL ADAPTIVE TIME TO LIVE MEKANISM (CASE STUDY: WEBSITE CURRENCY, BMKG, AND ANNOUNCEMENT)

AHSANUL KHULUQ SAIFULLAH
NRP 05211 4400 00106

SUPERVISOR:
Radityo Prasetianto Wibowo, S.Kom., M.Kom..

DEPARTMENT OF INFORMATION SYSTEMS
Faculty of Information Technology and Communication
Institut Teknologi Sepuluh Nopember
Surabaya 2018

LEMBAR PENGESAHAN

PEMBUATAN APLIKASI UNTUK QUERY ELEMEN TABEL HTML DENGAN MENGGUNAKAN MEKANISME *INCREMENTAL ADAPTIVE TIME TO LIVE* (STUDI KASUS: WEBSITE NILAI TUKAR MATA UANG, BMKG DAN PENGUMUMAN)

TUGAS AKHIR

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

Oleh:

AHSANUL KHULUQ SAIFULLAH
NRP. 05211 4400 00106

Surabaya, 13 Juli 2018

KETUA
DEPARTEMEN SISTEM INFORMASI

Dr. Ir. Aris Tjahyanto, M.Kom.
NIP. 19650310 199102 1 001

LEMBAR PERSETUJUAN

PEMBUATAN APLIKASI UNTUK QUERY ELEMEN TABEL HTML DENGAN MENGGUNAKAN MEKANISME *INCREMENTAL ADAPTIVE TIME TO LIVE* (STUDI KASUS: WEBSITE NILAI TUKAR MATA UANG, BMKG DAN PENGUMUMAN)

TUGAS AKHIR

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

Oleh:

AHSANUL KHULUQ SAIFULLAH
NRP. 05211 4400 00106

Disetujui Tim Penguji: Tanggal Ujian : 10 Juli 2018
Periode Wisuda : September 2018

Radityo Prasetianto Wibowo, S.Kom., M.Kom. (Pembimbing I)

Renny Pradina Kusumawardani, S.T., M.T. (Penguji I)

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

PEMBUATAN APLIKASI UNTUK QUERY ELEMEN TABEL HTML DENGAN MENGGUNAKAN MEKANISME *INCREMENTAL ADAPTIVE TIME TO LIVE* (STUDI KASUS: WEBSITE NILAI TUKAR MATA UANG, BMKG, DAN PENGUMUMAN)

Nama Mahasiswa : Ahsanul Khuluq Saifullah
NRP : 05211 4400 00106
Jurusan : Sistem Informasi FTIK-ITS
Pembimbing 1 : Radityo Prasetianto Wibowo, S.Kom., M.Kom.

ABSTRAK

Internet merupakan media penyebaran informasi yang berkembang sangat pesat dengan jumlah penggunanya yang terus meningkat seiring dengan berkembangnya teknologi. Tingginya jumlah website aktif yang diimbangi dengan meningkatnya jumlah pengguna internet menunjukkan pertukaran informasi melalui website begitu besar. Meluasnya penggunaan website telah menimbulkan beberapa permasalahan pengelolaan data baru, seperti mengesktrak data dari halaman web dan membuat database dapat diakses dari browser, dan telah memperbarui minat pada masalah yang telah muncul dalam konteks lain sebelumnya, seperti query grafik, data semi terstruktur dan dokumen terstruktur. Kebanyakan masalah ini pada dasarnya adalah restrukturasi data, seperti melakukan perubahan struktur data tertentu untuk membuat representasi lain.

Permasalahan lain juga terjadi dalam bagaimana melakukan penyimpanan data pada tempat penyimpanan sementara untuk menyimpan hasil query data. Kapan waktu yang tepat melakukan pembaharuan terhadap penyimpanan sementara tersebut. Jika frekuensi waktu pembaharuan penyimpanan terlalu cepat akan mengakibatkan komputasi overhead. Sedangkan jika

frekuensi waktu pembaharuan terlalu lama, maka data yang tersimpan akan usang.

Pada tugas akhir ini akan dikembangkan aplikasi yang akan merestrukturasi data pada tabel yang terdapat didalam web yang disimpan dan dapat diakses pada aplikasi lain. Kemudian untuk melakukan pengaksesan data menggunakan SQL agar bisa dilakukan proses query projection, selection dan order pada data yang ada. Pengaksesan data ini menggunakan mekanisme Incremental Adaptive Time to Live. Sehingga data tersebut dapat dilakukan pembaharuan sesuai dengan waktu time to live data tersebut.

Mekanisme Incremental Adaptive Time to Live menambahkan nilai TTL berdasarkan data hasil query yang ada. Ketika data yang diambil dari sumber data ternyata memiliki kesamaan dengan data sebelumnya, maka nilai time to live akan ditambahka sesuai dengan fungsi yang dipilih pengguna. Ketika data yang diambil oleh sumber data berbeda dengan penyimpanan sementara, maka nilai time to live akan diatur menjadi nilai paling rendah.

Hasil menunjukkan bahwa dalam melakukan pengambilan data yang memiliki perubahan cepat (perubahan dalam menit) lebih baik menggunakan pendeketan tipe TTL Linear tetapi jika website dengan perubahan yang cepat dan dilakukan pengambilan data dengan waktu sedang bisa menggunakan seluruh tipe kenaikan TTL yang ada. Sedangkan untuk data yang memiliki perubahan sedang (perubahan dalam jam) lebih baik menggunakan pendekatan tipe TTL Linear, dan untuk website yang jarang memiliki perubahan data lebih baik menggunakan pendekatan TTL Exponensial.

Kata Kunci: Website, Data semi struktural, Dokumen tersturktur, Incremental Adaptive Time to Live, Projection, Selection, Order.

DEVELOP APPLICATION FOR QUERY TABLE ELEMENT HTML WITH INCREMENTAL ADAPTIVE TIME TO LIVE MEKANISM (CASE STUDY: WEBSITE CURRENCY, BMKG, AND ANNOUNCEMENT)

Student Name : Ahsanul Khuluq Saifullah
NRP : 05211 4400 00106
Department : Sistem Informasi FTIK-ITS
Supervisor 1 : Radityo Prasetianto Wibowo, S.Kom., M.Kom.

ABSTRACT

The Internet is a medium of information spread that is growing very rapidly with the number of users who continue to increase along with the development of technology. The high number of active websites being offset by the increasing number of internet users shows the exchange of information through websites is so great. Widespread use of the website has led to several new data management issues, such as extracting data from web pages and making databases accessible from browsers, and has renewed interest in issues that have surfaced in other previous contexts, such as graphical queries, semi-structured data and structured documents. Most of these problems are essentially data restructuring, such as changing certain data structures to make other representations.

Other problems also occur in how to perform data storage on the temporary storage to store the data query results. When is the right time to update the temporary storage. If the frequency of storage update time is too fast will result in overhead computing. Whereas if the update time frequency is too long, then the stored data will be obsolete.

In this final project will be developed an application that will restructure the data in the table contained within the web that is stored and can be accessed on other applications. Then to make

accessing data using SQL in order to be able to process projection query, selection and order on existing data. Accessing this data using Incremental Adaptive Time to Live mechanism. So that data can be updated according to the time to live the data.

Incremental Adaptive Time to Live mechanism adds TTL values based on existing query data. When data retrieved from a data source has in common with previous data, the time to live value will be added in accordance with the function selected by user. When data captured by a data source is different from temporary storage, the time to live value will be set to the lowest value.

The results show that in making data with quick change (change in minutes) it is better to use Linear TTL type shortcut but if the website with fast change and do the data retrieval with medium time can use all type of TTL increase there. As for data that has a moderate change (change in hours) it is better to use Linear TTL type approach, and for websites that rarely have better data changes using the Exponential TTL approach.

Keywords: Website, Semi-structural Data, Structured Document, Incremental Adaptive Time to Live, Projection, Selection, Order.

KATA PENGANTAR

Puji dan syukur penulis tuturkan ke hadirat Allah SWT, Tuhan Semesta Alam yang telah memberikan kekuatan dan hidayah-Nya kepada penulis sehingga penulis mendapatkan kelancaran dalam menyelesaikan tugas akhir dengan judul:

PEMBUATAN APLIKASI UNTUK QUERY ELEMEN TABEL HTML DENGAN MENGGUNAKAN MEKANISME *INCREMENTAL ADAPTIVE TIME TO LIVE* (STUDI KASUS: WEBSITE NILAI TUKAR MATA UANG, BMKG, DAN PENGUMUMAN)

yang merupakan salah satu syarat kelulusan pada Departemen Sistem Informasi, Fakultas Teknologi Informasi dan Komunikasi, Institut Teknologi Sepuluh Nopember Surabaya.

Terima kasih penulis sampaikan kepada pihak-pihak yang telah mendukung, memberikan saran, motivasi, semangat serta bantuan baik berupa materiil maupun moril demi tercapainya tujuan pembuatan tugas akhir ini. Tugas akhir ini tidak akan mernah terwujud tanpa bantuan dan dukungan dari berbagai pihak yang sudah meluangkan waktu, tenaga dan pikirannya dalam membantu penulis menyelesaikan tugas akhir. Secara khusus penulis menyampaikan ucapan terima kasih yang sebanyak-banyaknya kepada:

1. Bapak Zaenal Arifin dan Ibu Mudjilah selaku kedua orang tua serta Sulthoni Zafrullah dan Nanang Khizbullah sebagai kakak penulis yang memberikan dukungan serta motivasi tiada henti untuk menyelesaikan tugas akhir ini.
2. Bapak Radityo Prasetianto Wibowo, S.Kom., M.Kom., selaku dosen pembimbing yang meluangkan waktu dalam memberikan ilmu, petunjuk, masukan serta motivasi untuk kelancaran tugas akhir.
3. Ibu Hanim Maria Astuti, S.Kom., M.Sc., selaku dosen wali penulis yang telah memberikan pengalaman serta nasehat

selama menempuh Pendidikan di Departemen Sistem Informasi.

4. Ibu Renny Pradina Kusumawardani, S.T., M.T., dan Bapak Faisal Johan Atletiko, S.Kom., M.T., selaku dosen penguji yang telah memberikan kritik, saran, serta masukan yang berharga sehingga dapat memperbaiki tugas akhir ini.
5. Seluruh dosen pengajar beserta staf dan karyawan di Departemen Sistem Informasi, Fakultas Teknologi Informasi dan Komunikasi ITS yang telah memberikan ilmu dan pengalaman yang berharga kepada penulis.
6. Jwalita Galuh Garini, Aditya Septa Setiabudi, Berlian Fajar Yusuf, Ferdian Widyatama, Abdul Azis dan Adam Ardiansyah selaku teman-teman yang membantu dengan memberikan pemikiran, masukan, serta pertolongan dalam menyelesaikan tugas akhir.
7. Teman-teman Wolfpack yang mau menjadi keluarga dalam menempuh kehidupan selama berada pada kehidupan kampus.
8. Teman-teman ISE! 2016 selaku partner dalam menyukseskan ISE! 2016 selama satu tahun dalam periode Februari 2016 – Februari 2017.
9. Rekan-rekan yang tergabung dalam Laboratorium ADDI yang senantiasa membantu dan memberikan semangat dan informasi dalam pengerjaan tugas akhir.
10. Mas Asep Saepudin selaku teman kerja di Bukalapak yang telah memberikan masukan dalam melakukan pengembangan aplikasi yang dikembangkan dalam tugas akhir ini.
11. Serta semua pihak yang telah membantu dalam pengerjaan tugas akhir ini yang belum mampu penulis sebutkan diatas.

Terima kasih atas segala bantuan, dukungan, serta doa yang telah diberikan selama ini. Penulis menyadari bahwa tugas akhir ini masih belum sempurna dan memiliki banyak kekurangan di

dalamnya. Oleh karena itu, penulis juga memohon maaf atas segala kesalahan penulis buat dalam buku tugas akhir ini. Penulis membuka pintu selebar-lebarnya bagi pihak yang inin memberikan kritik maupun saran, serta penelitian selanjutnya yang ingin menyempurnakan hasil dari penelitian tugas akhir ini. Semoga buku tugas akhir ini bermanfaat bagi seluruh pembaca.

Surabaya, Juni 2018

Penulis

Halaman ini sengaja dikosongkan

DAFTAR ISI

ABSTRAK v
ABSTRACT vii
KATA PENGANTAR ix
DAFTAR ISI xiii
DAFTAR GAMBAR xv
DAFTAR KODE xvii
DAFTAR TABEL xix
BAB I PENDAHULUAN 1
1.1. Latar Belakang Masalah 1
1.2. Perumusan Masalah 2
1.3. Batasan Masalah 2
1.4. Tujuan Penelitian 3
1.5. Manfaat Penelitian 3
1.6. Relevansi 4
BAB II TINJAUAN PUSTAKA 5
2.1. Penelitian Sebelumnya 5
2.2. Produk Sejenis 6
2.3. Landasan Teori 8
2.3.1. *Parser* 8
2.3.2. *Dokumen Object Model* (DOM) 9
2.3.3. Metadata 10
2.3.4. JavaScript Object Notation (JSON) 11
2.3.5. Structured Query Language (SQL) 13
2.3.6. Protokol REST 14
2.3.7. Mekanisme *Adaptive Time to Live* 15
BAB III METODOLOGI PENELITIAN 19
3.1. Studi Literatur 19
3.2. Perancangan dan Pengembangan Aplikasi 20
3.2.1. Perencanaan 21
3.2.2. Desain 22
3.2.3. Konstruksi 23
3.2.4. Development Delivery dan Feedback 23
3.3. Pengujian 23
3.4. Penyusunan Buku Tugas Akhir 24
BAB IV PERANCANGAN 27

4.1. Perancangan Sistem 27
4.1.1. Identifikasi Pengguna 27
4.1.2. Identifikasi Kebutuhan Pengguna 27
4.1.3. Arsitektur Sistem 28
4.2. Rancangan Database 36
4.3. Rancangan Proses Sistem.....38
4.4. Rancangan Antar Muka 40
4.5. Rancangan Pengujian.....42
4.5.1. Pengujian Fungsional.....42
4.5.2. Pengujian Tipe Kenaikan TTL 44
BAB V IMPLEMENTASI 47
5.1. Perangkat Penelitian.....47
5.2. Pengembangan Sistem 48
5.2.1. Pembuatan HTML Ekstraktor.....48
5.2.2. Pembuatan SQL Engine.....56
BAB VI HASIL DAN PEMBAHASAN.....73
6.1. Hasil Pengujian 73
6.1.1. Pengujian Fungsional.....73
6.1.2. Pengujian Tipe Kenaikan TTL 81
6.2. Pembahasan.....84
6.2.1. Analisa Hasil Pengujian Fungsional 84
6.2.2. Analisa Hasil Pengujian Tipe Kenaikan TTL 84
BAB VII KESIMPULAN DAN SARAN 89
7.1. Kesimpulan 89
7.2. Saran 90
DAFTAR PUSTAKA.....91
BIODATA PENULIS.....93
Lampiran A.....1
Lampiran B.....1

DAFTAR GAMBAR

Gambar 2.1 Pohon Struktur Frase 8
Gambar 2.2 Struktur Tabel pada HTML 9
Gambar 2.3 Bentuk DOM 10
Gambar 2.4 Contoh Teknik Get dengan HTTP Params 15
Gambar 3.1 Tahapan Pelaksanaan Tugas Akhir 19
Gambar 3.2 Metode Prototyping 21
Gambar 4.1 Use Case Diagram Kebutuhan Pengguna 28
Gambar 4.2 Arsitektur Sistem 29
Gambar 4.3 Contoh SQL Query 30
Gambar 4.4 Contoh SQL Tree 31
Gambar 4.5 Contoh Data JSON yang Dihasilkan Sistem 33
Gambar 4.6 Struktur Table pada HTML 34
Gambar 4.7 Bentuk DOM 35
Gambar 4.8 Hubungan Antar Collection 36
Gambar 4.9 Flowchart Alur Berjalannya Sistem 38
Gambar 4.10 Rancangan Tampilan Aplikasi 41
Gambar 5.1 Proses Mendapatkan Data yang Memiliki Colspan dan Rowspan 55
Gambar 5.2 Contoh Data yang Dihasilkan Method get_where 59
Gambar 5.3 Contoh Hasil Data pada Method get_where 60
Gambar 5.4 Contoh Hasil HTTP Params yang Dihasilkan Method SQL Parser 61
Gambar 5.5 Flowchart Komponen Translator 61
Gambar 5.6 Hasil Translasi Where Condition pada Method get_column_with_where 65

Halaman ini sengaja dikosongkan

DAFTAR KODE

Kode 2.1 Contoh Data Bertipe JSON Object 11
Kode 2.2 Contoh Data Bertipe JSON Array yang Memiliki Dua Object 12
Kode 2.3 Contoh Query Select 13
Kode 2.4 Contoh Query Select dengan Where-Clause 14
Kode 4.1 Contoh HTTP Request 32
Kode 5.1 Class SchemaController 49
Kode 5.2 Method createDom pada ParserController 50
Kode 5.3 Method is_new pada SchemaController 50
Kode 5.4 Method create_dom pada SchemaController 50
Kode 5.5 Method schema_definition pada SchemaController 51
Kode 5.6 Method get_headers_data pada SchemaController. 52
Kode 5.7 Method update_table pada SchemaController 52
Kode 5.8 Method update_column pada SchemaController 53
Kode 5.9 Method get_body_data pada SchemaController 53
Kode 5.10 Method get_data_table pada SchemaController ... 54
Kode 5.11 Helper fix_numeric_data 56
Kode 5.12 Method sql_parser Untuk Merubah Query Menjadi HTTP Params 57
Kode 5.13 Method get_where Untuk Mendapatkan Nilai Where Dari Query Yang Dilakukan 58
Kode 5.14 Contoh Query Where-Clause Dengan Operator And dan Order 58
Kode 5.15 Method get_order Untuk Mendapatkan Nilai Order pada Query 60
Kode 5.16 Method getTable Sebagai Method Utama Komponen Translator 62
Kode 5.17 Method get_clumn Untuk Melakukan Pengambilan Data Kolom 63
Kode 5.18 Method dynamic_select Untuk Melakukan Filtering Kolom 64
Kode 5.19 Method get_column_with_where Untuk Melakukan Pengambilan Data yang Memiliki Where 65

Kode 5.20 Method get_column_collection Untuk Melakukan Pengambilan Data.....66
Kode 5.21 Method dynamic_where, first_where, add_where, dan choose_where 68
Kode 5.22 Method where_between_condition, where_in_condition dan where_condition.....69
Kode 5.23 Method query_and, query_or, query_between dan query_in.....70
Kode 5.24 Method add_order Untuk Mentranslasikan Array Order Menjadi Query Builder.....71

DAFTAR TABEL

Tabel 2.1 Penelitian Sebelumnya 5
Tabel 2.2 Penjelasan Singkat Produk Microsoft Power BI 6
Tabel 2.3 Penjelasan Singkat Produk Microsoft Excel 6
Tabel 2.4 Perbandingan dengan Produk Sejenis 7
Tabel 2.5 Fungsi *F*(.) 17
Tabel 4.1 Penjelasan Collection pada Database 36
Tabel 4.2 Penjelasan Data pada Collection 37
Tabel 4.3 Rancangan Pembuatan Halaman Antar Muka 40
Tabel 4.4 Rancangan Pengujian Fungsional 42
Tabel 4.5 Pengaturan Nilai C Pada Setiap Tipe Kenaikan TTL 45
Tabel 4.6 Skenario Pertama Testing Pengambilan Data 45
Tabel 4.7 Skenario Kedua Testing Pengambilan Data 45
Tabel 5.1 Spesifikasi Perangkat Keras dan Lunak Dalam Melakukan Pengembangan Aplikasi 47
Tabel 5.2 Spesifikasi Perangkat Keras dan Lunak Dalam Melakukan Development Delivery 48
Tabel 6.1 Hasil Pengujian Fungsional Komponen Downloader 73
Tabel 6.2 Hasil Pengujian Fungsional Komponen HTML Parser 75
Tabel 6.3 Hasil Pengujian Fungsional Komponen SQL Parser 77
Tabel 6.4 Hasil Pengujian Fungsional Komponen Translator 78
Tabel 6.5 Hasil Pengujian Tipe Kenaikan TTL Interval Pertama 81
Tabel 6.6 Hasil Pengujian Tipe Kenaikan TTL Interval Kedua 82
Tabel 6.7 Nilai TTL Terakhir Saat Pengujian Interval Pertama 83
Tabel 6.8 Nilai TTL Terakhir Pengujian Interval Kedua 83
Tabel 6.9 Rangkuman Hasil Pengujian Fungsional 84
Tabel 6.10 False Positive Interval Pertama 85
Tabel 6.11 False Positive Interval Kedua 85

Tabel 6.12 Nilai *False Positive Ratio (FPr) Interval Pertama* 85
Tabel 6.13 Nilai *False Positive Ratio (FPr)* Interval Kedua..85
Tabel 6.14 Nilai *Stale Trafic Ratio (STr)* 86
Tabel 6.15 Nilai FPr dan STr Website Nilai Tukar Mata Uang 86
Tabel 6.16 Nilai FPr dan STr Website BMKG 87
Tabel 6.17 Nilai FPr dan STr Website Pengumuman.....88

BAB I
PENDAHULUAN

Pada bab ini akan dijelaskan gambaran umum mengenai tugas akhir. Hal tersebut meliputi latar belakang, rumusan masalah, batasan tugas akhir, tujuan tugas akhir, manfaat tugas akhir dan relevansi tugas akhir dengan keilmuan Sistem Informasi.

1.1. Latar Belakang Masalah

Internet merupakan media penyebaran informasi yang berkembang sangat pesat dengan jumlah penggunanya yang terus meningkat seiring dengan berkembangnya teknologi. Begitu pula yang terjadi di Indonesia, menurut survei yang dilakukan oleh Asosiasi Penyelenggara Jaringan Internet Indonesia (APJII) yang dilakukan sepanjang tahun 2016 menemukan bahwa 132,7 juta penduduk Indonesia telah terhubung ke internet [1]. Jumlah tersebut mengalami peningkatan sebesar 51,8 persen yang telah dibandingkan dengan hasil survey APJII di tahun 2014. Salah satu media penyebaran informasi paling besar di internet adalah website. Menurut survey yang telah dilakukan netcraft pada Oktober 2017 yang menunjukkan jumlah hostname yang ada pada seluruh dunia sebesar 1,815 miliar dengan website aktif sebesar 172 juta [2]. Tingginya jumlah website aktif yang diimbangi dengan meningkatnya jumlah pengguna internet menunjukkan pertukaran informasi melalui website begitu besar.

Meluasnya penggunaan website telah menimbulkan beberapa permasalahan pengelolaan data baru, seperti mengesktrak data dari halaman web membuat database dapat diakses dari browser, dan telah memperbarui minat pada masalah yang telah muncul dalam konteks lain sebelumnya, seperti query grafik, data semi terstruktur dan dokumen terstruktur [3]. Kebanyakan masalah ini pada dasarnya adalah restrukturasi data, seperti melakukan perubahan struktur data tertentu untuk membuat representasi lain. Pada tugas akhir ini akan dikembangkan aplikasi yang akan merestrukturasi data

pada tabel yang terdapat didalam web dengan data yang diambil merupakan data yang sifatnya bukan kredensial.

Aplikasi tersebut akan menyimpan data yang sudah dilakukan restrukturasi dan dapat diakses pada aplikasi lain. Kemudian untuk melakukan pengaksesan data menggunakan SQL agar bisa dilakukan proses query projection, selection dan order pada data yang ada. Pengaksesan data ini menggunakan mekanisme *Adaptive Time to Live*.

1.2. Perumusan Masalah

Rumusan masalah yang akan diselesaikan dalam tugas akhir ini adalah:

1. Bagaimana melakukan ekstraksi data pada website?
2. Bagaimana melakukan query data terhadap data yang tidak terstruktur pada website?
3. Bagaimana mendeteksi tipe data numerik yang tepat untuk data yang tidak terstruktur pada website?
4. Bagaimana memperbarui data yang tersimpan agar sesuai dengan sumber data?
5. Apa tipe kenaikan TTL terbaik untuk setiap studi kasus?

1.3. Batasan Masalah

Batasan masalah pada tugas akhir ini adalah:

1. Website yang dilakukan untuk ekstraksi data hanya sebatas satu halaman.
2. Tabel yang dilakukan ekstraksi data tidak memiliki atribut rowspan.
3. Data numerik yang dihasilkan pada saat ekstraksi data hanya bertipe integer dan float.
4. Query yang digunakan dalam penelitian ini hanya sebatas query sederhana (tanpa join dan *nested* query).
5. Kondisi pada query yang bisa digunakan hanya sebatas “where” dan “order”.

6. Tabel yang bisa dilakukan ekstraksi adalah tabel yang seluruh datanya dimuat pada satu halaman tanpa javascript.
7. Tabel yang bisa dilakukan ekstrasi adalah tabel yang tidak memiliki tag tabel didalam tag tabel (tidak *nested* tag tabel).
8. Tabel yang bisa dilakukan ekstrasi adalah tabel yang menggunakan *tag table* html (<table></table).
9. Website yang dilakukan untuk studi kasus adalah www.bmkg.go.id/cuaca/prakiraan-cuaca-indonesia.bmkg, www.x-rates.com, dan www.pengumuman.mlcunair.com/hasil?jenjang=sma

1.4. Tujuan Penelitian

Tujuan dari pengerjaan tugas akhir ini adalah:

1. Mengetahui cara melakukan pengambilan data yang tidak terstruktur pada halaman website.
2. Membuat aplikasi untuk melakukan query pada data yang tidak terstruktur pada website.
3. Mengidentifikasi tipe data numerik yang tepat pada data yang tidak terstruktur.
4. Mengetahui cara penyesuaian data yang tersimpan dengan sumber data.
5. Mengetahui tipe kenaikan TTL terbaik untuk setiap studi kasus yang ada.

1.5. Manfaat Penelitian

Manfaat yang dihasilkan dari pengerjaan tugas akhir ini adalah:

1. Peneliti dapat memberikan pengetahuan mengenai cara melakukan pengambilan data yang tidak terstruktur pada website.
2. Peneliti dapat memberikan pengetahuan cara melakukan query terhadap data yang tidak terstruktur pada website.
3. Peneliti dapat memberikan cara penyimpanan data secara sementara pada database.

4. Pengambilan data dari halaman website dapat digunakan untuk penelitian-penelitian selanjutnya.

1.6. Relevansi

Tugas Akhir ini memiliki relevansi terhadap mata kuliah yang terdapat pada laboraturium ADDI, karena melakukan implementasi mata kuliah pada bidang keilmuan seperti:

a. Pengantar Basis Data
b. Manajemen Administasi Basis Data
c. Pemrograman Berbasis Web

Tugas akhir ini juga bersifat menyelesaikan masalah dalam Data Curation yang melakukan pengambilan data dari berbagai macam sumber.

BAB II
TINJAUAN PUSTAKA

Bab ini akan menjelaskan penelitian sebelumnya dan dasar teori yang menjadi landasan teori dalam pengerjaan tugas akhir. Landasan teori akan menjelaskan gambaran umum apa saja yang digunakan dalam pembuatan tugas akhir ini.

2.1. Penelitian Sebelumnya

Berikut ini penelitian-penelitian sebelumnya yang dijadikan acuan dalam pengerjaan tugas akhir yang disajikan dalam tabel di bawah ini.

Tabel 2.1 Penelitian Sebelumnya

Judul	WebOQL: Exploiting Document Structure in Web Queries (1997)
Identitas Peneliti	Gustavo O. Arocena
Kesimpulan	Pada penelitian ini peneliti menggunakan metode *Hypertrees* sebagai *parser* untuk membuat hubungan heirarkis di antara komponen potongan teks terstruktur dalam bentuk yang independen dari tata bahasa yang digunakan untuk mengurai teks. Tetapi ketika membuat *Hypertrees* dari website yang memiliki struktur informasi yang tidak sepenuhnya tercermin dalam markup metode ini akan kesulitan membuat hubungan antar komponen. Penelitian ini juga berfokus pada bagaimana mendapatkan komponen yang ada sesuai dengan query yang dilakukan pengguna. Serta tidak terdapat penyimpanan data yang dilakukan.
Sumber	[3]

2.2. Produk Sejenis

Berikut ini merupakan penelitian produk sejenis yang dapat melakukan pengambilan elemen tabel pada HTML.

Tabel 2.2 Penjelasan Singkat Produk Microsoft Power BI

Produk	Microsoft Power BI
Pemilik	Microsoft
Kesimpulan	Pada produk Microsoft power BI terdapat sumber data dari website. Sumber data tersebut mengambil elemen tabel pada HTML dan menambahkannya menjadi sumber data power BI. Tetapi power BI harus menggunakan DAX sebagai query datanya. Fokus dari produk ini adalah pengolahan data dan memvisualisasikan data yang telah diperoleh dan diolah.
Sumber	[4]

Tabel 2.3 Penjelasan Singkat Produk Microsoft Excel

Produk	Microsoft Excel
Pemilik	Microsoft
Kesimpulan	Pada produk Microsoft excel terdapat pengambilan data dari web. Sumber data tersebut mengambil elemen tabel pada HTML dan nampilkannya pada sheet Microsoft Excel. Microsoft Excel menggunakan Power query dan visual dalam mengolah data yang ada. Fokus dari aplikasi ini adalah mengolah data
Sumber	[5]

Dari produk sejenis diatas yang telah dijelaskan pada Tabel 2.1, Tabel 2.2 dan Tabel 2.3 dapat dilakukan perbandingan dari beberapa aspek, yaitu:

Tabel 2.4 Perbandingan dengan Produk Sejenis

Project	Mekanisme pembaharuan data terhadap sumber data	Objek	Struktur Pengambilan data	Mampu mendeteksi tipe data numeric	Antarmuka	Mengirimkan Data Bertipe json
WebOQL	Tidak ada informasi	HTML	Hypertrees	Tidak ada informasi	SQL	Tidak ada informasi
PowerBI	Berbasis interval dan manual	Table HTML	Tidak ada informasi	Tidak	UI, DAX	Tidak
Excel	Berbasis interval dan manual	Table HTML	Tidak ada Informasi	Tidak	UI, Power Query (DAX)	Tidak
Projek sekarang	Adaptive Time to Live	Table HTML	DOM	Ya	SQL	Ya

Untuk produk WebOQL tidak dijelaskan bagaimana waktu pembaharuan data pada aplikasi. Untuk Microsoft power BI dan Microsoft excel terdapat beberapa pilihan untuk memperbaharui data yang mereka ambil, yaitu automatic refresh setiap beberapa menit dan atau setiap file tersebut dibuka. Kedua produk tersebut dalam melakukan pengolahan data dapat menggunakan UI dan DAX (untuk power BI), Power Query (untuk excel).

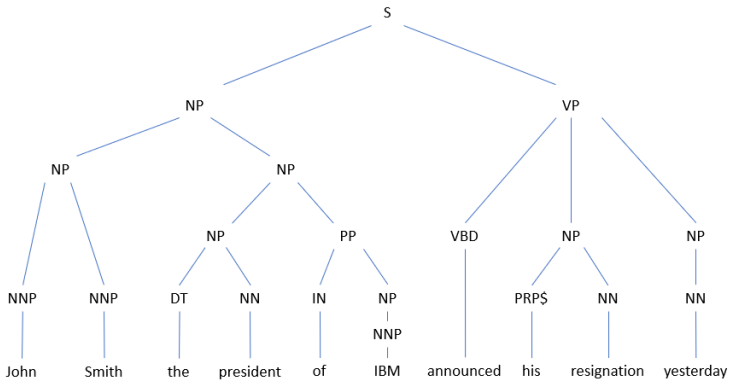
Untuk produk Power BI dan Microsoft Excel juga belum mampu dalam membedakan data bertipe numeric dan

produk sekarang mampu mendeteksi data tersebut. Produk sekarang dalam melakukan pengambilan data menggunakan mekanisme *Adaptive Time to Live* yang akan dijelaskan pada bagian 2.3.7. Serta menggunakan SQL dalam melakukan query untuk mengolahan data. Serta dapat memberikan data bertipe json pada hasil query.

2.3. Landasan Teori

2.3.1. *Parser*

Parser bertujuan untuk mengambil kalimat yang ditandai sebagai masukan (input) dan menghasilkan pohon struktur frase sebagai output [6]. Sebagai contoh: input kalimat adalah "John Smith the president of IBM, announced his resignation yesterday". Kemudian kalimat tersebut akan ditandai menjadi "John/NNP Smith/NNP, the/DT president/NN of/IN IBM/NNP, announced/VBD his/PRP$ resignation/NN yesterday/NN". Setelah itu parser akan merubah kalimat yang telah ditandai tersebut menjadi pohon struktur frase sebagai berikut:



Gambar 2.1 Pohon Struktur Frase

Pada tugas akhir ini parser berguna untuk melakukan pengecekan terhadap query yang dilakukan pengguna apakah

query tersebut sudah benar atau belum. Serta parser digunakan dalam melakukan pengambilan data melalui DOM yang dihasilkan dari *source code* halaman website yang telah diunduh.

2.3.2. *Dokumen Object Model* (DOM)

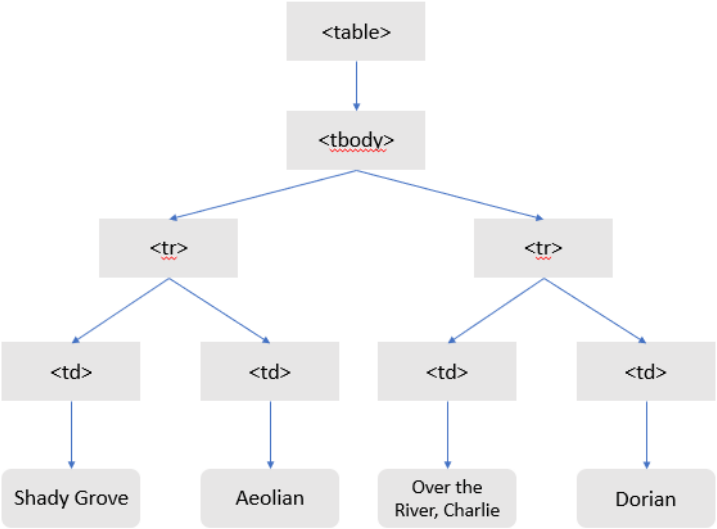
Dokumen Object Model (DOM) adalah platform yang memungkinkan program dan skrip untuk mengakses dan memperbarui konten, struktur, dan gaya dokumen secara dinamis. Dokumen tersebut dapat diproses lebih lanjut serta hasil pemrosesan dapat ditampilkan kembali pada halaman web yang sedang ditampilkan [7]. Sebagai contoh pada pendefinisian struktur tabel pada HTML ditunjukkan pada Gambar 2.2.

```
<TABLE>
<TBODY>
<TR>
<TD>Shady Grove</TD>
<TD>Aeolian</TD>
</TR>
<TR>
<TD>Over the River, Charlie</TD>
<TD>Dorian</TD>
</TR>
</TBODY>
</TABLE>
```

Gambar 2.2 Struktur Tabel pada HTML

Pada Gambar 2.2 pendefinisian tabel dimulai dengan membuat tag <table> pada awalan dan tag </table> sebagai penutup. Yang berarti semua tag didalamnya merupakan bagian dari tag tabel (child). Begitu juga dengan tag yang lain. Sehingga DOM yang dihasilkan oleh Gambar 2.2 akan menjadi seperti Gambar 2.3. DOM mirip sekali dengan struktur *tree* dimana DOM seperti “hutan” yang memiliki banyak *tree*. Dimana struktur *tree* membentuk titik-titik (*nodes*) yang

merepresentasikan element yang ada pada *document* [7]. Untuk mengakses data dapat dilakukan dengan cara “tree-walking”. Dimana cara tersebut mengharuskan kita mengakses *parent nodes* terlebih dahulu untuk mengakses *child nodes* yang ada. Dengan cara tersebut sistem dapat melakukan pengambilan data yang spesifik sesuai dengan *nodes* yang akan diambil datanya. Karena sistem hanya perlu mengakses *nodes* teratas dari setiap *node* table. Dari *node* tersebut sistem bisa mendapatkan data yang dibutuhkan secara spesifik.



Gambar 2.3 Bentuk DOM

2.3.3. Metadata

Metada memiliki tujuan utama untuk menggambarkan, mendefinisikan dan atau membubuhi keterangan data lain yang menyertainya [8]. Data yang menyertainya dapat berada di unit fisik yang sama dengan metadata seperti file di disk yang sama, atau di unit logis yang sama seperti pada database yang sama. Informasi yang disimpan dalam metadata mungkin dalam bentuk yang dapat dipahai oleh manusia non-teknis, atau dalam bentuk yang dapat digunakan oleh perangkat lunak [8].

Pada tugas akhir ini, metadata dibuat untuk dapat merepresentasikan data yang telah diambil pada website sumber dan mampu ditampilkan didalam aplikasi. Sehingga pengguna aplikasi dapat melakukan proses *projection*, *selection* dan *order* pada metadata tersebut.

2.3.4. JavaScript Object Notation (JSON)

JavaScript Object Notation (JSON) adalah format pertukaran data yang ringan, berbasis teks, bahasa-independen. JSON berasal dari Standar Bahasa Pemrograman ECMAScript. JSON mendefinisikan sekumpulan kecil aturan pemformatan untuk representasi portabel dari data terstruktur [9].

JSON dapat memiliki empat tipe data primitif (string, angka, boolean, dan null) dan dua tipe data struktural (objek dan array). Untuk JSON yang memiliki structural objek memiliki pasangan ‘key’ dan ‘value’. Dimana nilai ‘key’ dan ‘value’ dipisahkan dengan titik dua (:). Untuk lebih jelasnya bisa dilihat contoh data berformat JSON pada Kode 2.1 dan Kode 2.2.

```
1.  {
2.      "Image": {
3.          "Width":  800,
4.          "Height": 600
5.      },
6.      "name": "Example"
7.  }
```

Kode 2.1 Contoh Data Bertipe JSON Object

```
1.  [
2.        {
3.           "precision": "zip",
4.           "Latitude":  37.7668,
5.           "Longitude": -122.3959,
6.           "Address":   "",
7.           "City":      "SAN FRANCISCO",
8.           "State":     "CA",
9.           "Zip":       "94107",
10.          "Country":   "US"
11.       },
12.       {
13.          "precision": "zip",
14.          "Latitude":  37.371991,
15.          "Longitude": -122.026020,
16.          "Address":   "",
17.          "City":      "SUNNYVALE",
18.          "State":     "CA",
19.          "Zip":       "94085",
20.          "Country":   "US"
21.       }
22. ]
```

Kode 2.2 Contoh Data Bertipe JSON Array yang Memiliki Dua Object

Pada Kode 2.1 data bertipe object yang memiliki ‘key’ image dan name. Dimana untuk image merupakan sebuah object yang didalamnya memiliki nilai key width dan height dengan nilai masing-masing 800 dan 600. Sedangkan pada Kode 2.2 JSON merupakan sebuah array yang memiliki dua object didalamnya. Untuk masing-masing object memiliki nilai yang sama yaitu precision, latitude, longitude, address, city, state, zip dan country. Tetapi memiliki nilai dari masing-masing key berbeda-beda.

Pada tugas akhir ini JSON digunakan sebagai output akhir untuk pengguna yang ingin mengolah data hasil query tersebut. JSON juga digunakan dalam berkomunikasi dari backend dan frontend.

2.3.5. Structured Query Language (SQL)

Structured Query Language adalah bahasa database komputer yang dirancang untuk mengelola data dalam sistem manajemen basis data relasional [10]. SQL digunakan untuk query, mengubah dan mendefinisikan database relasional menggunakan pernyataan deklaratif. SQL dapat melakukan perintah untuk:

- Mendapatkan data dari database
- Memasukkan record ke database
- Memperbarui record pada database
- Menghapus record pada database
- Membuat database baru pada database
- Membuat table baru pada database
- Membuat stored procedures pada database
- Membuat views pada database
- Dapat mengatur izin pada table, procedure, dan view

SQL menggunakan template yang konsisten untuk ekspresi pernyataan sederhana [11]. Sebagai contoh dalam mendapatkan data pengguna harus menspesifikasikan kolom mana yang diinginkan untuk diambil, dari tabel mana dan kondisi dimana baris harus dikembalikan.

```
1.  SELECT DEPT FROM EMP
```

Kode 2.3 Contoh Query Select

Pada Kode 2.3 query tersebut mendapatkan seluruh daftar dari kolom DEPT yang ada pada tabel EMP. Karena pada query tersebut tidak menyebutkan kondisi dimana barus tersebut harus dikembalikan. Sebaliknya query pada Kode 2.4 memiliki where-clause akan mengembalikan data pada kolom NAME, SAL dari tabel EMP dimana memiliki nilai DEPT adalah ‘TOY’ dan MGR adalah ‘ANDERSON’.

```
1. SELECT NAME, SAL FROM EMP WHERE DEPT = 'TOY' AND MGR =
   'ANDERSON '
```

Kode 2.4 Contoh Query Select dengan Where-Clause

Pada tugas akhir ini, SQL digunakan pengguna untuk melakukan proses pengambilan data pada tabel yang disediakan oleh sistem. Sehingga query yang dilakukan dibatasi hanya query SELECT untuk melakukan pengambilan data.

2.3.6. Protokol REST

Representational State Transfer (REST) memiliki konsep yang berfokus pada penggunaan HTTP untuk mengambil representasi resource (sumber daya) pada berbagai state (keadaan) [12] Pada REST menggunakan URI (*Uniform Resource Identifier*) sebagai teknik identifikasi sumber daya. Dengan teknik tersebut menyajikan teknik komunikasi yang seragam yang terdiri dari GET, PUT, POST dan DELETE untuk menghubungkan operasi semantik CRUD (Creating, Reading, Updating dan Deteling) [13].

Pada tugas akhir ini REST Protokol digunakan untuk menghubungkan http request yang dibentuk berdasarkan query yang dilakukan pengguna dengan sumber daya pada database. Terdapat tiga sumber daya yang digunakan pada tugas akhir ini, yaitu url, table dan column. Ketiga sumber daya ini berupa document yang ada pada collection database MongoDB.

Dalam melakukan operasi semantik CRUD pada sumber daya diperlukan URI setidaknya berupa “URL/nama_sumberdaya/id_sumberdaya”. Sebagai contoh dalam mengakses sumberdaya website yang memiliki id 1 dapat berupa “http://localhost/website/1”. Kemudian untuk teknik komunikasi yang digunakan sebagai berikut:

- GET: mengambil sumber daya
- POST: membuat sumber daya
- PUT: meperbaharui sumber daya
- DELETE: menghapus sumber daya

Dalam tugas akhir ini, lebih banyak menggunakan teknik GET dalam mengambil sumber daya yang ada. Pada method get juga dapat kita tambahkan http params sebagai indikator tambahan dalam melakukan pengambilan data. Sebagai contoh sederhana dapat dilihat pada Gambar 2.4.

Gambar 2.4 Contoh Teknik Get dengan HTTP Params

Http params yang ditambahkan adalah ‘ttl=1’ dimana kita ingin memberikan tambahan indikator variable ‘ttl’ dengan nilai 1. Sehingga sistem dapat mengolah variable tersebut.

2.3.7. Mekanisme *Adaptive Time to Live*

Pada mekanisme *Time to Live* mengasumsikan bahwa pada sistem *caching* hasil query (q) di-cache yang dilihat pada waktu t(q) dan waktu kadaluwarsa diatur menjadi t(q) + T, dimana T merupakan nilai *Time to Live* (TTL) yang sama dengan semua query [14]. Yang berarti setiap query yang diberikan kembali sebelum mencapai waktu kadaluwarsa, maka hasil dari query tersebut akan dilayani oleh *cache*. Jika hasil dari query yang ada pada *cache* telah kadaluwarsa, maka query akan dijalankan ulang oleh *backend* [14]. Pada mekanisme TTL menyebabkan query *traffic ratio* yang kadaluwarsa dan redundant query *evaluation cost* berlebihan [14]. Secara umum jika nilai TTL besar menyebabkan hasil query akan sangat kadaluwarsa dan bila nilai TTL kecil menyebabkan *overhead* pemrosesan yang lebih tinggi. Oleh karena itu dibutuhkan nilai TTL yang dapat merepresentasikan kapan waktu yang tepat untuk menjalankan query ulang oleh *backend.*

Pada *Adaptive Time to Live* bertujuan untuk memilih nilai TTL secara adaptif berdasarkan query basis guna

meningkatkan keterkinian data yang disuguhkan dan mengurangi pemrosesan query yang *overhead*. Terdapat tiga alternatif strategi yang dapat diterapkan, yaitu *average* TTL yang mengambil nilai TTL dari rata-rata keseluruhan TTL pada *cache*, *incremental* TTL yang mengambil nilai TTL berdasarkan nilai TTL saat ini, dan *machine-learned* TTL yang mengambil nilai TTL berdasarkan hasil pembelajaran model yang diekstraksi dari data training query log.

Pada tugas akhir ini akan menerapkan strategi *incremental* TTL karena cara tersebut merupakan cara sederhana namun efektif dalam menetapkan nilai TTL dengan cara menyesuaikan nilai TTL berdasarkan nilai TTL saat ini [14]. Pada strategi *incemental* TTL setiap hasil query yang memiliki nilai TTL kadaluwarsa diminta, hasil query baru akan dibandingkan dengan hasil yang ada di cache untuk menentukan apakah hasil query tersebut kadaluwarsa atau masih baru. Nilai TTL akan ditentukan dengan cara [14]:

1. Jika hasil pada cache sudah kadaluwarsa, maka akan ditentukan nilai TTL baru dengan nilai TTL paling rendah $T_{\min}$. Alasan penuruan nilai TTL adalah dengan menghindari melewatkan periode update data yang meledak.
2. Jika hasil pada *cache* masih baru (hasil perbandingan query cache dan query baru sama), maka akan ditambahkan nilai TTL dari query sesuai dengan output dari fungsi kenaikan TTL.

Fungsi kenaikan TTL ditentukan dengan menggunakan persamaan (2.1).

$$T = T' + F(T') \tag{2.1}$$

Keterangan :
$T =$ nilai TTL baru
$T' =$ nilai TTL sekarang
$F(T') =$ fungsi eksponensial

Dengan nilai tambah $F(T')$ dihitung sesuai dengan fungsi F(.) pada Tabel 2.5.

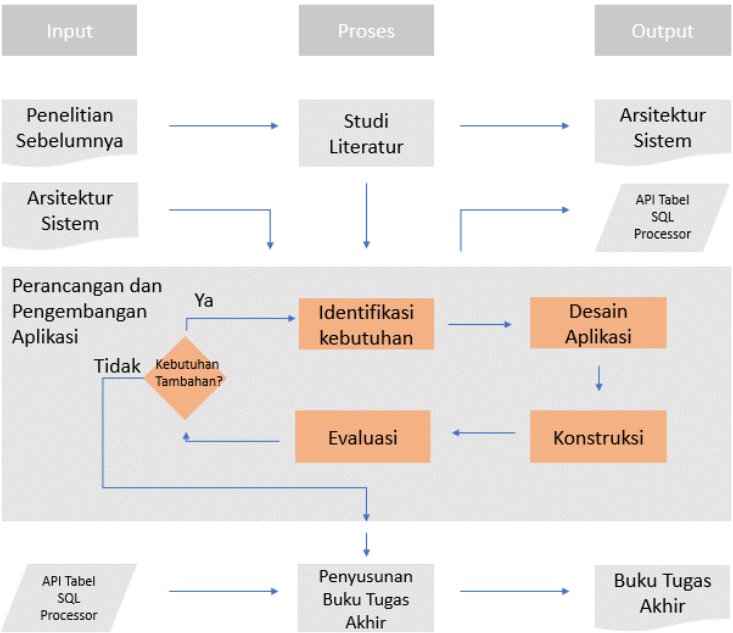
Tabel 2.5 Fungsi *F*(.)

Type	F(.)	Nilai Parameter (c)
Linear	$c \times T'$	1/10, 1/5, ½
Polynomial	$(T')^c$	-3, -2, -1, -½, ½
Exponential	$c^{T'}$	½, 1, 2

Halaman ini sengaja dikosongkan

BAB III
METODOLOGI PENELITIAN

Bab ini akan dijelaskan mengenai tahapan-tahapan apa saja yang dilakukan dalam menyelesaikan penelitian tugas akhir.



Gambar 3.1 Tahapan Pelaksanaan Tugas Akhir

Berikut merupakan penjelasan dari setiap tahapan yang ada pada metodologi yang digunakan, yaitu:

3.1. Studi Literatur

Tahapan awal pengerjaan Tugas Akhir ini adalah studi literatur. Studi literatur digunakan untuk menentukan topik dengan cara menganalisa latar belakang permasalahan informasi publik di media sosial kota surabaya dan menentukan rumusan permasalahan yang diangkat. Pada tahapan ini pun juga ditentukan batasan masalah, tujuan, serta manfaat penentuan topik penelitian.

Untuk mendukung latar belakang permasalahan dan rumusan masalah yang sesuai dengan topik, dilakukan peninjauan pustaka mulai dari studi penelitian sebelumnya terkait dengan penggunaan *Named Entity Recognition* dan juga dasar teori mengenai *Natural Language Processing*, *Named Entity Recognition*, algoritma pendekatan seperti *Conditonal Random Field* (CRF), *Performance Measure* dan *Confusion Matrix*, serta media sosial sebagai referensi pustaka. Adapun literatur yang digunakan berasal dari jurnal ilmiah nasional maupun internasional, makalah penelitian, *e-book*, maupun buku-buku yang dapat dijadikan bahan acuan pengerjaan tugas akhir ini.

3.2. Perancangan dan Pengembangan Aplikasi

Pada tahap ini dilakukan perancangan dan pengembangan aplikasi dimulai dari analisis kebutuhan, mendesain aplikasi, implementasi, dan pengujian aplikasi. Tahapan perancangan dan pengembangan aplikasi menggunakan metode *prototyping* dimana proses pengerjaan aplikasi dari satu fase ke fase lainnya secara cepat. Sehingga aplikasi secara cepat merepresentasikan aplikasi yang diinginkan dan mendapatkan *feedback* dari pengguna. Kemudian *feedback* tersebut akan digunakan untuk memperbaiki kebutuhan/perencanaan selanjutnya [15].

Tahapan kontruksi aplikasi dilakukan berdasarkan proses yang ada guna mendukung berjalannya aplikasi. Dimana urutan proses pengembangan dari aplikasi ini sebagai berikut:

1. Pembuatan HTML Ekstraktor
2. Pembuatan SQL Engine

Gambar 3.2 Metode Prototyping

3.2.1. Perencanaan

Setelah mengetahui konsep dari aplikasi yang ingin dibuat, tahapan selanjutnya adalah perencanaan terkait dengan perencanaan kebutuhan yang diperlukan sehingga aplikasi ini dapat berjalan sesuai dengan alur yang telah ditentukan.

Kebutuhan pada aplikasi ini dibedakan berdasarkan proses yang ada, yaitu:

1. Proses HTML Ekstraktor

   Pada proses ini terjadi pengolahan file HTML untuk dilakukan pembentukan HTML Tree. Sehingga dari HTML Tree tadi dapat diambil data tag table. Pada proses ini terdiri dari tiga komponen yang terlibat, yaitu:

   a. HTML Parser
   b. Downloader
   c. Metadata

Sehingga *output* dari proses ini adalah *metadata* yang dimasukkan (*input*) ke dalam *storage* dan selanjutnya akan diproses oleh SQL Engine

2. SQL Engine

   Pada proses ini akan berjalan proses pengambilan data dari SQL yang telah dituliskan pengguna untuk mengolah data pada storage untuk di *projection, selection* dan *order*. Sehingga query yang diinputkan oleh pengguna menghasilkan output data yang tepat. Pada proses ini terdiri dari dua komponen yang dibutuhkan, yaitu:

   a. SQL Parser sebagai pengecekan query oleh pengguna
   b. Translator sebagai penerjemah query pengguna dan melakukan pengambilan data sesuai dengan query pengguna.

   Sehingga *output* dari proses ini adalah data hasil query yang tepat.

3.2.2. Desain

Berdasarkan hasil identifikasi kebutuhan pada tahap sebelumnya akan digunakan untuk membuat gambaran desain dari fitur, fungsionalitas dan alur dari aplikasi. Pada tahap ini akan dibuat desain berupa:

1. Desain tampilan antar muka

   Pada tahap ini akan dilakukan desain tampilan antar muka yang nantinya digunakan untuk tampilan aplikasi. Aplikasi akan menampilkan data-data tabel yang berhasil dilakukan ekstraksi. Selain itu, aplikasi juga akan menampilkan data yang telah berhasil dilakukan proses *projection, selection* dan *order*.

2. Desain alur sistem

   Pada tahap ini akan dilakukan desain alur berjalannya sistem akan berjalan. Dimulai dari pengguna mengakses

aplikasi atau sistem sampai dengan pengguna menerima data hasil pengolahan sistem.

3. Desain Basis Data

   Pada tahap ini akan dilakukan desain bagaimana struktur data akan disimpan. Sehingga dapat memudahkan sistem dalam melakukan pengaksesan dan pengolahan data.

3.2.3. Konstruksi

Pada tahap ini dilakukan konstruksi sistem berdasarkan desain dan perencanaan yang telah dibuat sebelumnya. Output dari tahap ini merupakan aplikasi Table Query Processor yang siap pakai.

3.2.4. Development Delivery dan Feedback

Pada tahap ini akan dilakukan development delivery dan evaluasi oleh pengguna untuk mendapatkan feedback yang digunakan untuk pengembangan lebih lanjut.

3.3. Pengujian

Pada tahap ini akan dilakukan dua pengujian, yang terdiri dari:

1. Pengujian fungsional

   Pengujian fungsional dilakukan untuk memastikan kualitas program yang telah dibuat layak untuk dipakai atau tidak terjadi *error*.

2. Pengujian tipe kenaikan TTL

   Pengujian ini dilakukan untuk mengetahui tipe kenaikan TTL mana yang paling tepat untuk website dengan interval pembaharuan data yang dikategorikan cepat, sedang dan tidak terdapat perubahan.

3.4. Penyusunan Buku Tugas Akhir

Pada tahap ini dilakukan penyusunan laporan tugas akhir sebagai bentuk dokumentasi pengerjaan tugas akhir ini. Dalam laporan tersebut meliputi:

1. Bab I Pendahuluan

   Pada bab ini akan dijabarkan identifikasi masalah penelitian meliputi latar belakang masalah, rumusan masalah, atasan masalah, tujuan, dan relevansi keilmuan.

2. Bab II Tinjauan Pustaka

   Pada bab ini akan dijabarkan dasar teori, temuan dan penelitian sebelumnya yang digunakan sebagai acuan dalam penyusunan tugas akhir.

3. Bab III Metodologi Penelitian

   Pada bab ini akan dijabarkan tahapan-tahapan apa saja yang dilakukan beserta jadwal dalam proses pengerjaan tugas akhir.

4. Bab IV Perancangan

   Pada bab ini akan dijelaskan proses perancangan pengembangan *Table Query Processor* meliputi perancangan sistem, perancangan data, perancangan proses, dan perancangan antar muka.

5. Bab V Implementasi

   Pada bab ini akan dijelaskan proses konstruksi Table Query Processor yang dikembangkan menggunakan metode *prototyping*.

6. Bab VI Analisis dan Pembahasan

   Pada bab ini akan dijelaskan hasil analisis dan pembahasan dari hasil yang didapatkan pada tugas akhir.

7. Bab VII Kesimpulan dan Saran

   Pada bab ini berisi tentang kesimpulan dan saran serta hal-hal yang dapat dikembangkan lebih lanjut.

Halaman ini sengaja dikosongkan

BAB IV
PERANCANGAN

Pada bab ini dijelaskan proses pembuatan rancangan penelitian tugas akhir yang meliputi perancangan sistem, data, proses dan tampilan antar muka.

4.1. Perancangan Sistem

4.1.1. Identifikasi Pengguna

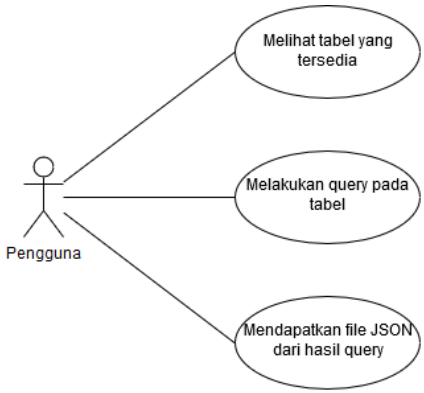
Aplikasi pada tugas akhir ini ditujukan untuk pengguna yang memenuhi kriteria sebagai berikut:

1. Mengetahui SQL query.
2. Membutuhkan pengambilan data pada tag tabel website untuk diolah oleh aplikasi lain.
3. Membutuhkan pengambilan data pada tag tabel website untuk dianalisis dalam bentuk csv atau excel.

Untuk kriteria pertama bersifat wajib, karena query yang dilakukan pengguna pada aplikasi ini adalah SQL query. Untuk kriteria kedua dan ketiga bersifat opsional atau dapat memenuhi salah satu ataupun kedua kriteria tersebut.

4.1.2. Identifikasi Kebutuhan Pengguna

Proses mengidentifikasi kebutuhan pengguna sangat penting untuk mengetahui fungsional apa saja yang akan diterapkan pada sistem. Berikut ini merupakan hasil identifikasi kebutuhan pengguna:

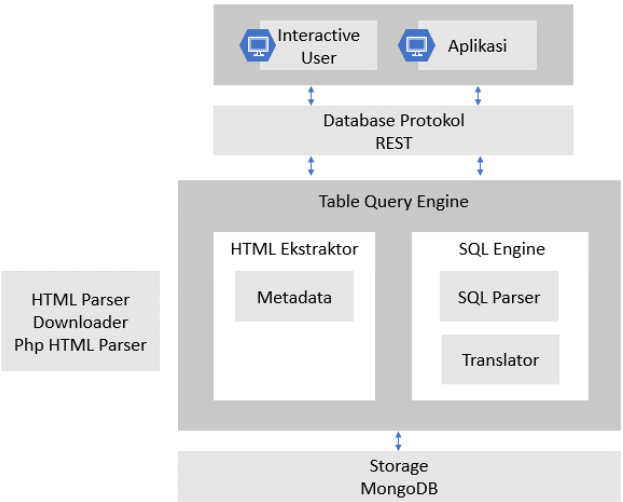


Gambar 4.1 Use Case Diagram Kebutuhan Pengguna

Dari use case diagram pada Gambar 4.1 pengguna dapat melihat tabel apa saja yang tersedia pada website. Sebagai contoh ketika sistem mendapatkan 3 tag HTML table pada sebuah website, maka sistem akan menampilkan ke 3 table tersebut beserta isinya. Pengguna juga dapat melakukan query pada tabel tersebut dan melihat hasil dari query tersebut. Pengguna juga dapat mengolah hasil query tersebut yang bertipe JSON dengan aplikasi lain.

4.1.3. Arsitektur Sistem

Pada sub bab ini akan dijelaskan mengenai arsitektur sistem yang akan dibuat. Rancangan arsitektur sistem adalah sebagai berikut:



Gambar 4.2 Arsitektur Sistem

Terdapat dua proses utama dalam Table Query Engine yaitu:

1. HTML Ekstraktor yang bertujuan untuk mengekstraksi data pada tabel yang terdapat dalam HTML dan melakukan *insert* data tersebut pada tempat penyimpanan data.
2. SQL Engine yang bertujuan untuk menerima query dari pengguna dan melakukan proses pengambilan data pada penyimpanan data yang telah dihasilkan oleh HTML Ekstraktor.

Berikut ini merupakan penjelasan setiap komponen yang terdapat didalam kedua proses Table Query Engine.

4.1.3.1. Metadata

Metadata merupakan data-data yang telah didapatkan dari hasil proses pengektrasian halaman website. Metadata ini digunakan sebagai input pada proses SQL Engine. Detail metadata yang didapatkan akan dijelaskan pada bagian 4.2

4.1.3.2. Downloader

Downloader merupakan komponen yang berfungsi untuk melakukan pengambilan keseluruhan halaman website dan membacanya menjadi sebuah data bertipe string dan membuat enkripsi md5 pada data tersebut. Kemudian data tersebut akan menjadi *input* pada komponen HTML Parser. Dan md5 akan digunakan untuk pembanding apakah halaman website memiliki perubahan atau tidak.

4.1.3.3. SQL Parser

SQL parser berfungsi membuat SQL query yang dilakukan pengguna menjadi sebuah SQL tree. Kemudian dari SQL tree tersebut akan dilakukan pengecekan apakah query tersebut benar.

```
Select * from table_a where column_b = 'Ab'
```

Gambar 4.3 Contoh SQL Query

Sebagai contoh pada Gambar 4.3 akan dirubah menjadi parser tree menjadi seperti Gambar 4.4.

Parser tree berfungsi untuk memahami tabel, fungsi dan operator mana yang direferensikan oleh query yang diberikan pengguna [16]. Sebagai contoh jika terdapat *node* yang sifatnya wajib diisi seperti *node* table pada *node* from berisi *null*. Maka akan sistem akan melakukan pemberitahuan bahwa query yang dilakukan pengguna salah. Untuk *nodes* dan atau kondisi yang bernilai wajib adalah sebagai berikut:

a. Query yang dilakukan harus query select
b. Nilai node table pada node from wajib terisi dan nama table tersebut ada.
c. Nilai nodes partition, group, having, limit, procedure, into, join, union, option, first, last akan dihiraukan.

```
SelectStatement {#192
  +expr: array:1 [
    0 => Expression {#194
      +database: null
      +table: null
      +column: null
      +expr: "*"
      +alias: null
      +function: null
      +subquery: null
    }
  ]
  +from: array:1 [
    0 => Expression {#195
      +database: null
      +table: "table_a"
      +column: null
      +expr: "table_a"
      +alias: null
      +function: null
      +subquery: null
    }
  ]
  +partition: null
  +where: array:1 [
    0 => Condition {#196
      +identifiers: array:2 [
        0 => "column_b"
        1 => "Ab"
      ]
      +isOperator: false
      +expr: "column_b = 'Ab'"
    }
  ]
  +group: null
  +having: null
  +order: null
  +limit: null
  +procedure: null
  +into: null
  +join: null
  +union: []
  +end_options: null
  +options: OptionsArray {#193
    +options: []
  }
  +first: 0
  +last: 14
}
```

Gambar 4.4 Contoh SQL Tree

Setelah query yang dilakukan pengguna benar, maka SQL parser akan merubah SQL *tree* menjadi parameter pada http request seperti berikut ini:

```
select%5B0%5D=%2A&from=table_a&where%5Barguments%5D%5B0%5D=co
lumn_b+%3D+%27Ab%27&where%5Bidentifier%5D%5B0%5D=column_b
```

Kode 4.1 Contoh HTTP Request

Dari parameter query tersebut akan diolah oleh komponen translator.

4.1.3.4. Translator

Translator berguna untuk mentranslasikan parameter http request yang diberikan oleh SQL parser menjadi query yang dapat diterima oleh sistem untuk melakukan query pada storage. Kemudian translator akan mengambil data pada storage sesuai dengan parameter http request yang ada dan memberikan kembalian data bertipe JSON. Sebagai contoh jika parameter pada http request benar (nama kolom benar, dan nilai where ada) maka sistem akan melakukan kembalian data seperti pada Gambar 4.5.

Pada data tersebut terdapat dua objek yaitu objek ‘table’ dan objek ‘columns’ dimana objek ‘table’ berisi tentang informasi nama dari tabel apa yang diquery dan tabel tersebut memiliki nilai header apa saja. Untuk objek ‘columns’ berisi matriks 2 dimensi berupa nilai kolom-kolom yang ada dan sesuai dengan hasil query yang ada.

Jika pada parameter http request nama header/kolom yang dilakukan proses selection tidak ada, maka nilai object ‘columns’ akan menjadi *error*. Dan jika pada proses selection sistem tidak dapat menemukan data yang tepat sesuai dengan parameter http request maka nilai pada objek ‘columns’ akan menjadi string kosong (‘’).

Translator juga berguna dalam melakukan penentuan nilai TTL baik bertambah ataupun mengatur menjadi minimum sesuai dengan kondisi query ditranslasikan oleh translator.

```
▼table:
    name:      "table_0"
  ▼header:
      0:       "kota"
      1:       "prakiraan_cuaca_dini_hari"
      2:       "suhu(_&deg;c_)"
      3:       "kelembaban(_%_)"
▼columns:
  ▼0:
      0:       "Aceh Barat Daya"
      1:       "Berawan"
      2:       "23 - 32"
      3:       "65 - 95"
  ▼1:
      0:       "Aceh Besar"
      1:       "Berawan"
      2:       "22 - 31"
      3:       "70 - 95"
  ▼2:
      0:       "Aceh Tengah"
      1:       "Berawan"
      2:       "16 - 28"
      3:       "80 - 100"
  ▼3:
      0:       "Banda Aceh"
      1:       "Berawan"
      2:       "23 - 32"
      3:       "65 - 95"
  ▼4:
      0:       "Bener Meriah"
      1:       "Berawan"
      2:       "16 - 28"
      3:       "75 - 100"
  ▼5:
      0:       "Bireun"
      1:       "Berawan"
      2:       "23 - 32"
      3:       "70 - 95"
```


Gambar 4.5 Contoh Data JSON yang Dihasilkan Sistem

4.1.3.5. HTML Parser

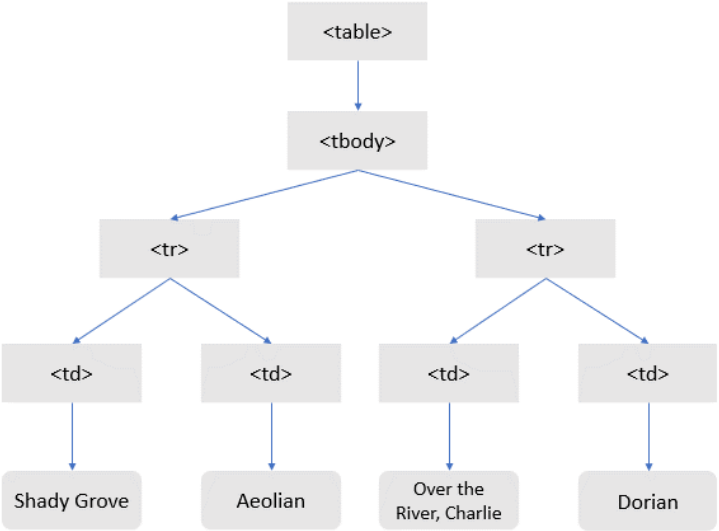
HTML parser digunakan untuk mendeteksi <tag> HTML pada halaman website dan merubah tag tersebut menjadi DOM. Selain itu, parser juga berguna untuk menghilangkan styling, atribut atau mengambil isi text dari sebuah tag. Sebagai contoh pada Gambar 4.6 terdapat struktur HTML sederhana yang digunakan untuk membuat table.

```
<TABLE>
<TBODY>
<TR>
<TD>Shady Grove</TD>
<TD>Aeolian</TD>
</TR>
<TR>
<TD>Over the River, Charlie</TD>
<TD>Dorian</TD>
</TR>
</TBODY>
</TABLE>
```

Gambar 4.6 Struktur Table pada HTML

HTML parser akan mengolah struktur tersebut menjadi sebuah DOM yang ditunjukkan pada Gambar 4.7. DOM tersebut bertujuan untuk memudahkan sistem dalam mengakses data yang ada pada setiap *nodes*. Dimana sistem hanya perlu mengakses *nodes* dan akan mengambil data pada *nodes* yang butuhkan.

Dari *nodes* yang telah terbentuk, sistem akan mengambil isi dari *nodes* tabel dan akan memberikan nama untuk setiap tabel dan membagi isi didalam *nodes* tabel menjadi header dan kolom.



Gambar 4.7 Bentuk DOM

4.1.3.6. Storage

Storage berguna untuk melakukan penyimpanan metadata. Sehingga metadata dapat diakses kembali oleh translator tanpa harus melakukan ekstraksi ulang.

4.1.3.7. Database Protokol

Database protocol merupakan jalan yang digunakan dalam melakukan query pada sistem dan mengembalikan data hasil query pada pengguna. Pada database protocol ini saya menggunakan protokol REST yang memiliki 4 metode yang digunakan memanajemen *resource* sebagai berikut:

- GET: mengambil resource
- POST: membuat resource
- PUT: meperbarui resource
- DELETE: menghapus resource

4.2. Rancangan Database

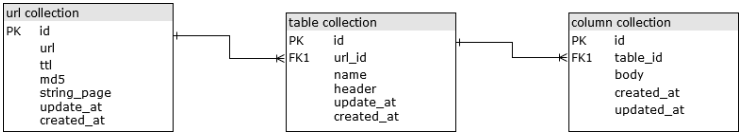
Aplikasi ini bertujuan untuk dapat menyimpan berbagai macam tipe data yang berhasil diambil oleh komponen HTML Parser. Karena setiap tabel pada website memiliki jumlah kolom dan baris yang berbeda-beda dan tipe data dari setiap barus berbeda-beda dengan tabel lain, maka untuk database yang digunakan adalah non-relational database MongoDB.

Data-data tersebut disimpan didalam tiga *collection* yang berbeda, yaitu:

Tabel 4.1 Penjelasan Collection pada Database

No	Collection	Penjelasan
1	Url	Pada collection ini meyimpan informasi apa saja yang berhubungan dengan url website yang dimasukan pengguna untuk mendapatkan elemen HTML table
2	Table	Pada collection ini menyimpan infromasi nama table apa dan header apa saja yang ada di dalam table tersebut.
3	Column	Pada collection ini menyimpan setiap baris dari isi table yang ada pada collection table.

Setiap collection memiliki hubungan dengan collection lain, hubungan ini dapat dilihat dari pada Gambar 4.8dibawah ini.



Gambar 4.8 Hubungan Antar Collection

Dibuatnya relasi one to many untuk url collection dengan table collection karena pada satu url yang ada pada user

dapat memiliki 0 atau lebih tag table. Untuk hubungan table collection dan column collection mempunyai hubungan one to many juga karena setiap table pasti memiliki satu atau lebih kolom. Karena satu dokumen pada column collection merepresentasikan satu row yang berfungsi untuk mempermudah sistem dalam melakukan query selection pada database.

Dari seluruh collection pada Tabel 4.1 dapat dijelaskan data-data yang disimpan adalah sebagai berikut:

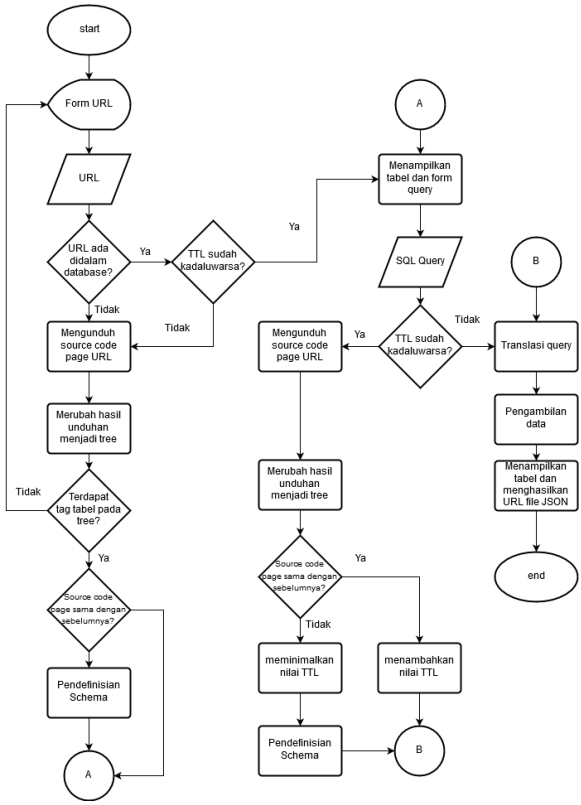
Tabel 4.2 Penjelasan Data pada Collection

No	Nama field	Tipe Data	Keterangan
1	url	String	Url yang dilakukan ekstraksi html
2	ttl	Integer	Nilai time to live dari data
3	md5	String	Enkripsi md5 dari string_table
4	string_table	String	String table dari url yang akan dilakukan ekstraksi data
5	url_id	String	Nomor unik (id) dokumen pada collection url
6	name	String	Nama tabel
7	header	Array	Header tabel yang diurutkan sesuai dengan urutan index array
8	table_id	String	Nomor unik (id) dokumen pada collection table
9	body	Array	Baris dari tabel yang diurutkan berdasarkan urutan index array
10	created_at	Timestamp	Waktu kapan dokumen tersebut dibuat

No	Nama field	Tipe Data	Keterangan
11	updated_at	Timestamp	Waktu kapan terakhir kali dokumen tersebut diupdate

4.3. Rancangan Proses Sistem

Pada bagian ini dijelaskan mengenai bagaimana proses sistem berjalan. Gambaran dari proses sistem tersebut dapat dilihat pada Gambar 4.9.



Gambar 4.9 Flowchart Alur Berjalannya Sistem

Sistem dimulai pada saat sistem menampilkan form untuk pengguna memasukan url yang ingin digunakan untuk query data. Setelah itu sistem akan melakukan proses mencari url tersebut didalam database apakah ada atau tidak. Sistem kemudian melakukan proses mengunduh source code dari halaman url yang ada dan menghasilkan md5 dari data tersebut. Sistem akan melakukan pengecekan apakah terdapat tag table pada url tersebut apa tidak. Jika tidak sistem akan kembali ke halaman form URL, jika tidak sistem akan melanjukan ke proses selanjut.

Proses berikutnya yaitu sistem akan melakukan perbandingan source code apakah source code tersebut sama atau tidak dengan source code sebelumnya. Jika sebelumnya URL tersebut tidak ada didalam database, maka source code tersebut dibandingkan dengan data null. Jika berbeda maka sistem akan melakukan proses pendefinisian schema.

Sistem akan melakukan pendefinisian schema dari hasil pengunduhan *source code* yang telah dilakukan pada proses sebelumnya. Ketika hasil perbandingan source code ternyata sama, maka sistem akan melakukan proses penampilan tabel apa saja yang tersedia didalam url tersebut begitu juga proses yang terjadi ketika setelah melakukan proses pendefinisian schema.

Kemudian pengguna bisa melakukan SQL query terhadap tabel yang telah ditampilkan oleh sistem. Sistem akan melakukan pengecekan terhadap query yang dimasukkan pengguna apakah query tersebut memiliki ttl yang sudah kadaluarsa atau belum. Jika query tersebut tidak kadaluarsa, sistem akan melakukan parsing dan mentranslasikan query tersebut menjadi query pada *storage* MongoDB. Tetapi jika tidak sistem akan melakukan proses mengunduh dan membandingkan source code sama dengan proses sebelumnya.

Perbedaannya disini adalah ketika hasil perbandingan source code ternyata sama, maka sistem akan menambahkan nilai ttl yang ada, dan jika hasilnya berbeda maka sistem akan mengatur ttl menjadi nilai minimum dan melakukan penyimpanan data baru serta melakukan pemrosesan data baru.

Kemudian sistem akan melakukan parsing dan mentraslasikan query tersebut menjadi query pada *storage* MongoDB. Kemudian sistem akan melakukan pengambilan data sesuai query yang telah ditranslasikan. Terakhir sistem akan menampilkan hasil dari query yang dilakukan pengguna.

4.4. Rancangan Antar Muka

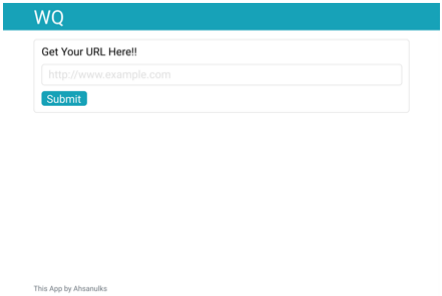
Pada tahap ini akan dilakukan rancangan pembuatan tampilan antar muka yang nantinya digunakan untuk tampilan aplikasi dari awal aplikasi diakses sampai dengan aplikasi terakhir diakses. Rancangan tersebut akan dijelaskan pada Tabel 4.3 berikut:

Tabel 4.3 Rancangan Pembuatan Halaman Antar Muka

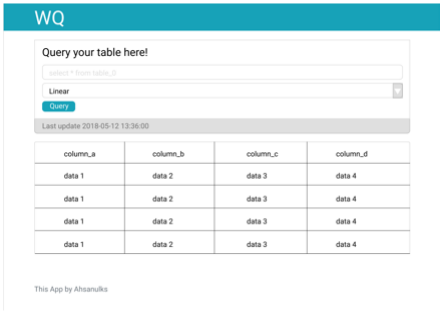
No	Tampilan	Fitur
1	Halaman Utama	Berisi *field* untuk memasukkan url oleh pengguna
2	Halaman Tabel (sebelum dilakukan query)	Berisi *field* untuk melakukan query pada tabel yang ditampilkan Manampilkan daftar list tabel beserta isinya yang bisa dilakukan query oleh pengguna
3	Halaman Tabel (setelah dilakukan query)	Berisi *field* untuk melakukan query pada tabel yang ditampilkan Berisi *field* url untuk mengakses hasil query bertipe JSON Manampilkan table hasil query

Pada Tabel 4.3 diatas, mempunyai contoh bentuk seperti pada Gambar 4.10 (a) pengguna akan memasukkan url halaman yang ingin dilakukan proses ekstraksi data. Setelah itu pengguna akan disuguhkan dengan data tabel apa saja yang tersedia pada halaman tersebut sesuai dengan Gambar 4.10 (b). Pada halaman ini pengguna dapat melakukan query yang dibutuhkan sesuai dengan data yang tersedia. Setelah pengguna

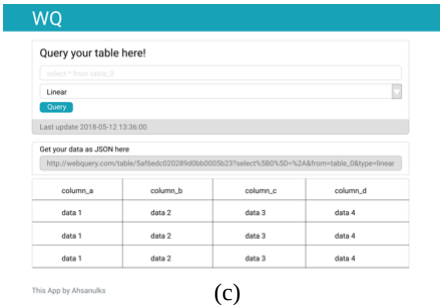
melakukan query, pengguna akan mendapatkan url untuk mengakses hasil query tersebut. Tabel yang berada dibawah akan berubah sesuai dengan hasil query yang dilakukan pengguna Gambar 4.10 (c).



(a)



(b)



(c)

Gambar 4.10 Rancangan Tampilan Aplikasi

4.5. Rancangan Pengujian

Pada tahap ini akan dilakukan rancangan pengujian sistem dari segi fungsional dan tipe kenaikan time to live yang paling tepat untuk website dengan waktu perbedaan pergantian data.

4.5.1. Pengujian Fungsional

Pengujian fungsional dilakukan dengan melakukan unit testing pada setiap unit atau komponen yang ada pada sistem. Pengujian ini menggunakan metode *black box* dan mencatat *error* yang terjadi disetiap unit. Detail rancangan pengujian pada setiap unit atau komponen dapat dilihat pada Tabel 4.4 dibawah ini.

Tabel 4.4 Rancangan Pengujian Fungsional

Komponen	Butir Uji	Cara Pengujian	Hasil yang Diharapkan
Downloader	D01	Mengunduh halaman dari url yang berawalan http://	Sistem berhasil mengunduh halaman
	D02	Mengunduh halaman dari url yang berawalan https://	Sistem berhasil mengunduh halaman
	D03	Mengunduh halaman dari url yang berawalan www.	Sistem berhasil mengunduh halaman
HTML Parser	H01	Mendeteksi terdapat node table	Jika tidak terdapat node table sistem kembali ke halaman awal

Komponen	Butir Uji	Cara Pengujian	Hasil yang Diharapkan
	H02	Membentuk metadata sesuai dengan table pada halaman website	Sistem menampilkan table sesuai dengan halaman website
SQL Parser	S01	Mendeteksi query bukan query select	Sistem menampilkan error
	S02	Mendeteksi error pada query yang tidak terdapat ‘from’	Sistem menampilkan error
	S03	Mendeteksi error pada query yang tabel tidak terdapat pada data	Sistem menampilkan error
Translator	T01	Mendapatkan data dengan query yang memiliki order	Sistem berhasil mendapatkan data sesuai dengan query
	T02	Mendapatkan data dengan query yang memiliki or	Sistem berhasil mendapatkan data sesuai dengan query
	T03	Mendapatkan data dengan query yang memiliki and	Sistem berhasil mendapatkan data sesuai dengan query
	T04	Mendapatkan data dengan query yang	Sistem berhasil

Komponen	Butir Uji	Cara Pengujian	Hasil yang Diharapkan
		memiliki kombinasi or, and dan order	mendapatkan data sesuai dengan query
	T05	Melakukan query pada data yang memiliki nilai TTL belum expired	Nilai TTL pada URL collection tetap
	T06	Melakukan query pada data yang memiliki nilai TTL sudah expired dan belum memiliki perubahan data	Nilai TTL pada URL collection bertambah
	T07	Melakukan query pada data yang memiliki nilai TTL sudah expired dan memiliki perubahan data	Nilai TTL pada URL collection menjadi minimum

Dari setiap butir uji yang ada, akan dilakukan pengecekan sesuai dengan hasil yang diharapkan. Jika hasil yang diharapkan sama dengan hasil pengujian, maka butir uji tersebut berstatus “Lolos”. Jika tidak sesuai maka butir uji tersebut berstatus “Gagal”.

4.5.2. Pengujian Tipe Kenaikan TTL

Pengujain dari tipe kenaikan TTL dilakukan dengan cara melakukan pengambilan data pada setiap website dengan setiap tipe kenaikan TTL di waktu yang bersamaan. Untuk pengaturan nilai C setiap tipe kenaikan TTL dapat dilihat pada Tabel 4.5:

Tabel 4.5 Pengaturan Nilai C Pada Setiap Tipe Kenaikan TTL

Tipe Kenaikan	Nilai c
Linear	1/5
Polynomial	-1
Exponential	1

Dalam melakukan pengambilan data, terdapat dua skenario. Pada skenario pertama pengambilan data dilakukan dengan interval kurang dari sama dengan waktu pembaharuan data pada website. Untuk lebih jelasnya dapat dilihat pada Tabel 4.6.

Tabel 4.6 Skenario Pertama Testing Pengambilan Data

Website	Waktu pembaharuan data	Interval Pengambilan I	Total Pengambilan Data	Perkiraan waktu selesai
Nilai tukar mata uang	± 1 Menit	1 Menit	20 Kali	20 Menit
BMKG	± 6 Jam	1 Jam	20 Kali	20 Jam
Pengumuman	Tidak Pernah	1 Hari	10 Kali	10 Hari

Pada skenario kedua pengambilan data dilakukan dengan interval waktu rata-rata kapan pembaharuan data pada website. Karena terdapat website yang tidak pernah dilakukan pembaharuan data, maka waktu interval pembaharuan datanya dianggap nol (0). Untuk lebih jelasnya dapat dilihat pada Tabel 4.7.

Tabel 4.7 Skenario Kedua Testing Pengambilan Data

Website	Waktu pembaharuan data	Interval Pengambilan II	Total Pengambilan Data	Perkiraan waktu selesai

Nilai tukar mata uang	$\pm$ 1 Menit	3 Jam	20 Kali	60 Jam
BMKG	$\pm$ 6 Jam	3 Jam	20 Kali	60 Jam
Pengumuman	Tidak Pernah	3 Jam	20 Kali	60 Jam

Dari setiap skenario akan dilakukan pengambilan data berupa:

1. Jumlah data yang *outdated* (data yang ditampilkan tidak sama dengan data pada website pada saat itu).
2. Jumlah pengambilan data baru yang terjadwal.

Dari kedua data tersebut akan dibuat binary classification sebagai berikut:

- True Positives: data outdated dan terdapat pengambilan data baru yang terjadwalkan.
- True Negatives: data tidak outdated dan tidak terdapat pengambilan data baru yang terjadwalkan.
- False Positives: data outdated dan tidak terdapat pengambilan data baru yang terjadwalkan
- False Negatives: data tidak outdated dan terdapat pengambilan data baru yang terjadwalkan.

Dari binary classification tersebut akan diambil ratio false positif yang dapat dihitung sesuai dengan persamaan (4.1).

$$FPr = \frac{\sum False\ Positive}{\sum query\ yang\ dilakukan} \quad \textbf{(4.1)}$$

FPr nanti akan dibandingkan dengan ratio seberapa banyak jumlah pengambilan data baru yang terjadwalkan atau bisa disebut *Stale Traffic Rasio (STr)*. STr dapat dihitung dari persamaan (4.2).

$$STr = \frac{\sum pengambilan\ data\ baru\ yg\ terjadwalkan}{\sum query\ yang\ dilakukan} \quad \textbf{(4.2)}$$

BAB V
IMPLEMENTASI

Bab ini berisi tentang proses implementasi dalam pembuatan aplikasi berdasarkan perancangan yang telah dibuat sebelumnya.

5.1. Perangkat Penelitian

Dalam melakukan implementasi pada tugas akhir ini memerlukan perangkat keras dan perangkat lunak pendukung dalam menyelesaika aplikasi. Adapun perangkat-perangkat yang dibutuhkan meliputi perangkat keras dan perangkat lunak beserta spesifikasinya dalam pengembangan aplikasi sebagai berikut:

Tabel 5.1 Spesifikasi Perangkat Keras dan Lunak Dalam Melakukan Pengembangan Aplikasi

Perangkat Keras	
CPU	Intel Core i5 5200U
RAM	12 GB
Data Disk	500 GB SSD
Perangkat Tambahan	Monitor, Keyboard, Mouse
Perangkat Lunak	
Sistem Operasi	Windows 10 Education64-bit
Webserver	Apache 2.4
Bahasa Pemrograman	PHP 7.0, Javascript
Database	MongoDB 3.4.9
Editor	Sublime Text, Visual Studio Code
Tools tambahan	Composer 1.5.2, NPM 5.5.1, Git 2.14.2

Dalam melakukan proses development delivery menggunakan server dengan spesifikasi perangkat keras dan perangkat lunak sebagai berikut:

Tabel 5.2 Spesifikasi Perangkat Keras dan Lunak Dalam Melakukan Development Delivery

Perangkat Keras	
CPU	2 Core
RAM	4 GB
Data Disk	8 GB SSD
Perangkat Lunak	
Sistem Operasi	Ubuntu 16.04 LTS
Web Server	Nginx 1.10.3
Bahasa Pemrograman	PHP 7.0
Database	MongoDB 3.2.20
Tools Tambahan	Composer 1.6.5, NPM 5.6.4, Git 2.7.4

5.2. Pengembangan Sistem

5.2.1. Pembuatan HTML Ekstraktor

Pada proses ini pembuatan terbagi dalam beberapa tahapan, yaitu:

5.2.1.1. Menyiapkan Database

Dalam menyiapkan database yang diakan digunakan sebagai tempat penyimpanan metadata pada proses HTML ekstraktor. Karena database menggunakan MongoDB tidak perlu dilakukan pembuatan database atau table. Yang diperlukan hanya menyambungkan aplikasi agar bisa terhubung kedalam database.

5.2.1.2. Pembuatan Komponen Downloader & HTML Parser

Karena komponen *downloader* sedikit dan hasil dari komponen downloader menjadi input untuk HTML Parser, maka kedua komponen ini dibuat didalam satu class yang sama. Yang nantinya bertujuan untuk membuat metadata yang akan disimpan didalam database.

Kedua komponen tersebut dibuat didalam satu class bernama “SchemaController”. Dimana class ini bertujuan untuk mengolah string url untuk di unduh dan menyimpan metadata dari nodes table yang ada.

```
23. Class SchemaController extends Controller
24. {
25.    private $url;
26.    private $dom;
27.    private $temp_md5;
28.    private $not_have_thead;
29.    private $type;
30.
31.    public function __construct($url, $type = '') {
32.            $this->url = Url::firstOrNew(['url' => $url]);
33.            $this->dom = new Dom();
34.            $this->dom->loadFromUrl($this->url->url)-
    >outerHTML;
35.            $this->url->string_table = (string) $this->dom-
    >find('table');
36.            $this->temp_md5 = md5($this->url-
    >string_table);
37.            $this->type = $type;
38. }
```

Kode 5.1 Class SchemaController

Saat class SchemaController dipanggil, class ini akan secara otomatis membuat beberapa instance variable yaitu:

- $url yang berisi object dari models Url (collection Url)
- $dom yang berisi class DOM

$temp_md5 yang berisi nilai md5 dari string hasil unduhan halaman url dapat dilihat pada kode 5.1.

Pada saat pengguna memberikan *input* sebuah url, maka sistem akan secara langsung memanggil class SchemaController, dimana sistem akan memeriksa apakah url tersebut baru atau tidak, jika baru maka akan membuat metadata yang akan disimpan didalam database. Tetapi jika tidak, maka sistem akan langsung menampilkan metadata yang ada didalam

database yang pernah ada yang dapat dilihat pada class ParserController pada method createDom seperti Kode 5.2.

```
1.  public function createDom(Request $req){
2.    $url    = str_replace('https://', 'http://', $req->url);
3.    $schema = new Schema($url);
4.    if ($schema->is_new()) $schema->create_dom();
5.    $url_id = $schema->get_url_id();
6.    return Redirect::to('/url/'.$url_id);
7.  }
```

Kode 5.2 Method createDom pada ParserController

Method is_new() berisi seperti pada Kode 5.3 dimana sistem akan membandingkan nilai md5 dari instance variable url dan nilai temp_md5. Jika kedua nilai tersebut sama maka method ini akan memberikan nilai kembalian false, tetapi jika nilai tersebut berbeda maka method akan memberikan nilai kembalian true.

```
1.  public function is_new(){
2.    return $this->url->md5 === $this->temp_md5 ? false :
    true;
3.  }
```

Kode 5.3 Method is_new pada SchemaController

Setelah itu sistem akan memanggil method create_dom pada class SchemaController yang memiliki kode seperti Kode 5.4.

```
1.  public function create_dom(){
2.      $this->url->md5 = $this->temp_md5;
3.      $this->url->ttl = Url::TTL_MIN;
4.      $this->url->save();
5.      $this->schema_definition();
6.  }
```

Kode 5.4 Method create_dom pada SchemaController

Pada method create_dom akan dilakukan penyimpanan data pada collection Url, dimana data yang disimpan adalah string_table, md5 dan nilai ttl minimum yang didefinisikan pada instance variable url. Kemudian sistem akan melakukan pendefinisian schema pada method schema_definition.

```
1.  private function schema_definition(){
2.      $tables = $this->dom->find('table');
3.      foreach ($tables as $key => $table) {
4.              $url_id = $this->url->id;
5.              $name           = 'table_'.$key;
6.              $header         = $this-
    >to_lower_and_underscore($this->get_headers_data($table));
7.              $table_id       = $this->update_table($url_id,
    $name, $header, $key);
8.              $this->update_column($table, $table_id);
9.      }
10. }
```

Kode 5.5 Method schema_definition pada SchemaController

Pada method schema_definition sistem akan mencari setiap nodes tabel, kemudian memberikan nama tabel untuk setiap node yang ada. Sistem juga akan mencari nodes tabel yang merepresentasikan header pada tabel website dengan menggunakan method get_headers_data seperti Kode 5.6.

```
1.  private function get_headers_data($dom){
2.      $thead = $dom->find('thead', 0);
3.      if (is_null($thead)){
4.              $this->not_have_thead    = true;
5.              $row = $dom->find('tr', 0);
6.              $th  = $this->get_data_table($row, 'th');
7.      }
8.      else {
9.              $row = $thead->find('tr');
10.             $th  = sizeof($row) == 0 ? $this-
    >get_data_table($thead, 'th') : $this-
    >get_data_table($row, 'th');
11.     }
12.     return $th;
```

```
13. }
```

Kode 5.6 Method get_headers_data pada SchemaController

Sistem akan mencari nilai nodes ‘th’ didalam nodes ‘theader’. Jika tidak ada, sistem akan mengambil nodes pertama ‘tr’ pada nodes ‘tbody’ untuk dijadikan header table. Kemudian sistem akan melakukan penyimpanan data pada collection table pada method update_table pada Kode 5.7.

```
1.  private function update_table($url_id, $name, $header,$i){
2.      if (sizeof($this->url->tables) > 0) {
3.              $this->url->tables[$i]->name     = $name;
4.              $this->url->tables[$i]->header   = $header;
5.              $this->url->tables[$i]->save();
6.              $id = $this->url->tables[$i]->id;
7.      }
8.      else{
9.              $data['url_id'] = $url_id;
10.             $data['name']   = $name;
11.             $data['header'] = $header;
12.             $table          = Tabel::create($data);
13.             $id             = $table->id;
14.     }
15.     return $id;
16. }
```

Kode 5.7 Method update_table pada SchemaController

```
1.  private function update_column($table, $table_id){
2.      $tbody = $table->find('tbody', 0);
3.      if (is_null($tbody)) {
4.              $rows = $table->find('tr');
5.      }
6.      else{
7.              $rows = $tbody->find('tr');
8.      }
9.      $data['tabel_id'] = $table_id;
10.
11.     $i = 0;
12.     if ($this->not_have_thead) $i = 1;
13.
```

```
14.     for ($i; $i < sizeof($rows); $i++) {
15.             $body[] = $this->get_body_data($rows[$i]);
16.     }
17.
18.     $this->delete_column($table_id);
19.     $chunk_body = array_chunk($body, 1);
20.     foreach ($chunk_body as $body) {
21.             $data['body'] = $body[0];
22.             Column::create($data);
23.     }
24. }
```

Kode 5.8 Method update_column pada SchemaController

Setelah itu, sistem akan meneruskan mencari data pada node ‘tbody’ untuk mendapatkan nilai kolom dari table yang telah disimpan sebelumnya. Sistem akan menjadi data pada note ‘td’ dimana data pada body tabel tersimpan seperti Kode 5.8.

```
1.  private function get_body_data($dom){
2.      $row_th = $dom->find('th', 0);
3.      if (is_null($row_th)) {
4.              $data[] = $this->get_data_table($dom, 'td');
5.      }
6.      else{
7.              $data[] = strip_tags($row_th->innerHtml);
8.              $data[] = $this->get_data_table($dom, 'td');
9.      }
10.     return flatten($data);
11. }
```

Kode 5.9 Method get_body_data pada SchemaController

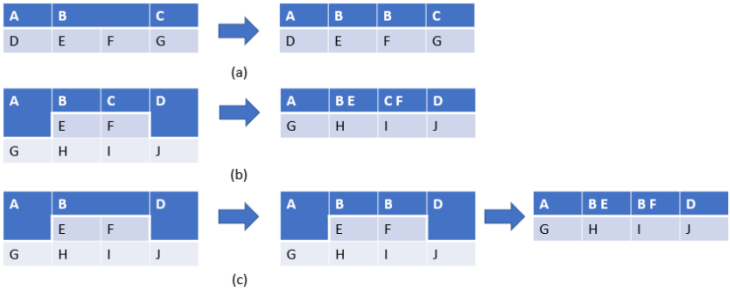
Sistem akan melakukan pengecekan, apakah terdapat nilai tag_th pada node ‘tbody’ body atau tidak. Jika ada sistem akan mengambil nilai node ‘th’ pertama node ‘tbody’ dan dilanjutkan mengambil nilai pada nodes ‘td’. Jika tidak sistem akan secara langsung mengambil nilai pada nodes ‘td’ seperti pada Kode 5.9. Kemudian sistem akan melakukan pengecekan apakah terdapat atribut rowspan atau colspan yang ada didalam

node td. Jika terdapat atribut colspan, sistem akan menyalin data sebelumnya sebanyak jumlah colspan yang ada. Jika terdapat atribut rowspan, sistem akan memasangkan data pada baris yang memiliki rowspan dengan data pada baris berikutnya seperti pada Kode 5.10.

```
1.  private function get_data_table($dom, $search, $i = 0){
2.      $datas = (sizeof((array) $dom) != 10) ? $dom[$i]-
    >find($search) : $dom->find($search);
3.      $array   = array();
4.      $rowspan = array();
5.      foreach ($datas as $data) {
6.              $colspantemp[]  = $data-
    >getAttribute('colspan');
7.              $colspan = $data->getAttribute('colspan');
8.              $rowspan[] = $data->getAttribute('rowspan');
9.              $array[] = $this->get_colspan_data($colspan,
    $data);
10.     }
11.     $uniq_rowspan = array_unique($rowspan);
12.     if (sizeof($uniq_rowspan) > 1 && in_array(null,
    $uniq_rowspan, TRUE)) {
13.             $index_rowspan = array_keys($rowspan, null,
    true);
14.             $temp = $this->get_data_table($dom, $search,
    $i+1);
15.             $array[$index_rowspan[0]] = $this-
    >set_colspan($array[$index_rowspan[0]], $temp,
    max($colspantemp));
16.     }
17.     return flatten($array);
18. }
```

Kode 5.10 Method get_data_table pada SchemaController

Untuk lebih mendapatkan gambaran bagaimana proses dalam pembentukan metadata yang memiliki atribut colspan dan rowspan dapat dilihat pada Gambar 5.1 Proses Mendapatkan Data yang Memiliki Colspan dan RowspanGambar 5.1.



Gambar 5.1 Proses Mendapatkan Data yang Memiliki Colspan dan Rowspan

Pada Gambar 5.1 (a) terjadi proses pembentukan metadata yang memiliki atribut colspan dimana sistem akan menyalin data sebanyak jumlah atribut colspan yang ada. Pada Gambar 5.1 (b) terjadi proses pembentukan metadata yang memiliki atribut rowspan. Dimana sistem akan memasangkan data yang tidak memiliki atribut rowspan pada baris yang sama dengan data pada baris berikutnya. Pada Gambar 5.1 (c) terjadi proses pembentukan metadata dimana tabel memiliki atribut colspan dan rowspan pada satu baris yang sama. Maka sistem akan menyalin data yang memiliki atribut colspan terlebih dahulu, kemudian data tersebut akan pasangkan dengan data pada baris berikutnya.

```
1.  function fix_numeric_data($number){
2.      $temp = str_replace('.', ',', $number);
3.      if (is_numeric($number) && strpos($temp, ',')) {
4.          $data = (float) $number;
5.      }
6.      elseif (is_numeric($number)) {
7.          $data = intval($number);
8.      }
9.      else{
10.         $data = $number;
11.     }
12.     return $data;
```

```
13. }
```

Kode 5.11 Helper fix_numeric_data

Ketika sistem melakukan pengambilan data pada node ‘td’, sistem juga melakukan pengecekan apakah tipe data pada node ‘td’ data bertipe string atau numeric yang dibantu oleh helper yang dapat dilihat pada Kode 5.11. Karena secara default tipe data yang ada pada node ‘td’ adalah string. Dan jika bertipe numeric sistem akan merubah data tersebut menjadi bertipe integer atau float. Untuk data numeric yang bertipe float, sistem hanya bisa mendeteksi jika data tersebut memiliki delimiter titik (.). Untuk lebih jelasnya terdapat contoh sebagai berikut:

- ‘2.5’ akan terdeteksi data bertipe float dengan nilai 2,5
- ‘2,5’ akan terdeteksi sebagai data bertipe string
- ‘2.0’ akan terdeteksi data bertipe float dengan nilai 2.0
- ‘2’ akan terdeteksi data bertipe integer

Setelah mendapatkan seluruh nilai kolom pada node ‘tbody’ maka sistem akan melakukan penyimpanan data tersebut pada column collection.

5.2.1.3. Validasi Metadata

Setiap sistem selesai melakukan penyimpanan schema, peneliti melakukan pembandingan data yang terbentuk oleh system dengan data yang ada pada website untuk memvalidasi metadata yang tersimpan. Apakah metadata tersebut sudah sesuai dengan rancangan data dan memiliki tipe data yang tepat.

5.2.2. Pembuatan SQL Engine

5.2.2.1. Pembuatan Komponen SQL Parser

Dalam pembuatan komponen SQL Parser, peneliti menggunakan library phpMyAdmin/sql-parser sebagai tools untuk merubah SQL yang dimasukkan pengguna menjadi SQL tree. Sehingga peneliti hanya berfokus melakukan pengambilan data pada SQL tree dan merubah query tersebut menjadi http params agar dapat diolah oleh komponen translator.

```
1.  function sql_parser(Request $req){
2.    $query  = $req->sql;
3.    $parser = new Parser($query);
4.    $flags  = Query::getFlags($parser->statements[0]);
5.    if ($flags['querytype'] == 'SELECT') {
6.      $from       = strtolower($parser->statements[0]-
    >from[0]->table);
7.      $expression = $parser->statements[0]->expr;
8.      $table      = Url::find($req->id)->tables-
    >where('name',$from)->first();
9.      foreach ($expression as $column) {
10.       $select[] = $column->expr;
11.     }
12.     $data['select'] = $select;
13.     $data['from']   = $from;
14.     if($parser->statements[0]->order) $data['order'] =
    $this->get_order($parser->statements[0]->order);
15.     $where = $parser->statements[0]->where;
16.     if ($where != null) {
17.       $data['where'] = $this->get_where($where);
18.     }
19.     $data['type'] = $req->type;
20.
21.     return url('/table/'.$table-
    >id."?".http_build_query($data));
22.   }
23.   else{
24.     echo "query denied";
25.   }
26. }
```

Kode 5.12 Method sql_parser Untuk Merubah Query Menjadi HTTP Params

Pada Kode 5.12 baris 1 dan 2 sistem akan menerima url yang dimasukan oleh pengguna. Kemudian system memanggil class Parser dari library phpMyAdmin/sql-parser untuk merubah query tersebut menjadi SQL tree. Sistem kemudian melakukan pengecekan apakah query yang dilakukan pengguna query ‘SELECT’ atau tidak pada Kode 5.12 baris 5. Karena system hanya membatasi query yang bisa dilakukan adalah query select. Kemudian system akan melakukan pengambilan

data untuk kolom mana yang dilakukan pengambilan, dari tabel mana dan memiliki kondisi apa saja untuk memenuhi kondisi yang telah diberikan pengguna.

```
1.  public function get_where($where){
2.    foreach ($where as $w) {
3.      if ($w->isOperator) {
4.        $data['operators'][]  = $w->expr;
5.      }
6.      else{
7.        $data['arguments'][]  = str_replace('\'', '', $w-
    >expr);
8.        $data['identifier'][] = $w->identifiers[0];
9.      }
10.   }
11.   return $data;
12. }
```

Kode 5.13 Method get_where Untuk Mendapatkan Nilai Where Dari Query Yang Dilakukan

Dalam mendapatkan where-clause yang dimasukan pengguna, system memerlukan pengecekan apakah where tersebut berupa operator (and, or). Karena komponen translator akan lebih mudah mengolah data where-clause jika dilakukan pemisahan antara operator, argument dan identifier yang digunakan pengguna. Sebagai contoh dapat dilihat pada query Kode 5.14.

```
1.  select * from table_0 where prakiraan_cuaca_dini_hari =
    'Hujan Lokal' and prakiraan_cuaca_siang = 'Cerah' order by
    Kota desc
```

Kode 5.14 Contoh Query Where-Clause Dengan Operator And dan Order

Pada query Kode 5.14 diatas akan menghasilkan data where seperti pada Gambar 5.2. Dari data tersebut dapat dilihat bahwa isi dari array dengan key arguments merupakan kondisi where-clause pada query. Karena query tersebut memberikan

dua kondisi dimana kondisi pertama adalah “prakiraan_cuaca_dini_hari = ‘Hujan Lokal’ dan kondisi kedua adalah “prakiraan_cuaca_siang = ‘Cerah’” maka array arguments memiliki dua index. Ketika kondisi yang ada pada query bertambah, maka index pada array arguments akan bertambah. Untuk array pada identifier juga berlaku demikian. Yang berbeda adalah pada array identifier hanya berisi kolom mana yang diberikan kondisi tersebut. Setiap index pada array identifier berpasangan dengan index yang sesuai dengan array arguments. Sebagai contoh index ke 0 para array arguments adalah pasangan dari array identifier pada index ke 0. Array operators akan menyimpan operator penghubung antar where-clause yang ada.

```
array:3 [
  "arguments" => array:2 [
    0 => "prakiraan_cuaca_dini_hari = Hujan Lokal"
    1 => "prakiraan_cuaca_siang = Cerah"
  ]
  "identifier" => array:2 [
    0 => "prakiraan_cuaca_dini_hari"
    1 => "prakiraan_cuaca_siang"
  ]
  "operators" => array:1 [
    0 => "AND"
  ]
]
```

Gambar 5.2 Contoh Data yang Dihasilkan Method get_where

Sedangkan untuk mendapatkan nilai order pada query Kode 5.14. Sistem akan memanggil method get_order seperti pada Kode 5.15

```
1.  function get_order($order){
2.    foreach ($order as $key => $order) {
3.      $data['arguments'][]  = $order->expr->expr;
4.      $data['type'][]       = $order->type;
```

```
    }
    return $data;
}
```

Kode 5.15 Method get_order Untuk Mendapatkan Nilai Order pada Query

Dalam mendapatkan nilai order prinsipnya sama dengan mendapatkan nilai where. Hanya saja disini hanya dibagi menjadi argument dan tipe yang dilakukan order pengguna. Untuk lebih jelas data yang dihasilkan dapat melihat pada Gambar 5.3. Pada gambar tersebut dapat dilihat bahwa array arguments adalah kolom yang akan dilakukan penyortiran dan array type adalah tipe penyortiran yang digunakan. Ketika query hanya menyebutkan kolom yang akan dilakukan penyortiran saja. Secara otomatis nilai array type adalah ‘ASC’ yang berarti *Ascending*.

```
array:2 [
  "arguments" => array:1 [
    0 => "Kota"
  ]
  "type" => array:1 [
    0 => "DESC"
  ]
]
```

Gambar 5.3 Contoh Hasil Data pada Method get_where

Dari seluruh data yang didapatkan oleh system. Akan dibentuk http (url) dengan params seperti pada Kode 5.15. Url tersebut akan mengakses komponen translator dan komponen translator akan melakukan pengambilan data sesuai dengan parameter yang diberikan.

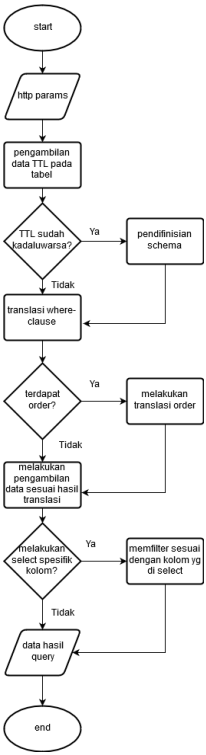
```
http://localhost/table/5b01b89220289d3024001a73?select%5B0%5D=%
2A&from=table_0&order%5Barguments%5D%5B0%5D=Kota&order%5Btype%5
D%5B0%5D=DESC&where%5Barguments%5D%5B0%5D=prakiraan_cuaca_dini_
hari+%3D+Hujan+Lokal&where%5Barguments%5D%5B1%5D=prakiraan_cuac
```

```
a_siang_hari+%3D+Cerah&where%5Bidentifier%5D%5B0%5D=prakiraan_c
uaca_dini_hari&where%5Bidentifier%5D%5B1%5D=prakiraan_cuaca_sia
ng_hari&where%5Boperators%5D%5B0%5D=AND&type=linear
```

Gambar 5.4 Contoh Hasil HTTP Params yang Dihasilkan Method SQL Parser

5.2.2.2. Pembuatan Komponen Translator

Komponen translator memiliki alur seperti pada Gambar 5.5.



Gambar 5.5 Flowchart Komponen Translator

Pertama kali translator akan menerima http params yang telah diberikan oleh komponen SQL parser. Kemudian

sistem akan melakukan pengecekan apakah nilai TTL dari URL yang ingin diambil datanya memiliki nilai expired atau tidak seperti Kode 5.16 pada baris ke 6. Ketika nilai TTL expired, maka sistem akan melakukan proses pendefinisian schema kembali.

```
1.  public function getTable($id, Request $req){
2.      if(!in_array($req->type, Url::TTL_TYPE)) return
    'Invalid Type';
3.      $table  = Table::select('name', 'header', 'url_id')-
    >where('_id', $id)->first();
4.      $url    = $table->url;
5.      $ttl    = new Ttl($url->url, $req->type);
6.      if ($ttl->is_expired()) {
7.          $schema = new Schema($url->url, $req->type);
8.          $schema->update_dom();
9.          $table  = Table::select('name', 'header',
    'url_id')->where('_id', $id)->first();
10.     }
11.     $headers        = $table->header;
12.     $table          = $this->get_table_and_header($table,
    $req->select);
13.     $data['table']  = $table;
14.     if (isset($req->where)){
15.         $data['columns'] = $this-
    >get_column_with_where($req->select, $id, $headers, $req-
    >where, $req->order);
16.     }
17.     else{
18.         $data['columns'] = $this->get_column($req->select,
    $id, $headers, $req->order);
19.     }
20.    return response()->json($data);
21. }
```

Kode 5.16 Method getTable Sebagai Method Utama Komponen Translator

Kemudian komponen translator akan melakukan translasi http params, dimulai dari apakah http params tersebut memiliki nilai where atau tidak.

Jika tidak, sistem melakukan memanggil method get_column yang dapat dilihat pada Kode 5.17.

```
1.  public function get_column($select, $table_id, $header,
    $order = false){
2.      if ($order) { if (!$this-
    >valid_order($order['arguments'], $header)) return
    'Error'; }
3.      if (!$this->valid_select($select, $header)) return
    'Error';
4.
5.      $query      = Column::query()->where('tabel_id',
    $table_id);
6.
7.      if ($order) $this->add_order($query, $order, $header);
8.      $result     = $this->get_body_column($query->get());
9.
10.     return $select[0] == '*' || $result == '' ? $result :
    $this->dynamic_select($result, $GLOBALS['selects']);;
11. }
```

Kode 5.17 Method get_clumn Untuk Melakukan Pengambilan Data Kolom

Pada method get_column akan melakukan pengambilan data kolom sesuai dengan table yang ingin diambil oleh pengguna. Kemudian sistem akan melakukan pengecekan apakah terdapat order atau tidak, jika ada maka sistem akan memanggil method add_order dan kemudian sistem akan melakukan pengambilan seluruh data sesuai dengan order yang ada. Sistem akan melakukan pengecekan apakah pengambilan data yang dilakukan pengguna merupakan pengambilan untuk semua kolom (“*”) atau untuk kolom yang spesifik. Ketika pengguna mengambil semua kolom maka sistem akan langsung meberikan respon sesuai dengan data yang diambil didalam database. Tetapi jika tidak, maka sistem akan melakukan filter menurut kolom yang dilakukan pengambilan melalui method dynamic_select yang dapat dilihat pada Kode 5.18.

```
1.  public function dynamic_select($columns_table,
    $column_index){
2.      $i = 0;
3.      foreach ($columns_table as $column) {
4.          foreach ($column_index as $col_index) {
5.              $columns[$i][] = $column[$col_index];
6.          }
7.          $i++;
8.      }
9.      return $columns;
10. }
```

Kode 5.18 Method dynamic_select Untuk Melakukan Filtering Kolom

Sedangkan jika http params memiliki where, maka sistem akan memanggil method get_column_with_where yang dapat dilihat pada Kode 5.19.

```
1.  public function get_column_with_where($selects, $id,
    $header, $where, $order = false){
2.      if (!$this->valid_where_and_select($selects, $where,
    $header)) return 'Error';
3.      if ($order) { if (!$this-
    >valid_order($order['arguments'], $header)) return
    'Error'; }
4.      foreach ($where['arguments'] as $key => $args) {
5.          $delimiter                          = $this-
    >explode_where($args);
6.          $GLOBALS['where_condition'][]       = $this-
    >get_condition($delimiter, $args);
7.          $GLOBALS['where_condition'][$key][] =
    trim($delimiter);
8.      }
9.      $GLOBALS['operators']   = isset($where['operators']) ?
    $where['operators'] : false;
10.     $results                = $this-
    >get_column_collection($id, $header, $order);
11.     $result                 = $this-
    >get_body_column($results);
12.     return $selects[0] == '*' || $result == '' ? $result :
    $this->dynamic_select($result, $GLOBALS['selects']);
```

```
13. }
```

Kode 5.19 Method get_column_with_where Untuk Melakukan Pengambilan Data yang Memiliki Where

Pada method get_column_with_where nilai where yang ada pada http params akan dilakukan translasi lebih spesifik kembali untuk mencari where condition yang dilakukan oleh pengguna. Sebagai contoh untuk query pada Kode 5.14 akan menghasilkan where seperti pada Gambar 5.6.

```
"where_index" => &49 array:2 [▼
  0 => 4
  1 => 2
]
"where_condition" => &50 array:2 [▼
  0 => array:3 [▼
    0 => "prakiraan_cuaca_dini_hari"
    1 => "Hujan Lokal"
    2 => "="
  ]
  1 => array:3 [▼
    0 => "prakiraan_cuaca_siang"
    1 => "Cerah"
    2 => "="
  ]
]
"operators" => &51 array:1 [▼
  0 => "AND"
]
```

Gambar 5.6 Hasil Translasi Where Condition pada Method get_column_with_where

Array where_index menyimpan index dimana kolom yang dilakukan where condition berada. Untuk array where_condition akan menyimpan hasil perubahan array arguments yang ada pada Gambar 5.2 menjadi lebih spesifik. Pada index kenol akan terisi nama kolom yang diberikan kondisi where, pada index kesatu berisi nilai yang harus dimiliki kolom tersebut. Dan pada index kedua berisi operator

kondisi yang ingin dicapai. Dan untuk array operators masih memiliki nilai yang sama seperti pada Gambar 5.2.

Setelah melakukan translasi where, method get_column_with_index akan memanggil method get_column_collection yang berguna untuk melakukan pengambilan data terhadap query yang diberikan. Method get_column_collection dapat dilihat pada Kode 5.20.

```
1.  public function get_column_collection($table_id, $header,
    $order){
2.      $result = Column::query()->where('tabel_id',
    $table_id)->where(function ($query){
3.                      $this->dynamic_where($query,
    $GLOBALS['operators'], $GLOBALS['where_index'],
    $GLOBALS['where_condition'], true);
4.                  });
5.      if ($order) $this->add_order($result, $order,
    $header);
6.      return $result->get();
7.  }
```

Kode 5.20 Method get_column_collection Untuk Melakukan Pengambilan Data

Pada method get_column_collection akan mentranslasikan where yang sebelumnya berupa array, akan dirubah kedalam query builder framework Laravel melalui method dynamic_where dan mentranslasikan array order menjadi query builder framework Laravel melalui method add_order. Method dynamic_where dapat dilihat pada

```
1.  public function dynamic_where($query, $operators,
    $where_index, $where_condition){
2.      for ($i=0; $i < sizeof($where_index) ; $i++) {
3.          if ($i == 0) {
4.              $this->first_where($operators, $query,
    $where_index, $where_condition, $i);
5.          }
6.          $this->add_where($operators, $query, $where_index,
    $where_condition, $i);
7.      }
```

```
8.  }
9.
10. public function first_where($operators, $query,
    $where_index, $where_condition, $i){
11.     if ($where_condition[$i][2] == 'between' ||
    $where_condition[$i][2] == 'not between') {
12.         $this->query_between($query, $where_index,
    $where_condition, $i);
13.     }
14.     elseif ($where_condition[$i][2] == 'in' ||
    $where_condition[$i][2] == 'not in'){
15.         $this->query_in($query, $where_index,
    $where_condition, $i);
16.     }
17.     else{
18.         $this->query_and($query, $where_index,
    $where_condition, $i);
19.     }
20. }
21.
22. public function add_where($operators, $query,
    $where_index, $where_condition, $i){
23.     if (isset($operators[$i-1])) {
24.         $this->choose_where($operators, $query,
    $where_index, $where_condition, $i);
25.     }
26. }
27.
28. public function choose_where($operators, $query,
    $where_index, $where_condition, $i){
29.     if ($where_condition[$i][2] == 'between' ||
    $where_condition[$i][2] == 'not between') {
30.         $this->where_between_condition($query, $operators,
    $i);
31.     }
32.     elseif ($where_condition[$i][2] == 'in' ||
    $where_condition[$i][2] == 'not in'){
33.         $this->where_in_condition($query, $operators, $i);
34.     }
35.     else{
36.         $this->where_condition($operators, $query,
    $where_index, $where_condition, $i);
37.     }
```

```
38. }
```

Kode 5.21 Method dynamic_where, first_where, add_where, dan choose_where

Method dynamic_where berfungsi sebagai menambahkan where sesuai banyaknya where index yang ada. Ketika index pertama dieksekusi oleh sistem, maka sistem akan memanggil method first_where yang berfungsi untuk melakukan translasi query apakah query tersebut merupakan query “where in”, “where between” atau hanya query where biasa. Sistem akan memanggil method query_between, query_in dan query_and sesuai dengan kondisi yang ada. Kemudian setelah memanggil method first_where, sistem akan memanggil method add_where yang akan melakukan pengecekan apakah terdapat operator dari index - 1 dari array_where index yang ada. Jika ada, maka sistem akan memanggil method choose_where yang fungsinya hampir sama dengan method first_where untuk memetakan kondisi where yang ada. Perbedaannya adalah pada method choose_where akan memanggil method where_between_condition, where_in_condition dan where_condition yang dapat dilihat pada .

```
1.  public function where_between_condition($query,
    $operators, $i){
2.      if ($operators[$i-1] == 'AND') {
3.          $this->query_between($query,
    $GLOBALS['where_index'], $GLOBALS['where_condition'], $i);
4.      }
5.      else{
6.          $GLOBALS['i'] = $i;
7.          $query->orWhere(function ($query2) {
8.              $this->query_between($query2,
    $GLOBALS['where_index'], $GLOBALS['where_condition'],
    $GLOBALS['i']);
9.          });
10.     }
11. }
12.
```

```
13. public function where_in_condition($query, $operators,
    $i){
14.     if ($operators[$i-1] == 'AND') {
15.         $this->query_in($query, $GLOBALS['where_index'],
    $GLOBALS['where_condition'], $i);
16.     }
17.     else{
18.         $GLOBALS['i'] = $i;
19.         $query->orWhere(function ($query2) {
20.             $this->query_in($query2,
    $GLOBALS['where_index'], $GLOBALS['where_condition'],
    $GLOBALS['i']);
21.         });
22.     }
23. }
24.
25. public function where_condition($operators, $query,
    $where_index, $where_condition, $i){
26.     if ($operators[$i-1] == 'AND') {
27.         $this->query_and($query, $where_index,
    $where_condition, $i);
28.     }
29.     elseif ($operators[$i-1] == 'OR') {
30.         $this->query_or($query, $where_index,
    $where_condition, $i);
31.     }
32. }
```

Kode 5.22 Method where_between_condition, where_in_condition dan where_condition

Pada ketiga method tersebut memiliki kesamaan yaitu melakukan pengecekan apakah index query yang masuk merupakan query yang memiliki operator OR atau AND. Ketika sudah teridentifikasi, maka masing-masing method akan memanggil method lain yang berguna untuk menambahkan query sesuai dengan kondisi operator yang ada.

```
1.  public function query_and($query, $where_index,
    $where_condition, $i){
2.      $query->where('body.'.$where_index[$i],
    $where_condition[$i][2],
    fix_numeric_data($where_condition[$i][1]));
```

```
3.  }
4.
5.  public function query_or($query, $where_index,
    $where_condition, $i){
6.      $query->orWhere('body.'.$where_index[$i],
    $where_condition[$i][2],
    fix_numeric_data($where_condition[$i][1]));
7.  }
8.
9.  public function query_between($query, $where_index,
    $where_condition, $i){
10.     if ($where_condition[$i][2] == 'between') {
11.         $query->whereBetween('body.'.$where_index[$i],
    $this->array_is_numeric($where_condition[$i][1]));
12.     }
13.     else{
14.         $data       = $this-
    >array_is_numeric($where_condition[$i][1]);
15.         if (is_numeric($data[0]) && is_numeric($data[1]))
    {
16.             $data[0]    = $data[0] - 1;
17.             $data[1]    = $data[1] + 1;
18.         }
19.         $query->whereNotBetween('body.'.$where_index[$i],
    $data);
20.     }
21. }
22.
23. public function query_in($query, $where_index,
    $where_condition, $i){
24.     if ($where_condition[$i][2] == 'in') {
25.         $query->whereIn('body.'.$where_index[$i], $this-
    >array_is_numeric($where_condition[$i][1]));
26.     }
27.     else{
28.         $query->whereNotIn('body.'.$where_index[$i],
    $this->array_is_numeric($where_condition[$i][1]));
29.     }
30. }
```

Kode 5.23 Method query_and, query_or, query_between dan query_in

Pada Kode 5.23 terdapat 4 method yang memiliki kesamaan. Dimana method-method tersebut berguna untuk

menambahkan query builder terhadap query yang akan dilakukan sistem. Perbedaannya terdapat translasi kembali pada method query_between dan query_in yang mengidentifikasi apakah query tersebut merupakan query yang mengandung “not” atau tidak. Ketika berhasil teridentifikasi maka sistem akan menambahkan query builder sesuai kondisi yang berhasil ditranslasikan. Pada Kode 5.20 sistem akan kembali lagi untuk mentranslasikan order menjadi query builder pada method add_order.

```
1.  public function add_order($query, $order, $header){
2.      $order['index'] = $this-
    >map_arguments($order['arguments'], $header);
3.      foreach ($order['index'] as $key => $order_index) {
4.          $query->orderBy('body.'.$order_index,
    $order['type'][$key]);
5.      }
6.  }
```

Kode 5.24 Method add_order Untuk Mentranslasikan Array Order Menjadi Query Builder

Pada Kode 5.24 terlihat sistem akan melakukan pengambilan seluruh order yang oleh array order. Dan mentranslasikan array tersebut menjadi query builder sesuai dengan tipe order yang dilakukan apakah Ascending atau Descending.

Halaman ini sengaja dikosongkan

BAB VI
HASIL DAN PEMBAHASAN

Pada bab ini akan dijelaskan hasil serta analisis terhadap hasil yang diperoleh dari proses implementasi yang telah dibahas pada bab sebelumnya.

6.1. Hasil Pengujian

Berdasarkan hasil percoban yang telah dilakukan diperoleh hasil pengujian sistem dari segi fungsional dan tipe kenaikan time to live yang paling tepat untuk website dengan waktu perbedaan pergantian data.

6.1.1. Pengujian Fungsional

Pengujian fungsional dilakukan dengan melakukan unit testing pada setiap unit atau komponen yang ada pada sistem. Komponen pengujian yang dilakukan terdiri dari downloader, HTML parser, SQL parser, dan translator, dimana masing-masing komponen terdiri dari beberapa butir ujir yang dijabarkan dibawah ini.

6.1.1.1. Komponen Downloader

Pada komponen ini dilakukan pengujian dalam mendapatkan halaman website yang akan dilakukan pengambilan data. Pada pengujian komponen ini berfokus pada website yang digunakan dalam studi kasus. Hasil pengujian komponen downloader dapat dilihat pada Tabel 6.1.

Tabel 6.1 Hasil Pengujian Fungsional Komponen Downloader

Butir Uji	Data	Keadaan Sekarang	Keadaan Setelah	Status
D01	http://www.bmkg.go.id/cuaca/prakiraan-cuaca-indonesia.bmkg	Object DOM kosong	Object DOM terisi string website	OK

Butir Uji	Data	Keadaan Sekarang	Keadaan Setelah	Status
	http://www.x-rates.com	Object DOM kosong	Object DOM terisi string website	OK
	http://www.pengumuman.mlcunair.com/hasil?jenjang=sma	Object DOM kosong	Object DOM terisi string website	OK
D02	https://www.bmkg.go.id/cuaca/prakiraan-cuaca-indonesia.bmkg	Object DOM kosong	Object DOM terisi string website	OK
	https://www.x-rates.com	Object DOM kosong	Object DOM terisi string website	OK
	https://www.pengumuman.mlcunair.com/hasil?jenjang=sma	Object DOM kosong	Object DOM terisi string website	OK
D03	www.bmkg.go.id/cuaca/prakiraan-cuaca-indonesia.bmkg	Object DOM kosong	Object DOM terisi string website	OK
	www.x-rates.com	Object DOM kosong	Object DOM terisi string website	OK

Butir Uji	Data	Keadaan Sekarang	Keadaan Setelah	Status
	www.pengumuman.mlcunair.com/hasil?jenjang=sma	Object DOM kosong	Object DOM terisi string website	OK

6.1.1.2. Komponen HTML Parser

Pengujian pada komponen HTML Parser bertujuan dalam melakukan validasi apakah sistem mampu mendeteksi *nodes* table dengan benar. Hasil pengujian komponen HTML Parser dapat dilihat pada Tabel 6.2.

Tabel 6.2 Hasil Pengujian Fungsional Komponen HTML Parser

Butir Uji	Data	Keadaan Sekarang	Keadaan Setelah	Status
H01	http://www.bmkg.go.id/cuaca/prakiraan-cuaca-indonesia.bmkg	Object DOM terisi string website	Terdeteksi terdapat nodes table	OK
	http://www.x-rates.com	Object DOM terisi string website	Terdeteksi terdapat nodes table	OK
	http://www.pengumuman.mlcunair.com/hasil?jenjang=sma	Object DOM terisi string website	Terdeteksi terdapat nodes table	OK
	http://www.detik.com	Object DOM terisi	Tidak terdeteksi nodes table,	OK

Butir Uji	Data	Keadaan Sekarang	Keadaan Setelah	Status
		string website	kembali ke halaman awal	
H02	http://www.bmkg.go.id/cuaca/prakiraan-cuaca-indonesia.bmkg	Object DOM terisi string website	Terbentuk Metadata yang sesuai table sumber	OK
	http://www.x-rates.com	Object DOM terisi string website	Terbentuk Metadata yang sesuai table sumber	OK
	http://www.pengumuman.mlcunair.com/hasil?jenjang=sma	Object DOM terisi string website	Terbentuk Metadata yang sesuai table sumber	OK

6.1.1.3. Komponen SQL Parser

Pengujian pada komponen SQL Parser berfokus pada pemvalidasian query yang dilakukan pengguna oleh sistem. Sehingga query yang akan diproses hanya query select yang memiliki where dan order. Untuk pengujian komponen ini digunakan data dari website www.bmkg.go.id/cuaca/prakiraan-cuaca-indonesia.bmkg. Untuk hasil pengujian komponen SQL parser dapat dilihat pada Tabel 6.3.

Tabel 6.3 Hasil Pengujian Fungsional Komponen SQL Parser

Butir Uji	Data	Keadaan Sekarang	Keadaan Setelah	Status
S01	Drop table table	Menampilkan seluruh daftar table	Menampilkan pesan error “Query denied”	OK
	insert	Menampilkan seluruh daftar table	Menampilkan pesan error “Query denied”	OK
	update	Menampilkan seluruh daftar table	Menampilkan pesan error “Query denied”	OK
	delete	Menampilkan seluruh daftar table	Menampilkan pesan error “Query denied”	OK
S02	Select *	Menampilkan seluruh daftar table	Menampilkan pesan error “Error on query structured”	OK
	Select * where	Menampilkan seluruh daftar table	Menampilkan pesan error “Error on query structured”	OK
S03	Select * from table_10	Menampilkan seluruh daftar table	Menampilkan pesan error “Table not found”	OK

6.1.1.4. Komponen Translator

Pada pengujian komponen translator data yang dipakai adalah data dari website www.bmkg.go.id/cuaca/prakiraan-cuaca-indonesia.bmkg. Berikut ini merupakan hasil pengujian dari komponen translator yang dijabarkan pada Tabel 6.4

Tabel 6.4 Hasil Pengujian Fungsional Komponen Translator

Butir Uji	Data	Keadaan Sekarang	Keadaan Setelah	Status
T01	Select * from table_0 order by kota	Menampilkan seluruh daftar table	Menampilkan data pada table_0 yang dilakukan order berdasarkan kolom kota secara Ascending	OK
	Select * from table_0 order by kota desc	Menampilkan seluruh daftar table	Menampilkan data pada table_0 yang dilakukan order berdasarkan kolom kota secara Descending	OK
	Select * from table_0 order by kota ASC	Menampilkan seluruh daftar table	Menampilkan data pada table_0 yang dilakukan order berdasarkan kolom kota secara Ascending	OK

Butir Uji	Data	Keadaan Sekarang	Keadaan Setelah	Status
	Select * from table_0 order by kota2	Menampilkan seluruh daftar table	Menampilkan pesan error karena kolom kota2 tidak ditemukan	OK
T02	select * from table_0 where prakiraan_cuaca_dini_hari = 'Hujan Lokal' or prakiraan_cuaca_dini_hari = 'Berawan'	Menampilkan seluruh daftar table	Menampilkan data pada table_0 yang memiliki prakiraan_cuaca_dini_hari adalah Hujan Lokal atau Berawan	OK
T03	select * from table_0 where prakiraan_cuaca_malam = 'Berawan' and prakiraan_cuaca_dini_hari = 'Berawan'	Menampilkan seluruh daftar table	Menampilkan data pada table_0 yang memiliki prakiraan_cuaca_malam adalah Berawan dan prakiraan_cuaca_dini_hari adalah Berawan	OK
T04	select * from table_1 where prakiraan_cuaca_pagi = 'Cerah Berawan' or prakiraan_cuaca_siang = 'Hujan Lokal'	Menampilkan seluruh daftar table	Menampilkan data pada table_1 yang memiliki prakiraan_cuaca_pagi adalah Cerah Berawan atau prakiraan_cuaca siang adalah	OK

Butir Uji	Data	Keadaan Sekarang	Keadaan Setelah	Status
	and prakiraan_cuaca_dini_hari = 'Berawan' order by kota		Hujan Lokal dan prakiraan_cuaca_dini_hari adalah Berawan yang diurutkan secara descending berdasarkan kota	
T05	Select * from table_0	Menampilkan seluruh daftar table dan nilai TTL belum expired	Menampilkan seluruh data pada table_0 dan tidak terdapat perubahan pada nilai TTL	OK
T06	Select * from table_0	Menampilkan seluruh daftar table tetapi nilai TTL sudah expired	Menampilkan seluruh data pada table_0 dan terdapat kenaikan pada nilai TTL sesuai dengan pendekatan yang dipakai	OK
T07	Select * from table_0	Menampilkan seluruh daftar table tetapi nilai TTL sudah expired	Menampilkan seluruh data pada table_0 dan nilai TTL menjadi nilai minimum 1	OK

6.1.2. Pengujian Tipe Kenaikan TTL

Pengujain tipe kenaikan TTL dilakukan sesuai dengan bagian 4.5.2. Untuk hasil pengujian pada interval pengambilan pertama dapat dilihat pada Tabel 6.5 dan untuk interval pengambilan kedua dapat dilihat pada Tabel 6.6.

Tabel 6.5 Hasil Pengujian Tipe Kenaikan TTL Interval Pertama

Website	Jumlah Data	Linear	Polynomial	Exponential
Nilai tukar mata uang	Outdated	19	19	19
	Pengambilan data baru yang terjadwal	14	14	14
	Outdated dan dilakukan pengambilan data baru	13	13	13
BMKG	Outdated	10	17	17
	Pengambilan data baru yang terjadwal	7	1	1
	Outdated dan dilakukan pengambilan data baru	3	0	0
Pengumuman	Outdated	0	0	0
	Pengambilan data	7	4	4

	baru yang terjadwal			
	Outdated dan dilakukan pengambilan data baru	0	0	0

Tabel 6.6 Hasil Pengujian Tipe Kenaikan TTL Interval Kedua

Website	Jumlah Data	Linear	Polynomial	Exponential
Nilai Tukar Mata Uang	Outdated	20	20	20
	Pengambilan data baru yang terjadwal	20	20	20
	Outdated dan dilakukan pengambilan data baru	20	20	20
BMKG	Outdated	15	18	18
	Pengambilan data baru yang terjadwal	15	6	6
	Outdated dan dilakukan pengambilan data baru	10	4	4
	Outdated	0	0	0

Pengumuman	Pengambilan data baru yang terjadwal	4	2	2
	Outdated dan dilakukan pengambilan data baru	0	0	0

Data outdated yang dimaksud adalah data yang dianggap berbeda dengan sumber website yang dilakukan query. Sedangkan untuk data outdated dan dilakukan pengambilan data baru adalah saat dimana data tersebut berstatus outdated dan terdapat pengambilan data baru yang dilakukan sistem pada sumber website yang ada.

Dari hasil pengujian yang dilakukan, didapatkan juga nilai TTL terakhir pada setiap pengujian yang dapat dilihat pada Tabel 6.7 untuk pengujian interval pertama dan Tabel 6.8 untuk pengujian interval kedua.

Tabel 6.7 Nilai TTL Terakhir Saat Pengujian Interval Pertama

Website	Linear	Polynomial	Exponential
Nilai Tukar Mata Uang	1.2	2	2
BMKG	1.2	2	2
Pengumuman	2,985984	3,244	4

Tabel 6.8 Nilai TTL Terakhir Pengujian Interval Kedua

Website	Linear	Polynomial	Exponential
Nilai Tukar Mata Uang	1	1	1
BMKG	1.2	1	1
Pengumuman	2,0736	2,5	3

6.2. Pembahasan

6.2.1. Analisa Hasil Pengujian Fungsional

Berdasarkan hasil pengujian blackbox testing pada bagian 6.2.1 dapat dirangkum seperti pada Tabel 6.9.

Tabel 6.9 Rangkuman Hasil Pengujian Fungsional

Komponen	Jumlah Pengujian	Jumlah Pengujian Berhasil	Jumlah Pengujian Gagal
Downloader	9	9	0
HTML Parser	7	7	0
SQL Parser	7	7	0
Translator	11	11	0

Dapat dilihat dari pengujian empat komponen sistem yaitu downloader, HTML parser, SQL parser dan translator seluruh pengujian yang dilakukan pada setiap komponen menunjukkan keberhasilan atau berjalan sesuai dengan apa yang diharapkan. Tidak terdapat juga pengujian yang mengalami kegagalan atau berjalan tidak sesuai harapan. Sehingga dapat disimpulkan untuk seluruh komponen yang dilakukan pengujian dapat berjalan dengan baik.

6.2.2. Analisa Hasil Pengujian Tipe Kenaikan TTL

Dari hasil pengujian yang dilakukan pada bagian 6.1.2 dapat dihitung nilai *false positive* dengan cara seperti pada persamaan (6.1).

$$\sum FP = \sum Ou - \sum Oupd \qquad \textbf{(6.1)}$$

Keterangan:
$FP = False\ Positive$
$Ou =$ data $Outdated$
$Oupd =$ Ou yang dilakukan pengambilan data baru

Sehingga nilai *false positive* pada interval pertama dan interval kedua sesuai dengan Tabel 6.10 dan Tabel 6.11.

Tabel 6.10 False Positive Interval Pertama

Website	Linear	Polynomial	Exponential
Nilai Tukar Mata Uang	6	6	6
BMKG	7	17	17
Pengumuman	0	0	0

Tabel 6.11 False Positive Interval Kedua

Website	Linear	Polynomial	Exponential
Nilai Tukar Mata Uang	0	0	0
BMKG	5	14	14
Pengumuman	0	0	0

Dari data pada Tabel 6.10 dan Tabel 6.11 dapat dihitung nilai *FPr* dengan persamaan (4.1) sehingga didapatkan nilai *FPr* seperti pada Tabel 6.12 dan Tabel 6.13.

Tabel 6.12 Nilai *False Positive Ratio (FPr) Interval Pertama*

Website	Linear	Polynomial	Exponential
Nilai Tukar Mata Uang	0,3	0,3	0,3
BMKG	0,35	0,85	0,85
Pengumuman	0	0	0

Tabel 6.13 Nilai *False Positive Ratio (FPr)* Interval Kedua

Website	Linear	Polynomial	Exponential
Nilai Tukar Mata Uang	0	0	0
BMKG	0,25	0,7	0,7

Pengumuman	0	0	0

Didapatkan juga nilai *STr* dari kedua pengujian pada bagian 6.1.2 dengan menggunakan persamaan (4.2) sehingga dihasilkan nilai *STr* seperti pada Tabel 6.14.

Tabel 6.14 Nilai *Stale Trafic Ratio (STr)*

Website	***STr* Interval Pertama**			***STr* Interval Kedua**		
	Linear	**Polynomial**	**Exponensial**	**Linear**	**Polynomial**	**Exponensial**
Nilai Tukar Mata Uang	0,7	0,7	0,7	1	1	1
BMKG	0,35	0,05	0,05	0,75	0,3	0,3
Pengumuman	0,35	0,2	0,2	0,2	0,1	0,1

Dibutuhkannya nilai *FPr* dan *STr* untuk mengetahui tingkat kesalahan dibandingkan dengan seberapa banyak frekuensi waktu pengambilan data baru yang terjadwalkan. Jika terlalu cepat akan mengakibatkan komputasi *overhead*. Sedangkan jika frekuensi waktu pembaharuan terlalu lama, maka data yang tersimpan akan usang. Semakin rendah nilai *FPr* dan *STr* maka akan semakin bagus karena sistem mampu mendeteksi kapan waktu yang tepat dalam melakukan pengambilan data dengan frekuensi pembaharuan data baru yang sedikit.

Tabel 6.15 Nilai FPr dan STr Website Nilai Tukar Mata Uang

Website nilai tukar mata uang	Interval Pertama		Interval Kedua	
	FPr	STr	FPr	STr
Linear	0,3	0,7	0	1

Polynomial	0,3	0,7	0	1
Exponential	0,3	0,7	0	1

Dari perbandingan nilai *FPr* dan *STr* pada website nilai tukar mata uang dapat dilihat pada Tabel 6.15. Nilai pengujian pada interval pertama dan interval kedua memiliki nilai yang sama untuk seluruh tipe kenaikan TTL. Sehingga dapat dikatakan untuk website dengan nilai tukar mata uang yang merepresentasikan website dengan perubahan data yang cepat dapat digunakan semua tipe kenaikan TTL yang ada. Tetapi jika dibutuhkan variabel lain maka dapat dilihat nilai TTL terakhir pada setiap pengujian yang dapat dilihat pada Tabel 6.7 dan Tabel 6.8. Dapat dilihat bahwa nilai terakhir TTL terakhir yang paling kecil pada kenaikan TTL linear. Karena perubahan data pada website nilai tukar mata uang cepat, sehingga nilai TTL terakhir yang paling baik adalah yang paling kecil karena akan memungkinkan nilai *false positive* yang lebih sedikit.

Tabel 6.16 Nilai FPr dan STr Website BMKG

Website BMKG	Interval Pertama		Interval Kedua	
	FPr	STr	FPr	STr
Linear	0,35	0,35	0,25	0,7
Polynomial	0,85	0,05	0,7	0,3
Exponential	0,85	0,05	0,7	0,3

Sedangkan untuk website BMKG yang memiliki perubahan sedang dapat dilihat nilai perbandingan *FPr* dan *STr* pada Tabel 6.16. Dari tabel tersebut terlihat bahwa nilai *FPr* untuk kenaikan TTL linear lebih kecil pada interval pengujian pertama dan kedua yang menunjukkan hasil yang lebih baik (sedikit terjadinya *false positive*) daripada kedua tipe kenaikan TTL yang lain. Meskipun memiliki nilai *STr* yang lebih besar daripada tipe kenaikan TTL lain data yang didapatkan cenderung lebih akurat atau lebih sedikit terjadinya *false positive* dalam melakukan pengambilan data. Sehingga dapat dikatan bahwa untuk website BMKG yang merepresentasikan

website dengan perubahan sedang lebih baik mengguakan tipe kenaikan TTL linear.

Tabel 6.17 Nilai FPr dan STr Website Pengumuman

Website Pengumuman	Interval Pertama		Interval Kedua	
	FPr	STr	FPr	STr
Linear	0	0,35	0	0,2
Polynomial	0	0,2	0	0,1
Exponential	0	0,2	0	0,1

Untuk perbandingan *FPr* dan *STr* pada website pengumuman dapat dilihat pada Tabel 6.17. Pada tabel tersebut nilai *FPr* untuk semua tipe kenaikan TTL sama yaitu 0. Karena tidak terdapat nilai *false positive* dikarenakan data pada website pengumuman tidak pernah mengalami berubahan sehingga membuat data yang tersimpan didalam sistem selalu sama dengan data pada website. Sehingga hanya menyisakan nilai *STr* yang menjadi indikator dalam menentukan mana tipe kenaikan TTL terbaik. Dari nilai *STr* yang ada, terlihat bahwa nilai *STr* pada tipe kenaikan TTL polynomial dan exponential merupakan yang paling kecil. Karena kedua tipe kenaikan TTL tersebut memiliki nilai penambahan yang paling besar sehingga membuat pengambilan data baru yang terjadwalkan pun semakin sedikit. Karena memiliki nilai yang sama, untuk mencari yang terbaik kita bisa melihat nilai TTL terakhir yang dapat dilihat pada Tabel 6.7 dan Tabel 6.8. Terlihat dari kedua tabel tersebut nilai TTL terakhir untuk tipe kenaikan TTL exponential paling besar. Sehingga dapat dikatan bahwa tipe kenaikan TTL exponential lebih jarang terjadi pengambilan data baru yang terjadwal karena nilai TTL lebih besar. Sehingga untuk website dengan data yang jarang berubah lebih baik menggunakan tipe kenaikan TTL exponential karena pengambilan data baru yang terjadwalkan paling sedikit dan data yang diambil selalu sama.

BAB VII
KESIMPULAN DAN SARAN

Pada bab ini dibahas mengenai kesimpulan dari semua proses yang telah dilakukan dan saran yang dapat diberikan untuk pengembangan yang lebih baik.

7.1. Kesimpulan

Kesimpulan yang didapatkan dari proses pengerjaan tugas akhir yang telah dilakukan antara lain:

1. Melakukan ekstraksi data dapat dilakukan dengan cara melakukan pengunduhan *source code* website dan melakukan *parsing* terhadap *source code* tersebut untuk didapatkan data yang diinginkan.
2. Melakukan query data yang tidak terstruktur dengan cara merubah data yang tidak terstruktur menjadi data yang terstruktur dan melakukan penyimpanan data tersebut kedalam sebuah tabel.
3. Mendeteksi data numerik yang tepat dapat dilakukan dengan cara mendeteksi apakah terdapat tanda baca titik (.) dalam sebuah data numerik atau tidak. Jika terdapat tanda baca tersebut maka dideteksi data numerik tersebut bertipe *float* jika tidak maka data numerik tersebut bertipe *integer*.
4. Memperbarui data yang tersimpan dengan sumber data dapat dilakukan dengan cara melakukan penjadwalan pengambilan data baru dari sumber sesuai dengan nilai TTL yang ada.
5. Untuk website dengan tingkat perubahan data yang cepat (berubah dalam hitungan menit atau kurang) dan dilakukan pengambilan data dengan interval yang cepat dapat menggunakan tipe kenaikan TTL linear.
6. Untuk website dengan tingkat perubahan data yang cepat (berubah dalam hitungan menit atau kurang) dan dilakukan pengambilan data dengan interval jauh lebih besar dari waktu perubahan sumber data maka dapat menggunakan semua tipe kenaikan TTL.

7. Untuk website dengan tingkat perubahan data yang sedang (berubah dalam hitungan jam) lebih baik menggunakan tipe kenaikan TTL linear.
8. Untuk website dengan tingkat perubahan data yang lambat atau tidak pernah berubah (berubah dalam hitungan bulan atau tahun) lebih baik menggunakan tipe kenaikan TTL Exponensial.

7.2. Saran

Berdasarkan hasil pengerjaan tugas akhir ini, maka saran yang dapat penulis berikan untuk penelitian selanjutnya antara lain:

1. Mengembangkan aplikasi agar dapat melakukan join antar tabel dan *nested query* sehingga pengguna dapat mengolah data didalam aplikasi ini dan mendapatkan data yang telah jadi.
2. Melakukan pengujian dengan frekuensi yang lebih banyak agar lebih terlihat perbedaan dari setiap tipe kenaikan TTL mana yang lebih baik untuk perubahan frekuensi data yang cepat, sedang dan jarang terjadi perubahan.
3. Mencari variable baru yang dapat digunakan dalam menentukan kapan data harus dilakukan pembaharuan dari sumber data.

DAFTAR PUSTAKA

[1] Y. H. Widiartanto, "Pengguna Internet di Indonesia Capai 132 Juta," Kompas, 2016. [Online]. Available: http://tekno.kompas.com/read/2016/10/24/15064727/2016.pengguna.internet.di.indonesia.capai.132.juta. [Accessed 22 Oktober 2017].

[2] Netcraft, "Web Server Survey," Netcraft, Oktober 2017. [Online]. Available: https://news.netcraft.com/archives/category/web-server-survey/. [Accessed 10 November 2017].

[3] A. G. O, "WebSQL: Exploiting Document Structure in Web Queries,," University of Toronto, Toronto, 1997.

[4] D. Seminger and A. Saxton, "Analyzing Web page data using Power BI Dekstop (tutorial)," Microsoft, 12 Juni 2017. [Online]. Available: https://docs.microsoft.com/en-us/power-bi/desktop-tutorial-importing-and-analyzing-data-from-a-web-page. [Accessed 24 Januari 2018].

[5] Microsoft, "Connect to a Web Page (Power Query)," Microsoft, [Online]. Available: https://support.office.com/en-us/article/connect-to-a-web-page-power-query-b2725d67-c9e8-43e6-a590-c0a175bd64d8. [Accessed 31 Januari 2018].

[6] M. J. Collins, "A New Statistical Parser Based on Bigram Lexical Dependencies," in *Annual Meeting of Association for Computational Linguistics*, 1996.

[7] W3C, "Document Object Model (DOM)," W3C, 2005. [Online]. Available: https://www.w3.org/DOM/. [Accessed 20 November 2017].

[8] P. M. Nadkarni, Metadata-driven Software Systems in Biomedicine, lONDON: Springer-verlag London limited, 2011.

[9] D. Crockfor, “The Application/JSON Media Type for JavaScript Object Notation (JSON),” Juli 2006. [Online]. Available: JSON.org.

[10] H.-P. Halvorsen, “Structured Query Language,” University College of Southeast Norway, Agustus 2016. [Online]. Available: http://home.hit.no/~hansha/.

[11] D. D. R. F. B. Chamberlin, “SEQUEL: A Structured English Query Language,” in *ACM SIGFIDET 1974 workshop on data description*, New York , 1974.

[12] G. Mulligan and D. Gracanin, “A Comparison of SOAP and REST Implementations of a Service Based Interaction Independence Middleware Framework,” in *WCS ’09 Winter Simulation Conference*, Texas, 2009.

[13] J. Espi-Beltran, “Enabling Distriuted Manufacturing Resources through SOA: The REST Approach,” *Robotics and Computer–Integrated Manufacturing*, vol. 46, no. 15, pp. 156-165, 2017.

[14] S. Alici and e. a. , “Adaptive Time-to-Live Strategies for Query Result Caching in Web Search Engines,” in *European Conference on Information Retrieval (ECIR) 2012 : Advance in Information Retrieval*, 2012, 2012.

[15] R. S. Pressman, Software Engineering A Practiotioner’s Approach, 7th ed., New York: McGraw-Hill, 2010.

[16] PostgreSQL, “The Parser Stage,” The PostgreSQL Global Development Group, 1996-2018. [Online]. Available: https://www.postgresql.org/docs/9.1/static/parser-stage.html#AEN84439. [Accessed 15 Januari 2018].

BIODATA PENULIS

Penulis bernama lengkap Ahsanul Khuluq Saifullah, lahir di Surabaya pada tanggal 29 Juli 1996. Merupakan anak terakhir dari 3 bersaudara. Penulis telah menempuh beberapa Pendidikan formal yaitu: SD Muhammadiyah 24 Surabaya, SMP Negeri 12 Surabaya, dan SMA Negeri 15 Surabaya.

Pada tahun 2014 pasca kelulusan SMA, penulis melanjutkan pendidikan dengan jalur Seleksi Bersama Masuk Perguruan Tinggi Negeri (SBMPTN) di Jurusan Sistem Informasi FTIK – Institut Teknologi Sepuluh Nopember (ITS) Surabaya dan terdaftar sebagai mahasiswa dengan NRP 05211440000106. Selama menjadi mahasiswa, penulis mengikuti berbagai kegiatan kemahasiswaan diantaranya beberapa kepanitiaan Integralistik Festival dan menjadi ketua pada Information System Expo 2016. Penulis juga pernah menjadi staf *Enterpreneurship* pada Badan Eksekutif Mahasiswa Fakultas Teknologi Informasi ITS pada tahun 2015 – 2016. Di bidang akademik, penulis aktif menjadi asisten dosen pada matakuliah Pemrograman Berbasis Web.

Penulis memiliki ketertarikan di bidang Akuisisi Data dan Desiminasi Informasi (ADDI) sehingga memilih topik *Database Management System* pada tugas akhir. Penulis dapat dihubungi melalui *email* pada ahsanulkh996@gmail.com

Halaman ini sengaja dikosongkan

Lampiran A

Hasil Pengujian Interval Pertama

Nilai tukar mata uaf	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	ttl akhir
Linear	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	sama	beda	beda	beda	beda	beda	beda	1.2
Polynomial	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	sama	beda	beda	beda	beda	beda	beda	2
Exponential	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	sama	beda	beda	beda	beda	beda	beda	2

BMKG	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	ttl akhir
Linear	sama	sama	sama	beda	beda	beda	sama	sama	sama	beda	beda	beda	sama	sama	beda	beda	beda	beda	sama	sama	1.2
Polynomial	sama	sama	sama	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	2
Exponential	sama	sama	sama	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	2

Pengumuman	1	2	3	4	5	6	7	8	9	10	ttl akhir
Linear	sama	sama	sama	sama	sama	sama	sama	sama	sama	sama	2,98598
Polynomial	sama	sama	sama	sama	sama	sama	sama	sama	sama	sama	3.244
Exponential	sama	sama	sama	sama	sama	sama	sama	sama	sama	sama	4

Pengambilan Data Baru

Lampiran B

Hasil Pengujian Interval Kedua

Nilai tukar mata uar	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	ttl akhir
Linear	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	1
Polynomial	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	1
Exponential	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	beda	1

BMKG	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	ttl akhir
Linear	sama	beda	beda	beda	sama	beda	beda	beda	beda	beda	sama	beda	beda	beda	sama	beda	beda	beda	sama	beda	1.2
Polynomial	sama	beda	beda	beda	beda	beda	beda	beda	beda	beda	sama	beda	beda	beda	beda	beda	beda	beda	beda	beda	1
Exponential	sama	beda	beda	beda	beda	beda	beda	beda	beda	beda	sama	beda	beda	beda	beda	beda	beda	beda	beda	beda	1

Pengumuman	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	ttl akhir
Linear	sama	sama	sama	sama	sama	sama	sama	sama	sama	sama	sama	sama	sama	sama	sama	sama	sama	sama	sama	sama	2
Polynomial	sama	sama	sama	sama	sama	sama	sama	sama	sama	sama	sama	sama	sama	sama	sama	sama	sama	sama	sama	sama	2.5
Exponential	sama	sama	sama	sama	sama	sama	sama	sama	sama	sama	sama	sama	sama	sama	sama	sama	sama	sama	sama	sama	3

	Pengambilan Data Baru