SKRIPSI

PERANCANGAN SISTEM PENCARIAN BERBASIS *FULL-TEXT SEARCH ENGINE* DENGAN *ELASTICSEARCH* STUDI KASUS: WEB DIGITAL *LIBRARY* FAKULTAS TEKNIK UNIVERSITAS HASANUDDIN

Disusun dan diajukan oleh

RAHMAT FIRMAN

D42114018

DEPARTEMEN TEKNIK INFORMATIKA

FAKULTAS TEKNIK

UNIVERSITAS HASANUDDIN

MAKASSAR

2021

LEMBAR PENGESAHAN SKRIPSI

PERANCANGAN SISTEM PENCARIAN BERBASIS *FULL-TEXT SEARCH ENGINE* DENGAN *ELASTICSEARCH*. STUDI KASUS: WEB DIGITAL LIBRARY FAKULTAS TEKNIK UNIVERSITAS HASANUDDIN

Disusun dan diajukan oleh

RAHMAT FIRMAN

D42114018

Telah dipertahankan di hadapan Panitia Ujian yang dibentuk dalam rangka Penyelesaian Studi Program Sarjana Program Studi Teknik Informatika Fakultas Teknik Universitas Hasanuddin pada tanggal 03 Agustus 2021 dan dinyatakan telah memenuhi syarat kelulusan.

Menyetujui,

Pembimbing Utama,	Pembimbing Pendamping,
Dr. Amil Ahmad Ilham, ST.,M.IT	Anugrayani Bustamin, S.T., M.T
Nip. 19731010 199802 1 001	Nip. 19901201 201807 4 001

Ketua Program Studi,

Dr. Amil Ahmad Ilham, S.T., M.IT
Nip. 19731010 199802 1 001

ABSTRAK

Perkembangan teknologi informasi yang pesat baik dari segi ukuran data maupun jenisnya membuat kebutuhan akan pengelolaan informasi juga ikut meningkat. Tidak terkecuali dalam dunia pendidikan misalnya Perguruan Tinggi. Ratusan bahkan ribuan karya ilmiah yang dihasilkan oleh mahasiswa maupun tenaga pendidik tiap tahunnya. Namun tidak banyak dari karya ilmiah tersebut berhasil diarsipkan secara digital. Sehingga dibutuhkan sebuah sistem yang dapat mengelola data tersebut agar dapat ditemukan secara cepat dalam hal ini adalah Sistem *Digital Library* dengan konsep pencarian data menggunakan *Full-Text Search Engine* dengan *Elasticsearch*. Untuk menguji sebuah sistem pencarian berbasis *Full-Text search* dengan *Elasticsearch,* pada penelitian ini digunakan metode *Black Box* untuk pengujian fungsional sistem dan *Precision*, *Recall*, *F-Measure* serta *Search Response Time* untuk pengujian kinerja sistem. Penggunaan kata kunci yang beragam dan *analyzer elasticsearch* dalam penelitian ini juga diharapkan dapat menambah tingkat akurasi dari hasil pengujian yang dilakukan. Dalam penerapannya, konsep pencarian data berbasis *Full-Text Search* dengan *Elasticsearch* dapat diterapkan dalam sistem *digital library* dan dipadukan dengan *relational database* seperti *MySQL* untuk mempermudah proses *management data.* Sistem yang dibangun terbukti efektif dalam melakukan pencarian data, hal ini dibuktikan dari hasil pengujian dengan perolehan nilai *F-Measure* diatas 80%.

Kata Kunci : *Search Engine*, *Elasticsearch*, *Full-Text Search*, *Digital Library*, *Management Data*

PERNYATAAN KEASLIAN

Yang bertanda tangan dibawah ini:

Nama	: RAHMAT FIRMAN
NIM	: D42114018
Program Studi	: TEKNIK INFORMATIKA
Jenjang	: S1/~~S2/S3~~

Menyatakan dengan ini bahwa karya tulisan saya berjudul

PERANCANGAN SISTEM PENCARIAN BERBASIS *FULL-TEXT SEARCH ENGINE* DENGAN *ELASTICSEARCH*. STUDI KASUS: WEB DIGITAL *LIBRARY* FAKULTAS TEKNIK UNIVERSITAS HASANUDDIN

Adalah karya tulis saya sendiri dan bukan merupakan pengambilan alihan tulisan orang lain bahwa skripsi yang saya tulis ini benar benar merupakan hasil karya saya sendiri.

Apabila dikemudian hari terbukti atau dapat dibuktikan bahwa sebagian atau keseluruhan skripsi ini hasil karya orang lain, maka saya bersedia menerima sanksi atas perbuatan tersebut.

Gowa, 11 Agustus 2021

Yang Menyatakan

Rahmat Firman

KATA PENGANTAR

Assalamu'alaikum Warahmatullahi Wabarakatuh

Segala puji dan syukur Penulis panjatkan ke hadirat Allah S.W.T, Tuhan Yang Maha Esa yang dengan limpahan rahmat dan hidayah-Nya sehingga tugas akhir dengan judul "*Perancangan Sistem Pencarian Berbasis Full-Text Search Engine dengan Elasticsearch. Studi Kasus: Web Digital Library Fakultas Teknik Universitas Hasanuddin"* ini dapat diselesaikan sebagai salah satu syarat dalam menyelesaikan jenjang Strata-1 pada Departemen Teknik Informatika Fakultas Teknik Universitas Hasanuddin. Selanjutnya *shalawat* serta salam Penulis panjatkan pula kepada sang revolusioner sejati, Nabi Muhammad S.A.W sebagai sosok teladan Penulis yang membawa zaman dari zaman jahiliyah ke zaman yang *In Shaa Allah* terang benderang ini, *aamiin*.

Dalam penyusunan penelitian ini disajikan hasil penelitian terkait judul yang telah diangkat dan telah melalui proses pencarian dari berbagai sumber baik jurnal penelitian, prosiding pada seminar-seminar nasional maupun internasional, buku, dan dari berbagai situs-situs terpercaya yang ada di internet.

Penulis menyadari bahwa tanpa bantuan dan bimbingan dari berbagai pihak, mulai dari masa perkuliahan sampai dengan penyusunan tugas akhir ini, sangatlah sulit untuk menyelesaikan tugas akhir ini. Oleh karena itu, pada kesempatan ini Penulis menyampaikan ucapan terima kasih sedalam-dalamnya kepada:

1) Allah S.W.T karena atas semua berkat, karunia serta pertolongan-Nya yang tiada batas, yang telah diberikan kepada Penulis di setiap langkah dalam pembuatan tugas akhir ini hingga penulisan laporan skripsi ini;

2) Kedua orang tua Penulis, Bapak Firman dan Alm. Ibu Hanisa yang selalu mendidik Penulis dan menjadi tempat untuk berkeluh kesah serta selalu memberikan dukungan, doa dan semangat kepada Penulis sejak kecil. Terima kasih untuk selalu ada untuk Penulis;
3) Bapak Dr. Amil Ahmad Ilham, ST., M.IT., selaku pembimbing I sekaligus Ketua Departemen Teknik Informatika dan Ibu Anugrayani Bustamin, S.T., M.T selaku pembimbing II yang selalu menyediakan waktu, tenaga, pikiran dan perhatian yang luar biasa untuk mengarahkan Penulis dalam penyusunan tugas akhir ini;
4) Bapak Robert, Bapak Zainuddin dan Ibu Yuanita serta segenap staff Departemen Teknik Informatika Fakultas Teknik Universitas Hasanuddin yang telah membantu kelancaran penyelesaian tugas akhir Penulis;
5) Saudara seperjuangan Penulis Teknik 2014 yang telah menemani perjalanan penulis dalam dunia pengkaderan sebagai anak teknik;
6) Keluarga besar Rectifier'14 yang juga menjadi rekan sekaligus tempat berbagi keluh dan kesah selama mengarungi dunia kampus sebagai anak teknik;
7) Teman-teman Adhyaksa khususnya Alam, Yakip, Syarif, Wawan, Andi, Jawa, Jaka, Ucup, Nono, Ulfah, Al, Abdi, dan segenap teman-teman Adhiyaksa yang selalu menemani Penulis dalam mengarungi segala rintangan dalam dunia perkuliahan sekaligus rumah untuk kembali pulang;

8) Seluruh pihak yang tidak sempat disebutkan satu persatu yang telah banyak meluangkan tenaga, waktu dan pikiran selama penyusunan laporan tugas akhir ini.

Wassalam

Gowa, 11 Agustus 2021

Penulis

DAFTAR ISI

	Halaman
HALAMAN JUDUL	i
LEMBAR PENGESAHAN	ii
ABSTRAK	iii
KATA PENGANTAR	iv
DAFTAR ISI	vii
DAFTAR GAMBAR	ix
DAFTAR TABEL	xi
BAB I PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Rumusan Masalah	4
1.3 Tujuan Penelitian	4
1.4 Batasan Masalah	4
1.5 Manfaat Penelitian	5
1.6 Sistematika Penulisan	5
BAB II TINJAUAN PUSTAKA	7
2.1 Konsep Dasar Sistem	7
2.2 Konsep Dasar Informasi	11
2.3 Digital *Library*	14
2.4 *Search Engine*	17
2.5 *Full-Text Search*	18
2.6 *Unstructured Data*	19

2.7 *Elasticsearch* 20

2.8 *Black Box Testing* 34

2.7 *Recall* dan *Precision* 35

BAB III METODOLOGI PENELITIAN 37

3.1 Tempat dan Waktu Penelitian 37

3.2 Instrumen Penelitian 37

3.3 Prosedur Penelitian 38

3.4 Gambaran Umum Sistem 43

3.5 Metode Pengujian 59

BAB IV HASIL DAN PEMBAHASAN 61

4.1 Hasil Pembuatan Sistem 61

4.2 Hasil Pengujian Sistem 62

BAB V PENUTUP 74

5.1. Kesimpulan 74

5.2. Saran 75

DAFTAR PUSTAKA 76

LAMPIRAN 80

DAFTAR GAMBAR

Gambar 2.1 Proses *Elasticsearch* 25

Gambar 2.2 Proses Komunikasi *elasticsearch* 26

Gambar 2.3 Proses *Indexing* 27

Gambar 2.4 Proses *Querying* 29

Gambar 2.5 Proses *Analyzer* dalam *Elasticsearch* 33

Gambar 3.1 Bagan Tahapan Penelitian 39

Gambar 3.2 Gambaran Umum Sistem 44

Gambar 3.3 Alur Proses *Searching* 47

Gambar 3.4 *Use Case Diagram System* 49

Gambar 3.5 *Activity Diagram* Input Data Literatur 51

Gambar 3.6 *Activity Diagram* Pencarian Data Literatur 52

Gambar 3.7 *Activity Diagram* Submit Data Tugas Akhir 53

Gambar 3.8 *Entity Relationship Diagram* 54

Gambar 3.9 *Default Setting Elasticsearch Node and Shard* 56

Gambar 3.10 Proses Pengambilan Data *Search Response Time* 60

Gambar 4.1 Halaman Pencarian Data Literatur 61

Gambar 4.2 Halaman Dashboard *Content Management System (CMS)* 62

Gambar 4.3 Grafik Hasil Pengujian *Search Response Time* 69

Gambar 4.4 Grafik Hasil Pengujian *Precision Standard Analyzer* dan *Custom Analyzer* 70

Gambar 4.5 Grafik Hasil Pengujian *Recall Standard Analyzer* dan *Custom Analyzer* 71

Gambar 4.6 Grafik Hasil Pengujian *F-Measure Standard Analyzer* dan *Custom Analyzer* 73

DAFTAR TABEL

Tabel 2.1 Model Data pada *Inverted Index Elasticsearch* 32

Tabel 2.2 Matriks *Recall* dan *Precision Lancaster* 36

Tabel 3.1 Daftar Kata Kunci 40

Tabel 3.2 Akuisisi Data Jurnal Ilmiah 41

Tabel 3.3 Perbandingan *Standard Analyzer* dan *Custom Analyzer* 57

Tabel 3.4 Perbandingan Hasil Pemrosesan *Standard Analyzer* dan *Custom Analyzer* 58

Tabel 4.1 Hasil Pengujian *Black Box Testing Login CMS* 62

Tabel 4.2 Hasil Pengujian *Black Box Testing add User CMS* 63

Tabel 4.3 Hasil Pengujian *Black Box Testing edit User CMS* 64

Tabel 4.4 Hasil Pengujian *Black Box Testing add Author/Writer CMS* 65

Tabel 4.5 Hasil Pengujian *Black Box Testing edit Author/Writer CMS* 66

Tabel 4.6 Hasil Pengujian *Black Box Testing add Journal CMS* 66

Tabel 4.7 Hasil Pengujian *Black Box Testing edit Journal CMS* 67

Tabel 4.8 Hasil Pengujian *Black Box Testing Login* Mahasiswa 67

Tabel 4.9 Hasil Pengujian *Black Box Testing* Submit Tugas Akhir 68

Tabel 4.10 Daftar Dokumen Relevan tidak Terambil 72

BAB I

PENDAHULUAN

1.1 Latar Belakang

Perkembangan teknologi dan informasi di internet sangat pesat. Didukung dengan perkembangan perangkat keras, *Google*, *Bing* dan *Yahoo* secara konsisten dilaporkan sebagai *website* yang digunakan sebagai mesin pencari. Mereka menguasai pasar pencarian data sebanyak 96%. *Google* menguasai 66,9%, *Bing* 18,1% dan *Yahoo* 11,1% (Shiri, 2014).

Di era teknologi yang serba canggih inilah banyak ditawarkan berbagai macam kemudahan. Termasuk dalam menemukan informasi yang diinginkan, di era ini hampir semua informasi dapat ditemukan dengan mudah di internet. Hal ini juga didukung dengan perkembangan *search engine* yang semakin canggih.

Selain itu di dunia pendidikan sekarang ini banyak tersedia buku-buku, artikel maupun hasil karya mahasiswa atau tenaga pengajar baik dalam bentuk fisik maupun elektronik. Jumlah data literatur yang dari tahun ke tahun semakin meningkat memerlukan penataan yang baik agar data tersebut mudah ditemukan terlebih lagi ketika data yang berbentuk fisik tersebut disimpan pada tempat yang sama. Data-data tersebut tentu juga memerlukan media penyimpanan yang besar (Novitasari, 2015).

Dalam rangka melaksanakan Tridharma Perguruan Tinggi (Pendidikan, Penelitian dan Pengabdian Masyarakat), institusi pendidikan telah banyak memanfaatkan teknologi informasi. Salah satu pemanfaatannya yaitu pengelolaan data literatur karya ilmiah pada institusi pendidikan seperti sistem digital *library*.

Sistem digital *library* ini dapat membantu dalam hal pengelolaan data atau dokumen agar lebih terstruktur dengan baik dan data dapat diakses dengan mudah dan cepat.

Salah satu bagian penting dalam pengelolaan data literatur karya ilmiah adalah proses pencarian informasi yang diinginkan oleh pengguna atau biasa disebut dengan temu kembali informasi (*information retrieval*). Proses pencarian informasi ini disesuaikan dengan kriteria data literatur baik buku, artikel maupun karya ilmiah.

Seiring dengan semakin banyaknya jumlah data literatur, begitupun dengan jenisnya. Maka pencarian data pun menjadi tantangan baru dalam pengelolaan data literatur. Untuk mengolah dan menemukan data literatur, dibutuhkan metode pencarian data yang tepat yang dapat memudahkan dalam mendapatkan referensi yang dibutuhkan, bukan hanya dari segi performa yang belakangan ini marak diperbincangkan dalam dunia pengembangan website tapi juga dari sisi keakuratan.

Berdasarkan penelitian terdahulu dari Aisyah (2017), pencarian dengan *system keyword matching on content* pada website digital *library* memungkinkan untuk diterapkan dengan menggunakan *Apache Solr.* Selain *Apache Solr*, platform *search engine full-text search* lainnya yang tidak kalah populer adalah *elasticsearch.* Keduanya sama-sama dibangun dari *Apache Lucene* namun berbeda perusahaan pengembang.

Ditinjau dari segi fitur yang disediakan, baik *Apache Solr* maupun *elasticsearch* sama-sama memiliki keunggulan masing-masing. Misalkan dari segi kemudahan instalasi, *elasticsearch* lebih mudah diimplementasikan dibandingkan

dengan *Apache Solr*. Dari segi format *response API*, *Apache Solr* menyediakan format *JSON* maupun *XML*, sedangkan *elasticsearch* hanya menyediakan format *JSON*. Dari segi popularitas, *elasticsearch* menempati posisi pertama sedangkan *Apache Solr* berada di posisi ketiga, hal ini dikarenakan *elasticsearch* lebih mudah dalam proses instalasi dan perawatan sistem, serta dukungan API yang lengkap. (Rafal. 2019).

Penelitian lainnya dilakukan oleh Novitasari (2015), dengan memanfaatkan *elasticsearch* pada sistem temu kembali informasi data literatur ilmiah. Dalam penelitiannya, *elasticsearch* dikombinasikan dengan *framework Hadoop* untuk penyimpanan dan pemrosesan file literatur ilmiah secara terdistribusi. Sebelum dimasukkan ke dalam *elasticsearch,* data literatur terlebih dahulu dilakukan *preprocessing* seperti pembersihan data dan *filtering* data. Proses pengujian melibatkan beberapa skenario dengan jumlah data dan keyword yang berbeda-beda. Sistem yang dibangun memiliki fitur yang sederhana untuk melakukan pencarian data literatur, hanya ada *field input keyword* dan tombol *search* serta pencarian dikhususkan untuk literatur berbahasa Indonesia. Hasil pencarian dari sistem berupa nama file literatur ilmiah beserta lokasi file yang disimpan dalam sistem hadoop.

Berdasarkan permasalahan di atas, penulis tertarik untuk mengimplementasikan sistem pencarian berbasis *Full-text Search Engine* dengan *elasticsearch* pada website digital *library* Fakultas Teknik dan menjadikan tugas akhir untuk dapat menyelesaikan studi pada Universitas Hasanuddin Makassar, dengan judul **"Perancangan Sistem Pencarian Berbasis *Full-Text Search***

***Engine* dengan *Elasticsearch*. Studi Kasus: Web Digital *Library* Fakultas Teknik Universitas Hasanuddin".**

1.2 Rumusan Masalah

Berdasarkan latar belakang di atas, penulis merumuskan permasalahan yaitu:

1. Bagaimana penerapan sistem pencarian berbasis *full-text search engine* dengan *elasticsearch* pada sistem website digital *library* ?
2. Bagaimana kinerja dari pengimplementasian sistem pencarian berbasis *full-text search engine* dengan *elasticsearch* pada sistem website digital *library*?

1.3 Tujuan Penelitian

Tujuan akhir dari penelitian ini yaitu:

1. Untuk menerapkan sistem pencarian berbasis *full-text search engine* dengan *elasticsearch* pada sistem website digital *library*.
2. Untuk meningkatkan kecepatan dan tingkat akurasi dalam pencarian informasi literatur pada sistem website digital *library*.

1.4 Batasan Masalah

1. Data jurnal yang digunakan dalam penelitian ini diambil dari website digital *library IEEE Xplore* dan hanya diperuntukkan sebagai data untuk pengujian sistem.
2. Atribut data jurnal yang digunakan untuk proses pencarian adalah judul dan abstrak.

3. Konfigurasi jumlah *node* dan *shard* pada *elasticsearch* dalam penelitian ini menggunakan *default setting* dengan jumlah *node* 1 dan *shard* sebanyak 5.

1.5 Manfaat Penelitian

Dengan dilakukannya penelitian ini, diharapkan dengan sistem pencarian berbasis *full-text search engine* dengan *elasticsearch* pada website digital *library* pengguna dapat dengan mudah mendapatkan literatur secara cepat dan sesuai dengan yang diinginkan.

1.6 Sistematika Penulisan

Laporan Tugas Akhir ini dibagi dalam lima bab yang tersusun secara sistematis, yaitu:

BAB I PENDAHULUAN

Bab ini menjelaskan tentang latar belakang pengambilan topik *full-text search engine* pada sistem website digital *library*, perumusan masalah, tujuan penelitian, batasan masalah, metode penulisan dan sistematika penulisan.

BAB II LANDASAN TEORI

Bab ini akan membahas mengenai penjelasan-penjelasan umum mengenai teori-teori yang berkaitan seperti konsep dasar dan pengertian tentang sistem *full-text search engine* pada website digital *library* yang mendukung terbentuknya suatu sistem *full-text search engine* untuk *library* yang berbasis web.

BAB III METODOLOGI PENELITIAN

Dalam bab ini membahas mengenai metode yang akan digunakan dalam penelitian, seperti prosedur penelitian, metode pengambilan data dan metode pengujian data yang akan digunakan.

BAB IV HASIL DAN PEMBAHASAN

Bab ini berisi tentang hasil penerapan sistem pencarian *full-text search* pada website *digital library*, hasil pengujian sistem dan pembahasan dari hasil pengujian yang dilakukan.

BAB V PENUTUP

Bab ini berisi tentang kesimpulan dan saran dari penelitian yang dilakukan serta saran-saran dan masukan untuk penelitian selanjutnya.

BAB II

TINJAUAN PUSTAKA

Dalam bab ini akan dibahas mengenai teori-teori yang dapat penunjang dan menjadi acuan dalam penelitian ini.

2.1 Konsep Dasar Sistem

Tata Sutabri (2012) dalam bukunya yang berjudul *Analisis Sistem Informasi* menjelaskan bahwa pada dasarnya sistem itu terdiri dari beberapa unsur atau variabel terorganisir yang saling berinteraksi antara satu dengan yang lainnya dan bersama-sama untuk mencapai suatu tujuan yang tertentu.

Sedangkan menurut Ladjamuddin (2013). Sistem lebih ditekankan sebagai kumpulan elemen yang secara bersama-sama menyelesaikan serangkaian tugas untuk menyelesaikan suatu target tertentu. Tugas-tugas tersebut berupa prosedur-prosedur yang saling terkait.

Berdasarkan dua pendapat dari para ahli di atas, maka sistem dapat diartikan sebagai gabungan/kumpulan dari elemen-elemen yang saling bekerja sama untuk mencapai suatu tujuan dan kumpulan suatu prosedur/jaringan yang saling berhubungan dan bekerja sama untuk mencapai suatu tujuan dari sebuah kegiatan.

2.1.1 Karakteristik Sistem

Menurut Ladjamudin (2013), Karakteristik dari suatu sistem adalah adanya komponen-komponen, batas sistem, lingkungan luar sistem, penghubung, *input, output,* pengolahan dan sasaran atau tujuan.

1. Komponen Sistem

   Di dalam sistem, terdapat berbagai komponen yang saling berhubungan dan bekerja sama membentuk satu kesatuan. Komponen-komponen dalam sistem dapat berupa sub-sub sistem atau bagian-bagian kecil pembentuk sistem.

2. Batasan Sistem

   Karena sistem merupakan satu kesatuan, maka batasan antara satu sistem dengan sistem lain bisa berupa lingkungan atau ruang lingkup dari masing-masing sistem itu sendiri.

3. Lingkungan Luar Sistem

   Lingkungan luar sistem adalah semua yang berada di luar batasan sistem yang. Lingkungan luar sistem bisa saja memberikan pengaruh positif terhadap sistem atau pengaruh negatif terhadap sistem. Lingkungan luar sistem yang mempunyai pengaruh positif terhadap sistem sebisa mungkin dipelihara agar dapat membantu keberlangsungan sistem yang ada, sedangkan lingkungan luar sistem yang memiliki pengaruh negatif sebisa mungkin ditahan atau dikendalikan agar tidak terlalu menghambat jalannya sistem yang ada.

4. Penghubung Sistem

   Untuk dapat bekerja, sub-sub sistem dalam sebuah sistem harus dapat saling terhubung agar dapat bekerja sama. penghubung antara sub-sub sistem ini bisa disebut sebagai penghubung sistem,

penghubung sistem ini merupakan sebuah media yang menjadi alat penghubung antara subsistem satu dengan subsistem yang lain yang ada dalam suatu sistem.

5. Masukan Sistem

   Untuk dapat berjalan, suatu sistem memerlukan suatu inputan, inputan dapat berupa energi ataupun sinyal input. energi digunakan sistem agar dapat berjalan sedangkan sinyal adalah energi yang telah diproses yang dikeluarkan oleh sistem.

6. Keluaran Sistem

   Keluaran sistem berupa energi yang telah diolah dan dianggap sebagai keluaran berguna. Keluaran sistem bisa saja menjadi inputan bagi subsistem yang lain.

7. Pengolahan Sistem

   Sebuah sistem setidaknya harus memiliki pengolahan sistem, pengolahan sistem ini bisa jadi merupakan sub sistem atau sistem itu sendiri. Pengolahan sistem inilah yang akan mengolah inputan menjadi keluaran.

8. Sasaran Sistem

   Sistem merupakan satu kesatuan yang mempunyai tujuan atau sasaran, sasaran atau tujuan ini menjadi syarat utama adanya sebuah sistem. Sasaran sistem juga dapat menjadi parameter keberhasilan sebuah sistem. Jika sistem dapat memenuhi sasaran maka dapat dikatakan bahwa sistem tersebut telah berhasil. Sasaran

suatu sistem dapat dipengaruhi oleh inputan dan keluaran dari sistem.

2.1.2 Klasifikasi Sistem

Menurut Ladjamudin (2013), berdasarkan sudut pandang maka sistem dapat diklasifikasikan diantaranya sebagai berikut:

1. Sistem dapat diklasifikasikan dari sudut pandang wujudnya. Sistem yang tidak tampak secara fisik dikategorikan sebagai sistem abstrak (*abstract system*) sedangkan sistem dengan fisik yang tampak dikategorikan sebagai sistem fisik (*physical system*). Contoh sistem abstrak adalah sistem teologi yang merupakan sistem yang berbicara tentang hubungan antara manusia dan tuhan, sistem ini secara fisik tidak dapat dilihat. Contoh sistem fisik adalah sistem komputer, sistem komputer dapat dilihat melalui perangkat-perangkat penyusun dari sistem komputer itu sendiri.
2. Sistem dapat diklasifikasikan dari sudut pandang proses terbentuknya. Sistem yang terbentuk akibat proses alam dan tidak dibuat oleh manusia secara sengaja seperti sistem perputaran bumi bisa kategorikan sebagai sistem alamiah (*natural system*). Sistem yang terbentuk dari rancangan manusia atau dibuat langsung oleh manusia seperti sistem informasi bisa dikategorikan sebagai sistem buatan manusia (*human made system*).
3. Sistem dapat diklasifikasikan dari sudut pandang perilaku sistem. Sistem dengan perilaku yang dapat diprediksi seperti sistem komputer

dapat dikategorikan sebagai sistem tertentu (*deterministic system*). Hal ini dikarenakan sistem komputer dapat diprediksi keluarannya berdasarkan dari program-program yang dijalankan. Sistem dengan perilaku yang tidak dapat diprediksi karena mengandung unsur probabilitas dapat dikategorikan sebagai sistem tak tentu (*probabilistic system*).

4. Sistem dapat diklasifikasikan dari sudut pandang sifatnya. Sistem dengan sifat yang tidak terpengaruh terhadap sistem luar dapat dikategorikan sebagai sistem tertutup (*closed system*). Sistem dengan sifat yang dapat dipengaruhi oleh sistem luar atau lingkungan luar dapat dikategorikan sebagai sistem terbuka (*open system*). Secara teori, sistem tertutup benar adanya namun kenyataannya tidak ada sistem yang benar-benar bersifat tertutup, yang ada adalah sistem dengan sifat tertutup secara relatif.

2.2 Konsep Dasar Informasi

Menurut Tata Sutabri (2012) informasi adalah data yang telah diklasifikasikan atau diolah atau diinterpretasikan untuk digunakan dalam proses pengambilan keputusan. Informasi juga disebut data yang diproses atau data yang memiliki arti. Informasi merupakan data yang telah diproses sedemikian rupa sehingga meningkatkan pengetahuan seseorang yang menggunakannya.

2.2.1 Siklus Informasi

Siklus informasi menggambarkan bagaimana suatu data diproses sehingga menjadi informasi yang bermanfaat bagi pengguna. Data akan melalui proses penginputan, pemrosesan dan akhirnya menjadi *output*. Data hasil pemrosesan berupa informasi yang bisa dijadikan sebagai *input*-an kembali untuk diproses dan seterusnya (Sutabri, 2012).

2.2.2 Kualitas Informasi

Menurut Tata Sutabri (2012) kualitas dari informasi tergantung dari tiga hal yaitu:

1. Akurat (*accurate*) Informasi tidak boleh mengandung ambiguitas yang bisa menyesatkan.
2. Tepat waktu (*timelines*) Kualitas dari informasi sangat bergantung dengan ketepatan waktu. Suatu informasi menjadi tidak berharga atau bernilai lagi ketika waktu penyampaian informasi tersebut terlambat.
3. Relevan (*relevance*) Kualitas informasi dapat dinilai dari kebermanfaatan informasi tersebut bagi pengguna. Kebermanfaatan ini tentunya berbeda-beda setiap pengguna.

Menurut Tata Sutabri (2012) nilai informasi didasarkan atas sepuluh sifat yaitu :

1. Mudah diperoleh

   Suatu informasi dapat dinilai dari segi kemudahan untuk mendapatkan informasi tersebut.

2. Luas dan lengkap

   Suatu informasi dapat dinilai dari luas tidaknya informasi yang ada, dan lengkap atau tidaknya informasi yang terkandung di dalamnya.

3. Ketelitian

   Suatu informasi harus bebas dari kesalahan, informasi tidak boleh mengandung unsur-unsur yang dapat merusak nilai informasi itu sendiri.

4. Kecocokan

   Setiap pengguna memiliki kecocokan yang berbeda-beda untuk setiap informasi yang didapatkan. Kecocokan suatu informasi dapat dilihat dari seberapa baik informasi tersebut bagi pengguna.

5. Ketepatan Waktu

   Setiap informasi memerlukan ketepatan waktu. Nilai dari informasi akan berkurang seiring dengan keterlambatan waktunya.

6. Kejelasan

   Suatu informasi hendaknya bebas dari ambiguitas yang dapat menyesatkan bagi pengguna.

7. Keluwesan

   Informasi dapat digunakan untuk membuat satu atau lebih keputusan, atau bisa digunakan lebih dari satu pengguna.

8. Dapat dibuktikan

Informasi dapat dinilai dari sejauh mana informasi tersebut bisa dibuktikan. Informasi dapat diuji oleh beberapa pengguna untuk melihat apakah mereka mendapatkan *output* atau kesimpulan yang sama atau tidak.

9. Tidak ada prasangka

   Suatu informasi dapat dinilai dari ada tidaknya prasangka dalam informasi tersebut yang dapat mengubah kesimpulan atau keputusan yang dibuat berdasarkan informasi tersebut.

10. Dapat diukur

    Hakikatnya informasi dapat diukur dengan melihat hasil dari sistem informasi formal.

2.3 Digital *Library*

Digital *Library* adalah sebuah sistem yang dimana di dalamnya terdapat berbagai jenis fitur dan kumpulan informasi yang dapat diakses melalui perangkat digital seperti *mobile*, *pc* atau komputer (Sismanto, 2008).

Hadirnya digital *library* diharapkan dapat mempermudah pagi para siswa, mahasiswa, guru maupun tenaga pelajar lainnya dapat mengakses informasi seperti dokumen, gambar dan sebagainya secara cepat. Sebuah perpustakaan digital tidaklah berdiri sendiri akan tetapi terhubung dengan berbagai sumber-sumber dan pelayananan informasi yang bersifat terbuka di seluruh dunia. Perpustakaan digital tidak hanya menyimpan informasi-informasi atau koleksi-koleksi digital pengganti fisiknya, ruang lingkup informasi yang ada dalam perpustakaan digital sangat luas

sampai dengan informasi-informasi yang tidak bisa digantikan dengan bentuk cetak atau fisiknya (Subrata, 2009).

Menurut Lesk (Pendit 2007) memandang bahwa secara umum digital *library* merupakan suatu sistem informasi yang di dalamnya terdapat berbagai jenis dokumen digital yang tertata. Arms (Pendit 2007) menambahkan lagi bahwa semua sistem tersebut diatur dan ditata menggunakan sistem penelusuran informasi.

Sismanto (2008) menegaskan bahwa gagasan digital *library* diikuti Kantor Kementerian Riset dan Teknologi lewat program Perpustakaan Digital yang tujukan untuk memberi kemudahan akses dokumentasi data ilmiah dan teknologi dalam bentuk digital secara terpadu dan lebih dinamis. Sistem ini dibangun dengan tujuan dapat menyediakan suatu media yang dapat menampung berbagai informasi terkait intelektual seperti disertasi, laporan penelitian, dan juga kebijakan yang terpublikasi.

Hasil pengamatan terhadap beberapa perpustakaan digital Indonesia yang dapat diakses lewat *internet* menunjukkan bahwa setidaknya perpustakaan digital yang ada di Indonesia memiliki beberapa fitur sebagai berikut (Winarko, 2009) :

1. Fitur Keanggotaan. Fitur ini dimaksudkan untuk membedakan hak akses antara pengguna terdaftar dengan pengguna lainnya. Pengguna yang terdaftar memiliki keuntungan berupa hak akses terhadap keseluruhan informasi yang tersedia. Salah satu perpustakaan yang memiliki fitur keanggotaan adalah ITB Central Library. Sedangkan perpustakaan digital yang belum menerapkan fitur ini adalah Perpustakaan Institut Bisnis dan Informatika Indonesia.

2. Fitur Pencarian. Fitur ini dapat digunakan oleh pengguna untuk mengakses informasi secara cepat dengan menggunakan *search engine* yang tersedia. Fitur ini dapat digunakan untuk mencari informasi baik dalam lingkup perpustakaan itu sendiri maupun lingkup global. Hampir semua perpustakaan memiliki fitur ini terkecuali perpustakaan islam.com yang belum menerapkan fitur ini ke dalam sistem mereka.
3. Fitur *Link* atau Pranala. Fitur ini memungkinkan untuk pengunjung perpustakaan digital untuk mengakses perpustakaan digital lainnya dengan cara menyediakan *link* menuju perpustakaan digital tersebut. *Link* memberikan keuntungan kepada pengunjung karena tidak perlu mencari link perpustakaan digital lainnya secara mandiri. Salah satu perpustakaan digital yang menerapkan fitur ini adalah ITB Central Library.
4. Fitur Dwi Bahasa. Fitur ini memberikan kemudahan bagi pengunjung perpustakaan digital yang tidak memiliki kemampuan berbahasa indonesia yang cukup. Menyediakan alternatif bahasa pada perpustakaan digital dapat meningkatkan jangkauan pengguna domestik maupun internasional. Salah satu perpustakaan digital yang memiliki fitur ini adalah Perpustakaan Digital UIN Sunan Kalijaga Yogyakarta.
5. Fitur Artikel. Fitur ini bervariasi antara satu perpustakaan digital dengan perpustakaan digital lainnya, mulai dari artikel yang bersifat ilmiah seperti hasil penelitian ataupun artikel yang bersifat populer seperti warta. Biasanya fitur ini digunakan untuk menampilkan artikel yang dihasilkan oleh lembaga

yang bersangkutan. Salah satu perpustakaan digital yang memiliki fitur ini adalah ITB Central Library.

6. Folder dan Arsip. Fitur folder dan arsip dimaksudkan untuk menyimpan file atau artikel yang biasanya bukan file terkini. Salah satu perpustakaan digital yang memiliki fitur folder dan arsip adalah Perpustakaan Digital UIN Sunan Kalijaga Yogyakarta.

2.4 *Search Engine*

Search Engine atau mesin penelusuran dapat diartikan sebagai suatu program atau aplikasi yang dibangun dalam bentuk website yang bertugas untuk melakukan penelusuran atau pencarian terhadap *keyword* yang kita masukkan pada *form* pencarian (Andrea Adelheid, 2012). Pada umumnya, *search engine* akan menampilkan hasil penelusuran dalam bentuk informasi dari berbagai sumber seperti *blog*, website, atau sumber lain seperti artikel. Halaman untuk menampilkan hasil pencarian dari *search engine* biasa juga dikenal sebagai *Search Engine Result Pages* (SERP) (Yonatan, 2018).

Mesin pencari dalam menampilkan hasil penelusuran melakukan berbagai proses ataupun algoritma, diantaranya adalah sebagai berikut (Maria Agustina S, 2009) :

a. Pencarian List

Algoritma pencarian list melakukan pencarian data di dalam list dengan cara mencari satu *key* secara linear. Kelebihan dari algoritma ini adalah hasil pencarian yang didapatkan lebih sedikit namun

kekurangannya adalah untuk melakukan pencarian membutuhkan waktu yang lebih lama karena pencariannya secara linear.

b. Pencarian *Tree*

Sesuai dengan namanya, algoritma pencarian *tree* mirip dengan cabang-cabang yang ada pada pohon, mencari data dimulai dari dataset paling banyak atau luas dan kemudian dipersempit hingga menghasilkan data yang lebih sedikit dan paling relevan.

c. Pencarian *SQL*

Algoritma pencarian ini menggunakan fitur *SQL* atau *Structured Query Language* untuk mencari data dalam dataset terstruktur. Kelebihan dari pencarian jenis ini adalah pengguna dapat mengambil langsung data subset dari keseluruhan data yang ada.

2.5 *Full-Text Search*

Full-text search adalah sebuah metode pencarian yang dilakukan komputer dengan cara mencocokan istilah atau *keyword* dalam *query* penelusuran dengan istilah dalam dokumen individual dalam database dan memberikan peringkat untuk setiap hasil pencarian menurut algoritma yang diterapkan (Beall, 2008).

Full-text search dalam proses pencarian data bekerja dengan cara mencari data dalam suatu data set yang tidak sama persis dengan *keyword* yang dimasukkan. Ketika pengguna memasukkan *keyword "Image Processing*", maka sistem yang menerapkan konsep *full-text search* akan mencari data dan menampilkan hasil yang mengandung salah satu atau kedua kata dari *keyword* (hanya "*image*" atau hanya "*processing*" atau "*image processing*" atau "*processing image*"). Hal ini

memungkinkan aplikasi dapat menebak data yang kemungkinan dicari oleh si pengguna (Santoso, 2017).

2.6 *Unstructured Data*

Secara umum, ada dua kategori data yaitu data terstruktur(*Structured Data*) dan data tidak terstruktur(*Unstructured Data*). Data terstruktur tersedia sekitar 20% dari seluruh data, serta direpresentasikan dalam bentuk relasi yang mudah dipetakan dan disimpan dalam database relasional. Sedangkan data tidak terstruktur tersedia sebanyak 80% dari seluruh data yang direpresentasikan dalam berbagai bentuk dokumen seperti laporan, artikel berita, *e-mail,* dan konten web (Knox, 2013).

Unstructured Data merupakan data yang tidak mempunyai struktur yang secara jelas atau dapat ditentukan. Jenis data ini kebanyakan ditemukan dalam bentuk teks atau paragraf, Dalam teks bisa saja terdapat berbagai jenis format data seperti tanggal, satuan, angka fakta ataupun pendapat. Jenis data seperti ini tidak mudah untuk diolah atau dipahami oleh komputer lewat program komputer tradisional. Diperlukan sebuah metode yang lebih *advanced* untuk mengekstrak informasi dari jenis data ini (Novitasari, 2015).

Menurut IDC dan EMC, diperkirakan pada akhir 2020 data akan bertambah pesat mencapai 40 *zettabytes*. Jika dibandingkan dengan jumlah data pada tahun 2010, maka jumlah data pada saat ini telah berkembang lebih dari 50 kali lipat. Saat ini, diperkirakan data tidak terstruktur sudah mencapai 70-80% dari semua data yang ada di internet saat ini (Novitasari, 2015).

Untuk mengolah atau mengekstrak informasi dari data tidak terstruktur maka diperlukan teknik-teknik khusus seperti *data mining*, analisis teks

menggunakan *Natural Language Processing* (NLP) juga merupakan salah satu metode untuk mengolah informasi sejenis ini. Metode yang paling umum digunakan saat ini untuk menata model data seperti ini yaitu melalui *tagging* atau penandaan manual, atau bisa juga dengan menggunakan *meta data* untuk penataan dalam dunia *data mining* (Afifanto, 2017).

Salah satu teknologi yang paling modern yang dapat menangani data seperti ini adalah *search engine*. *Search engine* adalah tempat dimana data tidak terstruktur banyak ditemukan, oleh karena itu, desain awal dari *search engine* memang diperuntukkan untuk menangani jenis data tidak terstruktur (Knox, 2013).

2.7 *Elasticsearch*

Elasticsearch diluncurkan pertama kali pada tahun 2010, dibangun dengan menggunakan *Apache Lucene* dan didistribusikan secara *open source*. Elasticsearch merupakan *search engine* yang dapat melakukan pencarian teks lengkap, analisis bisnis ataupun *log* (Divya, 2013).

elasticsearch dibangun menggunakan bahasa Java dan merupakan database *standalone server*. *Elasticsearch* menyimpan data dalam bentuk format canggih yang dikhususkan untuk pencarian data teks secara optimal. Setelah meng-i*nstall*, *elasticsearch* dapat diakses dengan mudah menggunakan *HTTP/JSON* (Novitasari, 2015).

elasticsearch dapat digunakan sebagai media penyimpanan data sekaligus sebagai media pencarian data. Elasticsearch sangat mudah untuk dikembangkan, salah satunya dikarenakan *elasticsearch* sangat *scalable,* pengguna dapat lebih

fokus kepada membangun sistem agar lebih optimal dan *elasticsearch* dapat di *upgrade* kapanpun sesuai dengan kebutuhan (Novitasari, 2015).

Konsep dan fitur-fitur dasar *elasticsearch* (Kuc, 2013):

1. *Indeks*

   Dalam *elasticsearch*, *index* dapat diumpamakan sebagai database. *index* dapat diatur dengan mengirim *query* ke *elasticsearch* lewat fitur *API*. Dalam *index* ini akan diatur jumlah *shard* yang ingin digunakan dan jumlah *replica* yang akan dibuat.

2. *Mapping*

   Mapping adalah bagian dimana pengguna mengatur bagaimana data akan disimpan dalam *elasticsearch*. Di bagian inilah *field-field* dokumen diatur beserta tipe datanya. Pengguna juga dapat mengatur bagaimana teks-teks yang diinputkan di tokenisasi, misalnya *tag-tag HTML* akan dibersihkan sebelum diinputkan ke dalam *index*. Dalam satu *index*, pengguna dapat menambahkan beberapa *mapping*. *Mapping* dapat diumpamakan sebagai tabel dalam *relational database*.

3. Dokumen

   Dokumen dalam *elasticsearch* merupakan *entity*, terdiri dari beberapa *field* dan setiap *field*-nya memiliki penamaan dan dapat menyimpan satu atau lebih nilai. Nantinya, dokumen dalam *elasticsearch* akan mempunyai *field* yang berbeda, beda halnya dengan *relational database* yang datanya telah terstruktur. Dokumen dalam *elasticsearch*

disimpan dalam format JSON. Sederhananya, dokumen dapat diumpamakan sebagai kolom dan datanya dalam *elasticsearch*.

4. *Type*

   Setiap dokumen yang ada dalam *elasticsearch* mempunyai tipe. Tipe dalam dokumen digunakan untuk mempermudah proses pencarian agar lebih cepat karena dengan tipe ini dokumen akan dikelompokkan ke dalam beberapa grup logika.

5. *Node*

   Node adalah instan dari sebuah *elasticsearch*. Elasticsearch memerlukan minimal satu *node* untuk dapat bekerja. Jika lebih dari satu *node*, maka penamaan *node* biasanya menggunakan urutan angka. *Node-node* ini akan saling berbagi data dan beban menyelesaikan suatu tugas yang diberikan. Satu *node* akan bertindak sebagai *node* master yang akan mengatur *node* yang lain seperti penambahan atau penghapusan *node*.

6. Cluster

   Kumpulan dari beberapa *node* dapat disebut sebagai *cluster*. Konsep penggunaan *cluster* adalah untuk mengurangi beban dari sistem dengan membagi tugas-tugas menjadi beberapa bagian. Kelebihan dari sistem *cluster* adalah ketika sistem mengalami kegagalan pada *node* tertentu maka sistem akan tetap beroperasi dengan mengalihkan kinerja *node* yang gagal ke *node* yang masih aktif sementara *node* yang gagal itu akan di *restart* agar dapat beroperasi lagi.

7. *Shard*

Data yang ada dalam *elasticsearch* akan dibagi-bagi menjadi beberapa bagian atau potongan dan akan disimpan ke dalam *shard*. Ketika *index* pertama kali dibuat, pengguna dapat mengatur jumlah *shard* yang akan digunakan. Ketika pengguna memasukkan data dalam *elasticsearch* (*indexing*), maka *elasticsearch* akan secara otomatis melakukan *sharding* dan memasukkan hasil *sharding* ke dalam *shard*. Pengguna tidak dapat mengetahui data yang diambil dari *elasticsearch* berasal dari *shard* mana. *Shard* dibagi menjadi dua macam :

1. *Shard* Primer

   Shard ini berfungsi untuk mengatur pembagian data. *Shard* ini sangat terbatas ukurannya dikarenakan beberapa faktor mulai dari spesifikasi perangkat keras yang digunakan, tingkat kompleksitas dokumen dll.

2. *Shard* Replika

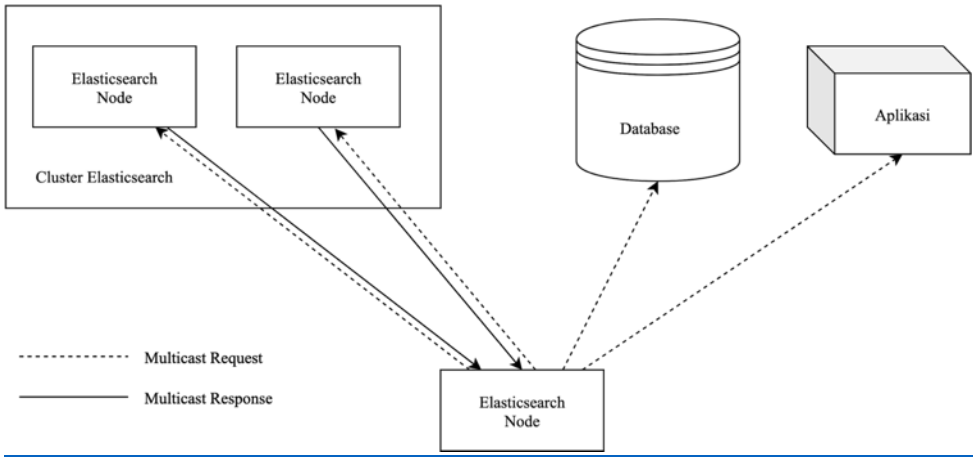
   Shard ini merupakan hasil *replica* dari *shard* primer. Setiap *shard* akan mempunyai *replica masing*-masing. Isi dari *shard replica* akan sama persis dengan *shard* primernya. *Shard replica* akan ditempatkan ke dalam *node* yang berbeda dengan *shard* primernya supaya ketika *node* dimana *shard* primernya mengalami gangguan dan tidak bisa beroperasi, maka data tetap akan utuh dikarenakan hasil *replica shard* primernya berada pada *node* lain yang masih beroperasi.

Dilihat dari segi arsitekturnya, fitur-fitur utama *elasticsearch* adalah :

1. *Default Setting.* Saat meng*install elasticsearch* untuk pertama kali, *elasticsearch* akan terkonfigurasi secara otomatis menggunakan konfigurasi standar seperti jumlah *shard*, *replica* dsb. Pengguna dapat langsung menggunakan atau mengakses *elasticsearch* tanpa harus melakukan pengaturan tambahan. Fitur ini termasuk fitur *built-in* dalam *elasticsearch*. Secara default, elasticsearch berjalan dengan 1 node dan 5 shard.
2. *elasticsearch* beroperasi secara terdistribusi secara otomatis dengan asumsi *node* dan *cluster* telah terkonfigurasi secara otomatis.
3. *Node* akan terkoneksi secara otomatis ke mesin lain dalam *cluster* untuk proses aliran data. Hal ini termasuk ke dalam replikasi *shard* yang berjalan secara otomatis. Fitur ini dikenal dengan SPOF atau Arsitektur *Peer-to-peer* tanpa titik tunggak kegagalan.
4. *elasticsearch* sangat mudah dalam hal pengukuran jumlah data dan kapasitas penyimpanan. Pengguna juga dapat menambahkan *node cluster* baru ke dalam *elasticsearch*.
5. *elasticsearch* sangat *scalable.* Tidak ada batasan mengenai jumlah data yang bisa dimasukan ke dalam *index elasticsearch*. Selain itu, pengguna dapat melakukan penyesuaian terhadap model data yang ada misalkan menambah *field* dsb. Elasticsearch juga dapat meng*handle* multiple jenis data dalam satu *index*.
6. *Near Real Time* (*NRT*). Elasticsearch dalam proses *index*ing dan pencarian data hampir *real time*. Waktu jeda yang dihasilkan

dikarenakan *elasticsearch* bekerja secara terdistribusi, sehingga proses penyesuaian data untuk setiap *shard, node* atau *cluster* memerlukan waktu.

2.7.1 Proses *elasticsearch*



Gambar 2.1 Proses *Elasticsearch*

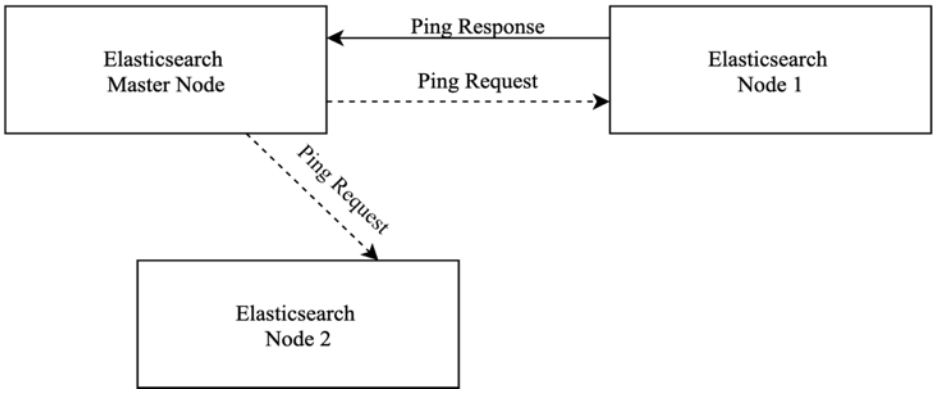
Sumber : Novitasari (2015)

Pada Gambar 2.1 dijelaskan proses *elasticsearch*. Saat *node* mulai bekerja, semua proses akan didistribusikan ke *node-node* yang ada pada *cluster elasticsearch*. Setiap *node* akan memiliki nama yang berbeda-beda dan diset saat pertama kali proses konfigurasi. Salah satu *node* yang ada dalam satu *cluster* akan berfungsi sebagai *node* master, *node* ini akan bertugas untuk mengatur jalannya perubahan topologi *cluster*. Ketika pengguna mengakses sistem *elasticsearch*, *elasticsearch* akan mengirimkan respon dari *node elasticsearch*, jika sistem dibangun ke dalam beberapa *node*, maka data akan dicari ke semua *node* yang aktif. Di dalam *node* inilah terdapat *shard*

yang menyimpan potongan-potongan dokumen. Respon dari *elasticsearch* inilah yang selanjutnya akan ditampilkan ke pengguna (Novitasari, 2015).

2.7.2 Komunikasi *elasticsearch*

Proses komunikasi dalam *elasticsearch* sepenuhnya diatur oleh *node* master. *Node* master akan memonitor semua *node* yang ada untuk memastikan status dari *node* apakah aktif atau mengalami masalah sehingga tidak aktif. Jika salah satu *node* tidak dapat bekerja dikarenakan suatu masalah maka *node* master secara otomatis melakukan pengalihan atau pergantian *node* yang gagal dan melakukan inisiasi *restart* untuk *node* yang gagal. *Node* yang lain akan bertanggung jawab untuk mengambil alih tugas atau beban dari *node* yang gagal sampai *node* yang gagal dapat bekerja kembali. Proses ini dikenal dengan proses *rebalancing cluster-shard* (Novitasari, 2015).



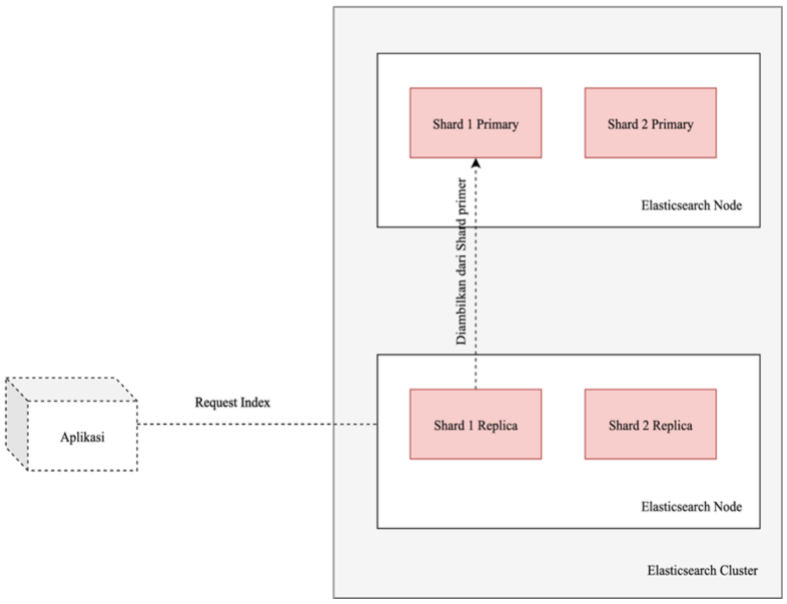
Gambar 2.2 Proses Komunikasi *Elasticsearch*

Sumber : Novitasari (2015)

Pada Gambar 2.2 dijelaskan proses komunikasi *elasticsearch*. Elasticsearch dibangun dengan menggunakan 3 *node*, 1 *node* master, dan sisanya sebagai *node* standar (*Node* 1 dan 2). Untuk memastikan setiap *node* aktif, maka *node* master akan melakukan *ping request* ke masing-masing *node*. *Node* akan mengirimkan *ping response* jika *node* tersebut berjalan atau aktif dan sebaliknya jika tidak aktif atau mengalami gangguan (Novitasari, 2015).

2.7.3 Indexing Data

Proses *indexing* data hanya terjadi pada *shard* primer. Jika proses *indexing* data dilakukan ke dalam *shard* yang tidak benar atau berisi *replica*, maka proses *indexing* akan diteruskan ke *shard* primer (Novitasari, 2015).



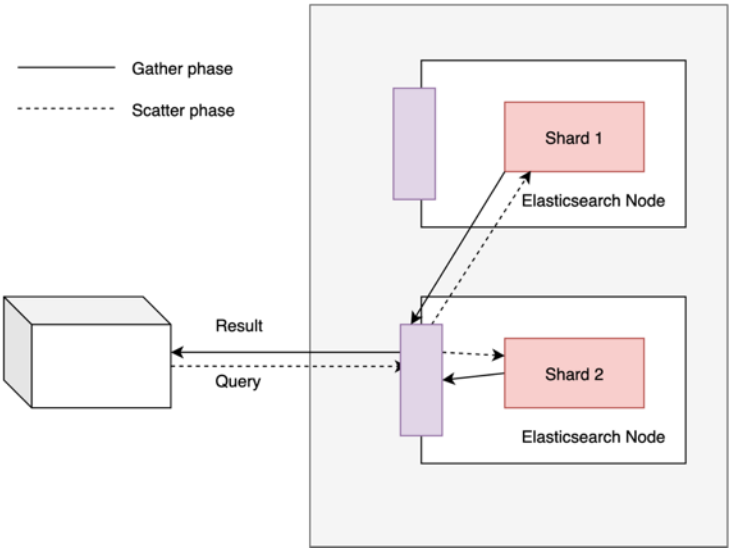
Gambar 2.3 Proses *Indexing*

Sumber : Novitasari (2015)

Pada Gambar 2.3 dijelaskan konsep dasar *indexing* pada *elasticsearch*. Proses *indexing* terjadi saat pengguna melakukan *request* penginputan data. Sebelum melakukan *indexing* data, nama *node*, *cluster*, dan *replica* telah dikonfigurasi terlebih dahulu untuk menghindari tumpang tindih pada proses *indexing*. Data yang akan di *indexing* akan dipisah-pisah atau dipotong agar dapat disimpan ke dalam *shard* yang sesuai secara otomatis. Setiap *node* terdiri dari dua *shard*, satu *node* terdiri dari *shard replica* dan *node* lain berisi dengan *shard* primer. Ketika pengguna melakukan *request* data, maka aplikasi akan memberikan *request* ke dalam *shard replica*. Dikarenakan konfigurasi yang diset adalah *request* pertama ke dalam *shard replica* maka *shard replica* akan meneruskan *request* ke *shard* primer selama *shard* primer terkoneksi dengan *shard replica*. Saat *replica* tidak dapat mengembalikan respon secara langsung jika dalam sistem masih ada *shard* primer. Baik *shard* primer maupun *replica* dapat dimasukkan ke dalam *node* yang sama (Novitasari, 2015).

2.7.4 *Querying* Data

Setelah proses *indexing* dilakukan, pengguna dapat melakukan *request* data ke *elasticsearch* dengan cara mengirim *query* dan *elasticsearch* akan mengirim respon ke pengguna.



Gambar 2.4 Proses *Querying*

Sumber : Novitasari (2015)

Pada Gambar 2.4 dijelaskan proses *querying* data, pada proses ini ada dua fase (Novitasari, 2015):

1. *Scatter phase* (fase pemecahan / pemisahan)

   Pada fase ini, pertama-tama pengguna akan mengirim *request* ke dalam sistem. *request* dari pengguna akan dikirim ke *cluster*, selanjutnya *cluster* akan melanjutkan *request* tersebut ke *node* dan *shard*, saat *request* diterima oleh *node*, *node* akan membagi *request* tersebut dan mengirimkan ke *shard* sesuai dengan jumlah *shard* yang ditentukan saat mengkonfigurasi *elasticsearch*. Selanjutnya *shard* akan mengirim respon kembali ke *cluster* untuk diteruskan ke *node*.

2. *Gather phase* (fase penggabungan)

   Pada fase ini, respon yang diterima dari *shard* akan dilakukan proses penggabungan oleh *node*. Jadi data yang ada di *node* adalah

data yang sudah mengalami penggabungan dari *shard-shard* yang ada dalam *node* tersebut. Jika sistem dibangun menggunakan lebih dari satu *node,* makan selanjutnya data dari semua *node* akan digabungkan dan dikirim ke *cluster* untuk dikirim ke pengguna sebagai bentuk respon. Semua proses penggabungan data terjadi secara otomatis dalam sistem.

Jumlah *node elasticsearch* dapat dikonfigurasi sesuai dengan kebutuhan pengguna. Spesifikasi seperti RAM, CPU, dan media penyimpanan dari *hardware* yang digunakan juga mempengaruhi jumlah *node* yang bisa ditambahkan oleh pengguna.

2.7.5 *Scoring Document*

Scoring Document dalam *elasticsearch* merupakan proses penentuan tingkat relevansi antara *document* dengan *query.* Setiap *document* yang dihasilkan dari proses *searching* akan memiliki *score* masing-masing yang menunjukkan tingkat relevansi *document* tersebut. *Score* pada dasarnya dipengaruhi oleh beberapa faktor diantaranya adalah *query, term frequency*, dan parameter-parameter lainnya. Faktor yang paling berpengaruh pada *scoring document* adalah *Term Frequency/Inverse Document Frequency (TF/TD) model*, *model* ini banyak digunakan untuk *system information retrieval, natural language processing*, dan algoritma *language analysis* lainnya. *Model* ini menarik kesimpulan bahwa *document* dengan *term frequency* yang lebih tinggi dan mengandung lebih banyak *term* unik yang tidak terdapat pada kebanyak *document* lain akan dianggap lebih relevan

daripada data yang lain. Sehingga, *document* yang memiliki *score* yang lebih tinggi akan dianggap lebih relevan dengan *query* dibandingkan dengan *document* yang memiliki *score* yang lebih rendah *(Bobriakov, 2017)*.

Dalam penerapannya, *document* dengan *score* yang lebih tinggi akan ditampilkan pada urutan pertama dalam hasil pencarian kemudian data yang lain sesuai urutan *score document*-nya masing-masing. Berikut formula untuk *scoring document* pada *elasticsearch (Bobriakov, 2017):*

$$score(q,d) = queryNorm(q) \times coord(q,d) \times \sum (tf(t\ in\ d) \times idf(t)^2 \times t.getBoost() \times norm(t,d))(t\ in\ q) \quad \text{(II.1)}$$

score (q,d)*:* skor relevan dokumen *d* terhadap *query q*

queryNorm (q)*:* faktor normalisasi *query*

coord (q,d)*:* faktor koordinasi *query*

tf (t in d)*:* frekuensi sebuah istilah *t* dalam dokumen *d*

idf (t)*:* inversi frekuensi dokumen untuk istilah t

t.getBoost()*:* nilai pengali untuk *query*

norm(t,d)*:* inversi akar kuadrat jumlah istilah dalam sebuah frasa.

2.7.6 *Analyzer*

Salah satu bagian dari *elasticsearch* yang dapat dikonfigurasi adalah *analyzer*. *Analyzer* berperan penting dalam menentukan kemampuan *elasticsearch* dalam menemukan dokumen relevan. *Analyzer* bertanggung

jawab terhadap bagaimana model dokumen akan disimpan ke dalam *elasticsearch* (Firdaus, 2020).

Analyzer memiliki kaitan yang erat dengan *inverted index. Inverted index* merupakan data struktur untuk menyimpan pemetaan (*mappings*) antara *tokens* dengan *document identifiers* yang mengandung istilah (*term*) terkait. Selain dari *document identifiers*, *inverted index* juga menyimpan posisi dimana istilah (*term)* tersebut berada dalam dokumen (Firdaus, 2020).

Berikut ditunjukkan salah satu contoh *inverted index* dengan 2 dokumen berbeda:

- Dokumen 1 : *"elasticsearch is fast"*
- Dokumen 2 : *"I want to learn elasticsearch"*

Jika dokumen tersebut di-*index* ke dalam *elasticsearch* sebagai *inverted index* maka model datanya akan seperti pada Tabel 1.1 di bawah ini:

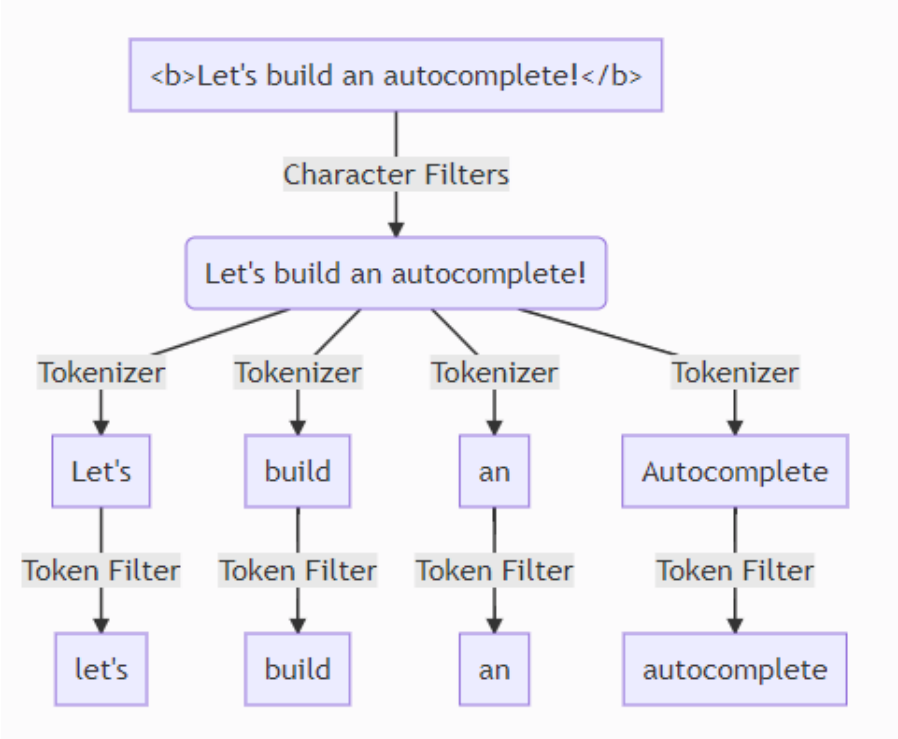
Tabel 2.1 Model Data pada *Inverted Index Elasticsearch*

Sumber : (Firdaus, 2020).

Tokens	*Count*	*Document : Position*
elasticsearch	2	1:1, 2:5
is	1	1:2
fast	1	1:3
i	1	2:1
want	1	2:2
to	1	2:3
learn	1	2:4

Pada Tabel 2.1, kedua dokumen diproses sehingga menjadi potongan-potongan istilah, frekuensi kemunculan serta posisi istilah tersebut dalam kedua dokumen. Proses dalam *inverted index* inilah yang dilakukan oleh *analyzer* dalam *elasticsearch* (Firdaus, 2020).

Pada saat proses *indexing data* ke dalam *elasticsearch*. Dokumen akan diproses dan dianalisis terlebih dahulu oleh *analyzer*. Dokumen terlebih dahulu dipecah-pecah menjadi potongan-potongan *tokens* sebelum dimasukkan ke dalam *inverted index.* Sebagai contoh, dokumen dengan teks *"Let's build an Autocomplete"* akan diubah menjadi *"let's", "build", "an",* dan *"autocomplete"* (Firdaus, 2020).



Gambar 2.5 Proses *Analyzer* dalam *Elasticsearch*

Sumber : (Firdaus, 2020).

Pada Gambar 2.5 ditunjukkan proses analisis oleh *analyzer*. Langkah pertama adalah proses *characters filters*, proses ini menghilangkan karakter *tag html* (,). Berikutnya adalah proses *tokenizer*, proses ini memecah teks ke dalam potongan kata/istilah. Dan terakhir adalah proses *token filter*, proses ini untuk melakukan *filter* terhadap token yang dihasilkan dalam ini adalah mengubah huruf besar menjadi huruf kecil (Firdaus, 2020).

Fitur *analyzer* dalam *elasticsearch* merupakan fitur *built-in*, sehingga pengguna dapat menggunakan *standard analyzer* yang sudah disediakan oleh *elasticsearch* sesuai dengan *use case* pada sistem yang dibangun. Selain itu, pengguna juga dapat membuat *custom analyzer* sendiri untuk keperluan yang lebih spesifik pada sistem (Firdaus, 2020).

2.8 *Black Box Testing*

Menurut Rosa dan Shalahuddin (2015) "*Black Box Testing* adalah proses menguji perangkat lunak dari segi spesifikasi fungsional tanpa menguji desain dan kode program".

Sedangkan menurut Rizky (2011), *Black Box Testing* adalah tipe *testing* yang memperlakukan perangkat lunak tanpa perlu mengetahui kinerja internalnya.

Adapun menurut Mustaqbal, dkk (2015), *Black Box Testing* lebih berfokus kepada spesifikasi fungsional dari suatu perangkat lunak, kumpulan dari kondisi input dan melakukan pengujian terhadap fungsional program.

Dari pendapat para ahli di atas, dapat ditarik kesimpulan bahwa *Black Box Testing* adalah sebuah metode yang digunakan untuk menguji kelancaran suatu

program atau aplikasi. Pengujian ini penting dilakukan terhadap suatu program untuk menghindari kesalahan alur program agar terhindar dari *error* atau *bug*.

2.9 *Recall, Precision* dan *F-Measure*

Sebuah sistem yang melakukan proses pencarian data tentunya diharapkan bisa memberikan hasil yang 100% benar. Namun kenyataannya, tidak bisa dipungkiri bahwa dalam penerapannya sistem tidak akan bisa 100% selalu memberikan hasil yang benar. Hal inilah yang membuat pengukuran kinerja dari suatu sistem harus dari berbagai aspek diantaranya adalah *Recall* dan *Precision* (Novitasari, 2015).

Recall dapat diartikan sebagai kemampuan suatu sistem dalam memanggil dokumen yang relevan. Nilai *Recall* menggambarkan perbandingan antara dokumen relevan yang berhasil ditemukan dengan jumlah keseluruhan dokumen relevan dalam suatu koleksi. Sedangkan *Precision* dapat diartikan sebagai kepersisan atau kecocokan antara permintaan pengguna dengan hasil yang diberikan oleh sistem. Nilai *Precision* menggambarkan perbandingan antara jumlah dokumen relevan yang berhasil ditemukan dengan jumlah seluruh dokumen yang ditemukan baik relevan maupun tidak relevan (Novitasari, 2015).

Lancaster (1979) merumuskan matriks *Precision* dan *Recall* seperti yang ditunjukkan pada Tabel 2.2

Tabel 2.2 Matriks *Recall* dan *Precision Lancaster*

	Relevant	*Not Relevant*	*Total*
Retrieved	a *(hits)*	b *(noise)*	a + b
Not Retrieved	c *(misses)*	d *(rejected)*	c + d
Total	a + c	b + d	a + b + c + d

Berdasarkan Tabel 2.2, rumus untuk mencari nilai *Recall* dan *Precision* adalah sebagai berikut:

$$Recall = \frac{a}{(a+c)} \times 100$$

(II.2)

$$Precision = \frac{a}{(a+b)} \times 100 \quad \text{(II.3)}$$

Dari nilai *Recall* dan *Precision* jika dikombinasikan akan menghasilkan perpaduan atau kombinasi yang disebut *harmonic mean,* atau biasa juga disebut *F-Measure* yang dapat dihitung menggunakan persamaan di bawah ini (Novitasari, 2015).

$$F - Measure = 2 \times \frac{(Precision \times Recall)}{(Precision + Recall)} \quad \text{(II.4)}$$