IMPLEMENTASI *ELASTICSEARCH* UNTUK PENCARIAN DAN MENENTUKAN *SIMILARITY* PADA DOKUMEN SKRIPSI DI FAKULTAS TEKNOLOGI INFORMASI UNIVERSITAS BALE BANDUNG

**SKRIPSI**

Karya Tulis sebagai Syarat untuk Memperoleh

Gelar Sarjan Komputer dari Fakultas Teknologi Informasi

Universitas Bale Bandung

Disusun oleh :

MAESURI FAUZIAH

NPM. C1A150028



**PROGRAM STARTA 1**

**TEKNIK INFORMATIKA**

**FAKULTAS TEKNOLOGI INFORMASI**

**UNIVERSITAS BALE BANDUNG**

**BANDUNG**

**2019**

# KATA PENGANTAR

Puji dan syukur penulis panjatkan atas kehadirat Allah SWT yang mana berkat limpahan Rahmat dan Karunia-NYA penulis dapat menyelesaikan Proposal ini dengan balik dan tepat pada waktunya. Dalam Laporan ini penulis dapat menyelesaikan skripsi yang berjudul ***“*IMPLEMENTASI *ELASTICSEARCH* UNTUK PENCARIAN DAN MENENTUKAN *SIMILARITY* PADA DOKUMEN SKRIPSI DI FAKULTAS TEKNOLOGI INFORMASI UNIVERSITAS BALE BANDUNG*”***dengan tepat pada waktunya. Skripsi ini disusun untuk menyelesaikan tugas akhir guna memperoleh gelar Sarjana Komputer.

Laporan ini dibuat dengan berbagai observasi, wawancara dan beberapa bantuan dari berbagai pihak yang membantu menyelesaikan proses pengerjaan Laporan ini. Oleh karena itu, penulis ucapkan terima kasih sebesar-besarnya kepada :

1. Allah SWT yang telah memberikan rahmat dan karunia-Nya dalam proses pengerjaan laporan skripsi ini.
2. Yudi Herdiana, S.T, M.T, selaku Dosen Pembimbing dan Dekan Fakultas Teknologi Informasi Universitas Bale Bandung.
3. Yaya Suharya, S.Kom, M.T, selaku Ketua Prodi Teknik Informatika Fakultas Teknologi Informasi Universitas Bale Bandung.
4. Mochammad Ridwan, S.T, Dosen Pembimbing Skripsi yang telah membantu dalam pembuatan laporan Skripsi.
5. Keluarga yang memberikan dukungan dan do’a dalam proses pengerjaan laporan. Dan juga teman-teman yang saling membantu dalam proses pengerjaan Laporan.

Penulis menyadari sepenuhnya bahwa laporan ini masih jauh dari kesempurnaan, oleh karena itu kritik dan saran dari semua pihak yang bersifat membangun selalu penulis harapkan demi kesempurnaa Laporan ini.

|  |  |
| --- | --- |
|  | Bandung, 12 Juni 2019  Penulis |

# ABSTRAK

Perkembangan Teknologi Informasi saat ini begitu pesat sehingga menyebabkan kita haus akan kecepatan, efisiensi dan akurasi yang tepat. Teknologi informasi yang paling sering digunakan adalah sistem informasi yang dapat diakses dari perangkat-perangkat TI seperti laptop atau smartphone selain cepat juga dapat di akses secara *real time*. Pada suatu perguruan tinggi data-data skripsi dari mahasiswa tahun-tahun sebelumnya dapat bermanfaat bagi mahasiswa yang membutuhkan referensi ilmiah dalam menyusun skripsi dan juga saat memulai penyusunan skripsi harus menyertakan pustaka yang jika tidak akan dianggap sebagi plagiat atau adanya kemiripan (*similarity)* dalam dokumen penelitian atau dokumen skripsi yang tentunya akan menghambat dalam penelitian. Dengan adanya *Elasticsearch* dapat membantu pengguna dalam pencarian dokumen skripsi, karena *Elasticsearch* merupakan mesin pencari *full text* yang *open source* dan sebagai alat analisis data juga dapat di skalakan (*scalable*). Selain itu *Elasticsearch* juga *Elasticsearch* di kembangkan dengan bahasa pemrograman Java yang ditenagai oleh Apache Lucene yang juga merupakan search engine database yang memiliki querylow level. *Elasticsearch* memiliki query yang lebih mudah untuk digunakan karena menggunakan *web service* RESTful dan juga merupakan salah satu database yang masuk ke dunia NoSQL.

**Kata kunci:** *Elasticsearch*, *web service, Skripsi, Full Text, RESTful, Java*, *NoSQL*, *Database,* Plagiat, *Similarity*

# *ABSTRACT*

*The development of Information Technology is now so rapid that it causes us to thirst for the right speed, efficiency and accuracy. Information technology that is most often used is information systems that can be accessed from IT devices such as laptops or smartphones, besides being fast can also be accessed in real time. In a college thesis data from students in previous years can be useful for students who need scientific references in preparing a thesis and also when starting the preparation of the thesis must include a library that otherwise would be considered as plagiarism or similarity in the research document or a thesis document that will certainly hinder the research. With Elasticsearch can help users in searching thesis documents, because Elasticsearch is a full-text full-text search engine and as a data analysis tool can also be scaled (scalable). Besides Elasticsearch also Elasticsearch was developed with the Java programming language that is powered by Apache Lucene which is also a database search engine that has querylow level. Elasticsearch has queries that are easier to use because they use RESTful web services and are also one of the databases that enter the NoSQL world.*

***Keywords****: Elasticsearch, web service, Thesis, Full Text, RESTful, Java, NoSQL, Database, Plagiarism, Similarity*

# DAFTAR ISI

[KATA PENGANTAR i](#_Toc11651567)

[ABSTRAK ii](#_Toc11651568)

[*ABSTRACT* iii](#_Toc11651569)

[DAFTAR ISI iv](#_Toc11651570)

[DAFTAR TABEL vi](#_Toc11651571)

[DAFTAR GAMBAR vii](#_Toc11651572)

[BAB I PENDAHULUAN 1](#_Toc11651573)

[1.1 Latar Belakang 1](#_Toc11651574)

[1.2 Rumusan Masalah 3](#_Toc11651575)

[1.3 Batasan Masalah 3](#_Toc11651576)

[1.4 Tujuan Penelitian 3](#_Toc11651577)

[1.5 Metodologi Penelitian 4](#_Toc11651578)

[1.6 Sistematika Penulisan 4](#_Toc11651579)

[BAB II TINJAUAN PUSTAKA 6](#_Toc11651580)

[2.1 Tinjauan Penelitian 6](#_Toc11651584)

[2.2 Dasar Teori 8](#_Toc11651585)

[2.2.1 *Elasticsearch* 8](#_Toc11651587)

[2.2.2 *Web Service* 12](#_Toc11651588)

[2.2.3 *Apache Lucene* 14](#_Toc11651589)

[2.2.4 *Search Engine* 14](#_Toc11651590)

[2.2.5 *Database* 15](#_Toc11651591)

[2.2.6 *NoSQL* 16](#_Toc11651592)

[2.2.7 *RESTful Application Programming Interface (API)* 18](#_Toc11651593)

[2.2.8 *MySQL* 19](#_Toc11651594)

[2.2.9 Java 20](#_Toc11651595)

[2.2.10 *JSON (Java Script Object Notation)* 21](#_Toc11651596)

[2.2.11 *Similarity* 22](#_Toc11651597)

[2.2.12 *Apache Maven* 24](#_Toc11651598)

[2.2.13 *Unified Modeling Language (UML)* 25](#_Toc11651599)

[2.2.14 *Postman* 29](#_Toc11651600)

[2.2.15 *Kibana* 30](#_Toc11651601)

[2.2.16 *NetBeans IDE* 30](#_Toc11651602)

[2.2.17 *Model Driven Development (MDD)* 31](#_Toc11651603)

[2.2.18 *Framework* *Spring* 32](#_Toc11651604)

[BAB III METODOLOGI PENELITIAN 34](#_Toc11651605)

[3.1 Kerangka Berfikir 34](#_Toc11651611)

[3.2 Deskripsi 35](#_Toc11651612)

[3.2.1 Investigasi Pendahuluan 35](#_Toc11651613)

[3.2.2 Analisis Masalah 37](#_Toc11651614)

[3.2.3 Analisis Kebutuhan 38](#_Toc11651615)

[3.2.4 Perancangan 40](#_Toc11651616)

[3.2.5 Kontruksi 41](#_Toc11651617)

[3.2.6 Implementasi 41](#_Toc11651618)

[3.2.7 Pembuatan Laporan 42](#_Toc11651619)

[DAFTAR PUSTAKA 43](#_Toc11651620)

# DAFTAR TABEL

[Tabel 2.1 Simbol-Simbol *Use Case Diagram* 26](#_Toc11652838)

[Tabel 2.2 Simbol-simbol *Activity Diagram* 27](#_Toc11652839)

[Tabel 2.3 Simbol-Simbol *Sequence Diagram* 28](#_Toc11652840)

# DAFTAR GAMBAR

[Gambar 2.1 Strategi Distribusi 11](#_Toc11652879)

[Gambar 2.2 Alur Kerja *Elasticsearch* 12](#_Toc11652880)

[Gambar 2.3 Contoh *Database* 20](#_Toc11652881)

[Gambar 2.4 Contoh *JSON* 21](#_Toc11652882)

[Gambar 2.5 *Model Driven Development* 32](#_Toc11652883)

[Gambar 3.1 Kerangka Berfikir 34](#_Toc11664829)

# BAB I PENDAHULUAN

## Latar Belakang

Skripsi adalah istilah yang digunakan di Indonesia untuk mengilustrasikan suatu karya tulis ilmiah berupa paparan tulisan hasil penelitian sarjana S1 yang membahas suatu permasalahan/fenomena dalam bidang ilmu tertentu dengan menggunakan kaidah-kaidah yang berlaku. Skripsi bertujuan agar mahasiswa mampu menyusun dan menulis suatu karya ilmiah, sesuai dengan bidang ilmunya. Mahasiswa yang mampu menulis skripsi dianggap mampu memadukan pengetahuan dan keterampilannya dalam memahami, menganalisis, menggambarkan, dan menjelaskan masalah yang berhubungan dengan bidang keilmuan yang diambilnya. Skripsi merupakan persyaratan untuk mendapatkan status sarjana (S1) di setiap Perguruan Tinggi Negeri (PTN) maupun Perguruan Tinggi Swasta (PTS) yang ada di Indonesia. (wikipedia).

Fakultas Teknologi Informasi adalah salah satu Fakultas di Universitas Bale Bandung yang sebelum mahasiswa memulai penyusunan skripsi tentunya mahasiswa diharapkan untuk mencari referensi-referensi ilmiah terkait topik atau judul, yang nantinya akan dijadikan sebagai bahan acuan pembuatan judul skripsi.

Pencarian skripsi sebagai referensi ilmiah memang sudah efisien dan pasti dapat ditemukan tetapi masih belum bisa secara cepat dan tepat, dari banyaknya dokumen skripsi di perpustakaan yang menyebabkan mahasiswa membutuhkan lebih banyak waktu untuk mencari referensi yang dirasa tepat dan sesuai dengan yang di inginkannya.

Memulai penyusunan skripsi sering ditemukan mahasiswa yang melakukan pengambilan pustaka tanpa menyertakan sumbernya yang mana akan dianggap sebagai plagiat atau pelanggaran hak cipta dari suatu penelitian atau skripsi yang telah dibuat. Tentunya mahasiswa harus mengetahui bahwa topik yang diambil haruslah murni hasil pemikiran dari mahasiswa yang bersangkutan dan tidak merupakan sebuah plagiat dari penelitian orang lain.

Berdasarkan pemaparan diatas teknologi yang tepat dalam mengatasinya yaitu mengimplementasikan *Elasticsearch* untuk pencarian dokumen karena kelebihan dari *Elasticsearch* dapat membantu pengguna dalam pencarian dokumen skripsi tahun sebelumnya yaitu mencari dengan menggunakan kata kunci abstrak, nama, judul skripsi, topik dan lain-iain yang dapat di akses secara real time dan merupakan mesin pencari *full text* yang *open source* dan sebagai alat analisis data juga dapat diskalakan (*scalable*). Keluaran dari *Elasticsearch* berupa dokumen yang sudah diberikan skor dan diurutkan berdasarkan relevansi dokumen dengan kata kunci yang diinputkan pengguna.

Sebuah aplikasi yang dapat mempermudah dalam pencarian topik atau judul skripsi dan pendeteksi kemoripan dokumen skripsi sangat dibutuhkan. Oleh karena itu judul yang diambil adalah “**Implementasi *Elasticsearch* Untuk Pencarian Dan Menentukan *Similarity* Pada Dokumen Skripsi Di Fakultas Teknologi Informasi”.**

## Rumusan Masalah

Berdasarkan pemaparan dari latar belakang di atas maka rumusan masalahnya adalah sebagai berikut:

1. Bagaimana implementasi *Elasticsearch* dalam pencarian dokumen skripsi?
2. Bagaimana pengaruh proses pencarian skripsi saat sebelum dan setelah diimplementasikannya *Elasticsearch*?
3. Bagaimana cara menentukan kemiripan (*similarity*) dokumen skripsi terhadap penelitian lain?

## Batasan Masalah

Berdasarkan rumusan masalah diatas permasalahan adan di batasi sebagai berikut :

1. Menggunakan *Elasticsearch* 6.5.4
2. Hanya mencari dokumen berupa skripsidi Fakultas Teknologi Informasi.
3. Menggunakan *web service RESTful API*.
4. Aplikasi berbasis *client* dengan menggunakan pemrograman Java.
5. Menggunakan *database NoSQL* dan *SQL*.

## Tujuan Penelitian

Tujuan dari diimplementasikannya *Elasticsearch* dalam pencarian dokumen skripsi secara cepat, efisien, dan akurat yaitu:

1. Untuk menghindari terjadinya plagiarisme dengan mencari kemiripan (*similarity*) dari suatu dokumen penelitian.
2. Untuk membantu mahasiswa dalam mencari referensi saat mulai menyusun skripsi atau membantu dosen dalam melakukan penelitian.

## Metodologi Penelitian

Metode yang digunakan untuk mengambangkan aplikasi ini menggunakan *Model Driven Development.* Yang terdiri dari tahap-tahap di bawah ini:

1. Investigasi Pendahuluan
2. Analisis Masalah
3. Analisis Kebutuhan
4. Perancangan
5. Kontruksi
6. Implementasi

## Sistematika Penulisan

Penggambaran secara umum dan singkat mengenai bab-bab yang ada dalam skripsi ini adalah sebagai berikut.

BAB I : PENDAHLUAN

Bab ini membahas tentang latar belakang, rumusan masalah, batasan masalah, tujuan, manfaat, metode penelitian, dan sistematika penulisan.

BAB II : TINJAUAN PUSTAKA

Bab ini membahas rangkuman informasi yang dihimpun dari pustaka yang relavan dengan masalah yang menjadi objek kajian untuk memperluas basis informasi dalam melakukan penelitian

BAB III : METODOLOGI

Bab ini membahas tentang waktu dan tempat penelitian, alat dan bahan, metode penelitian, metode pengumpulan data, metode perancangan

BAB IV : ANALISIS DAN PERANCANGAN

Bab ini menerangkan tentang analisis yang digunakan sebagai dasar implementasi Elasticsearch pada pencarian skripsi. Disamping itu juga pemodelan sistem yang menggambarkan muatan dan aliran informasinya.

BAB V : IMPLEMENTASI DAN PENGUJIAN

Bab ini berisi implementasi dan pengujian serta hasil pengamatan secara keseluruhan Elasticsearch pada pencarian skripsi.

BAB VI : KESIMPULAN DAN SARAN

Bab ini berisi kesimpulan dan saran terhadap uraian yang telah diberikan pada bab-bab sebelumnya.

# BAB II TINJAUAN PUSTAKA



## Tinjauan Penelitian

Tinjauan penelitian berisi ringkasan dari beberapa jurnal terkait judul dan objek penelitian yang diambil, berikut ini adalah beberapa referensi judul jurnal yang digunakan dalam proses penelitian, yaitu:

1. Pemanfaatan *Elasticsearch* Untuk Temu Kembali Informasi Tugas Akhir (Ardian Prima Atmajaa Dan Susilo Veri Yuliantob, 2018, Jurnal Nasional Teknologi Dan Sistem Informasi)

Penelitian ini membahas tentang bagaimana mengembangkan Sistem temu kembali informasi atau mesin pencari dokumen Tugas Akhir yang terintegasi dengan pusat data yang sudah ada sehingga dapat diakses oleh pihak-pihak yang memerlukan menggunakan perangkat yang terhubung dengan internet, sehingga untuk mengakses informasi yang lebih lengkap hanya menggunakan otentifikasi satu akun yang dimiliki oleh user dari tempat penelitian. Metode rekayasa perangkat lunak yang digunakan yakni *Rapid Aplication Development* (RAD) yang merupakan sebuah model proses pengembangan perangkat lunak yang menekankan pada siklus pengembangan yang sangat pendek dan menghasilkan komponen yang bersifat *reuse* sehingga sangat mengakomodasi pengembangan perangkat lunak secara fleksibel. Hasil pencarian dapat ditampilkan dengan cepat sesuai dengan kata kunci yang diberikan serta dari hasil pencarian tersebut dapat ditampilkan kembali detailnya.

1. Implementasi *Web Service* Pada Sistem Pengindeksan Dan Pencarian Dokumen Tugas Akhir, Skripsi, Dan Praktik Kerja Lapangan (A.A. Gede Yudhi Paramartha; Gusti Ketut Suryaningsih Dan Kadek Yota Ernanda Aryanto, 2016, Jurnal Sains Dan Teknologi).

Penelitian ini yaitu membahas tentang mengembangkan atau penyempurnaan sistem pengindeksan dan pencarian dokumen yang telah dibuat sebelumnya yang berbasis *web*. Pengembangan atau penyempurnaan yang dilakukan yaitu dengan memanfaatkan teknologi. *Web service* pada sistem yang untuk proses pengindeksan dan pencarian dokumen. Penelitian ini menggunakan model pengembangan *waterfall*. Model *waterfall* ini melakukan pendekatan secara terurut yang dimulai dari tahap kebutuhan sistem menuju ke tahap analisis, *design*, *coding*, *testing* atau *verification*, dan *maintenance*. Dengan adanya *web service* pencarian dokumen, maka akan lebih memudahkan pengembang dalam membangun sistem yang ingin memanfaatkan layanan pencarian dokumen. Hasil atau keluaran dari layanan pencarian dokumen tersebut berupa daftar dokumen yang sudah diberikan skor dan diurutkan berdasarkan relevansinya dengan *keyword* pencarian pengguna.

1. Penerapan Metode Term Frequency Inverse Document Frequency (Tf-Idf) Dan Cosine Similarity Pada Sistem Temu Kembali Informasi Untuk Mengetahui Syarah Hadits Berbasis Web (Studi Kasus: Syarah Umdatil Ahkam) (Ria Melita; Victor Amrizal; Hendra Bayu Suseno Dan Taslimun Dirjam, 2018, Jurnal Teknik Informatika Vol 11 No. 2)

Penelitian ini adalah untuk mengetahui tentang penerapan algoritma *similarity* yaitu tf/idf dan cosine similarity untuk pada sistem temu kembali informasi untuk mengetahui syarah hadits berbasis web dan juga menghitung nilai akurasi pada *Stemming* Nazief dan Adriani yang digunakan dalam *text preprocessing.* Metode pengembangan sistem yang digunakan adalah model *Rapid Aplication Development* (RAD). Berdasarkan rumusan masalah yang telah dikemukakan, maka kesimpulan dalam penelitian ini adalah bahwa metode *term frequency inverse document frequency* (tf/idf) dan *cosine similarity* telah berhasil diterapkan dalam sistem dengan baik dimana sistem dapat memberikan *output* berupa dokumen yang relevan yaitu *syarah hadits* sesuai dengan *query* yang di input. kan, dengan melalui 3 tahapan teks *preprocessing* yaitu *tokenizing, stopword removal* atau *filtering*, dan *stemming*.

## Dasar Teori



### *Elasticsearch*

*Elasticsearch* adalah mesin pencari *fulltext* yang *open source* dan alat analisis data yang dikembangkan dengan bahasa pemrograman Java, berbasiskan *Apache Lucene,* dan dapat diskalakan. (Hüseyin Akdoğan, 2015). *Elasticsearch* memanupilasi data dengan menggunakan *REST API* yang dapat digunakan untuk berbagai tugas, yaitu mengelola indeks, mengubah parameter instance, memeriksa node dan status cluster, indeks data, cari data, atau ambil dokumen melalui GET API yang berkonsentrasi pada penggunaan bagian CRUD (*create-retve-update-delete*) dari API, yang memungkinkan untuk menggunakan Elasticsearch dengan cara yang mirip dengan menzggunakan database *NoSQL* (Rafał Kuć dan Marek Rogoziński, 2013). Bagaimanapun, *elasticsearch* jauh lebih baik dari pada hanya Lucene dan lebih dari sekedar “hanya” pencarian full-text. Hal ini juga dapat digambarkan sebagai berikut:

1. Toko dokumen real-time terdistribusi dimana setiap *field* di indeks dan dicari
2. Mesin pencari terdistribusi dengan analisis *real-time*
3. Mampu melakukan penskalaan ke ratusan *server* dan petabyte yang terstruktur dan data tidak terstruktur.

*Elasticsearch* memiliki *query* yang lebih mudah untuk digunakan karena berbasis RESTful. Di *Elasticsearch* tidak ada *transaction* dan dapat membuat struktur indeks tergantung dengan kebutuhan kita. Selain itu dapat diatur untuk menjadi sebuah sistem terdistribusi terhadap sejumlah *server*. (Rafał Kuć dan Marek Rogoziński, 2013). Konsep dasar arsitektur *Elasticsearch*:

1. Index

Indeks adalah tempat logis dimana *Elasticsearch* menyimpan data logis, sehingga dapat dibagi menjadi potongan-potongan kecil. *Elasticsearch* dapat menyimpan banyak indeks yang terletak disatu mesin atau tersebar dibanyak *server*. Setiap indeks dibangun dari satu atau lebih pecahan, dan setiap pecahan dapat memiliki banyak replika.

1. Dokumen

Entitas utama yang disimpan dalam *Elasticsearch* adalah sebuah dokumen. Menggunakan analogi untuk *database relasional*, dokumen adalah deretan data dalam tabel *database*. Dokumen di *Elasticsearch* harus memiliki jenis yang sama untuk semua bidang umum. Ini berarti bahwa semua dokumen dengan bidang yang disebut judul harus memiliki tipe data yang sama untuk itu, misalnya string. Dokumen perlu memiliki pengidentifikasi unik sehubungan dengan tipe dokumen. Ini berarti bahwa dalam indeks tunggal, dua dokumen dapat memiliki pengidentifikasi unik yang sama jika bukan dari jenis yang sama.

1. Tipe Dokumen

Dalam *Elasticsearch*, satu indeks dapat menyimpan banyak objek dengan tujuan berbeda. Jenis dokumen memungkinkan kita dengan mudah membedakan antara objek dalam indeks tunggal. Setiap dokumen dapat memiliki struktur yang berbeda, tetapi dalam penyebaran di dunia nyata, membagi dokumen menjadi beberapa tipe sangat membantu dalam manipulasi data. Perlu diperhatikan bahwa tipe dokumen yang berbeda tidak dapat menetapkan tipe yang berbeda untuk properti yang sama.

1. *Mapping*

Di bagian tentang dasar-dasar pencarian *full-text* dituliskan tentang proses analisis persiapan input teks untuk pengindeksan dan pencarian. Setiap bidang dokumen harus dianalisis dengan benar tergantung pada jenisnya. *Elasticsearch* menyimpan informasi tentang bidang-bidang dalam mapping. Setiap tipe dokumen memiliki pemetaannya sendiri, bahkan jika kita tidak mendefinisikannya secara eksplisit.

Konsep kunci dari Elasticsearch:

1. Node dan cluster

*Elasticsearch* dapat berfungsi sebagai server pencarian tunggal yang berdiri sendiri. Namun demikian, untuk dapat memproses set data yang besar dan untuk mencapai toleransi kesalahan dan ketersediaan tinggi, Elasticsearch dapat dijalankan pada banyak server yang bekerja sama. Secara kolektif, server ini disebut cluster, dan setiap server yang membentuknya disebut node.

1. Shard

Ketika memiliki dokumen dengan jumlah yang besar di dalam satu node kemungkinan tidak cukup baik RAM , kapasitas harddisk, maupun pemrosesan yang tidak memadai, dan ketidakmampuan untuk menanggapi permintaan klien dengan cukup cepat. Dalam kasus seperti itu, data dapat dibagi menjadi bagian yang lebih kecil yang disebut pecahan (di mana setiap pecahan adalah indeks Apache Lucene yang terpisah). Setiap pecahan dapat ditempatkan di server yang berbeda, dan dengan demikian, data Anda dapat tersebar di antara node cluster. Saat Anda kueri indeks yang dibuat dari beberapa pecahan, Elasticsearch mengirimkan kueri ke setiap pecahan yang relevan dan menggabungkan hasilnya sedemikian rupa sehingga aplikasi Anda tidak tahu tentang pecahan. Selain itu, memiliki beberapa pecahan dapat mempercepat naik pengindeksan.

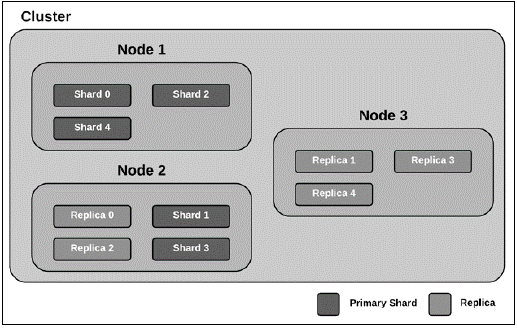
1. Replika

Replika hanyalah salinan tepat pecahan (shard), dan setiap pecahan dapat memiliki nol atau lebih replika. Dengan kata lain, Elasticsearch dapat memiliki banyak pecahan identik dan salah satunya dipilih secara otomatis sebagai tempat operasi yang mengubah indeks diarahkan. Pecahan khusus ini disebut pecahan primer (shard primer), dan yang lain disebut pecahan replika.

1. Gateway

*Elasticsearch* menangani banyak node. Keadaan cluster dipegang oleh gateway. Secara default, setiap node menyimpan informasi ini secara lokal, yang disinkronkan di antara node.

*Elasticsearch* mendistribusikan data ke lebih dari satu indeks Lucene secara fisik. Indeks ini disebut pecahan dan proses distribusi ini disebut Shard. Shard dikelola secara otomatis oleh *Elasticsearch*. Ini adalah unit pekerja tingkat rendah.



Gambar 2.1 Strategi Distribusi

Berikut adalah alur kerja *Elasticsearch*:



Gambar 2.2 Alur Kerja *Elasticsearch*

### *Web Service*

*Web service* adalah aplikasi sekumpulan data (*database*), perangkat lunak (*software*) atau bagian dari perangkat lunak yang dapat diakses secara *remote* oleh berbagai piranti dengan sebuah perantara tertentu. Secara umum, *web service* dapat diidentifikasikan dengan menggunakan URL seperti hanya *web* pada umumnya. Namun yang membedakan *web service* dengan web pada umumnya adalah interaksi yang diberikan oleh *web service*. Berbeda dengan URL *web* pada umumnya, URL *web service* hanya menggandung kumpulan informasi, perintah, konfigurasi atau sintaks yang berguna membangun sebuah fungsi-fungsi tertentu dari aplikasi.

*Web service* dapat diartikan juga sebuah metode pertukaran data, tanpa memperhatikan dimana sebuah *database* ditanamkan, dibuat dalam bahasa apa, sebuah aplikasi yang mengkonsumsi data, dan di platform apa sebuah data itu dikonsumsi. *Web service* mampu menunjang interoperabilitas. Sehingga *web service* mampu menjadi sebuah jembatan penghubung antara berbagai sistem yang ada.

Menurut W3C *Web services Architecture Working* Group pengertian *Web service* adalah sebuah sistem *software* yang di desain untuk mendukung interoperabilitas interaksi mesin ke mesin melalui sebuah jaringan. *Interface web service* dideskripsikan dengan menggunakan format yang mampu diproses oleh mesin (khususnya WSDL). Sistem lain yang akan berinteraksi dengan *web service* hanya memerlukan SOAP, yang biasanya disampaikan dengan HTTP dan XML sehingga mempunyai korelasi dengan standar Web (Web Services Architecture Working Group, 2004).

*Web* pada umumnya digunakan untuk melakukan respon dan request yang dilakukan antara client dan server. Sebagai contoh, seorang pengguna layanan web tertentu mengetikan alamat url web untuk membentuk sebuah request. Request akan sampai pada server, diolah dan kemudian disajikan dalam bentuk sebuah respon. Dengan singkat kata terjadilah hubungan client-server secara sederhana.

Sedangkan pada *web service* hubungan antara *client* dan server tidak terjadi secara langsung. Hubungan antara client dan server dijembatani oleh file *web service* dalam format tertentu. Sehingga akses terhadap database akan ditanggani tidak secara langsung oleh server, melainkan melalui perantara yang disebut sebagai *web service*. Peran dari *web service* ini akan mempermudah distribusi sekaligus integrasi database yang tersebar di beberapa server sekaligus.

### *Apache Lucene*

*Apache Lucene* adalah *library* pencarian Java kuat yang dengan mudah menambahkan pencarian ke aplikasi apa pun. Dalam beberapa tahun terakhir, Lucene menjadi sangat populer dan sekarang menjadi *library* pengambilan informasi yang paling banyak digunakan: ini memperkuat fitur pencarian di balik banyak situs web dan aplikasi desktop. Meskipun ditulis dalam Java, terima kasih untuk itu popularitas dan penentuan pengembang yang bersemangat yang sekarang Anda miliki dapat Anda gunakan sejumlah port atau integrasi ke bahasa pemrograman lain. (Michael McCandless, Erik Hatcher dan Otis Gospodnetic, 2010).

*Lucene* adalah pencarian teks berfitur lengkap. Ini berarti, cukup sederhana: suatu program mencari serangkaian dokumen teks untuk satu atau lebih istilah yang telah ditentukan pengguna. Ini menunjukkan bahwa Lucene tidak semata-mata digunakan dalam konteks world wide web, bahkan jika pencarian sebagian besar ditemukan di sini. Lucene juga dapat digunakan untuk arsip, perpustakaan, atau bahkan di PC desktop. Itu tidak hanya mencari dokumen HTML, tetapi juga berfungsi dengan email dan file PDF.

*Lucene* juga melakukan normalisasi ketika menganalisis data yang tokenisasi adalah bagian. Ini berarti bahwa istilah-istilah tersebut ditulis dalam bentuk standar, mis. semua huruf kapital ditulis dalam huruf kecil. Lucene juga berhasil memilah mereka. Ini berfungsi melalui berbagai algoritma, mis. melalui TF-IDF. Sebagai pengguna, Anda mungkin ingin mendapatkan hasil yang paling relevan atau terbaru terlebih dahulu - algoritma mesin pencari memungkinkan ini.

### *Search Engine*

*Search Engine* adalah program komputer yang dirancang untuk melakukan pencarian atas berkas-berkas yang tersimpan dalam layanan www, ftp, publikasi milis, ataupun *news group* dalam sebuah ataupun sejumlah komputer dalam suatu jaringan. Mesin pencari merupakan perangkat penelusur informasi dari dokumen-dokumen yang tersedia. Hasil pencarian umumnya ditampilkan dalam bentuk daftar yang seringkali diurutkan menurut tingkat akurasi ataupun rasio pengunjung atas suatu berkas yang disebut sebagai hits. Informasi yang menjadi target pencarian bisa terdapat dalam berbagai macam jenis berkas seperti halaman situs *web*, gambar, ataupun jenis-jenis berkas lainnya. Beberapa mesin pencari juga diketahui melakukan pengumpulan informasi atas data yang tersimpan dalam suatu basis data ataupun direktori *web*.

*Open source search engines* adalah kelas penting lain dari sistem yang memiliki tujuan desain yang agak berbeda dari mesin pencari komersial. Tiga sistem yang menarik adalah Lucene, Lemur, dan Galago. Tiga sistem yang diminati adalah Lucene, Lemur, dan Galago. (Croft, Metzler, dan Strohman, 2015).

### *Database*

*Database* adalah kumpulan data yang disimpan secara sistematis di dalam komputer yang dapat diolah atau dimanipulasi menggunakan perangkat lunak (program aplikasi) untuk menghasilkan informasi. Pendefinisian basis data meliputi spesifikasi berupa tipe data, struktur data dan juga batasan-batasan pada data yang kemudian disimpan.

*Database* merupakan aspek yang sangat penting dalam sistem informasi karena berfungsi sebagai gudang penyimpanan data yang akan diolah lebih lanjut. Basis data menjadi penting karena dapat mengorganisasi data, menghidari duplikasi data, menghindari hubungan antar data yang tidak jelas dan juga *update* yang rumit.

Menurut(Robin Nixon, 2014). *Database* adalah kumpulan catatan atau data yang terstruktur yang disimpan dalam sistem komputer dan diatur sedemikian rupa sehingga dapat dengan cepat dicari dan informasi dapat dengan cepat diambil.

Menurut(Ramakrishnan, R. dan Gehrke, J., 2012). *Database* adalah kumpulan data, biasanya menggambarkan kegiatan satu atau lebih banyak organisasi terkait. Misalnya, database universitas mungkin berisi informasi tentang berikut ini:

* Entitas seperti siswa, fakultas, kursus, dan ruang kelas.
* Hubungan antara entitas, seperti pendaftaran siswa dalam kursus, kursus mengajar fakultas, dan penggunaan kamar untuk kursus

### *NoSQL*

*NoSQL* adalah istilah yang dikenal dalam teknologi komputasi untuk merujuk kepada kelas yang luas dari sistem manajemen basis data yang di identifikasikan dengan tidak mematuhi aturan pada model sistem manajemen basis data relasional yang banyak digunakan. *NoSQL* dibuat dengan tujuan khusus untuk model data spesifik dan memiliki skema fleksibel untuk membuat aplikasi modern. *Database NoSQL* dikenal secara luas karena kemudahan pengembangan, fungsionalitas, dan kinerja dalam berbagai skala. *NoSQL* menggunakan berbagai model data, termasuk dokumen, grafik, nilai kunci, dalam memori, dan pencarian. Halaman ini termasuk sumber daya untuk membantu Anda memahami lebih baik database *NoSQL* dan mulai menggunakannya. Selama berpuluh tahun, model data utama yang digunakan untuk pengembangan aplikasi adalah model data relasional yang digunakan oleh database relasional seperti Oracle, DB2, SQL Server, MySQL, dan PostgreSQL. Hingga pada pertengahan hingga akhir tahun 2000 model data lain mulai mendapatkan adopsi dan penggunaan yang signifikan. Untuk membedakan dan mengategorikan kelas database dan model data baru ini, istilah “NoSQL” diciptakan. Sering kali istilah “*NoSQL*” digunakan secara bergantian dengan “nonrelasional”.

*Database* *NoSQL* menggunakan berbagai model data untuk mengakses dan mengelola data, seperti dokumen, grafik, nilai kunci, dalam memori, dan pencarian. Jenis database ini dioptimalkan secara khusus untuk aplikasi yang memerlukan volume data besar, latensi rendah, dan model data fleksibel, yang dicapai dengan mengurangi pembatasan konsistensi data dari database lainnya. Keuntungan dari database NoSQL adalah sebagai berikut:

1. Fleksibilitas: Database NoSQL umumnya menyediakan skema fleksibel yang memungkinkan pengembangan yang lebih cepat dan lebih berulang. Model data fleksibel membuat database NoSQL ideal untuk data yang semi terstruktur dan tidak terstruktur.
2. Skalabilitas: Database NoSQL umumnya didesain untuk meningkatkan skala dengan menggunakan klaster perangkat keras yang terdistribusi alih-alih meningkatkan skala dengan menambah server yang mahal dan robust. Beberapa penyedia layanan cloud menangani aktivitas di balik operasi ini sebagai layanan yang dikelola sepenuhnya.
3. Kinerja tinggi: Database NoSQL dioptimalkan untuk model data spesifik (seperti dokumen, nilai kunci, dan grafik) dan pola akses yang memberikan kinerja yang lebih tinggi dibandingkan jika Anda mencoba mendapatkan fungsionalitas yang mirip dengan database relasional.
4. Fungsionalitas tinggi: Database NoSQL menyediakan API dan jenis data fungsional yang dibuat secara khusus untuk setiap model data yang sesuai.

**Jenis Database NoSQL:**

1. Nilai-kunci: Database nilai kunci dapat dipartisi dan memungkinkan pengembangan horizontal pada skala yang tidak dapat dicapai oleh jenis database lain.
2. Dokumen: Dalam kode aplikasi, data sering diwakilkan sebagai sebuah objek atau dokumen seperti JSON karena ini merupakan model data yang efisien dan intuitif untuk pengembang.
3. Grafik: Database grafik bertujuan agar membuat dan menjalankan aplikasi yang berjalan dengan dataset yang selalu terhubung menjadi lebih mudah.
4. Dalam memori: Aplikasi gaming dan teknologi iklan memiliki kasus penggunaan seperti leaderboard, penyimpanan sesi, dan analisis real-time yang memerlukan waktu respons milidetik dan dapat setiap saat memiliki puncak lalu lintas yang besar.
5. Pencarian: Beberapa output aplikasi dicatat untuk membantu pengembang untuk memecahkan masalah.

### *RESTful Application Programming Interface (API)*

REST (REpresentational State Transfer) merupakan standar arsitektur komunikasi berbasis web yang sering diterapkan dalam pengembangan layanan berbasis web. Umumnya menggunakan HTTP (Hypertext Transfer Protocol) sebagai protocol untuk komunikasi data. REST pertama kali diperkenalkan oleh Roy Fielding pada tahun 2000.

Pada arsitektur REST, REST server menyediakan resources(sumber daya/data) dan REST client mengakses dan menampilkan resource tersebut untuk penggunaan selanjutnya. Setiap resource diidentifikasi oleh URIs (Universal Resource Identifiers) atau global ID. Resource tersebut direpresentasikan dalam bentuk format teks, JSON atau XML. Pada umumnya formatnya menggunakan JSON dan XML.

Menurut (Gormley & Tong, 2015). Semua bahasa lain dapat berkomunikasi dengan *Elasticsearch* melalui port 9200 menggunakan RESTful API, dapat diakses dengan *client* web favorit Anda. Bahkan, seperti yang telah Anda lihat, Anda bahkan dapat berbicara dengan *Elasticsearch* dari *command line* perintah dengan menggunakan perintah *curl*.

*Request* ke *Elasticsearch* terdiri dari bagian yang sama dengan permintaan HTTP:



Bagian yang ditandai dengan < > di atas adalah:

* *VERB*

Metode atau kata kerja HTTP yang sesuai: *GET, POST, PUT, HEAD,* dan *DELETE*. Berikut adalah penjelasan menurut (Alexandros Dallas,2014):

* GET: Untuk mengembalikan representasi sumber daya ntuk mengembalikan representasi resource
* POST: untuk membuat resource
* PUT: untuk memperbarui resource
* DELETE: untuk menghapus resource
* HEAD: Metode HEAD meminta tanggapan yang identik dengan permintaan GET, namun tanpa respon body.
* PROTOKOL

Baik http atau https (jika memiliki proxy https di depan Elasticsearch.)

* HOST

Hostname dari sembarang simpul dalam gugus Elasticsearch Anda, atau localhost untuk suatu simpul di mesin lokal.

* PORT

Port yang menjalankan layanan HTTP Elasticsearch, yang standarnya adalah 9200.

* QUERY\_STRING

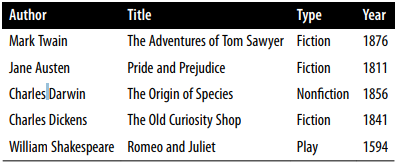
Parameter string-string opsional (misalnya? Cantik akan cukup mencetaknya Respons JSON untuk membuatnya lebih mudah dibaca.)

### *MySQL*

MySQL adalah sebuah perangkat lunak system manajemen basis data SQL (DBMS) yang multithread, dan multi-user. MySQL adalah implementasi dari system manajemen basisdata relasional (RDBMS). Pada saat ini MySQL merupakan database server yang sangat terkenal di dunia, semua itu tak lain karena bahasa dasar yang digunakan untuk mengakses database yaitu SQL. SQL (*Structured Query Language*) pertama kali diterapkan pada sebuah proyek riset pada laboratorium riset San Jose, IBM yang bernama system R. Kemudian SQL juga dikembangan oleh Oracle, Informix dan Sybase.

Menurut(Robin Nixon, 2014). *Database* MySQL berisi satu atau lebih tabel, masing-masing berisi catatan atau baris. Di dalam baris ini terdapat berbagai kolom atau bidang yang berisi data itu sendiri.

Gambar tabel di bawah ini menjukkan isi contoh *database* dari lima publikasi yang merinci dari penulis, judul, jenis, dan tahun publikasi



Gambar 2.3 Contoh *Database*

Setiap baris dalam tabel sama dengan baris dalam tabel MySQL, dan setiap elemen dalam satu baris sama dengan bidang MySQL. Untuk mengidentifikasi secara unik basis data ini, saya akan merujuknya sebagai basis data publikasi dalam contoh-contoh berikut. Dan, seperti yang akan Anda amati, semua publikasi ini dianggap klasik sastra, jadi saya akan memanggil tabel di dalam basis data yang menyimpan detail klasik.

### Java

Java adalah bahasa pemrograman dan platform komputasi pertama kali dirilis oleh *Sun Microsystems* pada tahun 1995. Java merupakan teknologi yang mendasari kekuatan program untuk utilitas, permainan, dan aplikasi bisnis. Java berjalan pada lebih dari 850 juta komputer pribadi di seluruh dunia, dan pada miliaran perangkat di seluruh dunia, termasuk ponsel dan perangkat TV. Salah satu karakteristik Java adalah portabilitas, yang berarti bahwa program komputer yang ditulis dalam bahasa Java harus dijalankan secara sama, pada setiap *hardware / platform* sistem operasi. Hal ini dicapai dengan menyusun kode bahasa Java ke sebuah Java *bytecode*. Pengguna aplikasi biasanya menggunakan Java RuntimeEnvironment (JRE) diinstal pada mesin mereka sendiri untuk menjalankan aplikasi Java, atau dalam *browser web* untuk applet Java. Untuk pembuatan dan pengembangan aplikasi berbasis Java diperlukan *Java Development Kit* (JDK), dimana saat ini pemilik lisensi dari JDK adalah *Oracle Corporation* yang telah secara resmi mengakuisisi *Sun Microsystem* pada awal tahun 2010. Ada beberapa Java *platform* untuk keperluan development, yaitu:

* + Java SE (*Standard Edition*), yang khusus digunakan untuk pengembangan aplikasi- aplikasi pada PC atau workstation.
  + Java ME *(Micro Edition*), yaitu khusus digunakan untuk pengembangan aplikasi-aplikasi yang ada di perangkat mobilespt HP, smartphone, PDA, tablet dsb.
  + Java EE (*Enterprise Edition*), yaitu khusus digunakan untuk pengembangan aplikasi skala besar (*enterprise*), dan aplikasi *web* berbasis java.

### *****JSON (Java Script Object Notation)*****

*JavaScript Object Notation (JSON)* adalah sebuah format data yang memungkinkan aplikasi untuk saling berkomunikasi di dalam sebuah jaringan, yang melalui RESTful API. JSON menyediakan semua bahasa pemrograman modern dengan bantuan untuk menghasilkan dan menerima data (Marss, 2017). Berikut adalah tampilan sederhana dari *JSON:*



Gambar 2.4 Contoh *JSON*

Pertukaran data dengan menggunakan format *JSON* sangat ideal karena *JSON* berbasis teks dan mudah dibaca oleh manusia. *JSON* sering digunakan untuk membagikan data yang terstruktur melalui sebuah jaringan yang disebut serialisasi. *JSON* terbentuk dari dua struktur diantaranya:

1. *Object* dimana data disimpan dengan pasangan *name:value*. *Object* dimulai dengan kurung kurawal buka ({) dan diakhiri dengan kurung kurawal tutup (}). Setiap nama diikuti dengan titik dua (:) dan setiap pasangan nama dan nilai dipisahkan oleh tanda koma (,).
2. *Array* adalah kumpulan data yang diserialisasikan di satu tempat. *Array* dimulai dengan kurung siku buka ([) dan diakhiri dengan kurung kotak tutup (]), dan setiap nilai dipisahkan oleh tanda koma (,).

Elasticsearch menggunakan *JavaScript Object Notation*, atau JSON, sebagai format serialisasi untuk dokumen. Pemrograman JSON didukung oleh sebagian besar bahasa pemrograman, dan telah menjadi format standar yang digunakan oleh gerakan NoSQL. Sederhana, ringkas, dan mudah dibaca. Mengkonversi objek ke JSON untuk pengindeks-an dalam Elasticsearch jauh lebih sederhana daripada proses yang setara dengan struktur tabel datar. Hampir semua bahasa memiliki modul yang akan mengubah struktur data atau objek menjadi JSON untuk, namun rinciannya spesifik untuk setiap bahasa (Gormley & Tong, 2015).

### *Similarity*

*Similarity* atau kemiripan dalam kata adalah mencari kesamaan antara kata inputan dengan dengan kata sumber. Kemiripan yang dimaksud bukan sinonim melainkan kata yang mendekati kata sumber (kata sebenarnya). Beberapa fungsi *similarity* di antaranya adalah *key tape error correction* atau koreksi kesalahan pengetikan.

*Similarity search* adalah istilah yang paling umum digunakan untuk berbagai mekanisme yang berbagi prinsip pencarian ruang (biasanya, sangat besar) objek di mana satu-satunya pembanding yang tersedia adalah kesamaan antara setiap pasangan objek. Ini menjadi semakin penting di zaman repositori informasi besar di mana objek-objek yang terkandung tidak memiliki tatanan alami, misalnya koleksi besar gambar, suara, dan objek digital canggih lainnya.

*Similarity search* melibatkan kumpulan objek (Dokumen, gambar) yang dikarakterisasi oleh koleksi fitur yang relevan dan diwakili sebagai titik dalam ruang atribut dimensi tinggi diberikan *query* dalam bentuk titik di ruang ini, kita diharuskan menemukan objek terdekat (*most similarity*) dengan pertanyaan.

Algoritma *Similarity* standar yang digunakan dalam *Elasticsearch*, yang disebut dengan istilah frekuensi / frekuensi dokumen terbalik atau TF / IDF yang berfungsi sebagai menghitung skor relevansi. Frekuensi istilah menghitung berapa kali suatu istilah muncul dalam bidang yang kita tanyakan dalam dokumen saat ini. Berikut adalah faktor-faktornya:

1. Frekuensi istilah (*Term frequency*)

Berfungsi untuk menunjukkan seberapa sering istilah itu muncul di lapangan. Semakin sering, semakin relevan. *Field* yang berisi lima sebutan untuk istilah yang sama lebih cenderung relevan daripada *field* yang hanya berisi satu penyebutan.

1. Frekuensi dokumen terbalik ( *Inverse document frequency*)

Berfungsi untuk menunjukkan seberapa sering setiap istilah muncul dalam indeks. Semakin sering, semakin tidak relevan. Istilah-istilah yang muncul di banyak dokumen memiliki bobot lebih rendah daripada persyaratan yang lebih umum.

1. Norma panjang lapangan *(Field-length norm)*

Berfungsi untuk menunjukkan berapa lama bidangnya. Semakin lama, semakin kecil kemungkinan bahwa kata-kata di lapangan akan relevan. Sebuah istilah yang muncul di *field* *tittle* pendek memiliki bobot lebih dari istilah yang sama yang muncul di *field* konten yang panjang.

**Model *similarity*  yang tersedia**

Tiga model kesamaan baru yang tersedia adalah sebagai berikut:

* + Model Okapi BM25: Model kesamaan ini didasarkan pada model probabilistik yang memperkirakan probabilitas menemukan dokumen untuk kueri yang diberikan. Untuk menggunakan kesamaan ini di Elasticsearch, Anda harus menggunakan nama BM25. Kesamaan Okapi BM25 dikatakan berkinerja terbaik ketika berhadapan dengan dokumen teks pendek di mana pengulangan istilah sangat merusak nilai keseluruhan dokumen. Untuk menggunakan kesamaan ini, Anda perlu mengatur properti kesamaan untuk bidang menjadi BM25. Kesamaan ini didefinisikan di luar kotak dan tidak perlu properti tambahan untuk ditetapkan.
  + *Divergence from randomness model*: Model kesamaan ini didasarkan pada model probabilistik dengan nama yang sama. Untuk menggunakan kesamaan ini di Elasticsearch, Anda harus menggunakan nama DFR. Dikatakan bahwa divergensi dari model kesamaan keacakan berkinerja baik pada teks yang mirip dengan bahasa alami.
  + Model berbasis informasi: Ini adalah model terakhir dari model kesamaan yang baru diperkenalkan dan sangat mirip dengan divergensi dari model keacakan. Untuk menggunakan kesamaan ini di elasticsearch, Anda harus menggunakan nama IB. Mirip dengan kesamaan DFR, dikatakan bahwa model berbasis informasi berkinerja baik pada data yang mirip dengan teks bahasa alami.

### *Apache Maven*

*Apache Maven* adalah manajemen proyek perangkat lunak dan alat pemahaman. Berdasarkan konsep model objek proyek (POM), *Maven* dapat mengelola pembangunan, pelaporan, dan dokumentasi proyek dari informasi utama..

**Tujuan *Maven***

Tujuan utama *Maven* adalah untuk memungkinkan pengembang untuk memahami keadaan lengkap dari upaya pengembangan dalam periode waktu singkat. Untuk mencapai tujuan ini, ada beberapa bidang yang perlu diperhatikan *Maven:*

* Membuat proses pembangunan menjadi mudah
* Menyediakan sistem pembangunan yang seragam
* Memberikan informasi proyek yang berkualitas
* Memberikan pedoman untuk pengembangan praktik terbaik
* Mengizinkan migrasi transparan ke fitur baru

### *Unified Modeling Language (UML)*

Menurut (Ivar Jacobson, 2000). *Unified Modeling Language* (UML) adalah bahasa standar untuk menulis cetak biru perangkat lunak. UML dapat digunakan untuk memvisualisasikan, menentukan, membuat, dan mendokumentasikan artefak dari sistem intensif perangkat lunak. UML hanya bahasa dan hanya satu bagian dari metode pengembangan perangkat lunak. Ada 3 *diagram* UML yang di gunakan:

1. *Class diagram*

*Class* adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek. *Class* menggambarkan keadaan (atribut/properti) suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (metoda/fungsi). *Class* memiliki tiga area pokok :

1. Nama (dan stereotype)
2. Atribut
3. Metoda

Atribut dan metoda dapat memiliki salah satu sifat berikut :

* + - 1. *Private*, tidak dapat dipanggil dari luar class yang bersangkutan
      2. *Protected*, hanya dapat dipanggil oleh class yang bersangkutan dan anak-anak yang mewarisinya
      3. *Public*, dapat dipanggil oleh siapa saja

1. *Use case diagram*

*Use Case Diagram* adalah gambaran dari beberapa atau semua *actor, use case*, dan interaksi diantaranya yang memperkenalkan suatu sistem*.* Simbol-simbol pada *Use Case Diagram*:

Tabel 2.1 Simbol-Simbol *Use Case Diagram*

|  |  |
| --- | --- |
| **Simbol** | **Keterangan** |
|  | *Actor* **:** Mempresentasikan seseorang atau sesuatu (seperti perangkat, sistem lain) yang berinteraksi dengan sistem. |
|  | *Use Case* **:** Adalah gambaran fungsionalitas dari suatu sistem, sehingga *customer* atau pengguna sistem paham dan mengerti mengenai kegunaan sistem yang akan dibangun. |
|  | *Association* **:** Menghubungkan link antar element. |
|  | <<Include>> **:**Yaitu kelakuan yang harus terpenuhi agar sebuah *event* dapat terjadi, dimana pada kondisi ini sebuah *use case* adalah bagian dari *use case* lainnya |

1. *Activity diagram*

*Activity Diagram* adalah diagram yang menggambarkan *worlflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis.

*Activity diagram* adalah teknik untuk mendeskripsikan logika prosedural, proses bisnis dan aliran kerja dalam banyak kasus. *Activity diagram* mempunyai peranan seperti halnya *flowchart*, akan tetapi perbedaannya dengan *flowchart* adalah *activity diagram* bisa mendukung perilaku pararel sedangkan *flowchart* tidak bisa. (Munawar, 2005).

Tabel 2.2 Simbol-simbol *Activity Diagram*

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Simbol** | **Nama** | **Deskripsi** |
| 1. |  | *Action State* | Menggambarkan keadaan dari suatu elemen dalam suatu aliran aktifitas |
| 2. |  | *State* | Menggambarkan kondisi suatu |
| 3. |  | *Flow Control* | Menggambarkan aliran aktifitas dari suatu elemen ke elemen lain |
| 4. |  | *Initial State* | Menggambarkan titik awal siklus hidup suatu elemen |
| 5. |  | *Final State* | Menggambarkan titik akhir yang menjadi kondisi akhir suatu elemen |

1. *Squence Diagram*

*Sequence diagram* menurut Munawar (2005) adalah grafik dua dimensi dimana obyek ditunjukkan dalam dimensi horizontal, sedangkan *lifeline* ditunjukkan dalam dimensi vertikal.

*Sequence diagram* menunjukan interaksi objek 28 dengan waktu yang direpresentasikan dalam grafik dua dimensi. Dimensi vertical menunjukan waktu, digambarkan melintang kebawah. Dimensi horizontal menunjukkan jenis peranan yang menggambarkan individu objek dalam diagram *collaboration*. Durasi aktivitas objek ditunjukkan oleh *lifeline* yang berupa garis putus-putus. Message ditampilkan sebagai panah dari satu *lifeline* sebuah objek ke *lifeline* objek yang lainnya.

Tabel 2.3 Simbol-Simbol *Sequence Diagram*

|  |  |  |
| --- | --- | --- |
| **Simbol** | **Nama** | **Keterangan** |
|  | Aktor / *Actor* | * Orang, proses, atau sistem lain yang berinteraksi dengan system informasi dan mendapat manfaat dari system. * Berpartipasi secara berurutan dengan mengirimkan atau mengirimkan pesan. * Di tempatkan di bagian atas diagram. |
|  | Objek / *Object* | * Berpartipasi secara berurutan dengan mengirimkan atau menerima pesan. * Di tempatkan di bagian atas diagram |
|  | Garis Hidup / *Lifeline* | * Menandakan kehidupan objek selama urutan. * Diakhiri tanda X pada titik di mana kelas tidak lagi berinteraksi. |
|  | Objek yang berinteraksi / *Activation* | * Fokus kontrol adalah persegi panjang yang sempit panjang ditempatkan di atas sebuah garis hidup. * Menandakan ketika suatu objek mengirim atau menerima pesan. |
|  | Pesan / *Message* | * Objek mengirim satu pesan ke objek lainnya. |
|  | Membuat / *Create* | * Menyatakan suatu objek membuat objek yang lain, arah panah mengarah pada objek yang dibuat. |
|  | Masukkan / *Message Send* | * Menyatakan bahwa suatu objek mengirim masukkan ke objek lainnya arah panah megarah pada objek yang dikirimi. |
|  | Keluaran / *Message Return* | * Objek atau metode menghasilkan suatu kembalian ke objek tertentu, arah panah mengarah pada objek yang menerima kembalian. |
|  | Hasil Akhir / *Destroy* | * Menyatakan suatu objek mengakhiri hidup objek yang lain, arah panah mengarah pada objek yang diakhiri, sebaiknya jika ada create maka ada destroy. |

### *Postman*

*Postman* adalah sebuah aplikasi HTTP *client* yang merupakan plugin dari browser Chrome. Fungsi Postman adalah untuk pengecekan web service. Postman dapat menampilkan hasil dari HTTP request yang kompleks sekalipun dengan cepat. (Alifa dan Alief, 2015)

***Postman***adalah sebuah aplikasi (berupa plugin) untuk browser chrome, fungsinya adalah sebagai REST Client atau istilahnya adalah aplikasi yang digunakan untuk melakukan uji coba REST API yang telah kita buat*.*

*Postman* adalah toolchain penting bagi pengembang API untuk berbagi, uji coba, dokumentasi dan memantau API. Lebih dari 3 juta insinyur dan pengembang di seluruh dunia menggunakan Postman untuk membangun perangkat lunak yang terhubung melalui API-cepat, mudah dan akurat. Ide untuk Postman muncul ketika para pendiri bekerja bersama-sama, dan frustrasi dengan alat yang ada untuk pengujian API. Mereka merasa harus ada bahasa yang lebih baik bagi para pengembang untuk berkomunikasi tentang API. Hal ini menyebabkan penciptaan *Postman. Postman* membantu pengembang membangun, menguji, dan mendokumentasikan lebih cepat.

### *Kibana*

*Kibana* adalah platform analisis dan visualisasi *open source* yang dirancang untuk bekerja dengan *Elasticsearch*. *Kibana* digunakan untuk mencari, melihat, dan berinteraksi dengan data yang disimpan dalam indeks *Elasticsearch*. Anda dapat dengan mudah melakukan analisis data tingkat lanjut dan memvisualisasikan data Anda dalam berbagai bagan, tabel, dan peta.

*Kibana* membuatnya mudah untuk memahami volume data yang besar. Antarmuka berbasis *browser* yang sederhana memungkinkan untuk dengan cepat membuat dan berbagi dasbor dinamis yang menampilkan perubahan pada permintaan *Elasticsearch* secara *real time*.

Menyiapkan *Kibana* sangat mudah. Anda dapat menginstal Kibana dan mulai menjelajahi indeks *Elasticsearch* Anda dalam hitungan menit - tanpa kode, tidak ada infrastruktur tambahan yang diperlukan.

### *NetBeans IDE*

*NetBeans IDE* adalah sebuah *Integrated Development Environment* untuk para pengembang software. Pengguna NetBeans IDE bisa mendapatkan segala tools yang diperlukan untuk membuat aplikasi-aplikasi desktop profesional, perusahaan, web, dan mobile dengan bahasa Java, C/C++, dan bahkan bahasa-bahasa dinamis seperti PHP, JavaScript, Groovy, dan Ruby. Netbeans IDE mudah diinstal dan digunakan langsung di luar kotaknya dan berjalan di banyak platforms termasuk Windows, Linux, Mac OS X dan Solaris.

Fungsi Java Netbeans IDE sendiri adalah untuk membuat dan mengembangkan sebuah aplikasi Desktop, contoh dari aplikasi netbeans adalah seperti aplikasi yang ada di toko toko sembako. selain untuk membuat dan mengembangkan sebuah aplikasi Netbeans juga digunakan oleh programer untuk mencompile, linker, debugger DLL. karena IDE sendiri secara global berarti “editor”. IDE adalah sebuah lingkungan terintegritas yang menyediakan semua kebutuhan programer.

Fitur – fitur yang terdapat pada Netbeans antara lain:

* *Smart Code Completion*, untuk mengusulkan nama variabel dari suatu tipe, melengkapi keyword dan mengusulkan tipe parameter dari sebuah method.
* *Bookmarking*, fitur yang digunakan untuk menandai baris yang suatu saat hendak kita modifikasi.
* *Go to commands*, fitur yang digunakan untuk jump ke deklarasi variabel, source code atau file yang ada pada project yang sama.
* *Code generator*, jika kita menggunakan fitur ini kita dapat meng-generate constructor, setter and getter method dan yang lainnya.
* *Error stripe*, fitur yang akan menandai baris yang *error* dengan memberi *highlight* merah.

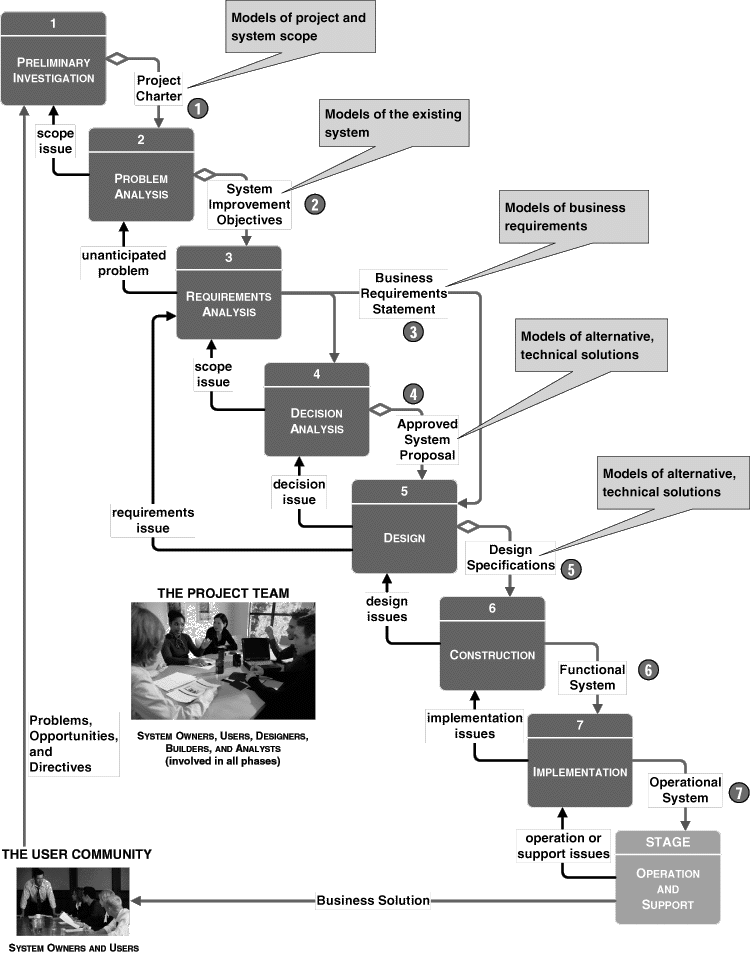
### *Model Driven Development (MDD)*

*Model Driven Development (MDD)* merupakan suatu teknik yang menekankan penggambaran model untuk memvisualisasikan dan menganalisis masalah, mendefinisikan proses bisnis, dan merancang sistem informasi. Ada beberapa pendekatan dalan teknik pemodelan:

1. *Structured Analysis Design* → berorientasi proses
2. *Information engineering (IE)* →berorientasi data
3. *Object-oriented analysis and design* (OOAD)→menggaungkan orientasi proses dan data ke dalam bentuk objek.

Keunggulan model-driven:

1. Spesifikasi persyaratan lebih menyeluruh dan didokumentasikan dengan baik
2. Persyaratan bisnis dan desain sistem lebih mudah divalidasi dengan gambar daripada dengan kata-kata
3. Lebih mudah mengidentifikasi, mengkonseptualkan, dan menganalisis solusi-solusi teknis alternatif
4. Spesifikasi desain cenderung solid, stabil, dapat beradaptasi, dan fleksibel karena berbasis model dan dianalisis lebih menyeluruh sebelum dibangun
5. Sistem dapat dikonstruksikan dengan lebih tepat pertama kali saat dibangun.



Gambar 2.5 *Model Driven Development*

### *Framework* *Spring*

*Framework spring* *adalah framework open source* berbasis java yang menyediakan infrastruktur yang komprehensif dalam mengembangkan aplikasi java dengan mudah dan cepat. *Spring* pertama kali ditulis dan dirilis oleh Rod Johnson dengan lisensi Apache 2.0 pada bulan Juni 2003. Spring akan membantu programmer dalam pengembangan aplikasi dengan build yang sederhana, portable, cepat dan sistem berbasis JVM yang fleksibel. *Spring* dapat digunakan untuk melakukan pengaturan deklarasi manajemen transaksi, remote access dengan menggunakan RMI atau layanan web lainnya, fasilitas mailing, dan beragam opsi untuk pengaturan data ke database. Spring juga memungkinkan kita menggunakan hanya modul-modul tertentu sehingga kita tidak perlu menggunakan semua modul spring dalam aplikasi apabila tidak diperlukan.

*Spring Framework* menggunakan teknik pemrograman yang sederhana, model pemrograman dengan Spring cukup mudah, namun rapi. Hal Ini memudahkan bagi para developer pemula untuk mempelajarinya. Jika mempelajari Spring, para developer berevolusi menjadi developer yang lebih baik. Hal ini karena framework Spring mendorong untuk membuat kode program yang modular dan independen. Hasilnya, kode program yang dibuat akan lebih rapi, mudah dites, dan terstruktur dengan baik.

Dengan menggunakan Spring Framework, developer dapat membuat aplikasi enterprise ataupun web. Selain itu juga, para developer dapat membuat aplikasi untuk keamanan dan aplikasi yang terkait dengan big data. Spring termasuk portabel karena aplikasi yang dikembangkan dapa tberjalan pada JVM manapun. Untuk menggunakan Spring, developer dapat menggunakan Eclipse atau Netbeans.

# BAB III METODOLOGI PENELITIAN



## Kerangka Berfikir

Berikut adalah langkah-langkah yang di lakukan untuk mencapai tujuan dari penelitian ini adalah sebagai berikut:



Gambar 3.1 Kerangka Berfikir

## 

## Deskripsi

### Investigasi Pendahuluan

1. Observasi

Penelitian ini merupakan kolaborasi dari sebuah *project* pengembangan sistem informasi *repository* skripsi di Fakultas Teknologi Informasi. Penelitian ini terdiri dari penelitian berjudul “Implementasi *Elasticsearch* Untuk Pencarian Dan Menentukan *Similarity* Pada Dokumen Skripsi Di Fakultas Teknologi Informasi” yang lebih menekankan pada pencarian dan menetukan kemiripan (*similarity*) pada suatu dokumen, dan penelitian “Aplikasi *Repositiry* Karya Ilmiah di Fakultas Teknologi Informasi Universitas Bale Bandung” yang lebih menekankan pada penyimpanan (*storaging*) dan *index*. Dalam pembuatan *project* dipimpin oleh bapa Moch. Ridwan, S.T, S.Kom.

*Project* ini diimplementasikan di Fakultas Teknologi Informasi Universitas Bale Bandung yang beralamat di Jl. Raden AA Wiranatakusumah No.7, Baleendah, Bandung, Jawa Barat 40375.

1. Wawancara

Wawancara dimulai dengan menemui divisi perpustakaan yaitu Bapak Yusuf Muharom, S.Kom, untuk meminta izin dalam melakukan wawancara tentang masalah yang ada di perpustakaan, penulis sudah menentukan bagian yang akan di jadikan objek penelitian yaitu tentang Skripsi. Saat melakukan wawancara penulis mengajukan beberapa pertanyaan di antaranya adalah sebagai berikut:

1. Apakah pengumpulan laporan skripsi masih manual?

Jawaban:

Masih manual jadi laporan skripsi di jilid lalu di kumpulkan ke perpustakaan, untuk kedepannya di kumpilkan ke *repository* juga.

1. Apakah pengumpulan laporan skripsi sudah tersusun rapih?

Jawaban:

Sudah oleh mahasiswa yang melakukan KP di FTI, di urutkan sampai tahun 2016 dari 2008, dan yang belum di kumpulkan yaitu angkatan 2017 dan 2018

1. Bagaimana cara mahasiswa mencari referensi ilmiah dari skripsi sebelum di buatkan aplikasi untuk membantu pencarian?

Jawaban:

Manual, langsung lihat daftar skripsi di perpus yang sesuai dengan judul yang mahasiswa inginkan dan tidak pakai aplikasi pendukung.

1. Apa permasalahan yang saat ini ada pada pengolahan data skripsi?

Jawaban:

Banyak, kesatu banyak skripsi yang belum di sahkan oleh penguji dan pembimbing, masih maen simpan sebelum ditanda tangan oleh pembimbing, dekan dan ketua prodi, kalau masalah mengurutkan di urutkan berdasarkan tahun saja

1. Bagaimana caranya mengetahui jika ada skripsi yang dianggap plagiat?

Jawaban:

Contohnya salah satu mahasiswa, dua jurnal di gabungkan jadi satu dan di klaim bahwa itu adalah proposalnya, makan nya di ulang pada saat sidang proposal skripsinya, berdasarkan pengakuan dosen yang menyidang mahasiswa tersebut, dan mahasiswa tersebut mengikuti sidang susulan. Dan kalau ketahuan plagiat kan di skripsinya juga ada lembar keterangan plagiat.

1. Studi Pustaka

Setelah melakukan wawancara penulis melakukan studi pustaka terhadap permasalahan yang ada di Fakultas Teknologi Informasi Universitas Bale Bandung, kemudia penulis melakukan studi pustaka yaitu dengan mencari beberapa jurnal terkait objek penelitian sebagai penunjang dan juga mencari beberapa referensi dari website, jurnal, buku digital (*ebook*), dll yang berkaitan dengan masalah yang ada di Fakultas Teknologi Informasi Universitas Bale Bandung. Berikut adalah judul dari referensi:

1. Pemanfaatan *Elasticsearch* Untuk Temu Kembali Informasi Tugas Akhir (Ardian Prima Atmajaa Dan Susilo Veri Yuliantob, 2018, Jurnal Nasional Teknologi Dan Sistem Informasi)
2. Implementasi *Web Service* Pada Sistem Pengindeksan Dan Pencarian Dokumen Tugas Akhir, Skripsi, Dan Praktik Kerja Lapangan (A.A. Gede Yudhi Paramartha; Gusti Ketut Suryaningsih Dan Kadek Yota Ernanda Aryanto, 2016, Jurnal Sains Dan Teknologi).
3. Penerapan Metode Term Frequency Inverse Document Frequency (Tf-Idf) Dan Cosine Similarity Pada Sistem Temu Kembali Informasi Untuk Mengetahui Syarah Hadits Berbasis Web (Studi Kasus: Syarah Umdatil Ahkam) (Ria Melita; Victor Amrizal; Hendra Bayu Suseno Dan Taslimun Dirjam, 2018, Jurnal Teknik Informatika Vol 11 No. 2)
4. Buku dengan judul “Elasticsearch in Action” karangan Radu Gheorghe, Matthew Lee Hinman dan Roy Russo
5. Buku dengan judul “Elasticsearch The Definitive Guide” karangan Clinton Gormley dan Zachary Tong
6. Buku dengan judul “JSON at Work” karangan Tom Marrs
7. Buku dengan judul ”RESTful Web Services with Dropwizard” karangan Alexandros Dallas
8. Buku dengan judul “Learning PHP, MySQL & JavaScript” karangan Robin Nixon. Pada buku ini topik yang diambil yaitu MySQL untuk melengkapi dasar teori tentang topik MySQL.

### Analisis Masalah

Berikut adalah hasil analisis masalah yang ada di Fakultas Teknologi Informasi Universitas Bale Bandung:

1. Mahasiswa saat memelakukan pencarian skripsi belum bisa secara cepat.
2. Kesalahan yang terjadi saat mahasiswa mencari referensi dari karya ilmiah orang lain dapat dianggap plagiat.

Berdasarkan masalah yang ada di Fakultas Teknologi Informasi Universitas Bale Bandung penulis dan rekan melakukan studi banding dengan pihak yang lebih berpengalaman, maka ada masukan bahwa *elasticsearch* yang lebih sesuai berdasarkan hasil studi banding, da melakukan diskusi dengan pembimbing maka penulis menemukan solusi bahwa *elasticsearch*-lah yang akan digunakan. Untuk memudahkan dalam manajemen *elasticsearch* dan mengembangakan aplikasi untuk *elasticsearch* dan juga membantu pelaporan yang dapat memvisulaisasikan dalam bentuk grafik dan sebagainya penulis disarankan pembimbing lapangan menggunakan kibana.

Jika di dalam tahap ini masih ada kekurangan maka akan kembali ke tahap investigasi awal sampai mendapat hasil yang tepat, kemudian di lanjutkan ketahat berikutnya

### Analisis Kebutuhan

1. Kebutuhan Prosedur

Prosedur sebelumnya mahasiswa mengimpulkan laporan skripsi dalam bentuk CD dan juga *hardcopy* untuk pengumpulan laporan juga mahasiswa di wajibkan untuk membuat file laporan skripsi dengan format pdf, kemudian mahasiswa mengunggah atau menyimpan laporan dengan file pdf ke *repository*.

Prosedur pencarian dokumen dan menentukan kesamaan (*similarity*) dokumen skripsi, dalam prosedur ini mahasiswa melakukan pencarian dengan cara mengaksesnya dari *repository* penyimpanan laporan skripsi dan untuk menentukan kemiripan dokumen akan di lakukan upload dokumen yang nantinya akan muncul persentasi kemiripan pada suatu dokumen.

1. Aplikasi Utama

Berikut adalah analisis kebutuhan yang di perlukan untuk menjawab semua pertanyaan yang ada di Fakultas Teknologi Informasi Universitas Bale Bandung, yaitu:

1. Membuat aplikasi yang dapat mencari skripsi atau penelitian secara cepat.
2. Membuat aplikasi pengecekan kemiripan (*similarity*) satu dokumen dengan dokumen penelitian yang lainnya.
3. Perankat Lunak Pendukung

Pembuat aplikasi utama dibutuhkan menggunakan aplikasi atau perangkat pendukung yakni sebagai berikut:

1. *Elasticsearch* adalah *engine* yang digunakan untuk mengindex, mencari dokumen.
2. *Kibana,* aplikasi pendukung yangdigunakan untuk ujicoba dalam mencari, melihat, dan berinteraksi dengan dokumen yang disimpan dalam indeks *Elasticsearch.*
3. *Postman,* digunakan untuk pengujian API yang cara kerjanya sama seperti *kibana*.
4. *Database MySQL*, digunakan untuk menyimpan *database* dari judul di setiap dokumen.
5. *Java* adalahbahasa pemrograman yang digunakan dalam pembuatan aplikasi utama. Menggunakan java pada NetBeans IDE karena library-nya lebih lengkap.
6. *Spring,* merupakan *framework* yang digunakan dalam pembuatan aplikasi di java.
7. *Apache Maven*, digunakan dalam konfigurasi proyek aplikasi Java.
8. NetBeans IDE, digunakan untuk membuat dan mengembangkan sebuah aplikasi dengan bahasa java.
9. Perangkat Keras Pendukung

Setelah melakukan analisis kebutuhan langkah berikutnya adalah melakukan installasi *elasticsearch* server dan kibana yang berperan membantu memanajemen *elasticsearch* dan mengembangkan aplikasi *elasticsearch* di laptop dengan spesifikasi minimal sebagai berikut:

1. Intel(R) Celeron(R) N4000 CPU @ 1.10GHz
2. RAM 4GB
3. HDD 500 GB

Dalam penelitian ini menggunakan spesifikasi minimal. Oleh karena nya dalam melakukan penelitian ini penulis dan rekan membatasi jumlah data yang tidak terlampau besar. Dibuktikan dengan sekian halaman dapat melakukan penyimpanan, memastikann bahwa berupa pdf bisa tersimpan.

### Perancangan

Pada tahap selanjutnya yaitu membuat desain aplikasi untuk pencarian dan pengecekan kemiripan (*similarity*) yang dibagi menjadi beberapa tahapan yaitu sebagai berikut:

1. Perancangan Aplikasi

Pada tahap ini perancangan di lakukan dengan membuat *Unified Modeling Language* (UML). Perancangan dengan *Unified Modeling Language* (UML) ini bertujuan untuk merancang aplikasi dengan mudah dalam proses pengembangan dan juga untuk memvisualisasikan, membangun dan mendokumentasikan dari sebuah sistem pembangunan perangkat lunak berbasis objek, maka dibuatlah:

1. *Use Case* untuk pencarian dokumen dan kemiripan dokumen
2. *Class Diagram* untuk pencarian berdasarkan autor, judul, kata kunci, abstrak, isi, tahun, topik, studi kasus, prodi.
3. *Activity Diagram* untuk pencarian dan menentukan kemiripan dokumen
4. *Sequence Diagram* untuk kemunculan menu pencarian dokumen dan menentukan kemiripan dokumen
5. Perancangan *Database*

Selanjutnya adalah dilakukan perancangan *database* yang bertujuan untuk menyimpan data laporan penelitian, agar dapat membantu saat pencarian. Setelahnya dilakukan perancangan tampilan untuk menggambarkan aplikasi yang akan di kembangkan.

Jika dalam tahap ini terdapat kesalahan atau kekurangan maka akan kembali ke tahap sebelumnya sampai mendapat hasil yang tepat, selanjutnya baru di lanjutkan ke tahap berikutnya.

### Kontruksi

Setelah tahap *Design* (Perancangan) selesai maka tahap selanjutnya adalah pembuatan aplikasi. Pada pembuatan aplikasi digunakan perangkat lunak dan bahasa pemrograman sebagai berikut:

1. *Elasticsearch,* digunakan untuk mengindex, mencari dokumen.
2. *Kibana,* digunakan sebagai visualisasi, penggunaan *command* API, pengujian API, dan berinteraksi dengan data yang disimpan dalam indeks *Elasticsearch.*
3. *Postman,* digunakan untuk pengujian API yang cara kerjanya sama seperti *kibana.*
4. *Database MYSQL*, digunakan untuk menyimpan data yang terdiri dari dokumen, *autor*, *publisher*.
5. *Java,* digunakan untuk memunculkan *user interface*, *coding* untuk menjalankan API yang ada di *elasticsearch,* penghubung antara *user interface* dengan *elasticsearch,* menghubungkan *elasticsearch*  dengan *database* di *MySQL*.
6. *Spring,* merupakan *framework* yang digunakan dalam pembuatan aplikasi di java
7. *Apache Maven*, digunakan dalam konfigurasi proyek aplikasi Java.
8. NetBeans IDE, digunakan untuk membuat dan mengembangkan sebuah aplikasi dengan menggunakan bahasa pemrograman Java.

### Implementasi

Sebelum implementasi dilakukan maka di awali melakukan penelitian terlebih dahulu terhadap aplikasi yang sudah dibuat. Pengujian terhadap aplikasi yang sudah dibuat, yaitu dilakukan pengujian dengan menggunakan metode *black box* yaitu untuk menguji fungsionalitas dari suatu aplikasi. Dalam pengujian API juga memakai kibana dan postman.

Jika di dalam tahap ini masih ada kekurangan maka akan kembali ke tahap kontruksi sampai mendapat hasil *coding* yang tepat, kemudian di lanjutkan ketahap berikutnya.

### Pembuatan Laporan

Pada tahap ini merupakan tahap terakhir dalam melakukan penelitian yang terdiri dari 6 bab. Berikut adalah sistematika penulisan:

BAB I : PENDAHLUAN

Bab ini membahas tentang latar belakang, rumusan masalah, batasan masalah, tujuan, manfaat, metode penelitian, dan sistematika penulisan.

BAB II : TINJAUAN PUSTAKA

Bab ini membahas rangkuman informasi yang dihimpun dari pustaka yang relavan dengan masalah yang menjadi objek kajian untuk memperluas basis informasi dalam melakukan penelitian

BAB III : METODOLOGI

Bab ini membahas tentang waktu dan tempat penelitian, alat dan bahan, metode penelitian, metode pengumpulan data, metode perancangan

BAB IV : ANALISIS DAN PERANCANGAN

Bab ini menerangkan tentang analisis yang digunakan sebagai dasar implementasi Elasticsearch pada pencarian skripsi. Disamping itu juga pemodelan sistem yang menggambarkan muatan dan aliran informasinya.

BAB V : IMPLEMENTASI DAN PENGUJIAN

Bab ini berisi implementasi dan pengujian serta hasil pengamatan secara keseluruhan Elasticsearch pada pencarian skripsi.

BAB VI : KESIMPULAN DAN SARAN

Bab ini berisi kesimpulan dan saran terhadap uraian yang telah diberikan pada bab-bab sebelumnya.

# DAFTAR PUSTAKA

*Pengertian Fungsi Netbeans*. (2017, Juli). Dipetik Mei 20, 2019, dari Freakprogrammer: http://www.freakprogrammer.com/2017/07/pengertian-fungsi-netbeans.html

*What is Maven?* (2019, 06 16). Dipetik Mei 2019, 2019, dari Apache Maven Project: http://maven.apache.org/what-is-maven.html

A.A. Gede Yudhi Paramartha, G. K. (2016). IMPLEMENTASI WEB SERVICE PADA SISTEM PENGINDEKSAN DAN PENCARIAN DOKUMEN TUGAS AKHIR, SKRIPSI, DAN PRAKTIK KERJA LAPANGAN. *Jurnal Sains dan Teknologi, Vol. 5, No. 2*, 775-784.

Akdoğan, H. (2015). *Elasticsearch Indexing.* Birmingham: Packt Publishing Ltd.

Alambiyah, W. (2015, Maret). *Wahidin Alambiyah*. Dipetik Mei 17, 2019, dari Pengertian Web Service dan Web Server: //wahidin-alambiyah-19.blogspot.com/2015/03/pengertian-web-service-dan-web-server.html

Allamaraju, S. (2010). RESTful Web Services Cookbook. Sebastopol: O’Reilly Media, Inc.

*Apa itu database NoSQL?* (t.thn.). Dipetik Juni 2019, 16, dari AWS: https://aws.amazon.com/id/nosql/

Ardian Prima Atmajaa, S. V. (2018). Pemanfaatan Elasticsearch untuk Temu Kembali Informasi Tugas Akhir. *Jurnal Nasional Teknologi dan Sistem Informasi, VOL. 04 NO. 03* , 160-167.

Clinton Gormley, Z. T. (2015). *Elasticsearch The Definitive Guide.* Sebastosol: O’Reilly Media, Inc.

Dallas, A. (2014). *RESTful Web Services with Dropwizard.* Brimingham: Packt Publishing Ltd.

Djuandi, F. (2015). Dipetik Mei 19, 2019, dari Belantara Foundation: https://www.belantara.or.id/document/resource/partnership-programs-in-west-kalimantan-province-resource.pdf

Fatimah, U. (2014, Maret). *UML : Activity Diagram.* Dipetik Mei 2019, 20, dari UMIALFAH: http://fatimahumi.blogspot.com/2014/03/uml-activity-diagram.html

Feridi. (2019, Januari 21). *Mengenal RESTful Web Services* . Dipetik Mei 2019, 13, dari Codepolitan: https://www.codepolitan.com/mengenal-restful-web-services

Gupta, Y. (2015). *Kibana Essentials.* Brimingham: Packt Publishing Ltd.

J. Whitten, L. B. (2007). *System Analysis and Design Methods.* New York: McGraw-Hill.

Johnson. Rod, a. a. (2014, Januari 23). *Spring Framework Documentation.* Dipetik Juni 16, 2019, dari http://docs.spring.io/spring/docs/current/springframework-reference/pdf/spring-framework-reference.pdf

Marrs, T. (2017). *JSON at Work.* Sebastopol: O’Reilly Media, Inc.

Melita, R., Amrizal, V., Suseno, H. B., & Dirjam, T. (2018). Penerapan Metode Term Frequency Inverse Document Frequency (Tf-Idf) Dan Cosine Similarity Pada Sistem Temu Kembali Informasi Untuk Mengetahui Syarah Hadits Berbasis Web (Studi Kasus: Syarah Umdatil Ahkam). *Jurnal Teknik Informatika, Vol 11 No. 2*.

*Mesin pencari web*. (t.thn.). Dipetik Mei 2019, 13, dari Wikipedia: https://id.wikipedia.org/wiki/Mesin\_pencari\_web

Michael Mccandless, E. H. (2010). *Lucene in Action.* Stamford,: Manning Publications Co.

Nixon, R. (2014). *Learning PHP, MySQL & JavaScript.* Sebastopol: O’Reilly Media.

*Pengertian Database*. (t.thn.). Dipetik Mei 15, 2019, dari TermasMedia: https://www.termasmedia.com/lainnya/software/69-pengertian-database.html

Radu Gheorghe, M. L. (2016). *Elasticsearch in Action.* New York: Manning Publications.

Rafał Kuć, M. R. (2014). *Elasticsearch Server.* Birmingham: Packt Publishing.

Robby Roynardo, F. E. (2017, Oktober 4). *Framework Spring Java*. Dipetik Juni 2019, 16, dari BINUS UNIVERSITY School of Computer Science: https://socs.binus.ac.id/2017/10/04/framework-spring-java/

Rohman, A. (2017, Februari 9). *Documentation & Testing API dengan Postman part 1*. Dipetik Juni 16, 2019, dari Medium.com: https://medium.com/skyshidigital/documentation-testing-api-dengan-postman-part-1-5d33e430dca7 Ardani Rohman

Sonatype. (2008). *Maven: The Definitive Guide.* Sebastopol: O’Reilly Media.

*Spring Framework - Overview*. (t.thn.). Dipetik Juni 2019, 16, dari Tutorials Point Simple Easy Learning: https://www.tutorialspoint.com/spring/spring\_overview.htm

Utomo, D. B. (2018, September 19). Dipetik Juni 11, 2019, dari Similarity: https://prezi.com/q6swh\_cn9xc8/similarity/

Utomo, D. B. (2018, September 19). *Similarity.* Dipetik Juni 2019, 11 , dari Prezi: https://prezi.com/q6swh\_cn9xc8/similarity/

W. Bruce Croft, D. M. (2015). *Search Engi nes Information Retrieval in Practice.* Pearson Education, Inc.

Wijaya, T. (.-D. (t.thn.). Dipetik 11 , dari Information System Lecture Notes: http://trisnadi169.blogspot.com/2009/09/model-driven-development.html

Wijaya, T. (2009, 09). *Model-Driven Development*. Dipetik Juni 11, 2019, dari Information System Lecture Notes: http://trisnadi169.blogspot.com/2009/09/model-driven-development.html