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

SKRIPSI

Karya Tulis sebagai Syarat untuk Memperoleh
Gelar Sarjana Komputer dari Fakultas Teknologi Informasi
Universitas Bale Bandung

Disusun oleh :

MAESURI FAUZIAH

NIM. C1A150028

PROGRAM STRATA 1

TEKNIK INFORMATIKA

FAKULTAS TEKNOLOGI INFORMASI

UNIVERSITAS BALE BANDUNG

BANDUNG

2019

LEMBAR PERSETUJUAN PEMBIMBING

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

Disusun oleh:

MAESURI FAUZIAH
NIM. C1A150028

Telah diterima dan disetujui untuk memenuhi persyaratan mencapai gelar
SARJANA KOMPUTER

Pada

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI
UNIVERSITAS BALE BANDUNG

Baleendah, Agustus 2019

Disetujui oleh:

Pembimbing 1	Pembimbing 2
Yudi Herdiana, S.T, M.T	Nurul Imamah, S.T., M.T
NIK. 04104808008	NIK. 04104808121

LEMBAR PERSETUJUAN PENGUJI

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

Disusun oleh:

MAESURI FAUZIAH
NIM. C1A150028

Telah diterima dan disetujui untuk memenuhi persyaratan mencapai gelar
SARJANA KOMPUTER

Pada

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI
UNIVERSITAS BALE BANDUNG

Baleendah, Agustus 2019

Disetujui oleh:

Penguji 1	Penguji 2
Yaya Suharya, S.KOM.,M.T.	Zen Munawar.S.T.,M.T
NIDN. 0407047706	NIDN. 04022037002

LEMBAR PENGESAHAN PROGRAM STUDI

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

Disusun oleh:

MAESURI FAUZIAH
NIM. C1A150028

SKRIPSI ini telah diterima dan disetujui untuk memenuhi persyaratan mencapai gelar

SARJANA KOMPUTER

Pada

PROGRAM STUDI TEKNIK INFORMATIKA

FAKULTAS TEKNOLOGI INFORMASI

UNIVERSITAS BALE BANDUNG

Baleendah, Agustus 2019

Mengetahui,	Mengesahkan,
Dekan,	Ketua Program Studi
Yudi Herdiana, S.T, M.T	Yaya Suharya, S.Kom, M.T
NIK. 04104808008	NIK. 01043170007

LEMBAR PERNYATAAN KEASLIAN

Saya yang bertanda tangan dibawah ini:

Nama : MAESURI FAUZIAH

NIM : C1A150028

Judul Skripsi : **IMPLEMENTASI *ELASTICSEARCH* UNTUK PENCARIAN DAN MENENTUKAN SKOR *SIMILARITY* PADA PROPOSAL SKRIPSI DI FAKULTAS TEKNOLOGI INFORMASI UNIVERSITAS BALE BANDUNG**

Menyatakan dengan sebenarnya bahwa penulisan skripsi ini berdasarkan hasil penelitian, pemikiran dan pemaparan asli dari saya sendiri, baik untuk naskah laporan maupun kegiatan *programming* yang tercantum sebagai bagian dari skripsi ini. Jika terdapat karya orang lain, saya mencantumkan sumber yang jelas.

Pernyataan ini saya buat dengan sesungguhnya dan apabila dikemudian hari terdapat penyimpangan dan ketidak benaran dalam pernyataan ini, maka saya bersedia menerima sanksi akademik berupa pencabutan gelar yang telah diperoleh karena karya tulis ini dan sanksi lain sesuai dengan peraturan yang berlaku di FAKULTAS TEKNOLOGI INFORMASI UNIVERSITAS BALE BANDUNG.

Demikian surat pernyataan ini saya buat dalam keadaan sadar tanpa paksaan dari pihak manapun.

Baleendah, Agustus 2019

Yang membuat pernyataan,

MAESURI FAUZIAH

NIM. C1A150028

ABSTRACT

The development of Information Technology nowdays is so rapid that make us thirst for the speed, efficiency and accuracy. In a college or university, the data of papers or theses from previous years students can be useful for students who need scientific references in composing paper or thesis. Also, before start to compose the paper or thesis, students must submit a proposal related to the title that has been taken. Therefore, the submitted proposal must be the result of the author's thought. Before starting the work it is necessary to determine the similarity of the thesis proposal to the previous paper or thesis document must be known. With Elasticsearch, it can help users in searching paper or thesis documents, because Elasticsearch is a full text search engine which open source and as a data analysis tool that can also be scaled (scalable). Moreover, Elasticsearch also was developed with the Java programming language powered by Apache Lucene which is also a search engine database that has a querylow level. Elasticsearch has an easier query to use because it uses the RESTful web service and is also one of the databases that enter the NoSQL world

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

ABSTRAK

Perkembangan Teknologi Informasi saat ini begitu pesat sehingga menyebabkan kita haus akan kecepatan, efisiensi dan akurasi yang tepat. 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 sebelum memulai penyusunan skripsi harus mengajukan proposal terkait judul yang akan diambil maka dari itu sebelumnya proposal yang diajukan haruslah hasil pemikiran penulis. Sebelum memulai pengerjaan perlu adanya penentuan kemiripan (*similarity) proposal* skripsi terhadap dokumen skripsi harus diketahui. 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,* Proposal, *Similarity*.

KATA PENGANTAR

Puji dan syukur penulis panjatkan atas kehadirat Allah SWT yang mana berkat limpahan Rahmat dan Karunia-NYA penulis dapat menyelesaikan Skripsi ini dengan balik dan tepat pada waktunya yang berjudul *"*IMPLEMENTASI ELASTICSEARCH UNTUK PENCARIAN DAN MENENTUKAN SKOR *SIMILARITY* PADA PROPOSAL SKRIPSI DI FAKULTAS TEKNOLOGI INFORMASI UNIVERSITAS BALE BANDUNG*"* dengan tepat pada waktunya 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. Nurul Imamah, S.T., M.T Dosen Pembimbing Skripsi yang telah membantu dalam pembuatan laporan Skripsi.
5. Keluarga yang sangat, amat, saya cintai yang telah memberikan dukungan dan do'a untuk kelancaran dalam proses pembuatan laporan ini.
6. Dan juga teman-teman yang saling membantu dalam senang maupun susah serta memberikan dukungan satusama lain.

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 kesempurnaan laporan ini.

Bandung, Agustus 2019

Penulis

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR LAMPIRAN

BAB I

PENDAHULUAN

1.1 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 merupakan persyaratan untuk mendapatkan status sarjana (S1) di setiap Perguruan Tinggi Negeri (PTN) maupun Perguruan Tinggi Swasta (PTS) yang ada di Indonesia. 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 secara langsung datang ke perpustakaan lalu menuju ke rak penyimpanan skripsi yang tersimpan dalam bentuk cetak atau *hardcopy*, dalam pencarian menelaah satu-persatu judul skripsi, untuk menemukan judul yang sesuai memerlukan waktu yang lama. Mahasiswa juga mencari tahu apakah penelitiannya sudah pernah dilakukan atau tidak. Proses awal saat 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 di atas maka perlu diterapkan *similarity* dengan menggunakan e*lasticsearch* sebagai alternatif dengan harapan diterapkannya e*lasticsearch* ini jauh lebih baik dibandingkan dengan menggunakan konsep dengan *database* pada umumnya. *Similarity* atau kemiripan dapat dilakukan dengan pencarian *fulltext* dengan menggunakan *elasticsearch.* Keluaran dari e*lasticsearch* berupa dokumen yang sudah diberikan skor yang didapat untuk menemukan nilai kemiripan dokumen dan diurutkan berdasarkan relevansi dokumen dengan kata kunci yang diinputkan pengguna.

Sebuah aplikasi yang menerapkan *elasticsearch* untuk mempermudah membangun aplikasi pencarian topik atau judul skripsi dan pendeteksi *similarity* dokumen proposal skripsi . Oleh karena itu judul yang diambil adalah "**Implementasi *Elasticsearch* Untuk Pencarian Dan Menentukan *Similarity* Pada Proposal Skripsi Di Fakultas Teknologi Informasi".**

1.2 Batasan Masalah

Berdasarkan rumusan masalah diatas permasalahan akan dibatasi sebagai berikut :

1) Menggunakan *Elasticsearch* 6.5.4
2) Hanya mencari dokumen berupa judul skripsi di Fakultas Teknologi Informasi.
3) Membandingkan kemiripan proposal skripsi terhadap skripsi terdahulu.
4) Menggunakan *web service RESTful API*.
5) Aplikasi berbasis *client* dengan menggunakan pemrograman Java.
6) Menggunakan *database NoSQL* dan *SQL*.
7) Hanya bisa menjalankan Client dalam satu waktu.
8) Hanya menampilkan skor *similarity* proposal skripsi terhadap skripsi yang sudah ada.

9) Hanya *single server* untuk uji coba dalam pencarian menggunakan *elasticsearch*

1.3 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*) proposal skripsi terhadap penelitian lain?

1.4 Tujuan Penelitian

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

1) Untuk mengetahui kemiripan (*similarity*) dari suatu proposal terhadap dokumen penelitian.
2) Untuk membantu mahasiswa dalam mencari referensi saat mulai menyusun skripsi atau membantu dosen dalam melakukan penelitian.

1.5 Metodologi Penelitian

1.5.1 Metode Pengumpulan Data

Metode pengumpulan data pada yang dilakukan dalam penelitian ini yaitu meliputi :

1) Observasi, peneliti melakukan pengamatan langsung ke Fakultas Teknologi Informasi Universitas Bale Bandung mengenai objek penelitian.

2) Wawancara, peneliti melakukan tanya jawab mengenai permasalahan yang ada pada objek penelitian.
3) Studi pustaka, peneliti mencari pustaka terkait objek penelitian.

1.5.2 Metode Pengembangan Sistem

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

1.6 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 tentang rangkuman informasi dari penelitian- penelitian yang telah dilakukan lalu menentukan dari pustaka yang relavan dengan masalah yang menjadi objek kajian untuk memperluas basis informasi dalam melakukan penelitian.

BAB III : METODOLOGI

Bab ini membahas tentang kerangka berfikir terhadap penelitian ini, dan mendeskripsikan krangka berfikir dari penelitian.

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

2.1 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, 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.

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).

   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.

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)

   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*.

2.2 Dasar Teori

2.2.1 *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 menggunakan 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) Penyimpanan 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).

1. 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.. Jika dalam basis data relasional, indeks seperti tabel. Namun, struktur indeks disiapkan untuk pencarian teks lengkap yang cepat dan efisien dan, khususnya, tidak menyimpan nilai asli.

2) 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.

3) 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.

4) 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.

2. 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.

    2) 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 ketidak mampuan 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 dapat tersebar di antara node cluster. Saat 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.

    3) Replika

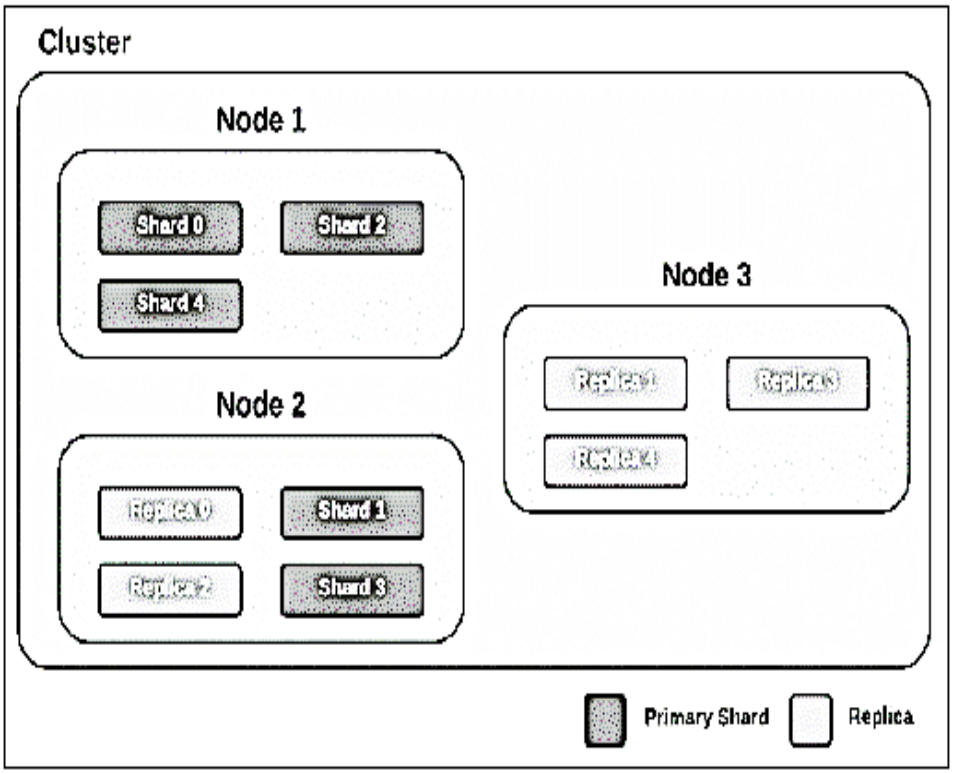
        Replika hanyalah salinan tepat pecahan (shard), dan setiap pecahan dapat memiliki nol atau lebih replika. Dengan kata lain, Elasticsearch dapat memiliki banyak pecahan identik dan salah satunya dipilih secara otomatis sebagai tempat operasi yang

mengubah indeks diarahkan. Pecahan khusus ini disebut pecahan primer (shard primer), dan yang lain disebut pecahan replika.

4) 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 Shard Elasticsearch

3. Pemetaan struktur indeks (*Index structure mapping*)

Setiap data memiliki struktur sendiri, beberapa sangat sederhana, dan beberapa termasuk hubungan objek yang rumit, dokumen turunan, dan properti bersarang. Dalam setiap kasus, kita perlu memiliki skema dalam Elasticsearch yang disebut pemetaan

yang menentukan bagaimana data terlihat. Tentu saja, kita dapat menggunakan skema-kurang dari Elasticsearch, tetapi kita bisa dan biasanya ingin menyiapkan pemetaan di muka, jadi kami tahu bagaimana data ditangani.

4. Querying Elasticsearch

Sejauh ini dalam melakukan *query* menggunakan *REST API* dan permintaan sederhana atau permintaan *GET*. Demikian pula, ketika mengubah indeks, kami juga menggunakan *REST API* dan mengirim data terstruktur *JSON* ke *Elasticsearch*. Terlepas dari jenis operasi, apakah itu perubahan pemetaan atau pengindeksan dokumen, menggunakan badan permintaan terstruktur *JSON* untuk menginformasikan *Elasticsearch* tentang detail operasi.

1) *Simple Query*

Cara paling sederhana untuk meminta *Elasticsearch* adalah dengan menggunakan permintaan permintaan URI. Misalnya, untuk mencari kata kejahatan di bidang judul, Anda dapat mengirim kueri menggunakan perintah berikut:

```
curl -XGET 'localhost:9200/library/book/_search?pretty'
```

Mengirim kueri menggunakan DSL permintaan agak berbeda, mengirim GET (POST juga diterima jika alat atau pustaka tidak mengizinkan badan permintaan pengiriman dalam permintaan GET HTTP) Permintaan HTTP ke titik akhir _search REST seperti sebelumnya dan menyertakan kueri dalam badan permintaan. Berikut ini adalah perintah pencarian dengan menggunakan *Query DSL* :

```
curl -XGET 'localhost:9200/library/book/_search?pretty' -d '{
   "query" : {
      "query_string" : { "query" : "title:crime" }
   }
}'
```

Berikut ini adalah hasilnya:

```
{
      "took" : 4,
      "timed_out" : false,
      "_shards" : {
            "total" : 5,
            "successful" : 5,
```

```
        "failed" : 0
    },
    "hits" : {
        "total" : 1,
        "max_score" : 0.5,
        "hits" : [ {
            "_index" : "library",
            "_type" : "book",
            "_id" : "4",
            "_score" : 0.5,
            "_source" : {
                "title" : "Crime and Punishment",
                "otitle" : "Преступле́ние и наказа́ние",
                "author" : "Fyodor Dostoevsky",
                "year" : 1886,
                "characters" : [ "Raskolnikov", "Sofia Semyonovna Marmeladova" ],
                "tags" : [ ],
                "copies" : 0,
                "available" : true
            }
        } ]
    }
}
```


2) *Paging* dan ukuran hasil

Elasticsearch memungkinkan untuk mengontrol berapa banyak hasil yang ingin di dapatkan (paling banyak) dan dari mana hasil yang ingin kita mulai. Berikut ini adalah dua sifat tambahan yang bias atur di badan permintaan:

- from: Properti ini menentukan dokumen dari mana ingin mendapatkan hasil. Nilai standarnya adalah 0, yang berarti bahwa ingin mendapatkan hasil dari yang pertama dokumen.
- size: Properti ini menentukan jumlah maksimum dokumen yang inginkan sebagai hasil dari satu permintaan (yang standarnya adalah 10). Misalnya, jika hanya tertarik pada hasil agregasi dan tidak peduli dengan dokumen yang dikembalikan oleh kueri, dapat mengatur parameter ini ke 0.

  Jika ingin permintaan mendapatkan dokumen mulai dari item kesepuluh dalam daftar dan mengambil 20 dokumen, berikut adalah perintahnya:

```
curl -XGET 'localhost:9200/library/book/_search?pretty' -d '{
    "from" : 9,
    "size" : 20,
    "query" : {
        "query_string" : { "query" : "title:crime" }
    }
```

```
}'
```

3) *Query Fuzzy*

Queri fuzzy memungkinkan untuk menemukan dokumen yang memiliki nilai yang mirip dengan yang di berikan dalam kueri. Kesamaan istilah dihitung berdasarkan algoritma edit jarak. Jarak pengeditan dihitung berdasarkan ketentuan yang di berikan dalam kueri dan terhadap dokumen yang dicari. Permintaan ini bisa memakan memori ketika datang ke Sumber CPU, tetapi dapat membantu saat membutuhkan pencocokan fuzzy; misalnya, saat pengguna membuat kesalahan ejaan. Dalam contoh, asumsi bahwa alih-alih kejahatan, pengguna memasukkan kata crme ke kotak pencarian dan ingin menjalankan bentuk fuzzy query yang paling sederhana. Kueri seperti itu akan terlihat seperti ini:

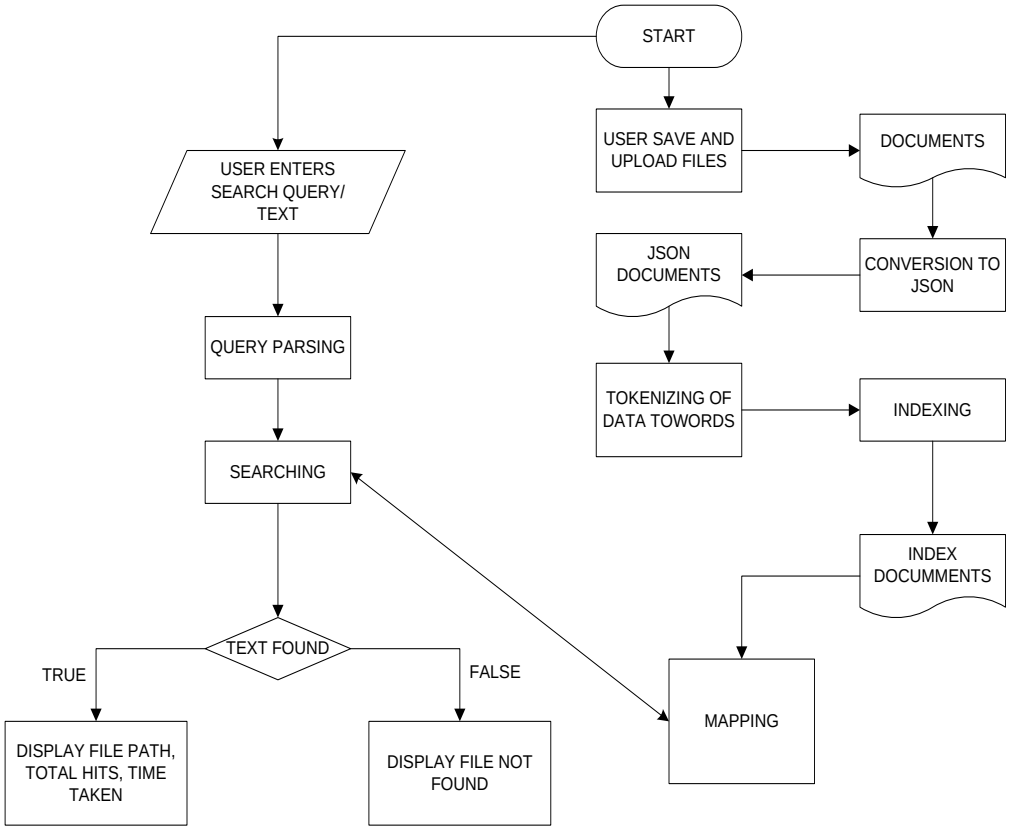
```
{
    "query" : {
        "fuzzy" : {
            "title" : "crme"
        }
    }
}
```

Berikut ini adalah respon hasil dari kueri:


```
{
    "took" : 81,
    "timed_out" : false,
    "_shards" : {
        "total" : 5,
        "successful" : 5,
        "failed" : 0
    },
    "hits" : {
        "total" : 1,
        "max_score" : 0.5,
        "hits" : [ {
            "_index" : "library",
            "_type" : "book",
            "_id" : "4",
            "_score" : 0.5,
            "_source" : {
                "title" : "Crime and Punishment",
                "otitle" : "Преступле́ние и наказа́ние",
                "author" : "Fyodor Dostoevsky",
                "year" : 1886,
                "characters" : [ "Raskolnikov", "Sofia Semyonovna Marmeladova" ],
                "tags" : [ ],
                "copies" : 0,
```

```
                "available" : true
        }
        } ]
}}
```

Berikut adalah alur kerja *Elasticsearch*:



Gambar 2.2 Alur Kerja *Elasticsearch*

2.2.2 *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.

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.

2.2.3 *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).

2.2.4 *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.

2.2.5 *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

2.2.6 *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.

2.2.7 *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:

```
curl -X<VERB> '<PROTOCOL>://<HOST>/<PATH>?<QUERY_STRING>' -d '<BODY>'
```

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.)

2.2.8 *XAMPP*

XAMPP adalah perangkat lunak gratis, yang mendukung banyak sistem operasi, merupakan kompilasi dari beberapa program untuk menjankan fungsinya sebagai *server* yang berdiri sendiri, yang terdiri atas program Apache *HTTP Server*, *MySQL database*, dan penterjemah bahasa yang ditulis dengan bahasa programmeraan PHP dan Perl. XAMPP adalah nama yang merupakan singkatan dari X (empat sistem operasi apapun), Apache, *MySQL*, PHP dan Perl. Program ini tersedia dalam *GNU General Public License* dan gratis. XAMPP merupakan *web server* yang mudah digunakan yang dapat mampu melayani halaman dinamis. Saat ini, XAMPP tersedia untuk sistem operasi *Microsoft Windows, Linux, Sun Solaris* dan *Mac OS*.

Sebagai suatu software yang bertindak sebagai web server layaknya hosting sesungguhnya tentu saja melibatkan banyak bagian – bagian penting yang ada pada XAMPP.

2.2.9 *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

Author	Title	Type	Year
Mark Twain	The Adventures of Tom Sawyer	Fiction	1876
Jane Austen	Pride and Prejudice	Fiction	1811
Charles Darwin	The Origin of Species	Nonfiction	1856
Charles Dickens	The Old Curiosity Shop	Fiction	1841
William Shakespeare	Romeo and Juliet	Play	1594

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.

2.2.10 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.

2.2.11 *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).

1. Sintak JSON

   Format data JSON menyajikan objek JSON sebagai tanda kurung kurawal dan daftar properti yang dipisahkan koma:

   {
   property1 ,
   ...
   propertyN

   }

   Berikut adalah tampilan sederhana dari *JSON:*

```
{
  "conference": "OSCON",
  "speechTitle": "JSON at Work",
  "track": "Web APIs"
}
```

Gambar 2.4 Contoh *JSON*

2. Type Data JSON
   - Number: Angka desimal bertanda tangan yang berisi bagian fraksional dan dapat menggunakan notasi eksponensial (E). JSON tidak mengizinkan bukan-bukan (seperti NaN), juga tidak membuat perbedaan antara integer dan titik apung. Lebih lanjut, JSON tidak mengenali format oktal dan heksadesimal. (Meskipun JavaScript menggunakan format floating-point presisi ganda untuk semua nilai numerik, bahasa lain yang menerapkan JSON dapat menyandikan angka secara berbeda.)
   - String: Urutan nol atau lebih karakter Unicode. String dibatasi dengan tanda kutip ganda dan dukungan a backslash lolos dari sintaks.
   - Boolean: Salah satu dari nilai-nilai itu benar atau salah.
   - Array: Daftar yang terurut dari nilai nol atau lebih, masing-masing yang mungkin dari jenis apa pun. Array menggunakan braket persegi

     notasi dengan elemen dipisahkan koma.

- Objek: Kumpulan properti tanpa urutan di mana nama (juga disebutkunci) adalah string. Karena objek dimaksudkan untuk mewakili array asosiatif, direkomendasikan, meskipun tidak diharuskan, bahwa setiap kunci menjadi unik dalam suatu objek. Objek dibatasi denganmenguatkan dan menggunakan koma untuk memisahkan masing-masing properti. Dalam setiap properti, karakter titik dua memisahkan kunci dari nilainya.
- Null: Nilai kosong, menggunakan kata kunci null.

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:

a. *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 (,).
b. *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).

2.2.12 *Similarity*

Fungsi Kemiripan atau s*imilarity* adalah menghitung kesamaan dan ketidak samaan antara dua objek yang diobservasi. Objek yang di maksud disini adalah komunitas yang saling berbeda. Ludwig & Reynolds (1988) menyatakan bahwa kemiripan suatu komunitas dengan komunitas lain dapat dinyatakan dengan *similarity coefficients. Similarity coefficients* memiliki nilai yang bervariasi antara 0 (jika komunitas benar-benar berbeda) hingga 1 (jika kedua komunitas identik).

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.

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.

2) 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.

3) 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.

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.

2.2.13 *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 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

2.2.14 *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 :

   a. Nama (dan stereotype)
   b. Atribut

c. Metoda

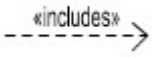
Atribut dan metoda dapat memiliki salah satu sifat berikut :

a. *Private*, tidak dapat dipanggil dari luar class yang bersangkutan
b. *Protected*, hanya dapat dipanggil oleh class yang bersangkutan dan anak-anak yang mewarisinya
c. *Public*, dapat dipanggil oleh siapa saja

2. *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

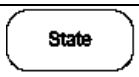
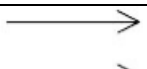
3. *Activity diagram*

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*

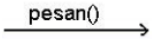
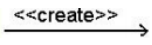
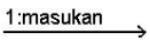
No	Simbol	Nama	Deskripsi
1.		*Action State*	Menggambarkan keadaan dari suatu elemen dalam suatu aliran aktifitas
2.		*State*	Menggambarkan kondisi suatu
3.		*Flow* *Control*	Menggambarkan aliran aktifitas dari suatu elemen ke elemen lain
4.		*Initial State*	Menggambarkan titik awal siklus hidup suatu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

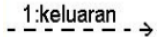
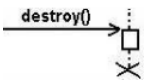
4. Squence Diagram

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 /	• Menyatakan bahwa suatu objek

	Message Send	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.

2.2.15 Kibana

Kibana adalah plugin visualisasi data open source untuk Elasticsearch. Ini memberikan kemampuan visualisasi di atas konten yang diindeks pada cluster Elasticsearch. Pengguna dapat membuat bar, garis, dan sebaran plot, atau diagram lingkaran dan peta di atas volume data yang besar.

Kibana juga menyediakan alat presentasi, yang disebut sebagai Kanvas, yang memungkinkan pengguna untuk membuat slide deck yang menarik data langsung langsung dari Elasticsearch.

Kombinasi Elasticsearch, Logstash, dan Kibana, disebut sebagai "Elastic Stack" tersedia sebagai produk atau layanan. Logstash menyediakan aliran input ke Elasticsearch untuk penyimpanan dan pencarian, dan Kibana mengakses data untuk visualisasi seperti dasbor. Elastic juga menyediakan paket "Beats" yang dapat dikonfigurasi untuk menyediakan visualisasi dan dashboard Kibana yang sudah dibuat sebelumnya tentang berbagai basis data dan teknologi aplikasi.

2.2.16 *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.

2.2.17 *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.

2.2.18 *Model Driven Development (MDD)*

Teknik pengembangan berbasis model (MDD) menekankan gambar model untuk membantu memvisualisasikan dan menganalisis masalah, mendefinisikan kebutuhan bisnis, dan merancang sistem informasi. Analisis dan desain sistem terstruktur - berpusat pada proses Teknik informasi (IE) - berpusat pada data Analisis dan desain berorientasi obyek (OOAD) - terpusat pada objek (integrasi data dan masalah proses) Rute model driven development.

Tahapan yang dipakai pada penelitian adalah sebagai berikut:

1) *Preliminary investigation* (investigasi awal)
2) *Problem analysis* (Analsis masalah)

3) *Requirements analysis* (Analisis Kebutuhan)
4) *Design* (Desain)
5) *Construction* (Kontruksi)
6) *Implementation* (implementasi)

1. *Preliminary Investigation Phase*

   Tahap ini merupakan tahap awal dari pengembangan sistem. Fase ini berisikan investigasi awal ketika ingin merancang sebuah sistem, seperti wawancara, tinjauan langsung dan mempelajari dokumen perusahaan.

   Tujuan dari tahap ini ialah menjawab pertanyaan mengenai apakah proyek ini cukup berharga untuk diperhatikan. Untuk menjawab pertanyaan ini perlu didefinisikan terlebih dahulu masalah, kesempatan, dan resiko-resiko dalam melanjutkan proyek. Kerangka kerja PIECES dapat digunakan untuk menjawab pertanyaan ini namun hasilnya bukanlah solusi permasalahan melainkan kategori-kategori masalah (dengan asumsi bahwa proyek ini berharga untuk diperhatikan) menetapkan rincian proyek yang akan menetapkan lingkup, kebutuhan dan hambatan proyek, anggota proyek, biaya, dan jadwal.

2. *Problem Analysis Phase*

   Problem Analysis ialah menganalisa masalah-masalah yang terdapat di lapangan. Tahap ini merupakan pengembangan dari tahap pertama. Pada tahap ini dilakukan analisis terhadap sistem yang telah ada saat itu. Tahap ini memberikan pemahaman yang lebih dalam bagi tim proyek mengenai permasalahan yang dihadapi. Analisis ini dilakukan untuk menjawab pertanyaan apakah keuntungan yang diperoleh setelah pemecahan masalah lebih besar daripada biaya yang dikeluarkan.

   Input utama dari tahap ini adalah project charter dari tahap sebelumnya. Informasi yang digunakan dalam memperlajari permasalahan yang dihadapi adalah fakta-fakta yang terdapat dalam

sistem, masalah, akibat, penyebab dari permasalahan, dan spesialis IT yang merancang sistem yang telah ada.

Output yang dihasilkan adalah system improvement objectives yang menyatakan kriteria bisnis yang akan digunkana untuk mengevaluasi sistem. Kadang-kadang dilakukan repesentasi pada tahap ini.

3. *Requirement Analysis Phase*

Requirement Analysis ialah melakukan analisa terhadap kebutuhan perusahaan. Pekerjaan pada tahap ini adalah mendefinisikan apa saja yang perlu dilakukan oleh sistem, apa yang dibutuhkan dan diinginkan oleh pengguna dari sistem baru.

Tahap ini memerlukan perhatian yang besar karena jika terjadi kesalahan dalam menerjemahkan kebutuhan dan keinginan pengguna sistem maka dapat mengakibatkan adanya rasa tidak puas pada sistem final dan perlu diadakan modifikasi yang tentunya akan kembali mengeluarkan biaya.

a. Input dari tahap ini adalah system improvement objectives yang dihasilkan pada tahap sebelumnya. Pada tahap ini, tim akan mengumpulkan dan mendiskusikan kebutuhan dan prioritas berdasarkan informasi yang diperoleh dari kuesioner, wawancara, dan rapat-rapat. Tantangannya adalah untuk memvalidasi semua kebutuhan informasi.
b. Output yang dihasilkan dari tahap ini adalah business requirement statement. Tahap ini pun merupakan tahap yang penting karena dapat menimbulkan ketidakpuasan dari pengguna sistem yang merasa kebutuhannya tidak terpenuhi. Tim proyek harus dapat membedakan antara apa yang dibutuhkan oleh pengguna dan bagaiman sebaiknya sistem yang baru bekerja.

4. *Desain Phase*

Setelah diperoleh proposal sistem yang disetujui, maka dapat mulai dilakukan proses desain dari sistem target. Tujuan dari tahap ini adalah untuk mentransformasikan business requirement statement menjadi spesifikasi desain untuk proses construksi. Dengan kata lain, tahap desain menyatakan bagaimana teknologi akan digunakan dalam

sistem yang baru. Tahap ini memerlukan ide dan opini dari pengguna, vendor, dan spesialis IT.

Pada akhir tahap ini masih terdapat beberapa alternatif keputusan mengenai proyek walaupun pembatalan proyek jarang dilakukan pada tahap ini (kecuali benar-benar over budget atau sangat terlambat dari jadwal). Perubahan lingkup menjadi lebih kecil masih dapat terjadi. Selain itu, mungkin juga terjadi perubahaan ulang jadwal untuk menghasilkan solusi yang lebih lengkap.

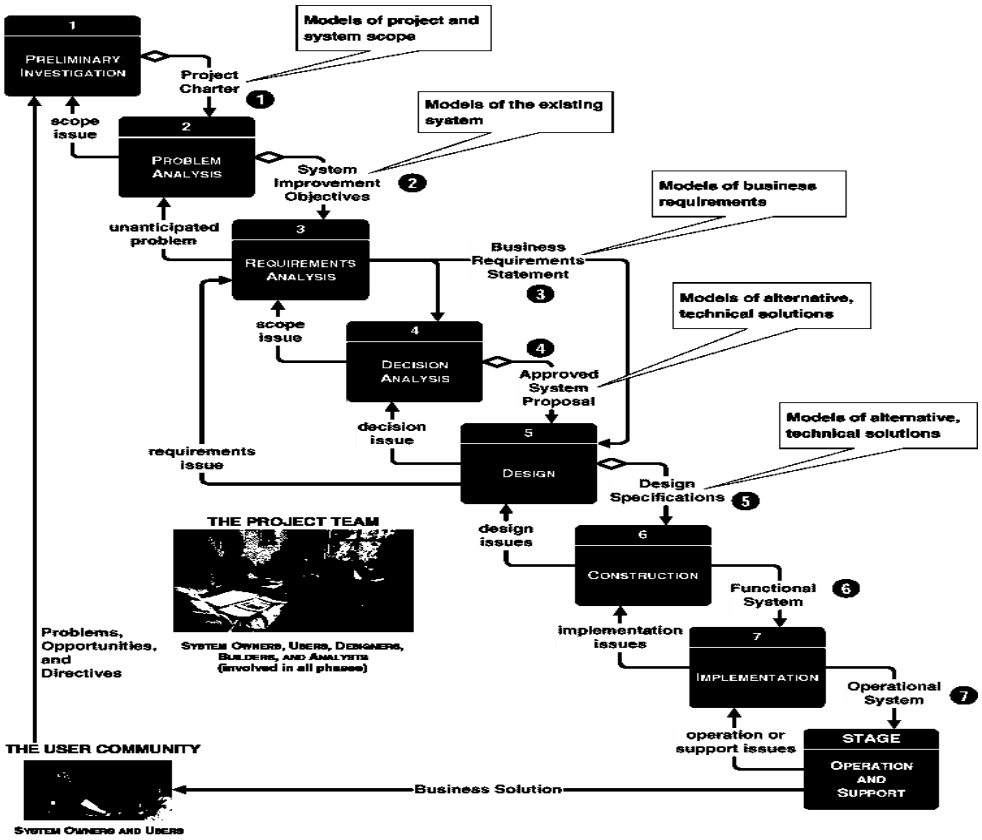
5. *Construction Phase*

Construction Phase ialah tahapan melaksanakan pengujian pada komponen sistem secara individu dan sistem secara keseluruhan.

Tujuan dari tahap ini adalah :

a) Membangun dan menguji sistem yang memenuhi business requirement dan spesifikasdi desain
b) Mengimplementasikan penghubung antara sitem baru dan sistem lama, termasuk instalasi dari software yang dibeli atau disewa
c) Pada tahap ini dilakukan konstruksi basis data, program aplikasi, dan penghubung antara sistem dan pengguna. Beberapa dari komponen ini telah ada sebelumnya.

Setelah dilakukan pengujian, maka sistem dapat mulai diimplementasikan



Gambar 2.5 *Model Driven Development*

2.2.19 *Flowmap*

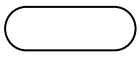
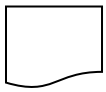
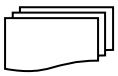
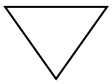
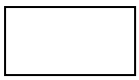
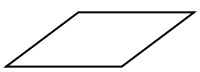
Flowmap atau bagan alir adalah bagan yang menunjukan aliran di dalam program atau prosedur sistem secara logika. Flowmap ini berfungsi untuk memodelkan masukan, keluaran, proses maupun transaksi dengan menggunakan simbol-simbol tertentu. Pembuatan flowmap ini harus dapat memudahkan bagi pemakai dalam memahami alur dari sistem atau transaksi. Adapun pedoman-pedoman dalam pembuatan flowmap adalah sebagai berikut:

1. *Flowmap* sebaiknya digambarkan dari atas ke bawah dan mulai dari bagian kiri dari suatu halaman.
2. Kegiatan di dalam *flowmap* harus ditunjukan dengan jelas.
3. Harus ditunjukan dari mana kegiatan akan dimulai dan dimana akan berakhir.

4. Masing-masing kegiatan didalam *flowmap* sebaiknya digunakan suatu kata yang mewakili suatu pekerjaan.
5. Masing-masing kegiatan didalam flow map harus didalam urutan yang semestinya.
6. Kegiatan yang terpotong dan akan disambung ditempat lain harus ditunjukan dengan jelas menggunakan simbol penghubung.
7. Gunakan simbol simbol *flowmap* yang standar.

Adapun simbol-simbol yang sering digunakan dalam *flowmap* dapat dilihat pada tabel 2.4 berikut ini:

Tabel 2.4 Simbol dan Keterangan *Flowmap*

Simbol	**Deskripsi**
	Simbol yang digunakan untuk menunjukkan awal atau akhir dari suatu proses
	Menunjukkan dokumen input dan output baik untuk proses manual mekanik atau komputer
	Menunjukkan pekerjaan manual
	Menunjukkan multi dokumen
	Pengarsipan Dat
	Menunjukkan Proses
	Simbol input/output digunakan untuk mewakili data input/output

Sumber: [Jogiyanto, 2001].

2.2.20 JavaFX

JavaFX adalah bahasa pemrograman modern yang memungkinkan untuk membuat aplikasi mencari dengan antarmuka pengguna grafis yang canggih. JavaFX adalah anggota baru dari keluarga besar Java, JavaFx merupakan sebuah tambahan baru untuk platform Java yang menjanjikan pemakaian konsisten dari desktop ke alat-alat portabel. JavaFX ini memungkinkan RIA (Rich Internet Application) untuk tampil di layar mobile device, desktop, televisi dan sebagainya.

JavaFX diintegrasikan sepenuhnya dengan Java Runtime Environment
(JRE). JavaFX aplikasi akan dijalankan pada desktop dan browser yang menjalankan dan JRE di atas ponsel menjalankan JavaME. JavaFX didasarkan pada konsep sebuah "common profil" yang ditujukan ke seluruh perangkat span didukung oleh JavaFX. Pendekatan ini memungkinkan para pengembang untuk menggunakan model pemrograman
Common mobile dan berbagi banyak kode, grafis dan konten aset antara desktop dan versi mobile.

2.2.21 Scene Builder

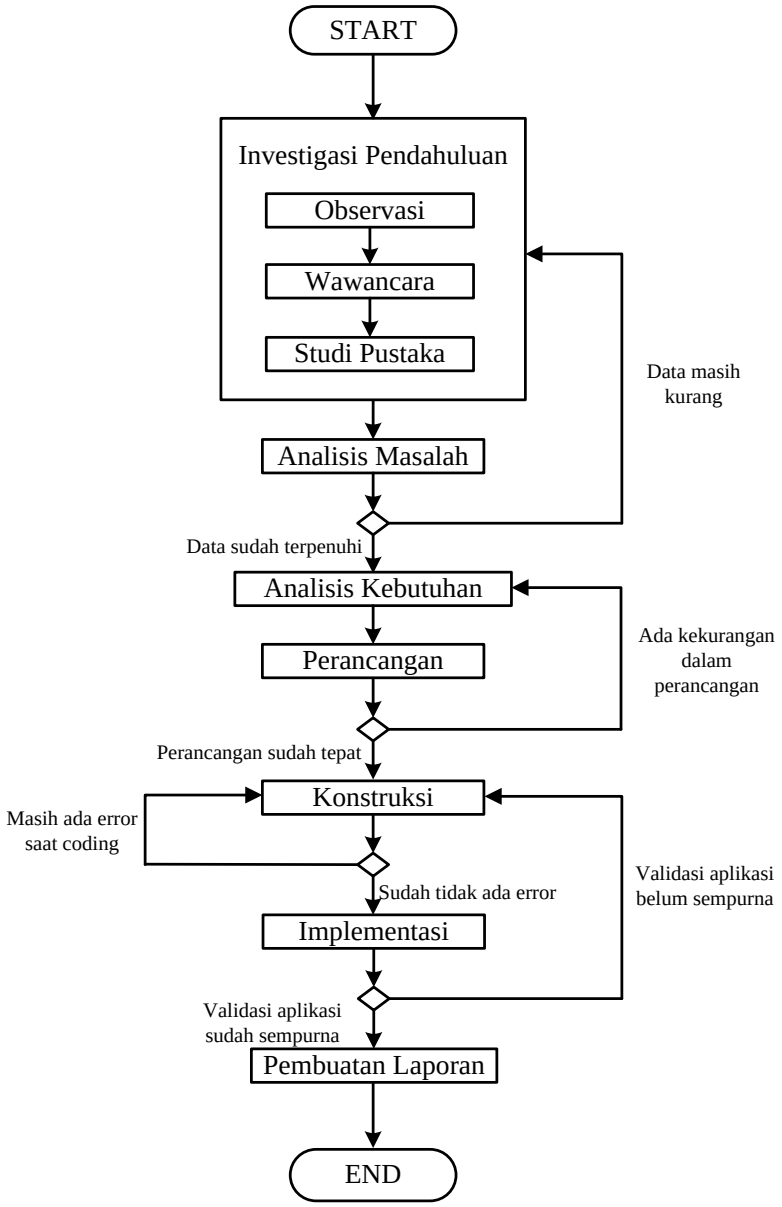
Scene Builder atau lebih singkatnya Screen Builder adalah Tools yang digunakan untuk membuat desain User Interface dari sebuah aplikasi. Javafx adalah penerus dari java swing, digunakan untuk membuat aplikasi java berbasis Grafical User Interface (GUI). Scene builder dapat membuat file fxml secara otomatis. Seperti yang kita ketahui, fxml adalah file yang berformat xml, yang isinya adalah layout dari aplikasi javafx.

BAB III

METODOLOGI PENELITIAN

3.1 Kerangka Berfikir

Berikut ini adalah langkah-langkah yang di lakukan untuk mencapai tujuan dari penelitian, ditunjukan dengan Gambar 3.1 dibawan ini:



Gambar 3.1 Kerangka Berfikir

3.2 Deskripsi

3.2.1 Investigasi Pendahuluan

Penelitian ini merupakan kolaborasi dari sebuah *project* pengembangan sistem informasi *repository* skripsi. Dalam pembuatan *project* dipimpin oleh bapa Moch. Ridwan, S.T, S.Kom. Yang diimplementasikan di Fakultas Teknologi Informasi Universitas Bale Bandung yang beralamat di Jl. Raden AA Wiranatakusumah No.7, Baleendah, Bandung, Jawa Barat 40375.

Untuk melaksanakan penelitian langkah awal yang dilakukan adalah:

1. Observasi

   Observasi dilakukan dengan mengamati sistem yang sedang berjalan saat ini bertujuan untuk mengetahui secara langsung bagaimana proses pencarian dan mencari kesamaan dokumen skripsi yang sedang berjalan.

   Tahap observasi ini peneliti mengamati ada beberapa fakta saat melakukan proses pencarian sebagai berikut:

   a) Mencarian skripsi secara langsung ke rak penyimpanan skripsi.
   b) Mencari dengan melihat satu-persatu judul skripsi dengan memakan waktu yang tidak sebentar.
   c) Mencari bahan untuk pembanding dengan penelitian yang akan di lakukan.

2. Wawancara

   Wawancara dimulai dengan menemui staff Tata Usaha yaitu bapak Yusuf Muharam S.Kom yang mengelola perangkat perpustakaan untuk meminta izin dalam melakukan wawancara tentang masalah yang ada saat pengelolaan perpustakaan , wawancara juga dilakukan kepada mahasiswa untuk mengetahui permasalahan apa yang ada pada saat melakukan pencarian skripsi . Hasil dari wawancara hasil di tunjukan dalam lampiran wawancara.

3. Studi Pustaka

Setelah melakukan wawancara penulis melakukan studi pustaka dengan mencari beberapa jurnal terkait "Implementasi *Elasticsearch* Untuk Pencarian Dan Menentukan Similarity Pada Proposal Skripsi Di Fakultas Teknologi Informasi Universitas Bale Bandung" sebagai penunjang dan juga mencari beberapa referensi dari website, jurnal, buku digital (*ebook*). Berikut adalah judul dari referensi:

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

Berdasarkan jurnal ini penulis mencoba untuk membuat pencarian dokumen tugas akhir berbasis *client* dengan menggunakan pemrograman Java dan *Elasticsearch* sebagai database pencarian dengan harapan hasil pencarian dapat ditampilkan dengan cepat sesuai dengan kata kunci yang diberikan serta dari hasil pencarian tersebut dapat ditampilkan kembali detailnya.

b) 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).

Berdasarkan jurnal ini penulis mengambil kasus pencarian dokumen dan mengimplementasikan *Elasticsearch* untuk hasil atau keluaran dari layanan pencarian dokumen tersebut berupa daftar dokumen yang sudah diberikan skor dan diurutkan berdasarkan relevansinya dengan *keyword* pencarian pengguna.

c) 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).

Berdasarkan penelitiian ini penulis mencoba untuk menerapkan similarity pada *Elasticsearch* utuk mengetahui tingkat kesamaan pada dokumen.

3.2.2 Analisis Masalah

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

1. Mahasiswa saat memelakukan pencarian dengan melihat satu-persatu judul skripsi dengan memakan waktu yang tidak sebentar dan masih di lakukan secara manual.
2. Mahasiswa belum bisa membandingkan kesamaan dari suatu dokumen penelitian yang akan diajukan sudah dilakukan atau belum.

Berdasarkan fakta 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.

3.2.3 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*) proposal skripsi, dalam prosedur ini mahasiswa melakukan pencarian dengan cara mengaksesnya dari *repository* penyimpanan laporan skripsi dan untuk menentukan kemiripan dokumen akan dilakukan upload dokumen yang nantinya akan muncul persentasi kemiripan pada suatu dokumen.

2. Kebutuhan Aplikasi Utama

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

   a) Membuat aplikasi yang dapat mencari skripsi atau penelitian secara cepat.
   b) 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:

   a) *Elasticsearch* adalah *engine* yang digunakan untuk mengindex, mencari dokumen.
   b) *Kibana,* aplikasi pendukung yang digunakan untuk ujicoba dalam mencari, melihat, dan berinteraksi dengan dokumen yang disimpan dalam indeks *Elasticsearch.*
   c) *Postman,* digunakan untuk pengujian API yang cara kerjanya sama seperti *kibana*.
   d) *Database MySQL,* digunakan untuk menyimpan *database* dari judul di setiap dokumen.
   e) *Java* adalah bahasa pemrograman yang digunakan dalam pembuatan aplikasi utama. Menggunakan java pada NetBeans IDE karena library-nya lebih lengkap.
   f) *Apache Tika,* digunakan unutuk konversi psd ke text,

g) *Apache Maven*, digunakan dalam konfigurasi proyek aplikasi Java.
h) NetBeans IDE, digunakan untuk membuat dan mengembangkan sebuah aplikasi dengan bahasa java.

4. 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:

a) Intel(R) Celeron(R) N4000 CPU @ 1.10GHz
b) RAM 4GB
c) HDD 500 GB

3.2.4 Perancangan

Pada tahap selanjutnya yaitu membuat desain aplikasi untuk pencarian dan pengecekan kemiripan (*similarity*) yang me;iputi struktur menu untuk memudahkan pengoperasian dalam aplikasi dan juga dibagi menjadi beberapa tahapan yaitu sebagai berikut:

1. Perancangan Aplikasi

Pada tahap ini perancangan dilakukan dengan menggunakan model *UML*. Perancangan dengan *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:

a) *Use Case* untuk pencarian dokumen dan pencarian *similarity* dokumen.
b) *Class Diagram* untuk pencarian berdasarkan autor, judul, kata kunci, abstrak, isi, tahun, topik, studi kasus, prodi.
c) *Activity Diagram* untuk pencarian dokumen dan pencarian *similarity* dokumen.
d) *Sequence Diagram* untuk kemunculan menu pencarian dokumen dan menentukan kemiripan dokumen

2. 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.

3.2.5 Konstruksi

Setelah tahap 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.

3.2.6 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.

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.

3.2.7 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 dalam penelitian.

BAB II : TINJAUAN PUSTAKA

Bab ini membahas tentang tinjauan dari penelitian yang telah di lakukan dan menjelaskan perbedaan pada setiap penelitian, dan juga membahas tentang dasar teori yang akan di pakai di dalam penelitian.

BAB III : METODOLOGI

Bab ini membahas tentang kerangka berfikir yang akan dilakukan saat mengerjakan penelitian, dan dijelaskan pula dengan deskripsi kerangka berfikir.

BAB IV : ANALISIS DAN PERANCANGAN

Bab ini menerangkan tentang instrument penelitian, analisis sistem, analisis kebutuhan dan hasil analisis. Dan dijelaskan perancangan untuk pembuuatan aplikasi.

BAB V : IMPLEMENTASI DAN PENGUJIAN

Bab ini berisi implementasi dan pengujian dari aplikasi yang telah dibuat untuk pencarian skripsi dan kesamaan (*similarity*).

BAB VI : KESIMPULAN DAN SARAN

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

BAB IV

ANALISIS DAN PERANCANGAN

4.1 Analisis

Berdasarkan hasil pengumpulan data Pencarian buku skripsi yang dilakukan di Fakultas Teknologi Informasi Universitas Bale Bandung masih berupa fisik dan dalam prosesnya masih secara konvensional dan proses pencarian dilakukan dengan melihat satu per satu judul buku skripsi yang tersimpan di rak tempat penyimpanan skripsi dan memerlukan waktu yang lama, maka penulis akan membuat dalam bentuk file. Dan juga untuk membandingkan bahwa penelitian yang akan diajukan sudah dilakukan atau belum, membutuhkan sebuah aplikasi yang dapat membantu dalam pencarian dan membandingkan kesamaan (*Similarity*) dokumen

Analisis merupakan tahap untuk pemahaman terhadap aplikasi yang akan dibuat. Pada tahap ini berisi Instrumen Penelitian, Analisis Sistem, Analisis Kebutuhan, Hasil Analisis dan Perancangan, dengan maksud untuk mengidentifikasikan segala permasalahan yang terjadi. Analisis juga memiliki tujuan untuk mengetahui bagaimana mekanisme sistem bekerja, mekanisme yang terlibat dalam sistem serta hubungan antara mekanisme tersebut. Analisa adalah tahap yang sangat penting karena jika ada suatu kesalahan dalam tahap ini maka akan mempengaruhi pada tahap berikutnya.

4.1.1 Instrumen Penelitian

Instrumen Penelitian merupakan alat bantu yang digunakan untuk membantu mengumpulkan data penelitian dilakukan dengan observasi, wawancara, studi pustaka dan pembuatan laporan diperlukan sebuah alat bantu sebagai instrument. Instrumen yang digunakan dalam penelitian ini yaitu meliputi Perangkat Keras, Perangkat Lunak dan Wawancara

1. Perangkat Keras

Perangkat keras yang digunakan dalam mengumpulkan data khususnya pada saat merekam wawancara, untuk mengukur waktu dalam melakukan pencarian dan mengambil data gambar menggunakan smartphone. Selain itu perangkat keras yang dipakai untuk menggabungkan berbagai data yang telah di kumpulkan menggunakan Laptop. Berikut ini merupakan instrumen penelitian pada perangkat keras yang dapat menjadi penunjang dalam proses penelitian:

Tabel 4.1 Instrumen Perangkat Keras

Perangkat Keras Laptop	
Tipe Laptop	Asus X441M-AGA041T
Prosesor	Intel(R) Celeron(R) N4000 CPU @ 1.10GHz
RAM	4GB
HDD	500GB
Perangkat Keras Smartphone	
Smartphone	Samsung M20
OS	Android Pie
RAM	3GB
Penyimpanan	32GB

2. Perangkat Lunak

Berikut ini merupakan instrumen penelitian pada perangkat lunak yang dapat menjadi penunjang dalam penelitian, berikut ini adalah beberapa perangkat lunak yang digunakan :

1) Sistem Operasi menggunakan Windows 10 64 bit
2) Membuat dokumen untuk mengumpulkan data menggunakan *Microsoft Word 2010*
3) Merekam wawancara menggunakan aplikasi Perekam Suara yang ada di smartphone.
4) Untuk dokumentasi foto menggunakan aplikasi Camera yang ada pada smartphone.

5) Mengukur waktu pencarian menggunakan fitur Stopwatch pada aplikasi Jam pada smartphone

3. Form Wawancara

Wawancara dilakukan peneliti dengan datang ke ruang Tata Usaha Fakultas Teknologi Informasi Universitas Bale Bandung, dan penulis mewawancarai mahasiswa dan juga staff yang pengelola perangkat perpustakaan dengan mengajukan beberapa pertanyaan.

Menurut penjelasannya dalam penyimpanan skripsi di Fakultas Teknologi Informasi Universitas Bale Bandung belum disimpan di *repository* , mencari judul skripsi masih dilakukan secara langsung dengan mencari di rak penyimpanan skripsi, dan masih belum bisa melakukan perbandingan kesamaan antara proposal skripsi terhadap dokumen skripsi.

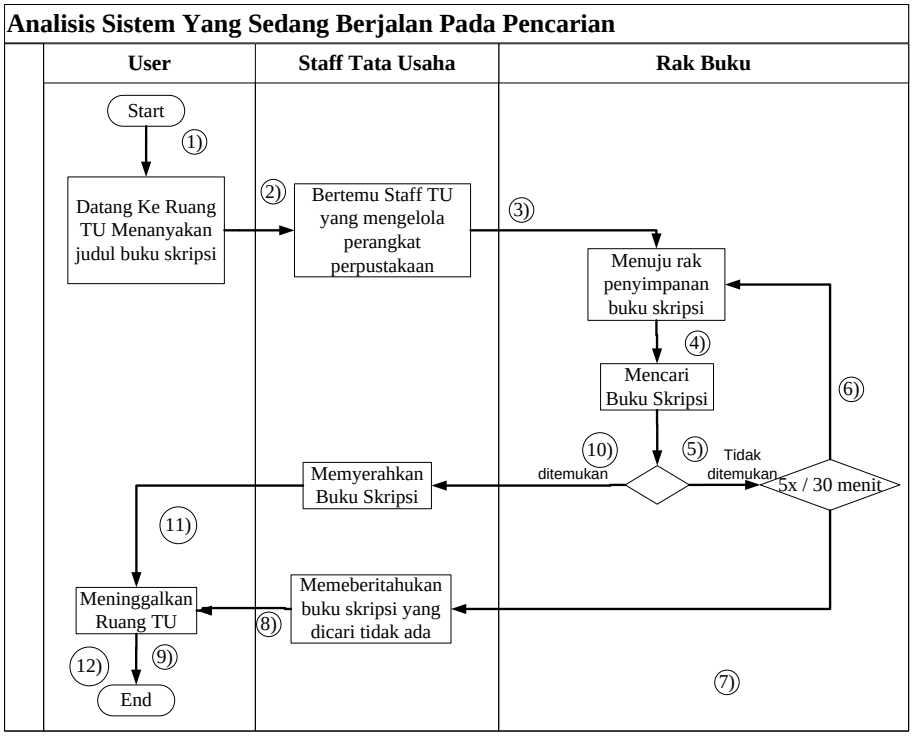
4.1.2 Analisis Sistem

Analisis sistem ini dilakukan untuk memberikan solusi terhadap permasalahan yang ada di Fakultas Teknologi Informasi Universitas Bale Bandung. Analisis sistem ini juga bertujuan untuk membandingkan sistem yang sedang berjalan saat ini dan sistem usulan yang akan dilakukan dalam penelitian ini. User meliputi mahasiswa, dosen dan Staff Tata Usaha Berikut ini adalah penjelasan Analisis Sistem yang meliputi Analisis Sistem Yang Sedang dan Analisis Sistem Usulan.

1. Analisis Sistem Yang Sedang Berjalan

Sistem yang sedang berjalan saat ini yaitu mencari dokumen dengan cara yang manual adapun mencari kesamaan dokumen yang akan menjadi usulan untuk penelitian ini. Cara manual yaitu dengan datang langsung ke ruang Tata Usaha dan nantinya bertemu dengan Staff Tata Usaha yang mengelola perangkat perpustakaan Lalu mencari langsung di rak penyimpanan buku skripsi. Pada analisis sistem ditunjukan dalam flowmap sebagai berikut:

Flowmap Analisis Sistem Yang Sedang Berjalan Pada Pencarian di tuliskan pada Gambar 4.1 dibawah ini.



Gambar 4.1 Analisis Sistem Yang Sedang Berjalan

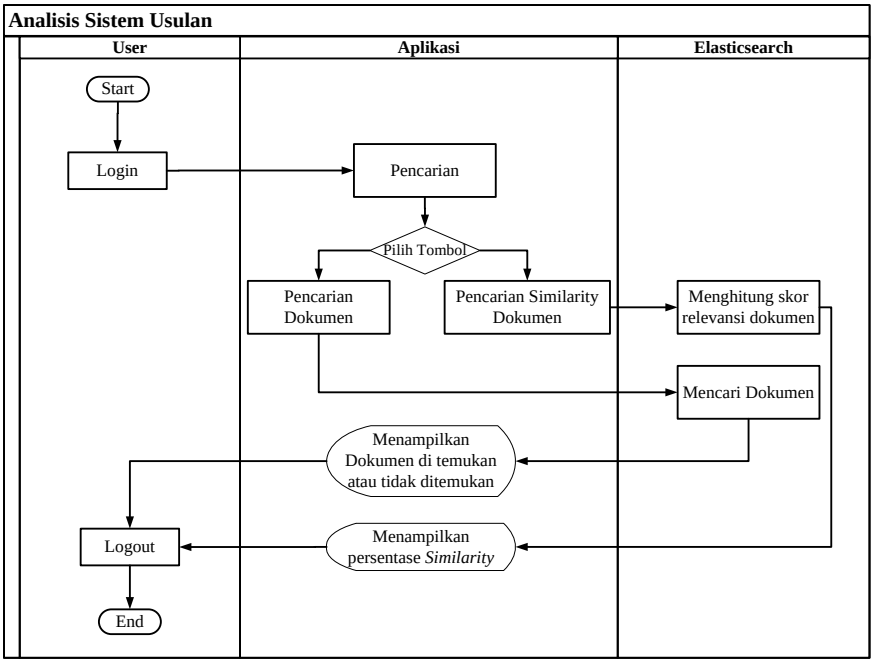
Berikut adalah penjalasan tentang Analisis Sistem Yang Sedang Berjalan:

1) User datang ke Ruang Tata Usaha dan untuk menanyakan judul buku skripsi untuk referensi ilmiah sebelum melakukan sebuah penelitian.
2) Bertemu dengan Staff Tata Usaha yang mengelola perangkat perpustakaan.
3) Setelah user bertemu Staff Tata Usaha dan menanyakan judul buku skripsi, lalu Staff Tata Usaha menuju rak penyimpanan buku skripsi.
4) Staff Tata Usaha mencari buku skripsi yang ditanyakan sebelumnya oleh user .
5) Jika buku tidak ditemukan di Rak Buku, maka user menuju kembali ke rak buku penyimpanan skripsi.
6) Dengan waktu dilakukan 5x atau 30 menit dalam pencariannya.

7) Jika selama 5x atau 30 menit tetap tidak ditemukan, maka Staff Staff Tata Usaha memberitahukan bahwa buku skripsi yang di minta tidak ditemukan
8) User meninggalkan ruang Tata Usaha
9) Selesai
10) Jika buku ditemukan, maka Staff Tata Usaha menyerahkan buku skripsi kepada user
11) Dan user keluar dari ruang TU.
12) Selesai atau langkah 9)

2. Analisis Sistem Usulan

Analisis Usulan dilakukan untuk memberikan usulan dalam analisis sistem ini agar didapatkannya sistem yang dapat membantu terkait permasalahan yang ada. Pada Analisis Sistem Usulan ini tentunya mempunyai kelebihan yaitu dengan memakai aplikasi yang dapat memudahkam pengguna dalam melakukan pencarian dan menentukan persentase *similarity*.



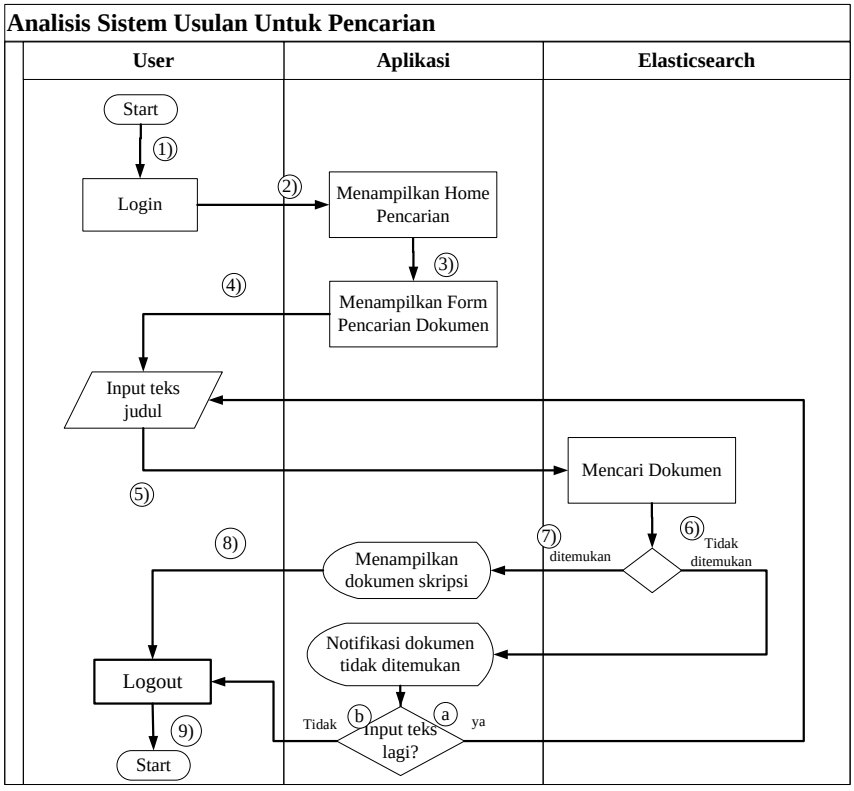
Gambar 4.2 Analisis Sistem Usulan

Berdasarkan gambar di atas User pada analisis yang sedang berjalan memiliki prilaku yang sama yaitu sebagai input. Staff Tata Usaha memiliki prilaku yang sama dengan Aplikasi pada sistem usulan yaitu sebagai output. Rak Buku memiliki prilaku yang sama dengan *Elasticsearch* pada sistem usulan yaitu sebagai proses.

Perbedaaan analisis sistem yang sedang berjalan dengan analisis sistem usulan adalah bahwa yang sebelumya membandingkan skripsi supaya tidak sama dilakukan dengan cara Staff Tata Usaha yang mengelola perpustakaan harus memiliki pengetahuan tentang skripsi yang ada, sedangkan pada sistem usulan pengetahuan tentang keragaman skripsi dilakukan oleh aplikasi pencarian.

Berikut ini adalah gambaran dari Analisis Sistem Usulan Pencarian:

a. Analisis Sistem Usulan Untuk Pencarian

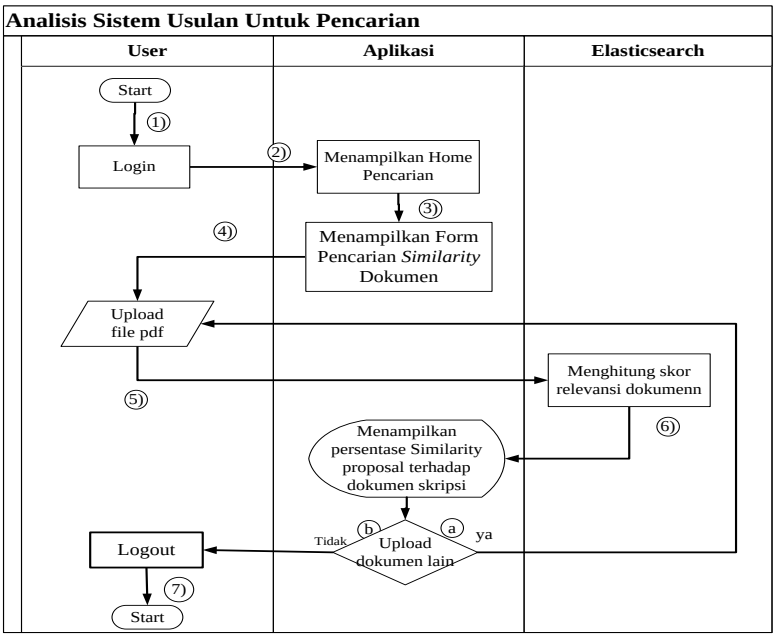


Gambar 4.3 Analisis Sistem Usulan Untuk Pencarian

Berikut ini adalah penjelasan dari gambar Analisis Sistem Usulan Untuk Pencarian:

1) User melakukan proses Login.
2) Lalu setelah melakukan login aplikasi menampilkan Form Home Pencarian, dan memilih tombol Pencarian Dokmen
3) Aplikasi menampilkan Form Pencarian Dokumen yang digunakan User untuk mencari judul dan menampilkan dokumen yang dicari.
4) Setelah itu User menginput judul apa yang akan di cari berdasarkan topik yang dipilih.
5) *Elasticsearch* akan mencari dokumen sesuai dengan hasil inputan User
6) Jika dalam *Elasticsearch* tidak temukan dokumen yang dicari, maka menampilkan notifikasi dokumen tidan ditemuka. Jika ingin menginput teks lagi maka kembali lagi ke langkah nomer 4). Jika tidak ingin menginput teks lagi maka logout
7) Jika dalam *Elasticsearch* temukan dokumen yang dicari, maka aplikasi akan menampilkan dokumen.
8) User melakukan logout
9) selesai.

b. Analisis Sistem Usulan Untuk *Similarity*



Gambar 4.4 Analisis Sistem Usulan Untuk *Similarity*

Berikut ini adalah penjelasan dari gambar Analisis Sistem Usulan Untuk *Similarity*:

1) User melakukan proses Login.
2) Lalu setelah melakukan login aplikasi menampilkan Form Home Pencarian, dan memilih tombol Pencarian Dokmen
3) Aplikasi menampilkan Form Pencarian Dokumen yang digunakan User untuk mencari judul dan menampilkan dokumen yang dicari.
4) Setelah itu User mengupload file proposal pdf yang akan dibandingkan dengan dokumen skripsi yang sudah ada.
5) *Elasticsearch* akan menghitung skor relevansi dokumen
6) Menampilkan persentase Similarity proposal terhadap dokumen skripsi
    a. Jika ingin upload dokumen proposal pdf lain maka kembali ke langkah nomer 4)
    b. Jika tidak user akan logout
7) Selesai.

4.1.3 Analisis Kebutuhan

Penulis melakukan analisis kebutuhan terhadap sistem yang akan dibangun berdasarkan kebutuhan yang dapat memenuhi proses pembuatan aplikasi yang meliputi kebutuhan software, kebutuhan hardware dan kebutuhan pengguna.

1. Kebutuhan Software

Kebutuhan software untuk membuat aplikasi dibutuhkan perangkat lunak yang meliputi kebutuhan software utama dan kebutuhan software pendukung yakni sebagai berikut:

a. Kebutuhan Software Utama
    1) *Elasticsearch,* digunakan sebagai *engine* yang dapat melakukan proses untuk mengindex, mencari dokumen. *Elasticsearch* juga merupakan server untuk menyimpan dokumen.

2) *Kibana,* digunakan untuk melakukan ujicoba API dalam mencari, melihat, dan berinteraksi dengan dokumen yang disimpan dalam *Elasticsearch.*
3) *Postman,* memiliki fungsionalitas yang sama dengan kibana yaitu digunakan untuk untuk melakukan ujicoba API dalam mencari, melihat, dan berinteraksi dengan dokumen yang disimpan dalam *Elasticsearch*. Dan *postman* ini berfungsi untuk mengecek apakan *elasticsearch* sudah berjalan atau belum
4) *Database MySQL*, digunakan untuk menyimpan data pelengkap seperti data login yang dapat membantu dalam mengakses aplikasi, dan juga data pelengkap dari setiap dokumen.

b. Kebutuhan Software Pendukung
1) *Java* adalah bahasa pemrograman yang digunakan dalam pembuatan aplikasi utama. Menggunakan java pada NetBeans IDE karena library-nya lebih lengkap.
2) *Spring,* merupakan *framework* yang digunakan dalam pembuatan aplikasi di java.
3) *Apache Maven*, library yang digunakan dalam mengkonfigurasi beberapa modul aplikasi class yang ada di java antara client dan server, maven disimpan di pom.xml.
4) NetBeans IDE, digunakan untuk membuat dan mengembangkan sebuah aplikasi pencarian ini dengan dengan bahasa java.
5) Pemodelan UML menggunakan StarUML V3.0.2
6) Pemodelan mockup *user interface* menggunakan Visio.
7) Pemodelan *user interface* pengguna *Scene builder*

2. Kebutuhan Hardware

Kebutuhan hardware yang digunakan untuk mendukung penelitian menggunakan sfesifikasi minimal. Berikut ini sfesifikasi laptop yang digunakan:

Tabel 4.2 Tabel Kebutuhan Hardware

Tipe	Asus X441M-AGA041T
Prosessor	Intel(R) Celeron(R) N4000 CPU @ 1.10GHz
RAM	4GB
Harddisk	500 GB

3. Kebutuhan Pengguna

Dalam aplikasi yang dibuat pengguna dibagi menjadi 3 hak akses, yaitu meliputi mahasiswa, dosen, maupun staff. Tujuan dari user yaitu mencari judul skripsi atau mengetahui kemiripan *(similarity)* proposal terhadap dokumen skripsi. Untuk menggunakan aplikasi ini pengguna wajib memilliki proposal skripsi dalam bentuk pdf yang nanti akan digunakan oleh aplikasi untuk diketahui persentase *similarity* nya.

4.1.4 Hasil Analisis

Hasil analisis dilakukan dengan membuat tabel dari beberapa kemungkinan *similarity* dari proposal skripsi terhadap skripsi yang sudah ada. Berikut ini adalah tabel kemungkinan yang telah dibuat:

Tabel 4.3 Tabel Hasil Analisis

Autor	Title	Publiser	Abstrak	Full Text	Keterangan
Y	Y	Y	Y	Y	*Similarity* Pasti
Y	Y	Y	Y	TP	*Similarity* Pasti
T	Y	T	T	T	*Similarity* Pasti
Y	Y	Y	T	T	*Kemungkinan*
T	T	T	Y	T	*Kemungkinan*
T	T	T	T	Y	*Tidak Mungkin*

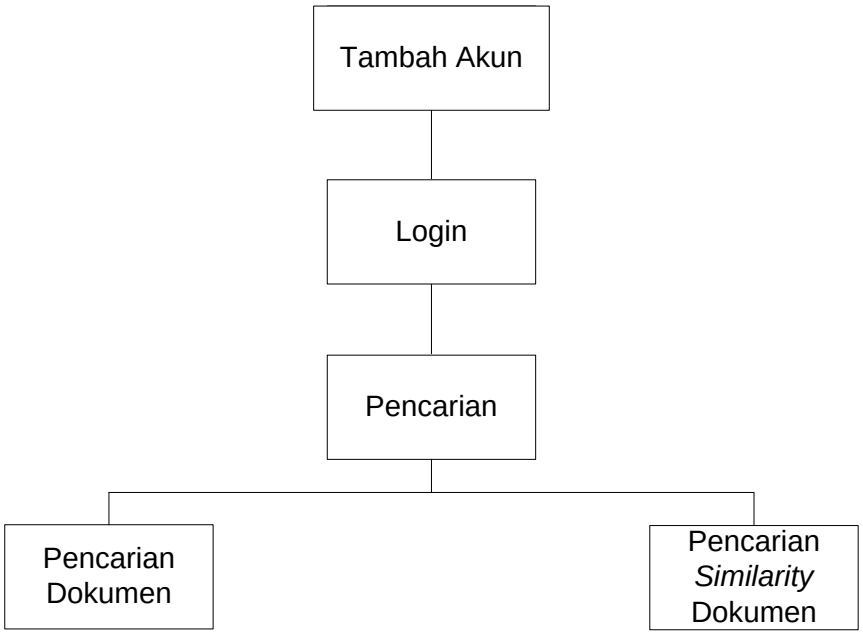
Ket: Y = Sama; T = Tidak sama; TP = Tidak Perlu

4.2 Perancangan

Proses perancangan aplikasi ini menggunakan *Unified Modeling Language (UML)* untuk pemodelannya, dan untuk memberikan gambaran pada penelitian ini yang meliputi Struktur Menu, *Use Case Diagram, Activity Diagram Sequence Diagram*, *Class Diagram* dan Perancangan *User Interface*.

4.2.1 Struktur Menu

Struktur menu disusun secara bertingkat dan dibuat untuk memudahkan pengoperasian dalam aplikasi. Berikut adalah struktur menu dari Analisis Sistem Usulan.



Gambar 4.5 Struktur Menu

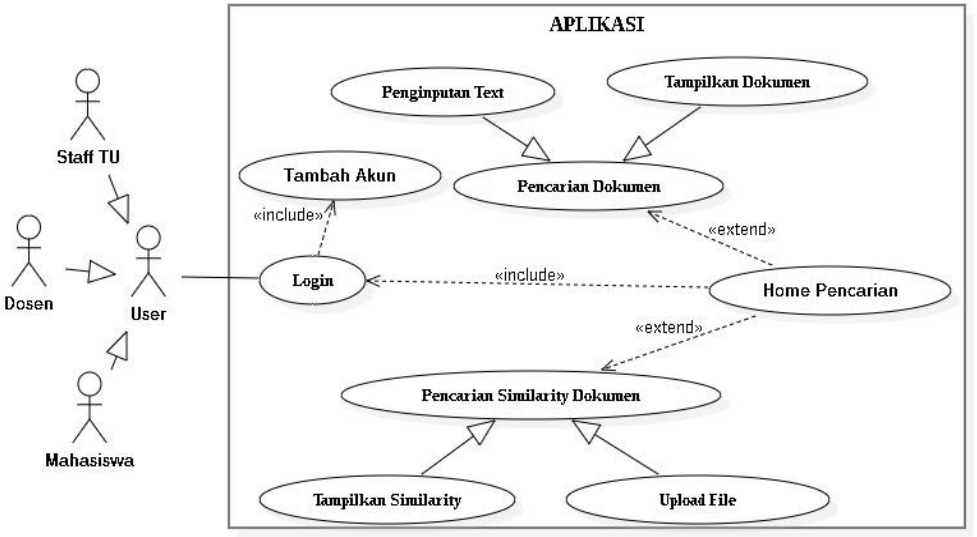
Berikut ini adalah penjelasan dari Struktur Menu:

1. Tambah Akun, menu ini dibuat dengan tujuan untuk membuat sebuah akun dengan mengisi data Npm, Nama, Email, Telepon, Username, Password dan Catatan untuk mendapatkan akses ke aplikasi.

2. Login, menu ini dibuat dengan tujuan sebagai pintu masuk awal untuk mengakses aplikasi.
3. Pencarian, menu ini dibuat dengan tujuan sebagai halaman utama dari aplikasi.
4. Pencaria Dokumen, menu ini dibuat sebagai tempat pencarian dalam mencari judul.
5. Pencarian *Similarity* Dokumen,, menu ini dibuat untuk mengetahui tingkat kemiripan *(similarity)* proposal skripsi terhadap dokujmen skripsi yang telah dilakukan.

4.2.2 *Use Case Diagram* Aplikasi Pencarian

Use Case Diagram bertujuan untuk memberikan gambaran interaksi aktor dengan sistem aplikasi pencarian. Berikut ini adalah *Use Case Diagram* pada aplikasi pencarian.



Gambar 4.6 *Use Case Diagram*

Berikut ini adalah penjelasan dari *Use Case Diagram* Aplikasi Pencarian:

a. Deskripsi Aktor

Deskripsi actor berfungsi untuk menjelaskan apa peran dari aktor yang terlibat dalam aplikasi yang akan dibangun. Berikut adalah deskripsi aktor dari *usecase*.

Tabel 4.4 Deskripsi Aktor

Aktor	**Deskripsi**
User	1. Melakukan Login untuk mengakses aplikasi 2. Mengakses informasi yang di cari 3. Bersifat general karena setiap user yang meliputi Mahasiswa, Dosen, dan Staff Tata Usaha memiliki prilaku yang sama.

b. Deskripsi *Use Case*

Deskripsi *Use Case* berfungsi untuk menjelaskan setiap proses yang di lakukan aplikasi. Berikut adalah deskripsi dari *usecase*.

Tabel 4.5 Deskripsi *Use Case*

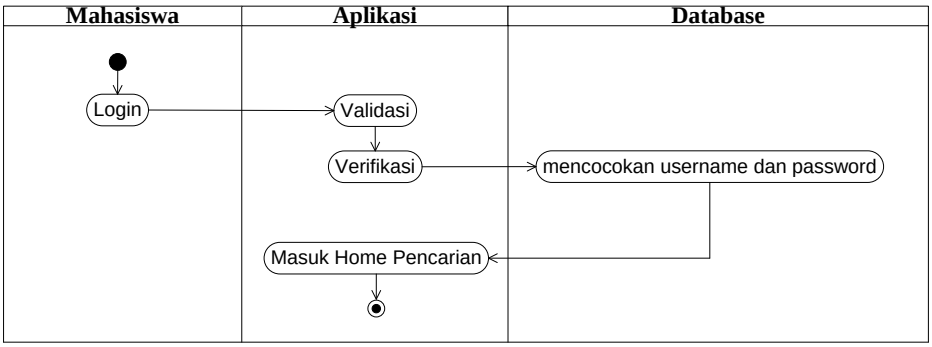
Use Case	**Deskripsi**
Login	Proses malakukan Login
Tambah Akun	Proses mengisi data untuk syarat melakukan login
Home Pencarian	Proses melakukan aktifitas pencarian
Pencarian Dokumen	Salah satu proses yang ada di dalam Home Pencarian, proses ini dilakukan untuk pecarian di dalam proses ini terdapat proses input teks dan proses menampilkan dokumen
Pencarian Similarity Dokumen	Salah satu proses yang ada di dalam Home Pencarian, proses ini dilakukan untuk

	mengetahui persentase kemiripan di dalam proses ini terdapat proses upload dokumen dan proses menampilkan similarity dokumen

4.2.3 *Activity Diagram*

Activity Diagram merupakan cara memodelkan aktifitas yang ada dalam suatu *use case* yang meliputi *Activity Diagram* Login, *Activity Diagram* Tambah Akun, *Activity Diagram* Home Pencarian, *Activity Diagram* Pencarian Dokumen dan *Activity Diagram* Pencarian Similarity Dokumen.

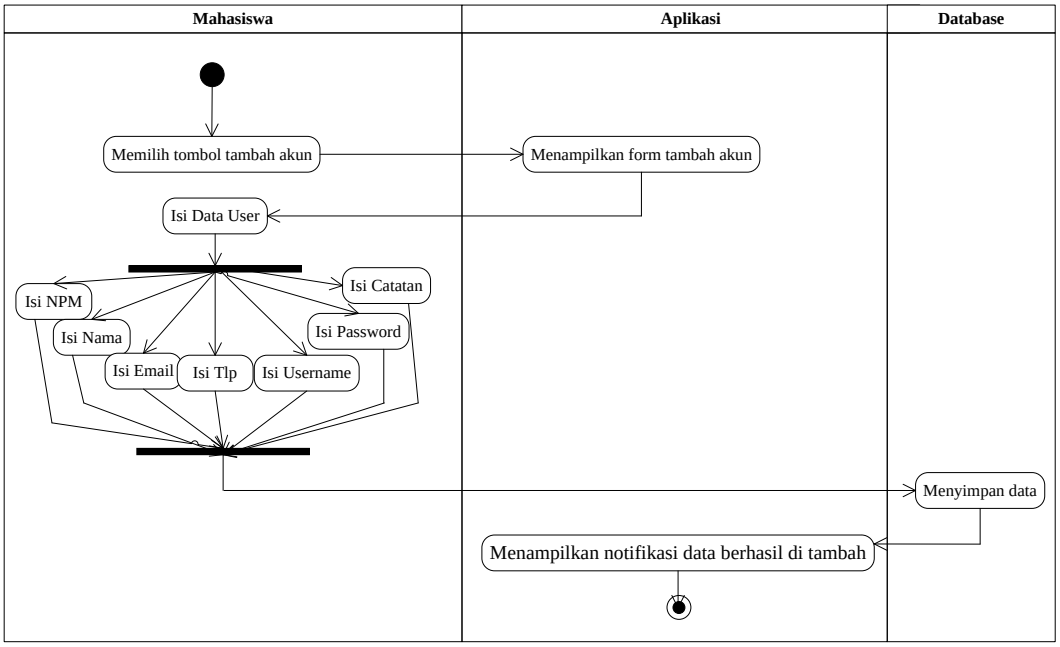
1. *Activity Diagram* Login



Gambar 4.7 *Activity Diagram* Login

Aktifitas ini dimulai dengan User melakukan interaksi dengan Aplikasi dengan melakukan aktifitas login. Di form login menginputkan username dan password. Setelah itu di validasi jika tidak valid user kembali menginputkan username dan password , jika valid langsung di verifikasi. lalu mencari username dan password yang cocok di *database*, jika ditemukan berarti data sah dan langsung masuk ke form Home Pencarian. Jika tidak di teukan muncul pesan data yang dimasukan tidak cocok.

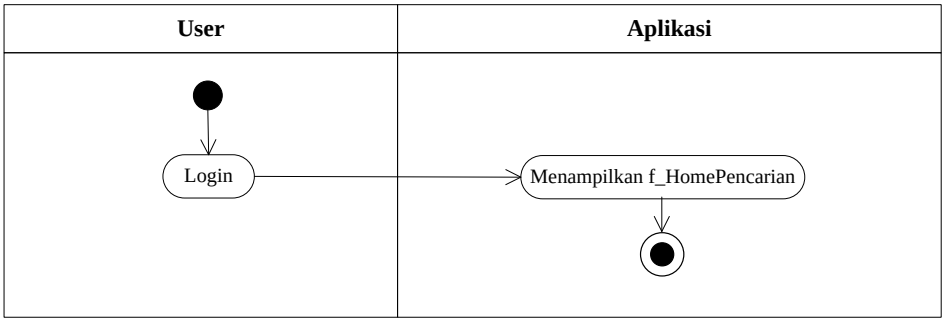
2. *Activity Diagram* Tambah Akun



Gambar 4.8 *Activity Diagram* Tambah Akun

Aktifitas tambah akun di mulai dengan User masuk ke form Login lalu memilih tombol Tambah Akun. Aplikasi merespon dengan menampilkan form Tambah Akun, dan user mengisi npm, nama, email, tlp, username, password dan pencarian. Setelah itu di simpan ke *database* da menampilkan pesan data berhasil di tambah.

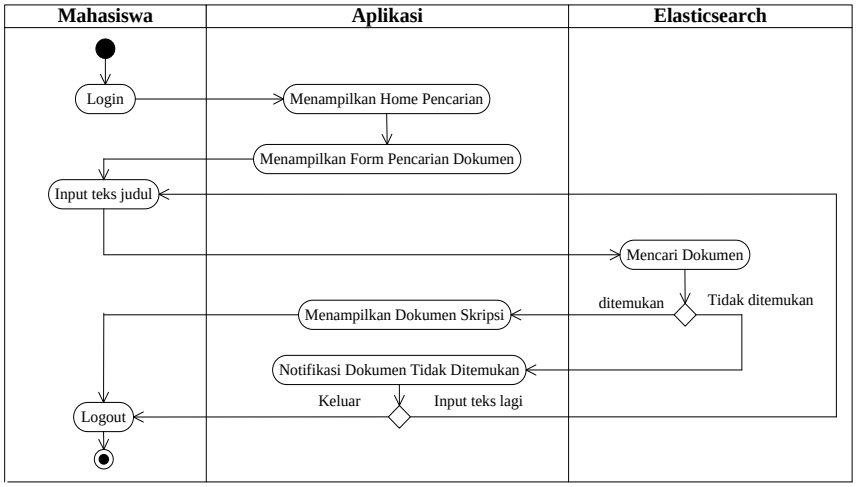
3. *Activity Diagram* Home Pencarian



Gambar 4.9 *Activity Diagram* Home Pencarian

Aktifitas di mulai dengan User melakukan aktifitas login yang sudah dijelaskan pada Gambar 4.7 *Activity Diagram* Login di atas, setelah itu Aplikasi akan, menampilkan form Home Pencarian.

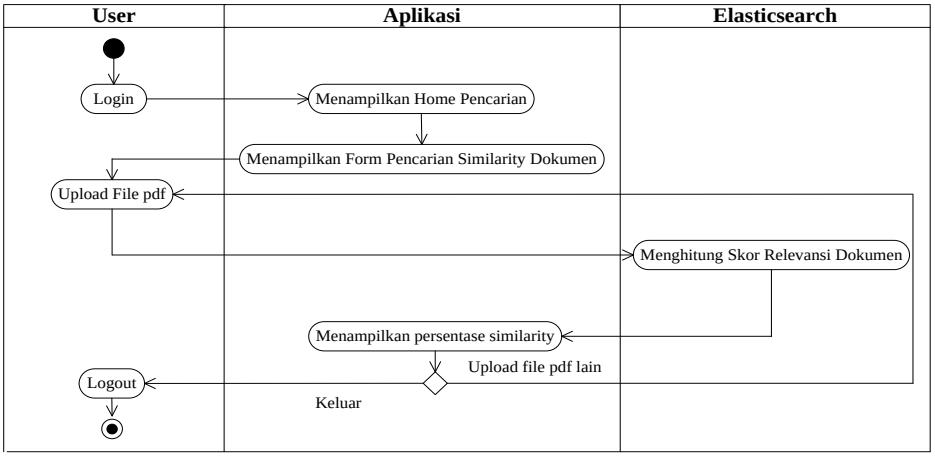
4. *Activity Diagram* Pencarian Dokumen



Gambar 4.10 *Activity Diagram* Pencarian Dokumen

Aktifitas di mulai dengan User melakukan aktifitas login. Setelah itu Aplikasi akan menampilkan form Home Pencarian, lalu User memilih Tombol Pencarian dokumen. Dan masuk ke form Pencarian Dokumen di Aplikasi. User menginput judul yang akan dicari. Dan *Elasticsearch* menerima inputan teks yang dicari. Lalu *Elasticsearch* mencari Judul buku skripsi. Jika ditemukan, maka akan menampilkan hasil dokumen di Pencarian Dokumen sesudah itu User logout. Atau jika dokumen tidak ditemukan, maka akan menampilkan pesan dokumen tidak ditemukan di Pencarian Dokumen sesudah itu User logout.

5. *Activity Diagram* Pencarian Similarity Dokumen



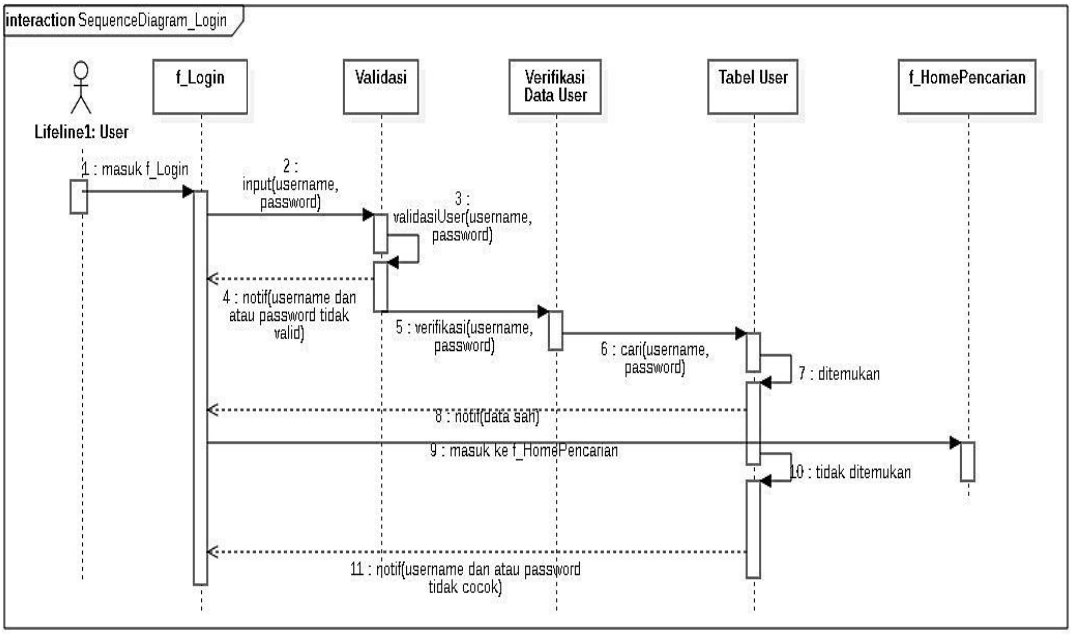
Gambar 4.11 *Activity Diagram* Pencarian Similarity Dokumen

Aktifitas di mulai dengan User melakukan login. Aplikasi akan menampilkan form Home Pencarian. User memilih Tombol Pencarian *Similarity* dokumen. Masuk ke form Pencarian Similarity Dokumen di Aplikasi. User melakukan Upload file proposal skripsi. Selanjutnya *Elasticsearch* menghitung skor relevansi dokumen. Dan setelah itu menampilkan persentase *similarity* proposal skripsi. User Logout dan selesai.

4.2.4 *Sequence Diagram*

Sequence Diagram merupakan diagram yang menggambarkan interaksi antar objek di sekitar aplikasi yang meliputi *Sequence Diagram Login*, *Sequence Diagram* Tambah Akun Baru, *Sequence Diagram* Pencarian Dokumen dan *Sequence Diagram* Pencarian *Similarity* Dokumen

1. *Sequence Diagram Login*



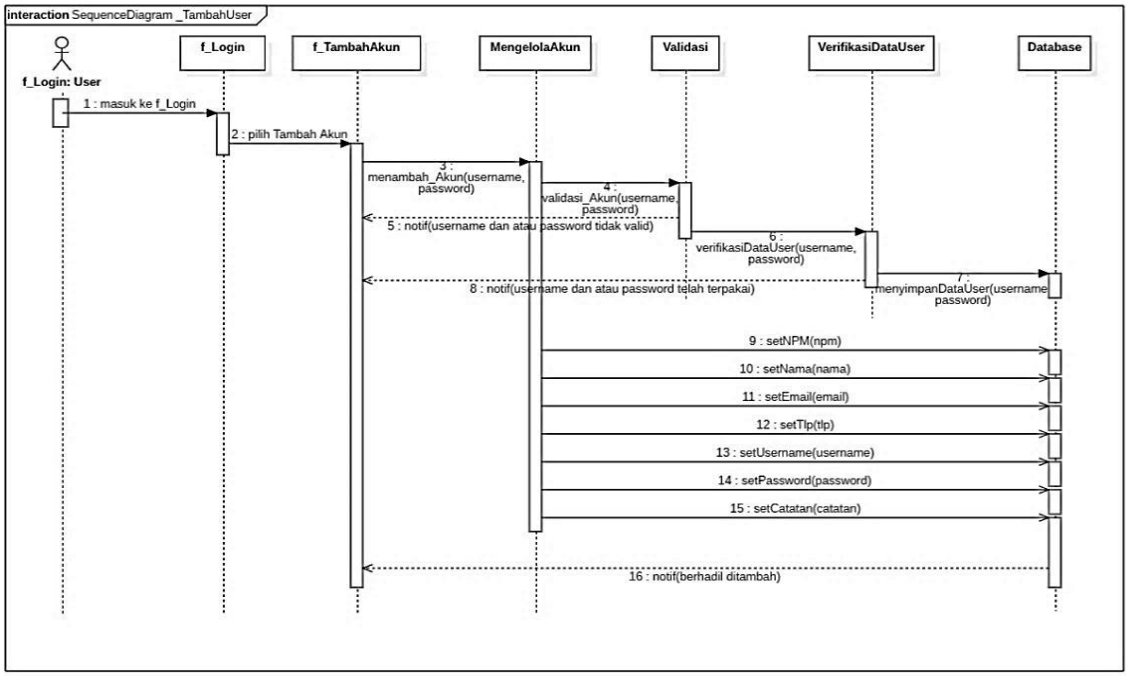
Gambar 4.12 *Sequence Diagram Login*

Berikut ini adalah penjelasan dari *Sequence Diagram Login*:

1) User masuk ko form login
2) User memasukan username dan password pada form Login.

3) Aplikasi melakukan validasi Username dan password.
4) Jika username atau password tidak valid maka atan menampilkan pesan username dan password tidak valid.
5) Jika username dan password maka aplikasi akan melakukan verifikasi data.
6) Username dan password di cari di Tabel user
7) Jika username dan password ditemukan.
8) Maka akan muncul notifikasi data sah.
9) Lalu masuk ke f_HomePencarian
10) Jika username dan password tidak ditemukan
11) Muncul username dan atau password tidak cocok.

2. *Sequence Diagram* Tambah Akun



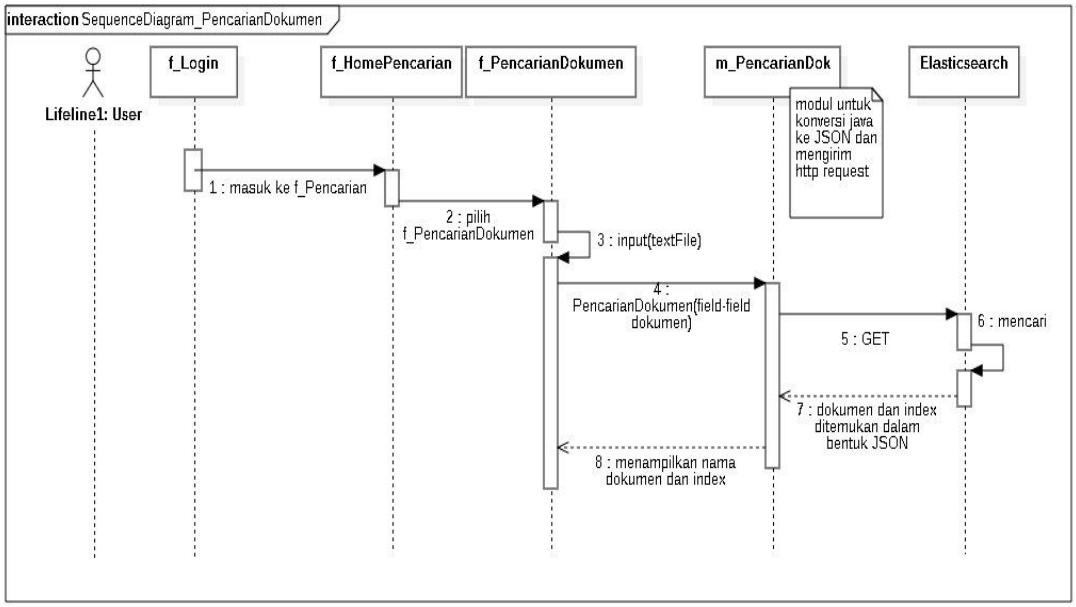
Gambar 4.13 *Sequence Diagram* Tambah Akun

Berikut ini adalah penjelasan dari *Sequence Diagram* Tambah Akun:

1) User memilih tombol tambah akun baru pada form login.
2) Kemudian memasukan data user pada form tambah akun baru.
3) Kemudian pilih tambah.

4) Aplikasi melakukan validasi data user
5) Jika data tidak valid maka akan menampilkan pesan data tidak valid.
6) Jika data user valid maka aplikasi akan melakukan verifikasi data user.
7) Jika username sudah pernah terpakai maka aplikasi menampilkan pesan username sudah terpakai.
8) Jika data user sudah sesuai maka aplikasi akan menyimpan data user dan menampilkan pesan akun berhasil ditambahkan.

3. *Sequence Diagram* Pencarian Dokumen



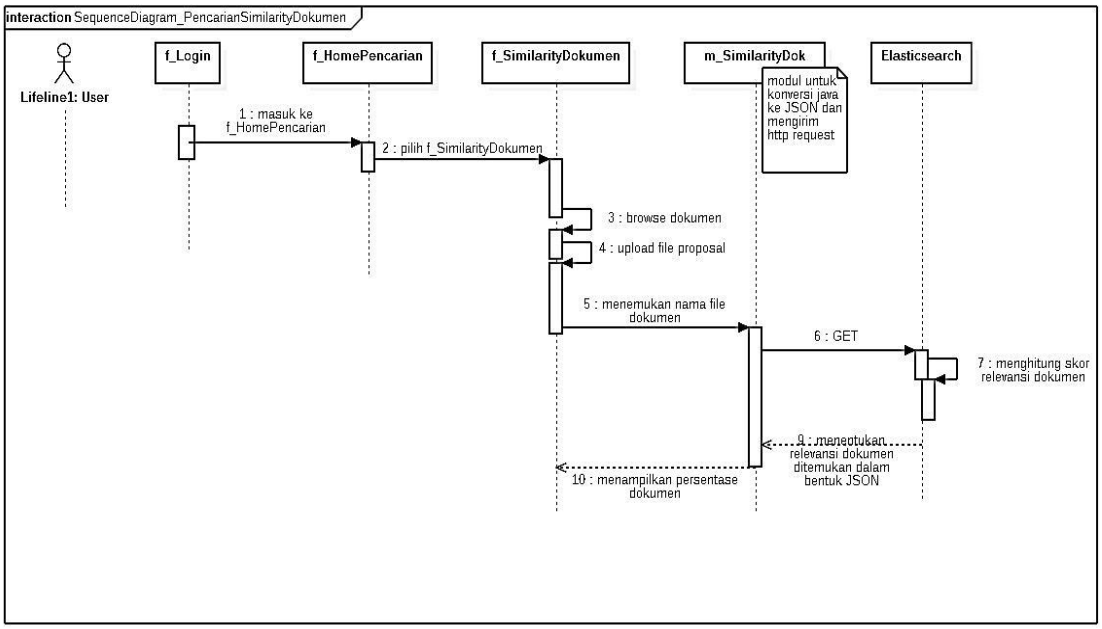
Gambar 4.14 *Sequence Diagram* Pencarian Dokumen

Berikut ini adalah penjelasan dari *Sequence Diagram* Pencarian Dokumen:

1) Sebelumnya user telah melakukan login dan masuk ke form pencarian
2) Pilih tombol Pencarian Dokumen
3) Setelah berada di form Pencarian Dokumen, user menginputkan teks
4) Teks yang di inputkan merupakan bentuk field, dan di konversi dalam bentuk dokumen JSON oleh modul m_PencarianDok
5) Dan melakukan http request GET
6) Melakukan pencarian

7) Jika di temukan dokumen dan index dalam bentuk JSON
8) JSON di rubah kembali oleh modul m_PencarianDok ke dalam Java
9) Dan menampilkan dokumen dan nama index sesuai yang dicari

4. *Sequence Diagram* Pencarian *Similarity* Dokumen



Gambar 4.15 *Sequence Diagram* Pencarian *Similarity* Dokumen

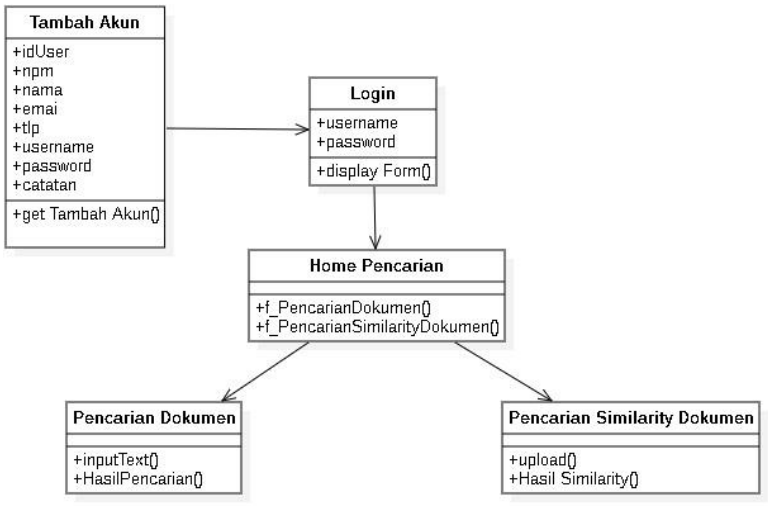
Berikut ini adalah penjelasan dari *Sequence Diagram* Pencarian Dokumen:

1) Sebelumnya user telah melakukan login dan masuk ke form pencarian
2) Pilih tombol Pencarian Dokumen
3) Setelah berada di form Pencarian Dokumen, user menginputkan teks
4) Teks yang di inputkan merupakan bentuk field, dan di konversi dalam bentuk dokumen JSON oleh modul m_PencarianDok
5) Dan melakukan http request GET
6) Menghitung skor relevansi dokumen
7) Jika di temukan dokumen dan index dalam bentuk JSON
8) JSON di rubah kembali oleh modul m_PencarianDok ke dalam Java

9) Dan menampilkan dokumen dan nama index sesuai yang dicari

4.2.5 *Class Diagram*

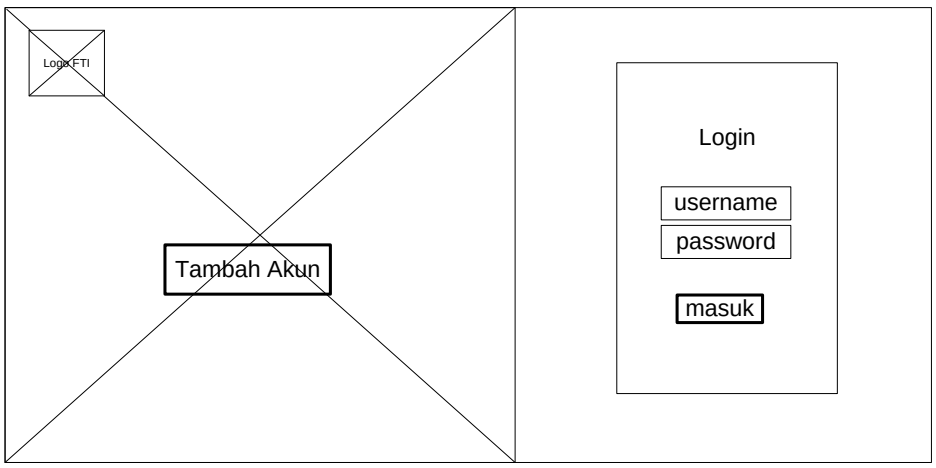
Class Diagram merupakan diagram untuk menampilkan beberapa kelas yang ada dalam aplikasi yang menunjukan hubungan antar kelas dalam aplikasi yang akan dibangun. Berikut digambar *Class Diagram* dari aplikasi pencarian.



Gambar 4.16 *Class Diagram*

4.2.6 Perancangan *User Interface*

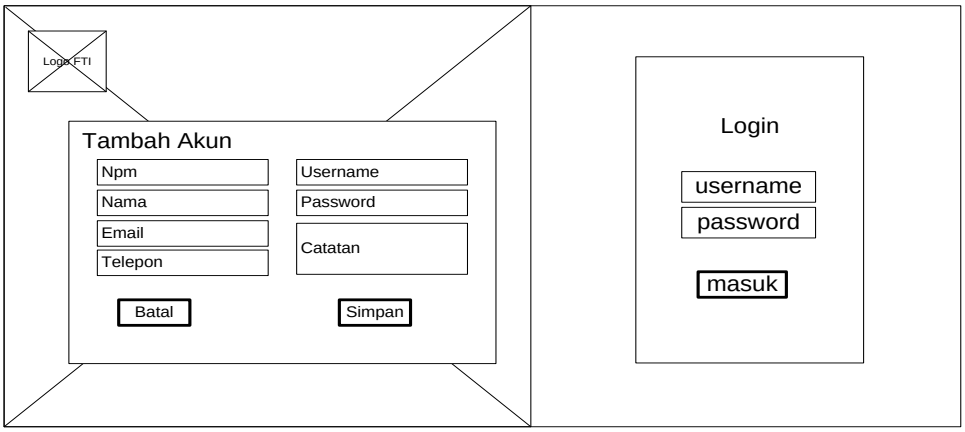
1. Perancangan *User Interface Form* Login



Gambar 4.17 Perancangan *User Interface* Login

Gambar diatas merupakan perancangan antar muka atau *user interface* Login. Di dalamnya terdapat fitur untuk melakukan login, dan terdapat fitur tombol Tambah akun.

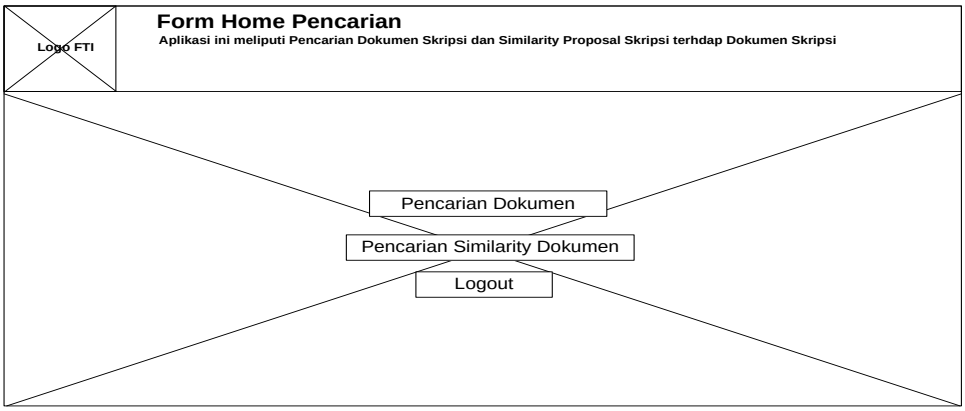
2. Perancangan *User Interface Form* Tambah Akun



Gambar 4.18 Perancangan *User Interface* Tambah Akun

Gambar diatas merupakan perancangan antar muka atau *user interface Form* Tambah Akun. Di dalamnya terdapat fitur untuk melakukan login, dan terdapat fitur untuk mengisi data untuk menambah Akun.

3. Perancangan *User Interface Form Home* Pencarian

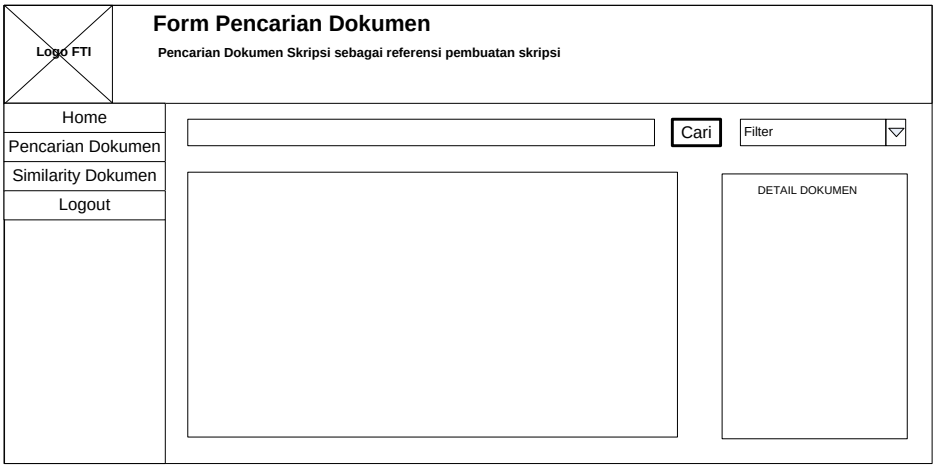


Gambar 4. 19 Perancangan *User Interface Form Home* Pencarian

Gambar diatas merupakan perancangan antar muka atau *user interface Form Home* Pencarian. Di dalamnya terdapat fitur tombol

Pencarian Dokumen, tombol Pencarian *Similarity* Dokumen dan tombol Logout.

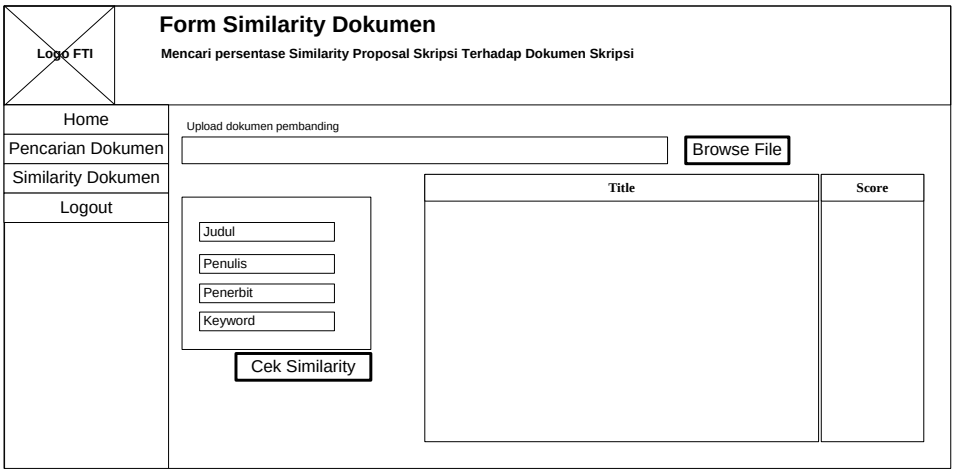
4. Perancangan *User Interface Form* Pencarian Dokumen



Gambar 4. 20 Perancangan *User Interface Form* Pencarian Dokumen

Gambar diatas merupakan perancangan antar muka atau *user interface Form Home* Pencarian. Di dalamnya terdapat fitur tombol Pencarian Dokumen, tombol Pencarian *Similarity* Dokumen dan tombol Logout. Selain itu terdapat *text field* dan tombol cari dan juga ada filter berdasarkan sesuai yang dipilih. Di bawah nya terdapat *listview* untuk menampilkan dokumendan disampingnya terdapat detail dokumen.

5. Perancangan *User Interface* Pencarian *Similarity* Dokumen



Gambar 4.21 Perancangan *User Interface* Pencarian *Similarity* Dokumen

Gambar diatas merupakan perancangan antar muka atau *user interface Form Home* Pencarian. Di dalamnya terdapat fitur tombol Pencarian Dokumen, tombol Pencarian *Similarity* Dokumen dan tombol Logout. Selain itu terdapat tombol Upload untuk mrnguplоad dokumen , hasil dokumen yang di *upload*, di bawahnya terdapat list view untuk menampilkan judul dokumen yang di *upload*. Dan ada tombol Cek *Similarity*, dan di bawahnya terdapat list view untuk menampilkan persentase *similarity*.

BAB V

IMPLEMENTASI DAN PENGUJIAN

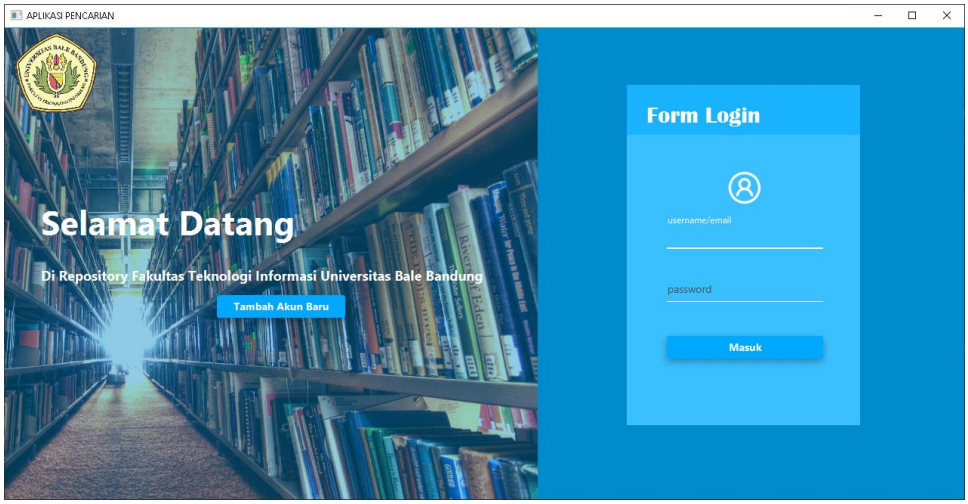
5.1 Implementasi

Impelentasi ini dilakukan sesuai dengan perancangan aplikasi yang di lakukan pada bab sebelumnya. Terdiri dari tampilan struktur menu dan pengujian apllikasi.

5.1.1 Implementasi Antar Muka

Implementasi antar muka merupakan perancangan yang telah dilakukan dan diketahui bahwa struktur menu bertujuan untuk memudahkan pengoperasian dalam aplikasi yang terdiri dari Tambah Akun, Login, Home Pencarian, Pencarian Dokumen, Pencarian Similarity Dokumen. Berikut ini adalah Tampilan Struktur Menu:

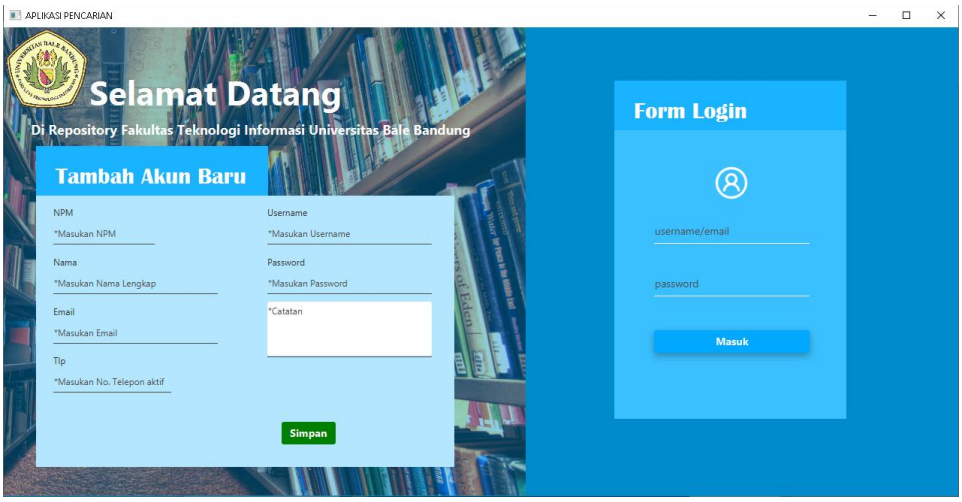
1. Login



Gambar 5.1 Tampilan Struktur Menu *Login*

Pada form login ini user diharuskan untuk memasukan username dan password yang sebelumnya telah mendaftar dengan menekan tombol tambah akun baru yang nantinya akan diarahkan ke Form Home Pencarian.

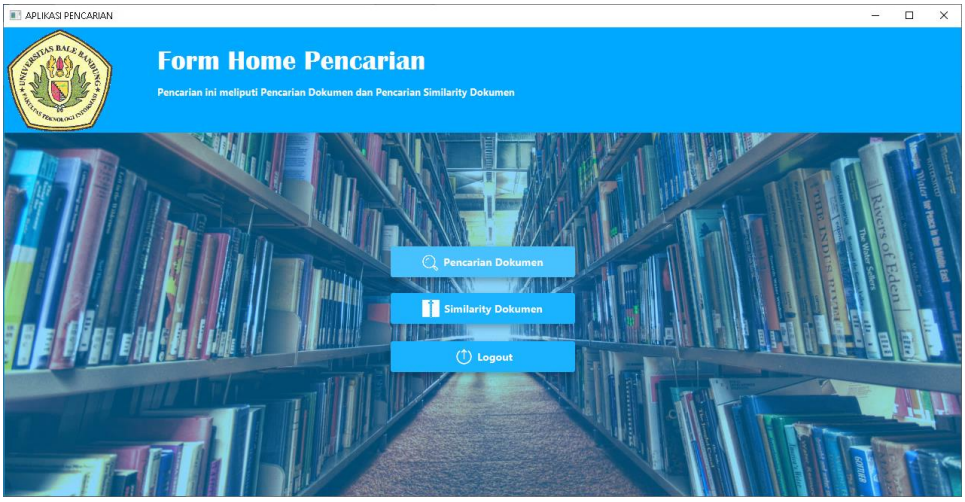
2. Tambah Akun



Gambar 5.2 Tampilan Struktur Menu Tambah Akun

Berdasarkan gambar diatas, sebelum melakukan login, user terlebih dahulu di haruskan untuk menambah akun baru untuk mengakses aplikasi pencarian yang nantinya akan digunakan untuk login.

3. Home Pencarian

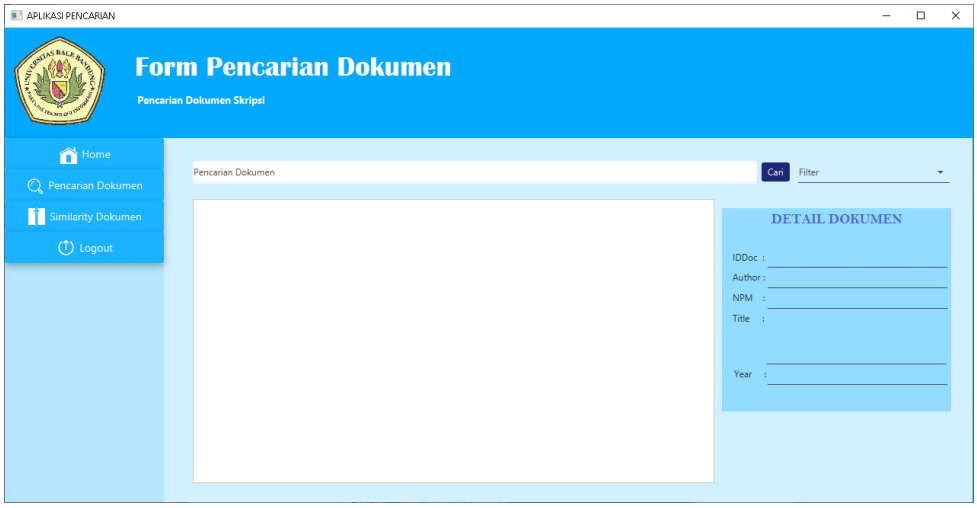


Gambar 5.3 Tampilan Struktur Menu Home Pencarian

Home Pencarian ini merupakan tampilan awal aplikasi pencarian dan di dalam nya terdapat tombol “Pencarian Dokumen”, “Pencarian Similarity Dokumen” dan ‘Logout”. Tombol “Pencarian Dokumen” digunakan untuk menuju ke Form Pencarian Dokumen. Tombol

"Pencarian Similarity Dokumen" digunakan untuk menuju ke Form *Similarity* Dokumen.

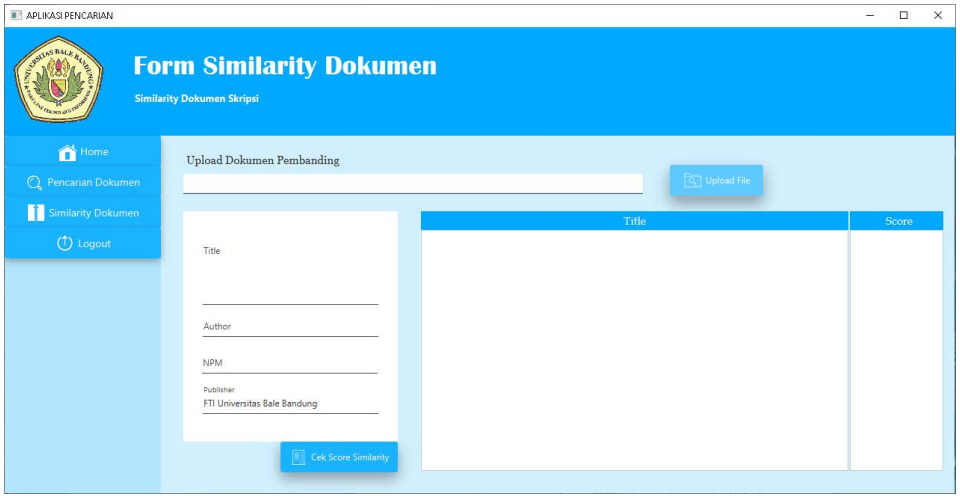
4. Pencarian Dokumen



Gambar 5.4 Tampilan Struktur Menu Pencarian Dokumen

Berdasarkan gambar diatas, Pencarian Dokumen ini dapat di akses dengan menekan tombol "Pencarian Dokumen" yang ada pada setiap form kecuali di Form Login dan Form Tambah Akun. Fungsi dari form ini yaitu untuk mencari judul skipsi sebagai referensi untuk menyusun skripsi.

5. *Similarity* Dokumen

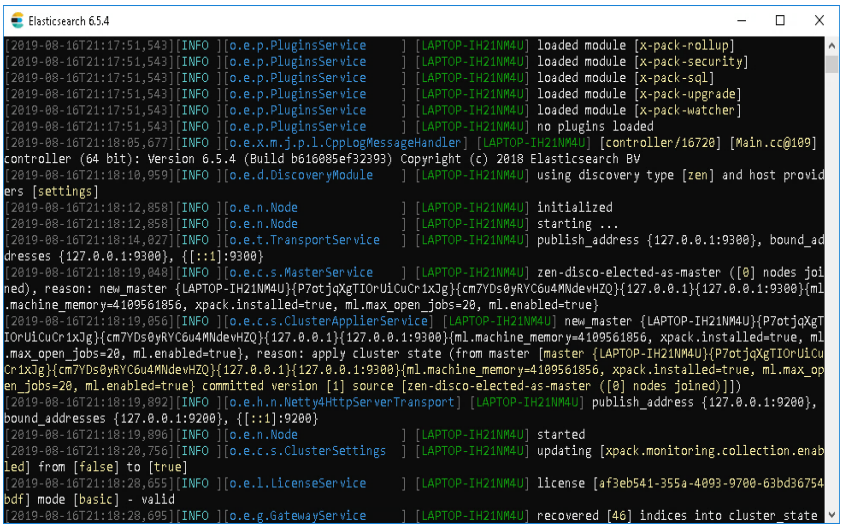


Gambar 5.5 Tampilan Struktur Menu *Similarity* Dokumen

Similarity Dokumen ini dapat di akses dengan menekan tombol "*Similarity* Dokumen" yang ada pada setiap form kecuali di Form Login dan Form Tambah Akun. Fungsi dari form ini yaitu untuk mencari persentase kesamaan (*Similarity*)dari proposal skripsi terhadap skipsi yang sudah ada.

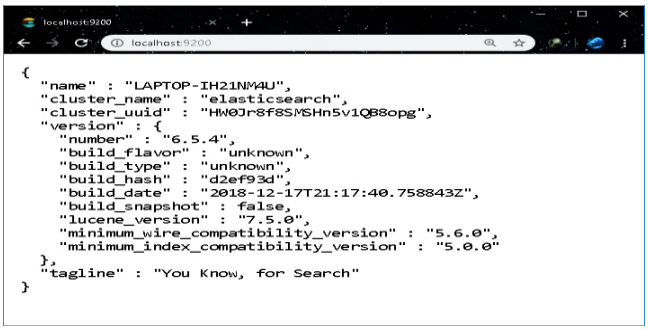
5.1.2 Implementasi Elasticsearch

Pertama *Elasticsearch* dijalankan untuk mengetahui keaktifan *Elasticsearch.* Berikut ini adalah tampilannya:



Gambar 5.6 Tampilan *Command Line Elasticsearch* Aktif

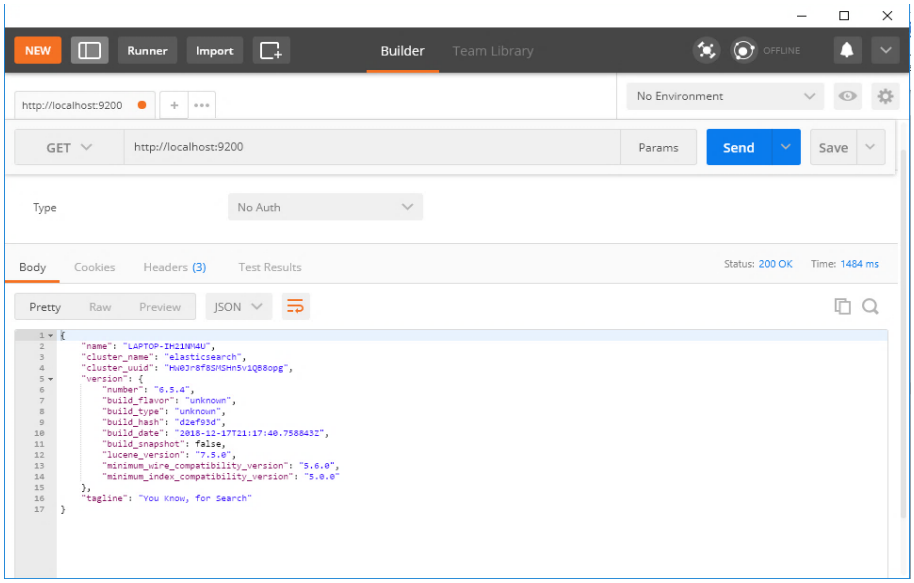
Gambar diatas menujukan bahwa *Elasticsearch* sudah aktif atau sudah siap untuk digunakan. Ditunjukan dengan munculnya kata "*started*" Berikut ini pengecekan untuk memastikan *Elasticsearch* sudah aktif:



Gambar 5.7 Tampilan Aktif *Elasticsearch* di *Browser*

Berdasarkan gambar diatas untuk memastikan bahwa *Elasticsearch* sudah aktif yaitu dengan cara mengetikan localhost:9200 pada *addres bar* pada browser lalu aka tampil JSON seperti yang ditunjukan pada gambar.

Berikut ini pengecekan untuk memastikan *Elasticsearch* sudah aktif melalui HTTP *Client Postman*:



Gambar 5.8 Tampilan Aktif *Elasticsearch* di *Postman*

Berdasarkan gambar diatas untuk memastikan bahwa *Elasticsearch* sudah aktif yaitu dengan cara melakukan HTTP *Request* GET localhost:9200 pada HTTP *Client* Postman lalu akan tampil JSON seperti yang ditunjukan pada gambar.

5.2 Pengujian

Pengujian dilakukan dengan menggunakan metode *blackbox* untuk mengetahui aplikasi berjalan dengan fungsinya dan sesuai dengan perancangan yang dilakukan pada bab sebelumnya. Dalam pengujian meliputi beberapa tahapan-tahapan yang akan mengetahui proses aplikasi berjalan. Berikut ini adalah pengujian dari beberapa tahap.

5.2.1 Rencana Pengujian

Rencana pengujian yang akan dilakukan dalam pengembangan aplikasi pencarian dokumen dan unruk menentukan *similarity* pada proposal ini menggunakan metode pengujian *blak box* yang menitik beratkan pada fingsi dari aplikasi. Berikut ini adalah rencana pengujian.

Tabel 5.1 Rencana Pengujian

No	Item Uji	Detail Uji	Jenis Uji
1.	Login User	Mengisi *username* dan *password* dan menekan tombol Masuk pada form Login	*Black Box*
		Menampilkan notifikasi saat mengisi *username* dan *password* yang salah.	*Black Box*
2.	Tambah Akun	Mengisi Npm, Nama, Email, Tlp, *Username*, *Password* dan Catatan dan menekan tombol Simpan	*Black Box*
		Melakukan Login di Form Tambah Akun	
3.	Halaman Utama Pencarian	Menekan Tombol Pencarian Dokumen	*Black Box*
		Menekan Tombol Pencarian Similarity Dokumen	*Black Box*
		Menekan Tombol Logout	*Black Box*
4.	Form Pencarian Dokumen	Mencari bersadarkan NPM	*Black Box*
		Mencari bersadarkan Year	*Black Box*
		Mencari bersadarkan Author	*Black Box*
		Mencari bersadarkan Title	*Black Box*
		Mencari bersadarkan	*Black Box*

		Keywords	
		Mencari bersadarkan IDDoc	*Black Box*
		Mencari bersadarkan Publisher	*Black Box*
		Mencari bersadarkan Status	*Black Box*
		Mencari bersadarkan Mounth	*Black Box*
		Mencari bersadarkan Tags	*Black Box*
		Klik Item hasil pencarian	*Black Box*
5.	Form *Similarity* Dokumen	Mengupload File Pdf	*Black Box*
		Menekan tombol Cek *Similarity*	*Black Box*

5.2.2 Kasus dan Hasil Pengujian

Kasus dan hasil pengujian berisi pemaparan dari rencana pengujian yang telah disusun pada rencana pengujian.

1. Kasus

Sebelum melakukan pengujian ada beberapa kasus, berikut ini adalah beberapa kasus:

- Untuk mengambil data dari server elasticsearch yaitu melakuakn http request ke elasticsearch, dengan uri yang sesuai dengan query untuk mencari berdasarkan filter yang di pilih.
- Ditemukannya format response dari http request adalah berbrntuk JSON sehingga harus dikonversi ke java object.
- Mengubah / parse JSON menjadi java object dengan library Jackson. Menemukan cara parsing JSON ke java object
- Dengan memakai library Jackson pada saat konversi lebih kompleks sehingga masih terjadi kesalahan/ error karna belum ditemukannya

format yang tepat untuk membuat sintaks penulisan, dengan menggunakan library JSON Simple lebih sederhana

- Menampilkan data java object ke listview di form pencarian
- Memecahkan masalah pengkonversi JSON menjadi java object ke listview
- Menganalisis data hasil report hasil dari perintah http://localhost:9200/_cat/indices?v yang menghasilkan
- Pengaturan clustering

2. Hasil Pengujian

Pengujian ini dilakukan secara *black box* dengan hanya memperhatikan masukan ke dalam sistem dan keluaran dari masukan tersebut, berikut ini pemaparan dari setiap nomor pengujian yang terdapat pada rencana pengujian:

Tabel 5.2 Pengujian Aplikasi

Pengujian Aplikasi				
No	**Item Uji**	**Skenario Uji**	**Hasil yang diharapkan**	**Hasil Pengujian**
1.	Login User	Mengisi *username* dan *password* dan menekan *button* Masuk	Jika username dan password benar maka akan muncul tampilan Form Home Pencarian	+Berhasil
		Menampilkan notifikasi saat mengisi *username* dan *password* yang salah.	Ketika memasukan username dan atau password yang salah akan muncul alert peringatan	+Berhasil
2.	Tambah Akun	Mengisi Npm, Nama, Email,	Ketika klik tombol simpan akan muncul	+Berhasil

		Tlp, *Username*, *Password* dan Catatan dan menekan tombol Simpan	alert pemberitahuan yang menyatakan data sudah tersimpan.	
		Melakukan Login di Form Tambah Akun	Ketika sudah menyimpan data di form tambah akun , dapat melakukan login di form yang sama dan telah di sediakan untuk melakukan login	+Berhasil
3.	Halaman Utama Pencarian	Menekan Tombol Pencarian Dokumen	Ketika menekan tombol pencarian dokumen maka akan di arahkan ke Form Pencarian Dokumen	+Berhasil
		Menekan Tombol Pencarian Similarity Dokumen	Ketika menekan tombol similarity dokumen maka akan di arahkan ke Form Similarity Dokumen.	+Berhasil
		Menekan Tombol Logout	Ketika menekan tombol logout akan keluar dari operasi pencarian dan akan di arahkan ke halaman awal saat mebuka aplikasi yaitu Form Login	+Berhasil

4.	Form Pencarian Dokumen	Mencari bersadarkan NPM	Akan menampilkan judul berdasarkan NPM yang di cari	+Berhasil
		Mencari bersadarkan Year	Akan menampilkan judul berdasarkan tahun (year) yang di cari	+Berhasil
		Mencari bersadarkan Author	Akan menampilkan judul berdasarkan Author yang di cari	+Berhasil
		Mencari bersadarkan Title	Akan menampilkan judul berdasarkan Title yang di cari	+Berhasil
		Mencari bersadarkan Keywords	Akan menampilkan judul berdasarkan Keywords yang di cari	+Berhasil
		Mencari bersadarkan IDDoc	Akan menampilkan judul berdasarkan IDDoc yang di cari	+Berhasil
		Mencari bersadarkan Publisher	Akan menampilkan judul berdasarkan Publisher yang di cari	+Berhasil
		Mencari bersadarkan Status	Akan menampilkan judul berdasarkan Status yang di cari	+Berhasil
		Mencari bersadarkan Mounth	Akan menampilkan judul berdasarkan Mounth yang di cari	+Berhasil
		Mencari bersadarkan	Akan menampilkan judul berdasarkan	+Berhasil

		Tags	Tags yang di cari	
		Klik Item hasil pencarian	Ketika klik salah satu item hasil pencarian maka akan muncul detail dari item yang di klik	+Berhasil
5.	Form *Similarity* Dokumen	Mengupload File Pdf	Ketika mengupload file, pdf akan di konversi menjadi teks	+Berhasil
		Menekan tombol Cek skor *Similarity*	Ketika menekan tombol cek skor similarity maka akan muncul title dan juga score	+Berhasil

Keterangan:

+ Berhasil

= Tidak Berhasil

5.2.3 Kesimpulan Hasil Pengujian

Berdasarkan hasil pengujian dengan metode *black box* dapat di simpulkan bahwa aplikasi ini secara fungsional mengeluarkan hasil yang sesuai dengan yang diharapkan yaitu dapat mencari judul berdasarkan filter yang di pilih dan dapat melakukan pencarian fulltext untuk mengetahui skor similarity akan tetapi masih terdapat banyak kekurangan saat melakukan pencarian fulltext yang masih terbatas serta query yang digunakan masih menggunakan query simple pada elasticsearch, sehingga membutuhkan proses *maintenance* untuk lebih mengetahui kekurangan dari aplikasi, sehingga dapat dikembangkan lebih lanjut.

BAB VI

KESIMPULAN DAN SARAN

Dari hasil analisis dan pengujian yang tlah di lakukan di bab 4 maka saya menyimpilkan bahwa

6.1 Kesimpulan

Berdasarkan hasil analisis, pengumpulan data, dan perancangan maka hasil yang tercapai adalah menyelesaikan laporan dan Aplikasi Pencarian Dan Menentukan *Similarity* Pada Proposal Skripsi Di Fakultas Teknologi Informasi Universitas Bale Bandung, penulis dapat menyimpulkan bahwa

1. Aplikasi ini dapat melakukan pencarian berdasarkan filter yang dipilih, dan berbeda dari yang sebelumnya masih melakukan pencarian secara konvensional.
2. Aplikasi ini dapat menampilkan skor berdasarkan relevansi setiap dokumen yang dibandingkan dari perbandingan proposal skripsi terhadap skripsi yang tersimpan dalam repository.
3. Aplikasi ini mengunakan *query simple elasticsearch* yang dipanggil dengan URL. Sehingga aplikasi ini terbatas saatmelakukan pencarian dengan *fulltext* .

6.2 Saran

Dalam proses pembuat Aplikasi Pencarian Dan Menentukan *Similarity* Pada Proposal Skripsi Di Fakultas Teknologi Informasi Universitas Bale Bandung penulis masih banyak sekali memiliki kekurangan dan jauh dari sempurna, sehingga penulis berharap untuk peneliti selanjutnya dapat mengembangkan lagi aplikasi ini, diantaranya

1. Aplikasi ini ditunjukan untuk menentukan kemiripan atau *similarity* pada proposal skripsi, tetapi sebenarnya bisa di terapkan untuk mencari

similarity secara penuh pada dokumen yang ada. Akan tetapi aplikasi ini jauh dari sempurna.

2. Aplikasi ini dapat di tambahkan fitur ataupun performansinya, selain itu dapat dikembangkan menjadi berbasis web maupun berbasis mobile maupun dengan bahasa pemrograman lain.
3. Aplikasi ini membutuhkan spesifikasi perangkat keras yang tinggi untuk memaksimalkan dalam pencarian fulltext.

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.

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.

C, A. (2017, Des 18). *Pengertian Bahasa Pemograman PHP Menurut Para Ahli*. Retrieved Mei 1, 2019, from mastekno.com/id/pengertian-bahasa-pemograman-php-menurut-para-ahli: www.mastekno.com

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.

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.

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 MengetahuiSyarah Hadits Berbasis Web (Studi Kasus: Syarah Umdatil Ahkam). *Jurnal Teknik Informatika, Vol 11 No. 2*.

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

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

Pasca Nugraha, R. M. (2011). Pengembangan Aplikasi QR Code Generator dan QR Code Reader dari Data Berbentuk Image. *KNIF*, 148-155.

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.

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

Rosa, M Shalahuddin. (2016). *Rekayasa Perangkat Lunak Terstruktur dan Berorientasi Objek.* Jakarta: Inforamtika Bandung.

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

Sugiarti, Y. (2013). *Analisis & Perancangan UML (Unified Modeling Language) bisa digunakan untuk:*. Yogyakarta: Graha Ilmu.

Surya, A. P. (2017). Aplikasi Pendeteksi Plagiarisme Dalam Karya Imliah Dengan Algoritma Rabin Krap. 17.

Tim Redaksi. (2008). *Kamus Bahasa Indonesia.* Jakarta.

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

LAMPIRAN

Lampiran 1 Form Wawancara

Narasumber : Yusuf Muharom, S.Kom

Hari / Tanggal : Kamis / 25 April 2019

Waktu : 15.00

Tempat : Ruang Tata Usaha

No	Pertanyaan	Jawaban	
		Ya	Tidak
1.	Apa pencarian laporan skripsi masih manual?	✓	
2.	Apakah perlu aplikasi untuk membantu pencarian skripsi?	✓	
3.	Apakah ada aplikasi untuk membantu dalam pencarian?		✓
4.	Apakah dengan akan diterapkannya aplikasi akan membantu mahasiswa?	✓	
5.	Apakah sudah ada aplikasi untuk menetukan kemiripan pada proposal?		✓
6.	Apakah pencarian yang cepat, efisien dan akurat dapat membantu?	✓	
7.	Apa perbedaan jika pencarian tanpa aplikasi dan pencarian memakai aplikasi?	✓	
8.	Apakah boleh jika mengambil topik yang sama dengan skripsi yang sudah ada?	✓	

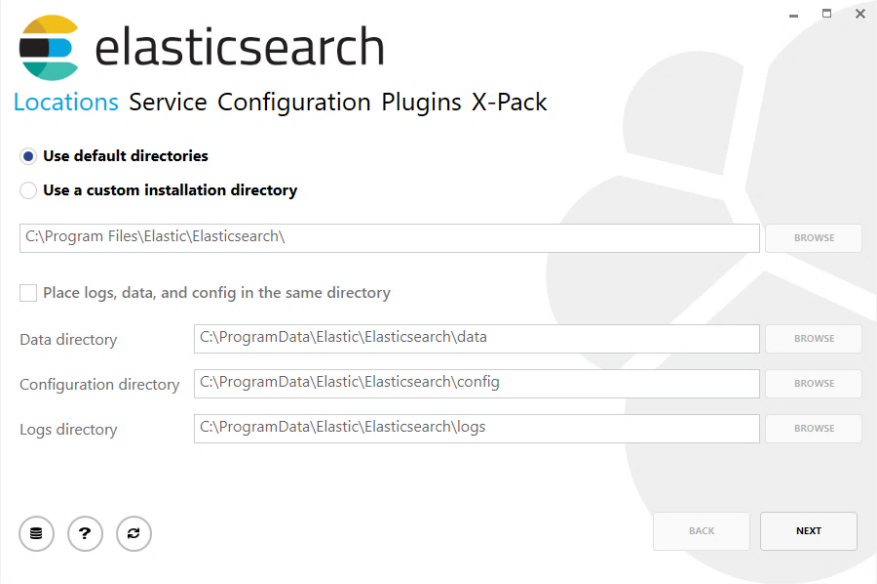
Bandung, 25 April 2019

Narasumber

Yusuf Muharom, S.Kom

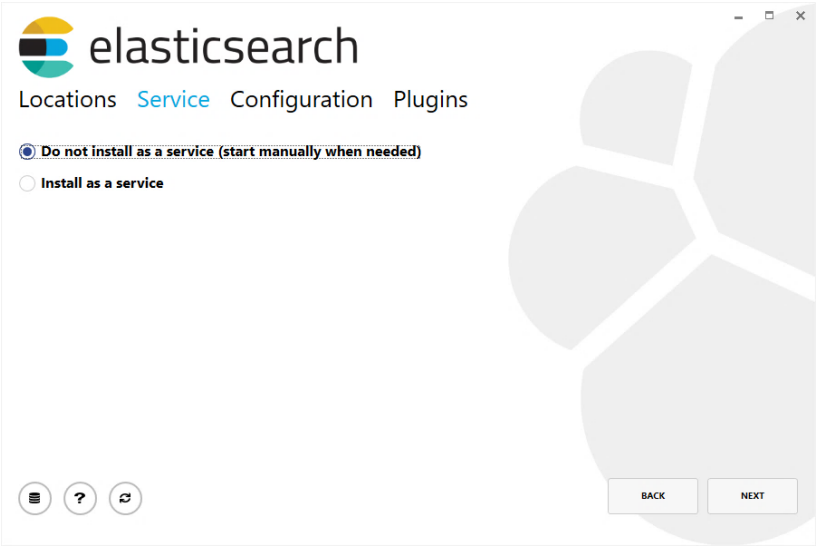
Lampiran 2 Instalasi Elasticsearch 6.5.4

1. Download Elasticsearch 6.5.4 MSI di: https://artifacts.elastic.co/downloads/elasticsearch/elasticsearch-6.5.4.msi.
2. Klik dua kali file yang diunduh untuk memunculkan GUI. Di dalam layar pertama, pilih direktori penempatan:



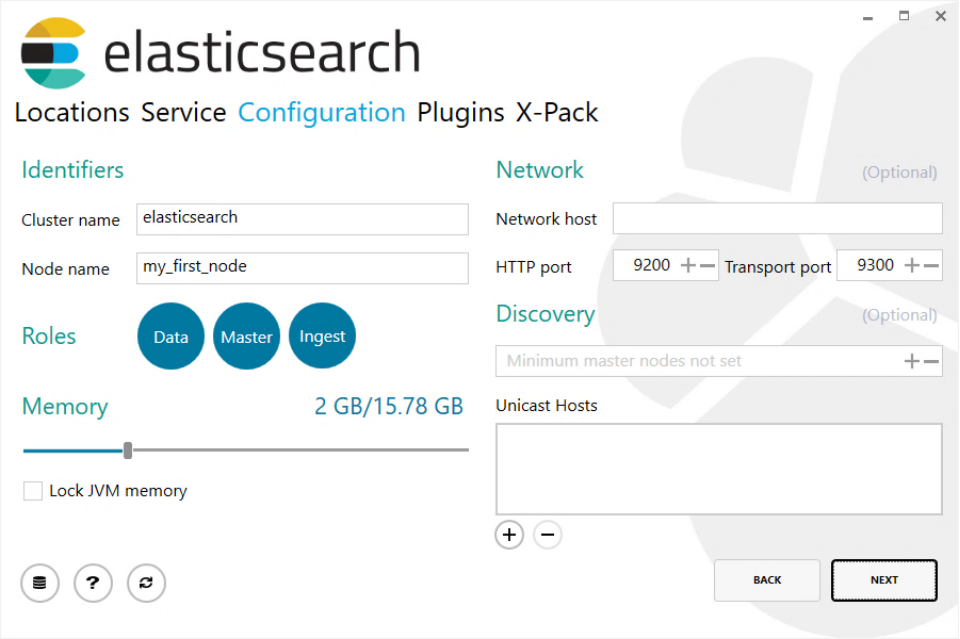
Gambar Lampiran 3.1 Lokasi Menginstall *Elasticsearch* MSI

3. Kemudian pilih "Do not install a service". Untuk menyelaraskan dengan contoh tar, pilih untuk tidak menginstal sebagai layanan:



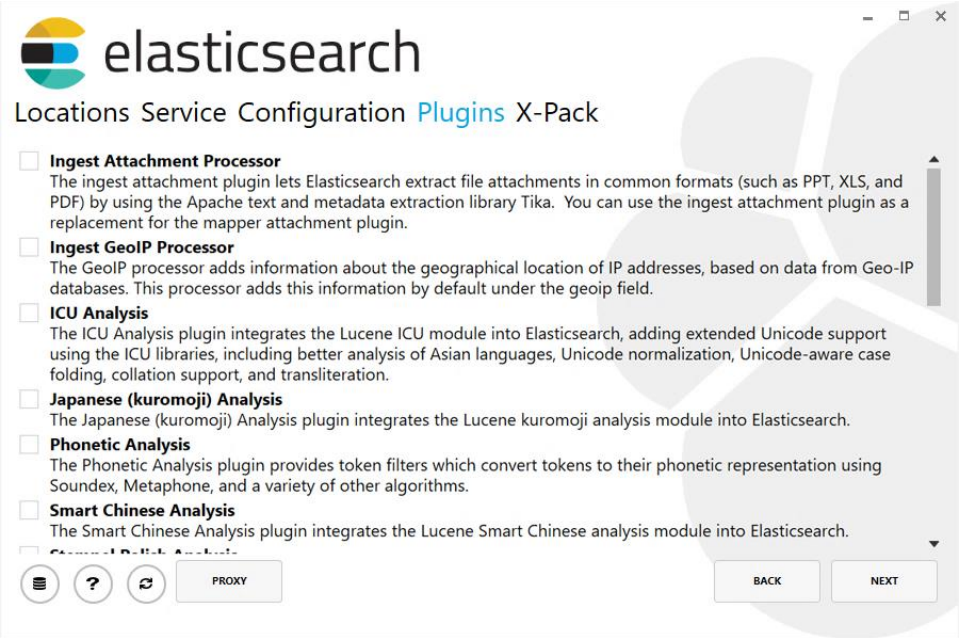
Gambar Lampiran 3.2 *Install Elasticsearch* MSI Tanpa *Service*

4. Untuk konfigurasi, cukup tinggalkan nilai default:



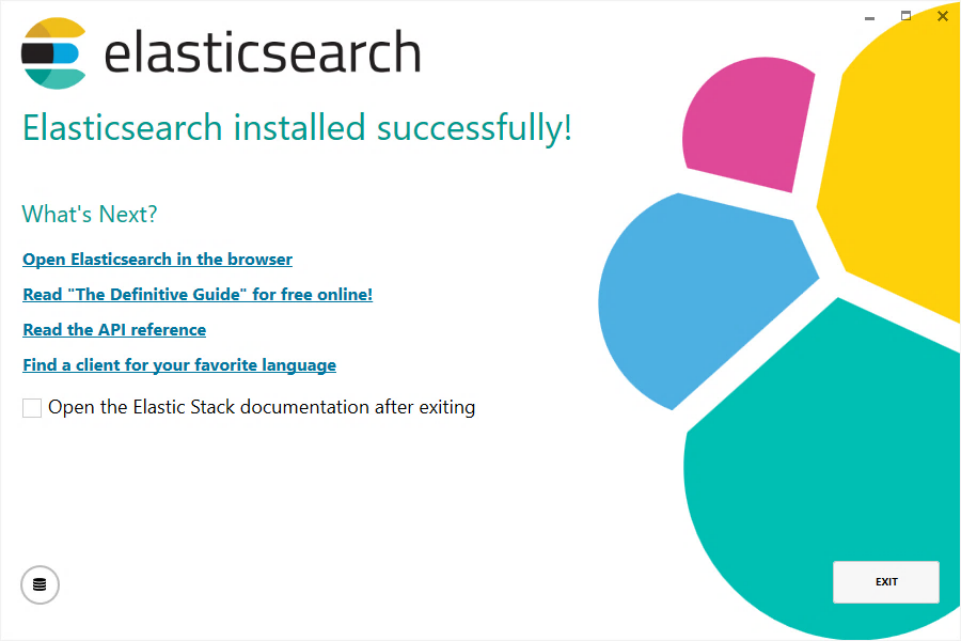
Gambar Lampiran 3.3 Konfigurasi *Install Elasticsearch* MSI

5. Sekali lagi, untuk menyelaraskan dengan contoh tar, hapus centang semua plugin untuk tidak menginstal plugins:



Gambar Lampiran 3. 4 Pligin *Install Elasticsearch* MSI

6. Setelah mengklik tombol install, Elasticsearch akan diinstal:

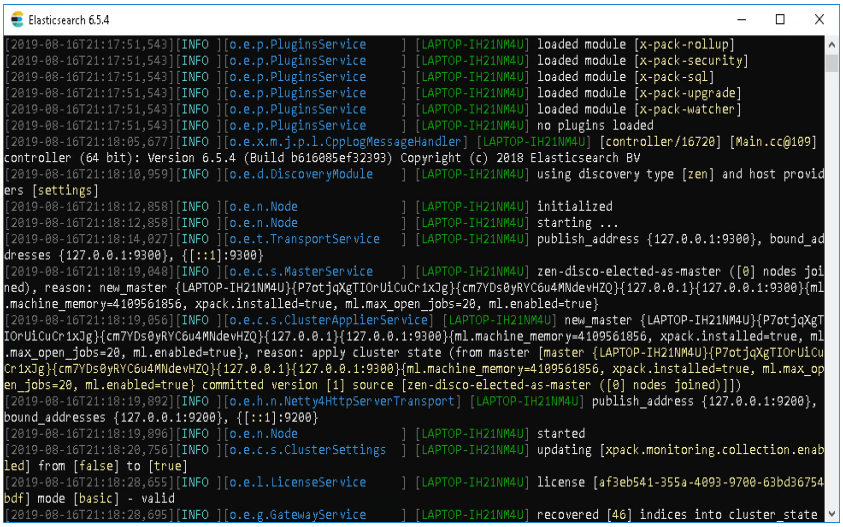


Gambar Lampiran 3.5 *Install Elasticsearch* MSI Sukses

7. Secara default, Elasticsearch akan diinstal di% PROGRAMFILES% \ Elastic \ Elasticsearch. Navigasi di sini dan masuk ke direktori bin sebagai berikut:

   cd %PROGRAMFILES%\Elastic\Elasticsearch\bin

8. Berhasil menjalankan node. Jika semuanya berjalan baik dengan instalasi, Anda akan melihat banyak pesan yang terlihat seperti di bawah ini:



Gambar Lampiran 3.6 Sukses Menjalankan *Node Elasticsearch*

Lampiran 3 Listing Program

F_LoginController.java

```
public class F_LoginController implements Initializable {
    private Label label;
    @FXML
    private AnchorPane formlogin;
    @FXML
    private ImageView close;
    @FXML
    private JFXTextField tfuser;
    @FXML
    private JFXPasswordField tfpass;
    @FXML
    private JFXButton btnMasuk;
    @FXML
    private JFXButton btntbakn;
    Stage dialogStage = new Stage();
    Scene scene;
    Connection connection = null;
    PreparedStatement preparedStatement = null;
    ResultSet resultSet = null;
    public F_LoginController() {
        connection = ConnectionUtil.connectdb();
    }
    @Override
    public void initialize(URL url, ResourceBundle rb) {
    }
    @FXML
    private void actionMsk(ActionEvent event) {
        String username = tfuser.getText().toString();
        String password = tfpass.getText().toString();
        String sql = "SELECT * FROM tabel_user WHERE username = ? and password = ?";
        try{
            preparedStatement = connection.prepareStatement(sql);
            preparedStatement.setString(1, username);
            preparedStatement.setString(2, password);
            resultSet = preparedStatement.executeQuery();
            if(!resultSet.next()){
                infoBox("Mohon Masikan Username dan Password yang Benar!!", null, "Gagal!");
            }else{
                infoBox("Login Sukses",null,"Sukses" );
                Node node = (Node)event.getSource();
                dialogStage = (Stage) node.getScene().getWindow();
                dialogStage.close();
                scene = new Scene((Parent)
FXMLLoader.load(getClass().getResource("/fxml/F_HomePencarian.fxml")));
                dialogStage.setScene(scene);
                dialogStage.show();
            }
        }
        catch(Exception e){
            e.printStackTrace();
        }
    }
    public static void infoBox(String infoMessage, String headerText, String title){
        Alert alert = new Alert(Alert.AlertType.CONFIRMATION);
```

```
        alert.setContentText(infoMessage);
        alert.setTitle(title);
        alert.setHeaderText(headerText);
        alert.showAndWait();
    }
    @FXML
    private void actionTbAkn(ActionEvent event) {
        formlogin.getChildren().clear();
            Node [] nodes = new  Node[15];
            for(int i = 0; i<10; i++){
                try {
                    nodes[i] =
(Node)FXMLLoader.load(getClass().getResource("/fxml/F_TambahAkun.fxml"));
                    formlogin.getChildren().add(nodes[i]);
                } catch (IOException ex) {
                        Logger.getLogger(F_LoginController.class.getName()).log(Level.SEVERE, null,
ex);
                }
            }
    }
}
```

F_TambahAkunController.java

```
public class F_TambahAkunController implements Initializable {
    @FXML
    private JFXTextField tfuser;
    @FXML
    private JFXPasswordField tfpass;
    @FXML
    private JFXButton btnMasuk;
    @FXML
    private JFXTextField npm;
    @FXML
    private JFXTextField nama;
    @FXML
    private JFXTextField email;
    @FXML
    private JFXTextField tlp;
    @FXML
    private JFXTextField username;
    @FXML
    private JFXPasswordField password;
    @FXML
    private JFXTextArea catatan;
    @FXML
    private JFXButton btnbatal;
    @FXML
    private JFXButton btnsimpan;
    @FXML
    private AnchorPane formtbhakn;
    Stage dialogStage = new Stage();
    Scene scene;
    Connection connection = null;
    PreparedStatement preparedStatement = null;
    ResultSet resultSet = null;
    public F_TambahAkunController() {
        connection = ConnectionUtil.connectdb();
```

```
    }
    @Override
    public void initialize(URL url, ResourceBundle rb) {
    }
    @FXML
    private void actionMsk(ActionEvent event) {
        String username = tfuser.getText().toString();
        String password = tfpass.getText().toString();
        String sql = "SELECT * FROM tabel_user WHERE username = ? and password = ?";
        try{
            preparedStatement = connection.prepareStatement(sql);
            preparedStatement.setString(1, username);
            preparedStatement.setString(2, password);
            resultSet = preparedStatement.executeQuery();
            if(!resultSet.next()){
                infoBox("Mohon Masikan Username dan Password yang Benar!!", null, "Gagal!");
            }else{
                infoBox("Login Sukses",null,"Sukses" );
                Node node = (Node)event.getSource();
                dialogStage = (Stage) node.getScene().getWindow();
                dialogStage.close();
                scene = new Scene((Parent)
FXMLLoader.load(getClass().getResource("/fxml/F_HomePencarian.fxml")));
                dialogStage.setScene(scene);
                dialogStage.show();
            }
        }
        catch(Exception e){
            e.printStackTrace();
        }
    }
    @FXML
    private void actionSimpan(ActionEvent event) {
        try {
            String sql = "INSERT INTO tabel_user VALUES
('"+npm.getText()+"','"+nama.getText()+"','"+email.getText()+"','"+tlp.getText()+"','"+username.g
etText()+"','"+password.getText()+"','"+catatan.getText()+"')";
            java.sql.Connection conn=(Connection)config.configDB();
            java.sql.PreparedStatement pst=conn.prepareStatement(sql);
            pst.execute();
            Alert alert = new Alert(Alert.AlertType.INFORMATION);
            alert.setContentText("Data berhasil disimpan !");
            alert.setTitle("Tambah Akun");
            alert.showAndWait();
        } catch (Exception e) {
            Alert alert = new Alert(Alert.AlertType.WARNING);
            alert.setContentText("Data gagal disimpan !");
            alert.setTitle("Tambah Akun");
            alert.showAndWait();
        }
    }
}
```

F_HomePencarianController.java

```
public class F_HomePencarianController implements Initializable {
    @FXML
    private ImageView close;
```

```
    @FXML
    private JFXButton btnsearch;
    @FXML
    private JFXButton btnsimilarity;
    @FXML
    private AnchorPane f_homepencarian;
    @FXML
    private JFXButton btnLogout;
    @Override
    public void initialize(URL url, ResourceBundle rb) {
    }
    @FXML
    private void actionPeDok(ActionEvent event) {
        f_homepencarian.getChildren().clear();
        Node [] nodes = new  Node[15];
        for(int i = 0; i<10; i++){
            try {
                nodes[i] =
(Node)FXMLLoader.load(getClass().getResource("/fxml/F_PencarianDokumen.fxml"));
                f_homepencarian.getChildren().add(nodes[i]);
            } catch (IOException ex) {
                Logger.getLogger(F_HomePencarianController.class.getName()).log(Level.SEVERE,
null, ex);
            }
        }
    }
    @FXML
    private void actionSimDok(ActionEvent event) {
        f_homepencarian.getChildren().clear();
        Node [] nodes = new  Node[15];
        for(int i = 0; i<10; i++){
            try {
                nodes[i] =
(Node)FXMLLoader.load(getClass().getResource("/fxml/F_PencarianSimilarityDokumen.fxml"));
                f_homepencarian.getChildren().add(nodes[i]);
            } catch (IOException ex) {
                Logger.getLogger(F_HomePencarianController.class.getName()).log(Level.SEVERE,
null, ex);
            }
        }
    }
    @FXML
    private void actionLogout(ActionEvent event) {
        Alert alert = new Alert(Alert.AlertType.CONFIRMATION);
        alert.setContentText("Anda yakin akan Logout?");
        alert.setTitle("Logout");
        alert.showAndWait();
        Node [] nodes = new  Node[15];
        for(int i = 0; i<10; i++){
            try {
                nodes[i] = (Node)FXMLLoader.load(getClass().getResource("/fxml/Login.fxml"));
                f_homepencarian.getChildren().add(nodes[i]);
            } catch (IOException ex) {
                Logger.getLogger(F_HomePencarianController.class.getName()).log(Level.SEVERE,
null, ex);
            }
        }
    }
```

```
}
```

config.java

```
public class config {
    private static Connection mysqlconfig;
    public static Connection configDB()throws SQLException{
        try {
            String url="jdbc:mysql://localhost:3306/repository"; //url database
            String user="root"; //user database
            String pass=""; //password database
            DriverManager.registerDriver(new com.mysql.jdbc.Driver());
            mysqlconfig=DriverManager.getConnection(url, user, pass);
        } catch (Exception e) {
            System.err.println("koneksi gagal "+e.getMessage()); //perintah menampilkan error pada
koneksi
        }
        return mysqlconfig;
    }
}
```

ConnectionUtil.java

```
public class ConnectionUtil {
    Connection conn = null;
    public static Connection connectdb()
    {
        try
        {
            Class.forName("com.mysql.jdbc.Driver");
            Connection conn =
DriverManager.getConnection("jdbc:mysql://localhost/repository","root","");
            return conn;
        }
        catch(Exception e)
        {
            JOptionPane.showMessageDialog(null, e);
            return null;
        }
    }
}
```

F_Login.fxml

```
<AnchorPane id="AnchorPane" fx:id="formlogin" prefHeight="623.0" prefWidth="1271.0"
style="-fx-background-color: #032a57;" xmlns="http://javafx.com/javafx/8.0.171"
xmlns:fx="http://javafx.com/fxml/1"
fx:controller="com.mycompany.aplikasipencarian.F_LoginController">
   <children>
      <ImageView fitHeight="623.0" fitWidth="1107.0" layoutX="-401.0" layoutY="1.0"
opacity="0.4" pickOnBounds="true" preserveRatio="true">
         <image>
            <Image url="@../images/872784.jpg" />
         </image>
      </ImageView>
      <ImageView fitHeight="113.0" fitWidth="114.0" layoutX="11.0" layoutY="5.0"
pickOnBounds="true" preserveRatio="true">
         <image>
```

```
          <Image url="@../images/Logo-FTI-UNIBBA.png" />
        </image>
      </ImageView>
      <Label layoutX="48.0" layoutY="226.0" text="Selamat Datang " textFill="WHITE">
        <font>
          <Font size="45.0" />
        </font>
      </Label>
      <Label layoutX="48.0" layoutY="315.0" text="Di Repository Fakultas Teknologi Informasi
Universitas Bale Bandung" textFill="WHITE">
        <font>
          <Font size="21.0" />
        </font>
      </Label>
      <AnchorPane layoutX="824.0" layoutY="142.0" prefHeight="384.0" prefWidth="309.0"
style="-fx-background-color: #3686f5;">
        <children>
          <ImageView fitHeight="59.0" fitWidth="52.0" layoutX="130.0" layoutY="44.0"
pickOnBounds="true" preserveRatio="true">
            <image>
              <Image url="@../images/user.png" />
            </image>
          </ImageView>
          <JFXTextField fx:id="tfuser" focusColor="WHITE" labelFloat="true" layoutX="53.0"
layoutY="119.0" prefHeight="30.0" prefWidth="207.0" promptText="username/email"
unFocusColor="WHITE">
            <font>
              <Font size="14.0" />
            </font>
          </JFXTextField>
          <JFXPasswordField fx:id="tfpass" focusColor="WHITE" labelFloat="true"
layoutX="53.0" layoutY="189.0" prefHeight="25.0" prefWidth="207.0" promptText="password"
unFocusColor="WHITE">
            <font>
              <Font size="14.0" />
            </font>
          </JFXPasswordField>
          <JFXButton fx:id="btnMasuk" buttonType="RAISED" layoutX="53.0" layoutY="266.0"
onAction="#actionMsk" prefHeight="25.0" prefWidth="207.0" style="-fx-background-color:
#064cdb;" text="Masuk" textFill="WHITE">
          </JFXButton>
        </children>
      </AnchorPane>
      <JFXButton fx:id="btntbakn" buttonType="RAISED" layoutX="281.0" layoutY="354.0"
onAction="#actionTbAkn" prefHeight="25.0" prefWidth="170.0" style="-fx-background-color:
#064cdb;" text="Tambah Akun Baru" textFill="WHITE">
      </JFXButton>
      <Pane layoutX="824.0" layoutY="76.0" prefHeight="66.0" prefWidth="309.0" style="-fx-
background-color: #1b81ff;" AnchorPane.bottomAnchor="482.0"
AnchorPane.leftAnchor="824.0" AnchorPane.rightAnchor="138.0"
AnchorPane.topAnchor="76.0">
        <children>
          <Label layoutX="26.0" layoutY="25.0" text="Form Login" textFill="WHITE">
            <font>
              <Font name="Britannic Bold" size="30.0" />
            </font>
          </Label>
        </children>
```

```
    </Pane>
  </children>
</AnchorPane>
```

F_TambahAkun.fxml

```
<AnchorPane id="AnchorPane" fx:id="formtbhakn" prefHeight="623.0" prefWidth="1271.0"
style="-fx-background-color: #032a57;" xmlns="http://javafx.com/javafx/8.0.171"
xmlns:fx="http://javafx.com/fxml/1"
fx:controller="com.mycompany.aplikasipencarian.F_TambahAkunController">
  <children>
    <ImageView fitHeight="623.0" fitWidth="1107.0" layoutX="-412.0" opacity="0.4"
pickOnBounds="true" preserveRatio="true">
      <image>
        <Image url="@../images/872784.jpg" />
      </image>
    </ImageView>
    <ImageView fitHeight="113.0" fitWidth="114.0" pickOnBounds="true"
preserveRatio="true">
      <image>
        <Image url="@../images/Logo-FTI-UNIBBA.png" />
      </image>
    </ImageView>
    <Label layoutX="114.0" layoutY="55.0" text="Selamat Datang " textFill="WHITE">
      <font>
        <Font size="45.0" />
      </font>
    </Label>
    <Label layoutX="36.0" layoutY="121.0" prefHeight="31.0" prefWidth="647.0" text="Di
Repository Fakultas Teknologi Informasi Universitas Bale Bandung" textFill="WHITE">
      <font>
        <Font size="21.0" />
      </font>
    </Label>
    <AnchorPane layoutX="813.0" layoutY="137.0" prefHeight="384.0" prefWidth="309.0"
style="-fx-background-color: #3686f5;">
      <children>
        <ImageView fitHeight="59.0" fitWidth="52.0" layoutX="130.0" layoutY="44.0"
pickOnBounds="true" preserveRatio="true">
          <image>
            <Image url="@../images/user.png" />
          </image>
        </ImageView>
        <JFXTextField fx:id="tfuser" focusColor="WHITE" labelFloat="true" layoutX="53.0"
layoutY="119.0" prefHeight="30.0" prefWidth="207.0" promptText="username/email"
unFocusColor="WHITE">
          <font>
            <Font size="14.0" />
          </font>
        </JFXTextField>
        <JFXPasswordField fx:id="tfpass" focusColor="WHITE" labelFloat="true"
layoutX="53.0" layoutY="189.0" prefHeight="25.0" prefWidth="207.0" promptText="password"
unFocusColor="WHITE">
          <font>
            <Font size="14.0" />
          </font>
        </JFXPasswordField>
        <JFXButton fx:id="btnMasuk" buttonType="RAISED" layoutX="53.0" layoutY="266.0"
```

```
onAction="#actionMsk" prefHeight="25.0" prefWidth="207.0" style="-fx-background-color:
#064cdb;" text="Masuk" textFill="WHITE" />
      </children>
    </AnchorPane>
    <Pane layoutX="813.0" layoutY="71.0" prefHeight="66.0" prefWidth="309.0" style="-fx-
background-color: #1b81ff;">
      <children>
        <Label layoutX="26.0" layoutY="25.0" text="Form Login" textFill="WHITE">
          <font>
            <Font name="Britannic Bold" size="30.0" />
          </font>
        </Label>
      </children>
    </Pane>
    <AnchorPane layoutX="43.0" layoutY="224.0" prefHeight="361.0" prefWidth="557.0"
style="-fx-background-color: #a5cfff;">
      <children>
        <Label layoutX="23.0" layoutY="14.0" text="NPM" />
        <JFXTextField fx:id="npm" layoutX="23.0" layoutY="38.0" promptText="*Masukan
NPM" />
        <Label layoutX="23.0" layoutY="80.0" text="Nama" />
        <JFXTextField fx:id="nama" layoutX="23.0" layoutY="104.0" prefHeight="25.0"
prefWidth="219.0" promptText="*Masukan Nama Lengkap" />
        <Label layoutX="23.0" layoutY="146.0" text="Email" />
        <JFXTextField fx:id="email" layoutX="23.0" layoutY="170.0" prefHeight="25.0"
prefWidth="219.0" promptText="*Masukan Email" />
        <Label layoutX="23.0" layoutY="211.0" text="Tlp" />
        <JFXTextField fx:id="tlp" layoutX="23.0" layoutY="235.0" prefHeight="25.0"
prefWidth="157.0" promptText="*Masukan No. Telepon aktif" />
        <Label layoutX="308.0" layoutY="14.0" text="Username" />
        <JFXTextField fx:id="username" layoutX="308.0" layoutY="38.0" prefHeight="25.0"
prefWidth="219.0" promptText="*Masukan Username" />
        <Label layoutX="308.0" layoutY="80.0" text="Password" />
        <JFXPasswordField fx:id="password" layoutX="308.0" layoutY="104.0"
prefHeight="25.0" prefWidth="219.0" promptText="*Masukan Password" />
        <JFXTextArea fx:id="catatan" layoutX="308.0" layoutY="141.0" prefHeight="72.0"
prefWidth="219.0" promptText="*Catatan" style="-fx-background-color: white;" />
        <JFXButton fx:id="btnsimpan" layoutX="327.0" layoutY="301.0"
onAction="#actionSimpan" prefHeight="25.0" prefWidth="63.0" style="-fx-background-color:
green;" text="Simpan" textFill="WHITE" />
      </children></AnchorPane>
    <Pane layoutX="43.0" layoutY="158.0" prefHeight="66.0" prefWidth="309.0" style="-fx-
background-color: #1b81ff;">
      <children>
        <Label layoutX="26.0" layoutY="25.0" text="Tambah Akun Baru" textFill="WHITE">
          <font>
            <Font name="Britannic Bold" size="30.0" />
          </font>
        </Label>
      </children>
    </Pane>
  </children>
</AnchorPane>
```

F_HomePencarian.fxml

```
<AnchorPane id="AnchorPane" fx:id="f_homepencarian" prefHeight="623.0"
prefWidth="1271.0" style="-fx-background-color: #004c7c;"
```

```
xmlns="http://javafx.com/javafx/8.0.171" xmlns:fx="http://javafx.com/fxml/1"
fx:controller="com.mycompany.aplikasipencarian.F_HomePencarianController">
  <children>
    <ImageView fitHeight="715.0" fitWidth="1271.0" layoutX="1.0" layoutY="-92.0"
opacity="0.18" pickOnBounds="true" preserveRatio="true">
      <image>
        <Image url="@../images/872784.jpg" />
      </image>
    </ImageView>
    <Pane layoutX="1.0" prefHeight="140.0" prefWidth="1271.0" style="-fx-background-color:
#004c7c;">
      <children>
        <ImageView fitHeight="140.0" fitWidth="146.0" pickOnBounds="true"
preserveRatio="true">
          <image>
            <Image url="@../images/Logo-FTI-UNIBBA.png" />
          </image>
        </ImageView>
        <Label layoutX="203.0" layoutY="26.0" text="Form Home Pencarian"
textFill="WHITE">
          <font>
            <Font name="Britannic Bold" size="36.0" />
          </font>
        </Label>
        <Label layoutX="203.0" layoutY="77.0" text="Pencarian ini meliputi Pencarian Dokumen
dan Pencarian Similarity Dokumen" textFill="WHITE">
          <font>
            <Font size="13.0" />
          </font>
        </Label>
      </children>
    </Pane>
    <JFXButton fx:id="btnsearch" buttonType="RAISED" layoutX="514.0" layoutY="291.0"
onAction="#actionPeDok" prefHeight="41.0" prefWidth="245.0" style="-fx-background-color:
#008bae;" text="Pencarian Dokumen " textFill="WHITE">
      <graphic>
        <ImageView fitHeight="27.0" fitWidth="29.0" pickOnBounds="true"
preserveRatio="true">
          <image>
            <Image url="@../images/icons8_search_128px.png" />
          </image>
        </ImageView>
      </graphic>
    </JFXButton>
    <JFXButton fx:id="btnsimilarity" buttonType="RAISED" layoutX="514.0" layoutY="353.0"
onAction="#actionSimDok" prefHeight="41.0" prefWidth="245.0" style="-fx-background-color:
#008bae;" text="Pencarian Similarity Dokumen " textFill="WHITE">
      <graphic>
        <ImageView fitHeight="27.0" fitWidth="29.0" pickOnBounds="true"
preserveRatio="true">
          <image>
            <Image url="@../images/icons8_similar_items_filled_100px.png" />
          </image>
        </ImageView>
      </graphic>
    </JFXButton>
     <JFXButton fx:id="btnLogout" buttonType="RAISED" layoutX="514.0" layoutY="417.0"
onAction="#actionLogout" prefHeight="41.0" prefWidth="245.0" style="-fx-background-color:
```

```
#008bae;" text="Logout" textFill="WHITE">
        <graphic>
          <ImageView fitHeight="27.0" fitWidth="29.0" pickOnBounds="true"
preserveRatio="true">
            <image>
              <Image url="@../images/icons8_logout_rounded_up_64px.png" />
            </image>
          </ImageView>
        </graphic>
      </JFXButton>
    </children>
</AnchorPane>
```

Lampiran 4 Biodata Penulis

Maesuri Fauziah lahir di Bandung, pada 7 Agustus 1997. Anak kedua dari empat bersaudara, yang terlahir dari pasangan Dudung Abdullah dan Enung Rofiqoh. Mulai mengenjang pendidikan di TK Mawar Putih (2003), kemudian dilanjutkan ke SDN Jalan Cagak (2004-2009), kemudian melanjutkan pendidikan ke SMP Islam Pacet (2009-2012), serta melanjutkan masa SMA di SMA Negeri 1 Ciparay (2012-2015). Untuk mendapatkan gelar sarjana penulis melanjutkan S1 penulis melanjutkan pendidikan di Universitas Bale Bandung Fakultas Teknologi Informasi Jurusan Teknik Informatika. Penulis juga pernah aktif sebagai anggota di UKM Bale Percussion di FTI UNIBBA pada tahun 2015-2016, selain itu penulis pernah aktif sebagai anggota HIMA-IF sampai tahun 2018.

“USAHA TAK AKAN MENGHIANATI HASIL, BERDO’A DAN IKHTIAR”

“TANAMLAH POHON WALAUPUN HARI ESOK AKAN MATI”