***Draft Seminar I***

**RANCANG BANGUN APLIKASI PENDETEKSI KESAMAAN DOKUMEN MENGGUNAKAN METODE *DICE SIMILARITY***

****

**ARIS AKHYAR ABDILLAH**

**H071171505**

|  |  |
| --- | --- |
| **Pembimbing Utama** | **: Dr. Hendra S.Si., M. Kom.** |
| **Pembimbing Pertama** | **: Edy Saputra R, S. Si., M. Si.** |
| **Penguji** | **:** |

**PROGRAM STUDI SISTEM INFORMASI**

**FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM**

**UNIVERSITAS HASANUDDIN**

**MAKASSAR**

**2022**

# DAFTAR ISI

[DAFTAR ISI ii](#_Toc110330315)

[DAFTAR GAMBAR iv](#_Toc110330316)

[DAFTAR TABEL v](#_Toc110330317)

[1 BAB I PENDAHULUAN 1](#_Toc110330318)

[1.1 Latar Belakang 1](#_Toc110330319)

[1.2 Rumusan Masalah 2](#_Toc110330320)

[1.3 Tujuan Penelitian 3](#_Toc110330321)

[1.4 Manfaat Penelitian 3](#_Toc110330322)

[1.5 Batasan Masalah 3](#_Toc110330323)

[2 BAB II TINJAUAN PUSTAKA 4](#_Toc110330324)

[2.1 Penelitian Terkait 4](#_Toc110330325)

[2.2 Landasan Teori 6](#_Toc110330326)

[2.2.1 Sistem Informasi 6](#_Toc110330327)

[2.2.2 *Hypertext Transfer Protocol Secure* (HTTPS) 6](#_Toc110330328)

[2.2.3 Javascript 7](#_Toc110330329)

[2.2.3.1 Karakteristik Javascript 7](#_Toc110330330)

[2.2.3.2 NodeJS 9](#_Toc110330331)

[2.2.3.3 ExpressJS 9](#_Toc110330332)

[2.2.4 *Database* 10](#_Toc110330333)

[2.2.4.1 SQL dan NoSQL 10](#_Toc110330334)

[2.2.4.2 MongoDB 10](#_Toc110330335)

[2.2.5 Jurnal 11](#_Toc110330336)

[2.2.6 Plagiarisme 12](#_Toc110330337)

[2.2.7 *Dice Similarity* 12](#_Toc110330338)

[2.2.7.1 Implementasi Dice Similarity 13](#_Toc110330339)

[3 BAB III METODE PENELITIAN 16](#_Toc110330340)

[3.1 Prosedur Penelitian 16](#_Toc110330341)

[3.2 Rancangan Sistem 16](#_Toc110330342)

[3.2.1 *Use Case Diagram* 16](#_Toc110330343)

[3.2.2 *Entity Relationship Diagram* (ERD) 17](#_Toc110330344)

[3.2.3 Desain Sistem 18](#_Toc110330345)

[3.2.3.1 Algoritma Input Dokumen ke Dalam *Database* 18](#_Toc110330346)

[3.2.3.2 Algoritma Perbandingan Dokumen 19](#_Toc110330347)

[3.3 Metode Penelitian 20](#_Toc110330348)

[3.3.1 Algoritma *Dice Similarity* 20](#_Toc110330349)

[3.4 Instrumen Penelitian 23](#_Toc110330350)

[DAFTAR PUSTAKA 24](#_Toc110330351)

# DAFTAR GAMBAR

[Gambar 3.1 *Use Case Diagram* 17](#_Toc110684024)

[Gambar 3.2 *Entity Relationship Diagram* (ERD) 18](#_Toc110684025)

[Gambar 3.3 *Flowchart* Input Dokumen ke Dalam Database 18](#_Toc110684026)

[Gambar 3.4 *Flowchart* Perbandingan Dokumen 19](#_Toc110684027)

[Gambar 3.5 *Flowchart* Implementasi *Dice Similarity* ke Dalam Sistem 22](#_Toc110684028)

# DAFTAR TABEL

[Tabel 3.1 Perbandingan Data X dan Data Y 20](#_Toc107246297)

[Tabel 3.2 Kamus Data 21](#_Toc107246298)

[Tabel 3.3 Perhtiungan di Dalam Kamus Data 22](#_Toc107246299)

# BAB I PENDAHULUAN

## Latar Belakang

Teknologi merupakan sarana untuk menyediakan barang-barang yang diperlukan untuk mempermudah kegiatan atau kehidupan manusia. Penggunaan teknologi oleh manusia dimulai dengan pengubahan sumber daya alam menjadi alat-alat sederhana. Penemuan di masa lampau tentang kemampuan mengendalikan api telah menaikkan ketersediaan sumber - sumber pangan, sedangkan penciptaan roda telah membantu manusia dalam bepergian dari satu tempat ke tempat lainnya. Perkembangan teknologi terbaru dimasa sekarang seperti mesin cetak, telepon, dan internet telah mengubah manusia untuk bekerja maupun berinteraksi dalam menjalankan kehidupannya.

Dengan adanya teknologi di saat ini, dapat dikatakan pekerjaan manusia semakin mudah. Seperti untuk akses informasi, sekarang sudah banyak sekali informasi yang tersebar di internet dan hampir semua yang dicari dapat ditemukan. Tidak seperti dahulu, untuk mengakses informasi, orang biasa pergi ke perpustakaan untuk mendapatkan informasi yang diinginkan. Hal itu memungkinkan karena adanya digitalisasi dokumen yang dulunya hanya berupa *hardcopy* (salinan cetak) menjadi *softcopy* (salinan digital). Dengan hal tersebut kita bisa mengakses semua dokumen kapan saja dan dimana saja. Dengan semua kemudahan yang diberikan terknologi seperti contoh diatas, timbul suatu masalah yaitu mudahnya dilakukan plagiarisme yang merupakan suatu tindakan yang tidak baik.

Plagiarisme berasal dari bahasa Latin yaitu “*plagiare*” yang berarti mencuri. Plagiarisme berasal dari kata plagiat yang berarti pengambilan karangan (pendapat dan sebagainya) orang lain dan menjadikannya seolah - olah karangan (pendapat dan sebagainya) sendiri, misalnya menerbitkan karya tulis orang lain atas nama dirinya sendiri. Sehingga dapat diartikan plagiarisme merupakan tindakan mencuri gagasan hasil penelitian orang lain, untuk kemudian disajikan seolah-olah milik sendiri (Ridhatillah, 2013). Tindakan plagiarisme merupakan salah satu tindakan yang melanggar hak cipta. Hak Cipta itu sendiri merupakan hak eksklusif untuk Pencipta ataupun penerima hak buat mengumumkan ataupun perbanyak Ciptaannya ataupun membagikan izin buat itu dengan tidak kurangi pembatasan- pembatasan bagi peraturan perundang- undangan yang berlaku. Jika terjadi pelanggaran tersebut, dapat dikenai pelanggaran hak cipta di Pasal 72 ayat UUHC dengan dipidana dengan pidana penjara pendek selama 1 bulan serta / ataupun denda sangat sedikitnya Rp1.000.000,00, ataupun pidana penjara lama 7 tahun serta / ataupun denda sebanyak Rp5.000.000. 000,00.

Untuk mengurangi hal tersebut, dapat dilakukan beberapa metode untuk mencegah plagiarisme seperti menggunakan cara manual ataupun menggunakan teknologi yang ada. Jika menggunakan cara manual, dilakukan dengan membandingkan sebuah dokumen yang akan dicek dengan dokumen yang akan dijadikan perbandingan dengan melihat kata yang digunakan ataupun tata penulisan dokumen tersebut. Cara ini kurang efektif karena jumlah dokumen yang akan dijadikan perbandingan jumlah tidak sedikit bisa mencapai ratusan bahkan ribuan dokumen lebih. Sedangkan untuk cepat menggunakan menggunakan mesin sebagi pembanding.

Ada beberapa metode yang dapat digunakan seperti *Cosine Similarity*, TF-IDF, *Jaccard Similarity*, *Dice Similarity*, *Word2Vec* dan sebagainya. Dengan menggunakan salah satu metode diatas, dapat temukan persentase kesamaan sebuah dokument. Salah satu metode yang dapat digunakan yaitu dengan metode *Dice Similarity*. *Dice Similarity* merupakan salah satu metode perbandingan antara dokumen dengan membandingkan kata yang ada diantara dokumen tersebut. Melihat permasalahan diatas…*(lanjut nanti)*, Dengan adanya penelitian ini, diharapkan dapat membangun sebuah Sistem Informasi berbasis Web yang mampu memberikan persentase kesamaan dari sebuah dokumen terhadap dokumen lainnya yang terdapat didalam *database* guna mengurangi tindak plagiarisme yang sering terjadi saat ini.

## Rumusan Masalah

Adapun beberapa point yang menjadi rumusan masalah penelitian ini yaitu :

1. Bagaimana merancang sebuah aplikasi berbasis web ?
2. Bagaimana mengimplementasikan metode *Dice Similarity* ?
3. Bagaimana menguji kesamaan dari sebuah dokumen dengan dokumen lainnya menggunakan metode *Dice Similarity* ?

## Tujuan Penelitian

Adapun beberapa point yang menjadi tujuan penelitian ini yaitu :

1. Merancang sebuah aplikasi berbasis web.
2. Mengimplementasikan metode *Dice Similarity*.
3. Menguji kesamaan dari sebuah dokumen dengan dokumen lainnya menggunakan metode *Dice Similarity*.

## Manfaat Penelitian

Penelitian ini merupakan pengimplementasian metode *Dice Similarity* pada sebuah aplikasi berbasis web guna mengetahui persentase kesamaan sebuah dokumen dengan dokumen lainnya sehungga dapat digunakan untuk mengurangi tindak plagiarisme dengan mudah.

## Batasan Masalah

Adapaun beberapa batasan masalah pada penelitian ini yaitu :

1. Sistem ini dibuat menggunakan ahasa pemrograman Javascript dengan runtime environment Nodejs, *framework* Express JS, dengan database MongoDB.
2. Menggunakan data berupa jurnal yang diinput oleh user dan memiliki ekstensi pdf (*Portable Document Format*).

# BAB II TINJAUAN PUSTAKA

## 2.1 Penelitian Terkait

Dalam Penelitian Mufti Ari Bianto, dkk (2018) dengan judul *Perancangan Sistem Pendeteksi Plagiarisme Terhadap Topik Penelitian Menggunakan Metode K-Means Clustering dan Model Bayesian,* hasil kemiripan diuji berdasarkan kesamaan pola kata dalam kalimat, dokumen dengan nilai *similarity* tertinggi diperoleh 1 dokumen dengan presentase kemiripan sebanyak 100 %, dan jika berdasarkan kesamaan *term*, berpengaruh terhadap hasil dokumen mirip yang dihasilkan, sehingga diperoleh 2 dokumen mirip. Meskipun uji berdasarkan kesamaan *term*, menghasilkan dokumen mirip yang lebih banyak namun belum cukup akurat menunjukkan adanya plagiasi, karena dalam menentukan plagiasi kesamaan rangkaian kalimat merupakan hal yang penting untuk diperhatikan.

Dalam penelitian Joko Priambodo (2018) dengan judul *Pendeteksian Plagiarisme Menggunakan Algoritma Rabin - Karb dengan Meotde Rolling Hash*, pendeteksian plagiarisme menggunakan algoritma *Rabin-Karp* dengan metode *rolling hash* dari hasil pengujian 30 dokumen teks yang menghasilkan tingkat akurasi yang terbesar yaitu 47.58 %. Hasil persentase tersebut termasuk dalam kategori tingkat plagiat 15 - 50 %, berarti menandakan dokumen tersebut termasuk plagiat tingkat sedang. Sedangkan tingkat akurasi yang terkecil yaitu 19.28 %, berarti menandakan dokumen tersebut termasuk plagiat tingkat sedang. Selain itu, berdasarkan analisis proses pendeteksian tingkat plagiarisme menggunakan algoritma *rabin - karp* dengan metode rolling hash bisa membaca karakter berupa huruf, simbol seperti titik (.), koma (,), dan lain - lain.

Dalam penelitian Pavel Stefanovic (2019) dengan judul *The N-Grams Based Text Similarity Detection Approach Using Self-Organizing Maps and Similarity Measures*, dengan pendekatan berdasarkan teks yang dipecah menjadi n-gram dan mengevaluasinya menggunakan *Self-Organizing Maps* (SOM) dan *Similarity Measure*. Deteksi teks serupa dilakukan dalam tiga langkah: (1) konversi kumpulan data teks ke numerik ekspresi menggunakan n-gram; (2) perhitungan ukuran kesamaan; (3) visualisasi dataset teks menggunakan SOM dan representasi kesamaan di atasnya. Pada langkah pertama, fokus utamanya adalah membuat sekantong n-gram dari semua dataset. Berbagai jumlah kata dalam n-gram dianalisis. Selain itu, filter yang berbeda diterapkan: penghapusan angka dan tanda baca, frekuensi kata, transformasi huruf besar, *stemming* algoritma, dll. Analisis menunjukkan filter dan ukuran n-gram mempengaruhi hasil akhir. Untuk ini dataset, ukuran n-gram dipilih dan sama dengan tiga untuk penyelidikan eksperimental. Pada langkah kedua, empat ukuran kesamaan dihitung: *Cosinus*, *Dice*, *Extends Jaccard*, dan *Overlap*. Hasil akhir menunjukkan bahwa persentase kemiripan tertinggi diperoleh dengan menggunakan *overlap*. Tiga nilai ukuran lainnya selalu sama dan lebih kecil. Penggunaan SOM menunjukkan bahwa SOM membantu untuk melihat hasil ringkasan kesamaan semua teks dalam bentuk visual dengan cepat. Dia sangat mudah untuk memahami teks mana yang mirip satu sama lain atau tidak. Dalam kasus kumpulan data yang dianalisis, SOM membantu mendeteksi kesamaan, dan cluster yang terbentuk dikorelasikan dengan kategoris yang diberikan deskripsi kumpulan data.

Dalam penelitian Uswatun Hasanah, dkk (2019) dengan judul *Perbandingan Metode Cosine Similarity dan Jaccard Similarity Untuk Penilaian Otomatis Jawaban Pendek*, dinilai belum mampu memberikan jawaban yang memuaskan i dikarenakan kedua metode hanya menilai kemiripan berdasarkan susunan leksikalnya. Sementara itu, jawaban mahasiswa juga sangat bervariasi dan menggunakan kata-kata yang jauh berbeda dari jawaban kunci, walaupun pada dasarnya memiliki makna semantik yang sama. Oleh karena itu, pada penelitian selanjutnya diperlukan metode lain yang mampu menangani makna semantik pada jawaban. Bagaimanapun, metode *Cosine Similarity* dan *Jaccard Similarity* masih dapat dipertimbangkan untuk menilai jawaban pendek secara otomatis, dengan batasan bahwa pertanyaan yang digunakan mengharuskan jawaban dalam format keyword sehingga tidak memunculkan kata-kata lain yang mampu menurunkan nilai kemiripan.

## 2.2 Landasan Teori

### 2.2.1 *Website*

*Website* adalah kumpulan halaman web yang saling terhubung dan seluruh file saling terkait. Web terdiri dari page atau halaman dan kumpulan halaman yang dinamakan homepage. *Homepage* berada pada posisi teratas dengan halaman-halaman terkait berada di bawahnya. Biasanya, setiap halaman di bawah *homepage* (*child page*) berisi *hyperlink* ke halaman lain dalam web (Gregorius, 2000).

Sejarah *website* dimulai oleh Tim Berners Lee, yang mengembangkan *World Wide Web* (WWW) pada tahun 1989. Pada bulan Oktober tahun 1990, Tim menggagas tiga teknologi dasar untuk membangun sebuah *website*, yaitu HTML, URL, dan HTTP. Awalnya, *website* dirancang hanya untuk berbagi informasi di kalangan para ilmuwan di CERN. Sampai akhirnya, Tim melihat potensi *website* sebagai sarana yang bisa digunakan oleh siapa saja untuk berbagai tujuan. Kemudian, ia merilisnya secara resmi pada 6 Agustus 1991.

*Website* yang pertama dirilis saat itu baru sebatas menampilkan teks sederhana tanpa variasi font, video, maupun gambar. Saat ini, webs sudah berkembang sangat pesat dan menjadikan *website* sebagai salah satu kebutuhan penting baik individu, organisasi, dan perusahaan diseluruh dunia hingga saat ini. Saat ini, website berkembang sangat pesat dengan muncul berbagai jenis *website* seperti pribadi, *e-commerce*, *blog*, dan media sosial.

### 2.2.2 *Hypertext Transfer Protocol Secure* (HTTPS)

*Hypertext Transfer Protocol Secure* (HTTPS) adalah sebuah *protocol* komunikasi dalam suatu jaringan internet dengan keamanan yang lebih terjamin. Disebut lebih aman karena suatu perintah atau data yang dikirim melalui HTTPS ini dilindungi dengan sistem enkripsi sehingga menyulitkan *hacker* untuk membobol atau mencurinya (Fakhri Aziz Firmansyah, 2019). HTTPS merupakan tingkatan dari HTTP dimana yang membedakan di HTTPS terdapat *(Secure Socket Layer*) SSL dan (*Transport Layer Security*) TSL yang digunakan untuk mengamankan data yang disimpan atau yang akan dikirim. Menurut SSL Labs pada April 2018, 33,2% dari 1.000.000 situs web teratas Alexa menggunakan HTTPS sebagai *default*, 57,1% dari 137.971 situs web paling populer di Internet memiliki implementasi HTTPS yang aman, dan 70% dari pemuatan halaman (diukur oleh *Firefox Telemetry*) menggunakan HTTPS. Jadi saat ini website sekarang sudah mulai beralih dari sebelumnya HTTP ke HTTPS karena di HTTPS lebih aman daripada HTTP yang sudah lama.

### 2.2.3 Javascript

Javascript adalah sebuah bahasa pemrograman tingkat tinggi yang dinamis, *scripting*, *untyped*, dan *interpreter*. Javascript sendiri dibuat oleh Brendan Eich dari perusahaan Nestcape pada tahun 1994 yang diberi nama Mocha pada saat itu, kemudian berganti menjadi Livescript. Karena saat itu browser yang populer adalah Nestcape, Microsoft berusaha untuk mengalahkan popularitas browser tersebut dengan Internet Explorer dan melakukan *Reverse Enginering* terhadap Livescript dan terciptalah JScript pada tahun 1996. Karena terdapat dua browser yang besar yang berbeda, maka dibuatlah satu standar agar mempermudah pembuatan website saat itu dan dibuatlah ECMAScript. Menurut Douglas Rockford, "*JavaScript, JScript, and ECMAScript 3 Silly Name for 1 Silly Language*" yang berarti bahwa ketiga nama tersebut adalah bahasa yang sama yaitu Javascript.

#### 2.2.3.1 Karakteristik Javascript

Untuk Ekstensi dari file Javascript menggunakan ekstensi \*.js. Contoh namaFile.js. Setiap akhir dari kode javascript dapat menggunakan “;” ataupun tidak. Sedangkan *block scope* di Javascript menggunakan { }.

Javascript merupakan salah satu bahasa pemrogram yang *untyped* / *dynamicly* *typed* yang berarti tidak mendefiniskan terlebih dahulu tipe variabel yang akan didefinisikan. Untuk penamaan variabel, terdapat beberapa *keyword* yaitu var, let, dan const. Untuk var, dan let, nilai variabel dapat berubah atau diisi ulang, sedangkan const nilai variabelnya tidak dapat diubah. Perbedaan var dan let sendiri terletak pada hoistingnya. Sedangkan untuk penamaan variabel, hampir sama dengan beebrapa bahasa pemrograman lainnya seperti karakter pertama variabel tidak boleh angka, menggunakan penulisan *camelCase*, dan untuk *keyword* const yang merupakan variabel const biasanya menggunakan *snake\_case* dan semuanya huruf kapital. Contoh :

var value = 10 // Contoh Penggunaan var

let nilai = 'Delapan' // Contoh Penggunaan let

const SPEED\_OF\_LIGHT = 3.14 // Contoh Penggunaan const

Javascript memiliki beberapa tipe data seperti *String*, *Number* atau *Integer*, *Boolean*, *Array*, *Object*, dan *Undifined*. Untuk mengetahui tipe data dari sebuah variabel dapat menggunakan *keyword typeof*. Contoh :

let name = 'Aris Akhyar Abdillah' // String

let age = 22 // Number atau Integer

let male = true // Boolean

let family = ['Father', 'Mother', 'Son', 'Girl'] // Array

let cv = {

    fullName : 'Aris Akhyar Addillah',

    nim : 'H071171505'

} // Object

let girlfreind = undefined // Undefined

*Synchronous* dan *Asynchronous*, Secara sederhana, *Syncronus* dan *Asynchronous* merupakan tahapan dalam mengeksekusi sebuah kode dimana Synchronous mengeksekusi sebuah kode perbaris sesuai urutan kode yang dituliskan. Sedangkan *Asynchronous*, tidak selalu seperti *Synchronous*, tapi melihat waktu proses dari kode tersebut. Penggunaan *Asynchronous* tidak akan menunggu suatu kode selesai dijalankan, tetapi berlanjut ke kode selanjutnya. Contoh :

// Menggunakan Synchronous

console.log('What Is Your Name ?')

console.log('Hello World')

console.log('My Name is Aris')

// Output :

// What Is Your Name ?

// Hello World

// My Name is Aris

// Menggunakan Asynchronous

setTimeout(function() {

  console.log('What Is Your Name ?')

}, 1500)

console.log('Hello World')

setTimeout(function() {

    console.log('My Name is Aris')

}, 1000)

// Output :

// Hello World

// My Name is Aris

// What Is Your Name ?

Jika melihat output dari dua program diatas antara menggunakan *Synchronous* dan *Asynchronous* terdapat perbedaan urutan dari hasil *output*, dimana *Synchronous* memiliki *output* sesuai dari urutan baris kode sedangkan *Asynchronous* berbeda dimana *output* yang dihasilkan berdasarkan jumlah waktu pengeksekusian dari kode tersebut yang disimulasikan menggunakan setTimeout() dari pengeksekusian paling cepat hingga yang paling lambat.

#### 2.2.3.2 NodeJS

Node.js adalah *runtime environment* untuk JavaScript yang bersifat *open-source* dan *cross-platform*. Dengan Node.js kita dapat menjalankan kode JavaScript dimana pun, tidak hanya terbatas pada lingkungan *browser*. Seperti diketahui bahwa Javascript hanya dapat berjalan pada sebuah *web browser*, kemudian Ryan Dahl, membuat sebuah *runtime environment* dengan mengeluarkan *engine* Javasscript dari Chrome yaitu *V8 Javascript Engine* menggunakan bahaasa C agar Javascript dapat dijalankan diluar *browser*. Akhirnya tercipta NodeJS pada tahun 2009. Dengan Begitu, Javascript yang sebelumnya hanya bisa di *client side* dengan adanya NodeJS bisa juga di *server side*. Beberapa fitur yang terdapat di NodeJS seperti *Asynchronous & Event-driven, Single Threaded but Highly Scalable*.

#### 2.2.3.3 ExpressJS

Menurut Website resmi dari Express JS adalah “*Fast, unopinionated, minimalist web framework for Nodejs*”. Express JS sendiri merupakan salah satu *web framework* khusus untuk NodeJS dimana kita dapat membuat sebuah *website* yang cepat, sederhana, dengan struktur yang tidak ditentukan atau tergantung dari pengguna ExpressJS sendiri.

### 2.2.4 *Database*

*Database* atau basis data adalah kumpulan informasi yang disimpan di dalam komputer secara sistematik sehingga dapat diperiksa menggunakan suatu program komputer untuk memperoleh informasi dari basis data tersebut (Andry Andaru, 2018). Berdasarkan definisi *database*, *database* berfungsi untuk menyimpan catatan atau sebuah data pada sebuah penyimpanan yang kemudian akan digunakan pada waktu lainnya. Pada umumnya, database sudah pasti memiliki *key* dan *value*, walaupun istilah ini berbeda ditiap - tiap jenis database yang ada. Adapun beberapa jenis database seperti *Operational Database*, *Database Warehouse*, *Distributed Database*, *Relational Database*, dan *End User Database*.

#### 2.2.4.1 SQL dan NoSQL

SQL dan NoSQL merupakan salah satu contoh dari *Relational Database* yang populer saat ini. SQL (*Structured Query Language*) merupakan bahasa yang digunakan untuk mengelola data secara *relational*. SQL sendiri memiliki ciri yaitu memiliki tabel yang terdiri dengan *row* atau *record* dan *field* yang bisa memiliki relasi dengan tabel lainnya. Untuk tiap data yang ada didalam tabel, harus memiliki skema yang sama untuk tiap recordnya. Contoh dari SQL seperti MySQL, PostgreSQL, dan MariaDB. Berbeda dengan NoSQL (*Not Only Structured Query Language*) yang merupakan bahasa untuk mengelola data secara *Non Relational*. Berbeda dengan SQL yang menggunakan tabel untuk menyimpan data. NoSQL memilik banyak jenis tempat untuk menyimpan data seperti *Document Database*, *Key-Value Database*, dan *Graph Database*. NoSQL juga tidak memiliki skema sehingga untuk menyimpan data bisa secara flexibel. Contoh dari NoSQL seperti MongoDB, Redis, Neo4j, dan Cassandra.

#### 2.2.4.2 MongoDB

MongoDB merupakan salah satu contoh dari NoSQL yang dikembangkan menggunakan bahasa pemrograman C++ yang rilis pertama kali pada tanggal 11 Februari 2009. MongoDB adalah salah satu contoh *Document Database* yang dimana tiap - tiap datanya merupakan sebuah JSON (*Javascript Object Notation*) atau dalam MongoDB disebut BSON (*Binary JSON*). Hingga saat ini, MongoDB sudah digunakan lebih dari 85 Juta pengguna diseluruh dunia dan sudah banyak perusahaan besar yang menggunakan database ini seperti EBay, Google, Adobe, dan EA. Salah satu contoh data yang terdapat di MongoDB :

{

  \_id : '91829jw1jooojo2',

  firstName : 'Aris',

  lastName : 'Akhyar',

  age : 20

  hobbies : ['eat', 'drink']

}

### 2.2.5 Jurnal

Jurnal merupakan bagian dari jenis terbitan berseri yang ada diperpustakaan, adapun pengertian jurnal menurut *High Beam* “*Journal is the collection and periodic publication or transmission of news and the result of research through media*”, artinya bahwa jurnal merupakan suatu koleksi dan terbitan berkala atau transmisi mengenai berita dan hasil-hasil penelitian mengenai media. Jurnal sendiri terbagi atas dua format yaitu tercetak dan digital (*e-journal*). Untuk format digital jurnal dikemas dalam dua format , yaitu bentuk CD-ROM dan dalam bentuk akses secara online melalui internet. *E-Journal* dipahami sebagai publikasi ilmiah dalam format elektronik dan mempunyai ISSN (International Standard Serial Number) yang format dokumennya biasanya pdf (Rusydi 2014).

Penggunaan kata jurnal untuk berbagai bidang juga memberi arti yang bervariasi, misalnya jurnal dalam bidang ekonomi menunjukan sistem pembukuan rangkap. Jurnal dalam bidang pelayaran diartikan sebagai logbook berarti buku untuk mencatat semua kejadian selama pelayaran. Jurnal sebenarnya merupakan publikasi ilmiah yang memuat informasi tentang hasil kegiatan dalam bidang ilmu pengetahuan dan teknologi minimal harus mencakup kumpulan atau kumulasi pengetahuan baru, pengamatan empiris dan pengembangan gagasan atau usulan. Dengan demikian jurnal merupakan representasi dari pengetahuan baru tentang perkembangan ilmu pengetahuan yang dilaksanakan secara empriris dan biasanya merupakan gagasan yang terbaru.

### 2.2.6 Plagiarisme

Plagiarisme berasal dari bahasa Latin “*plagiare*” yang berarti mencuri. Plagiarisme berasal dari kata plagiat yang berarti pengambilan karangan (pendapat dan sebagainya) orang lain dan menjadikannya seolah - olah karangan (pendapat dan sebagainya) sendiri, misalnya menerbitkan karya tulis orang lain atas nama dirinya sendiri. Sehingga dapat diartikan plagiarisme merupakan tindakan mencuri gagasan hasil penelitian orang lain, untuk kemudian disajikan seolah-olah milik sendiri (Ridhatillah, 2013). Tindakan plagiarisme merupakan salah satu tindakan yang melanggar hak cipta. Hak Cipta itu sendiri merupakan hak eksklusif untuk Pencipta ataupun penerima hak buat mengumumkan ataupun perbanyak Ciptaannya ataupun membagikan izin buat itu dengan tidak kurangi pembatasan- pembatasan bagi peraturan perundang- undangan yang berlaku. Jika terjadi pelanggaran tersebut, dapat dikenai pelanggaran hak cipta di Pasal 72 ayat UUHC dengan dipidana dengan pidana penjara pendek selama 1 bulan serta / ataupun denda sangat sedikitnya Rp1.000.000,00, ataupun pidana penjara lama 7 tahun serta / ataupun denda sebanyak Rp5.000.000.000,00.

### 2.2.7 *Dice Similarity*

*Dice Similarity* atau *Sørensen–Dice coefficient* merupakan salah satu algoritma yang mengukur kesamaan antara dua set data. Algoritma ini banyak digunakan dalam validasi algoritma segmentasi gambar yang dibuat dengan *Artificial Intelligence* atau AI, tetapi ini adalah konsep yang jauh lebih umum yang dapat diterapkan pada kumpulan data untuk berbagai aplikasi termasuk *Natural Language Processing* (NLP). Untuk penggunaan rumus dari *Dice Similarity*, ada beberapa penggunakan rumus tergantung datanya, jika terdapat dua set data diskrit, dengan X dan Y adalah data yang ingin dibandingkan, maka menggunakan persamaan :

|  |  |  |
| --- | --- | --- |
|  |  | (2.1) |

Jika menggunakan data *boolean*, maka menggunakan definisi *True Positive* (TP), *False Positive* (FP), and *False Negative* (FN) dengan rumus yaitu :

|  |  |  |
| --- | --- | --- |
|  |  | (2.2) |

Jika menggunakan data *vector*, dengan A dan B sebagai *vector*, maka rumusnya yaitu :

|  |  |  |
| --- | --- | --- |
|  |  | (2.3) |

#### 2.2.7.1 Implementasi Dice Similarity

Untuk pengimplementasian dari metode *Dice Similarity*, diambil contoh dari definisi dari komputer menurut para ahli yaitu Menurut Robert H. Blissmer, “*komputer adalah suatu alat elektronik yang mampu melakukan beberapa tugas seperti menerima input, memproses input tadi sesuai dengan programnya, menyimpan perintah - perintah dan hasil pengolahan, serta menyediakan output dalam bentuk informasi”.* Sedangkan menurut Arief Susanto, “*komputer adalah sekelompok alat elektronik yang terdiri atas perintah input, alat yang mengolah input, dan peralatan output yang memberikan informasi serta bekerja secara otomatis”*.

Setelah mendapatkan kedua data diatas, kemudian dilakukan pembagian. Untuk pengertian dari Robert H. Blissmer akan menjadi data dari database sebagai X, kemudian untuk pengertian dari Arief Susanto akan menjadi data yang akan dibandingkan dengan data dari database sebagai Y. Sebelum diolah, kedua data diatas akan dilakukan penghapusan *Stopword* dan karakter yang tidak berguna dalam perhitungan nantinya sehingga menjadi :

|  |  |
| --- | --- |
| **Data X** | **Data Y** |
| alat elektronik tugas menerima input memproses input sesuai programnya menyimpan perintah - perintah hasil pengolahan menyediakan output bentuk informasi | sekelompok alat elektronik perintah input alat mengolah input peralatan output informasi otomatis |

Tabel 2.1 Perbandingan Data X dan Data Y

Setelah dilakukan penghapusan *Stopword* dan karaktek yang tidak berguna, kemudian dilakukan pembuatan kamus kata lalu dilakukan perhutngan jumlah kata terhadap kamus data yang ada sehingga menjadi :

| **Kamus Data** | **Data X** | **Data Y** |
| --- | --- | --- |
| alat | 2 | 1 |
| bentuk | 0 | 1 |
| elektronik | 1 | 1 |
| hasil | 0 | 1 |
| infromasi | 1 | 1 |
| input | 2 | 2 |
| memproses | 0 | 1 |
| menerima | 0 | 1 |
| mengolah | 1 | 0 |
| menyediakan | 0 | 1 |
| menyimpan | 0 | 1 |
| otomatis | 1 | 0 |
| output | 1 | 1 |
| pengolahan | 0 | 1 |
| perintah | 1 | 2 |
| perlatan | 1 | 0 |
| programnya | 0 | 1 |
| sekelompok | 1 | 0 |
| sesuai | 0 | 1 |
| tugas | 0 | 1 |

Tabel 2.2 Kamus Data

Setelah itu, data diatas akan diubah menjadi vektor satu dimensi sehingga menjadi :

|  |  |  |
| --- | --- | --- |
|  |  | (2.4) |
|  |  | (2.5) |

Kemudian kedua vektor diatas dimasukkan kedalam rumus *Dice Similarity* pada persamaan 2.3. Untuk mempermudah perhitungan, dilakukan beberapa penambahan *row* dan *field* sehingga menjadi :

| **Kamus Data** | **Data X** | **Data Y** | **X . Y** | **X2** | **Y2** |
| --- | --- | --- | --- | --- | --- |
| alat | 2 | 1 | 2 | 4 | 1 |
| bentuk | 0 | 1 | 0 | 0 | 1 |
| elektronik | 1 | 1 | 1 | 1 | 1 |
| hasil | 0 | 1 | 0 | 0 | 1 |
| infromasi | 1 | 1 | 1 | 1 | 1 |
| input | 2 | 2 | 4 | 4 | 4 |
| memproses | 0 | 1 | 0 | 0 | 1 |
| menerima | 0 | 1 | 0 | 0 | 1 |
| mengolah | 1 | 0 | 0 | 1 | 0 |
| menyediakan | 0 | 1 | 0 | 0 | 1 |
| menyimpan | 0 | 1 | 0 | 0 | 1 |
| otomatis | 1 | 0 | 0 | 1 | 0 |
| output | 1 | 1 | 1 | 1 | 1 |
| pengolahan | 0 | 1 | 0 | 0 | 1 |
| perintah | 1 | 2 | 2 | 1 | 4 |
| perlatan | 1 | 0 | 0 | 1 | 0 |
| programnya | 0 | 1 | 0 | 0 | 1 |
| sekelompok | 1 | 0 | 0 | 1 | 0 |
| sesuai | 0 | 1 | 0 | 0 | 1 |
| tugas | 0 | 1 | 0 | 0 | 1 |
| **Total** | **12** | **18** | **11** | **16** | **22** |

Tabel 2.3 Perhtiungan di Dalam Kamus Data

Setelah memperoleh hasil seperti di table 2.3, kemudian nilai tersebut dimasukkan kedalam rumus *Dice Similarity* sehingga menjadi :

|  |  |  |
| --- | --- | --- |
|  |  | (2.6) |
|  |  | (2.7) |
|  |  | (2.8) |
|  |  | (2.9) |

Sehingga diperoleh nilai *Dice Similarity* adalah 0,5789473684 atau jika dipersentasekan menjadi 57.8 %. Jadi, Persentase kesamaan antara antara Data X dengan Data Y adalah sebesar 57.8 %

# BAB III METODE PENELITIAN

## 3.1 Prosedur Penelitian

Prosedur Penelitian merupakan serangkaian kegiatan yang dilaksanakan secara sistematis serta teratur agar mencapai tujuan dari penelitian ini. Adapun prosedur penelitian sebagai berikut :

1. Analisa Kebutuhan Sistem

Dalam tahap ini, dilakukan analisa terhadap kebutuhan sebuah sistem seperti bagaimana tampilan sistem, bagaimana cara kerja sistem, ataupun penggunaan bahasa pemrograman yang tepat.

1. Studi Literatur

Dalam tahap ini, dilakukan pengumpulan teori - teori ataupun landasan dasar dalam proses membangun sebuah sistem yang akan dibuat.

1. Perancangan Sistem

Dalam tahap ini, dilakukan pembangunan Sistem Pendeteksi Kesamaan Teks Menggunakan Metode *Dice Similarity* dengan menggunakan metode, teori, ataupun bahasa pemrograman yang telah ditentukan sebelumnya.

1. Pengujian Sistem

Dalam tahap ini, dilakukan testing atau pengujian terhadap sistem.

## 3.2 Rancangan Sistem

### 3.2.1 *Use Case Diagram*

*Use Case Diagram* adalah suatu jenis diagram yang menggambarkan hubungan atau interaksi dari sebuah sistem dengan pengguna. Dengan adanya *Use Case Diagram*, kita dapat mendeskripsikan tipe atau jenis yang dilakukan pengguna terhadap sebuah sistem. Berikut use case diagram dari Sistem Pendeteksi Kesamaan Teks Menggunakan Metode *Dice Similarity*.

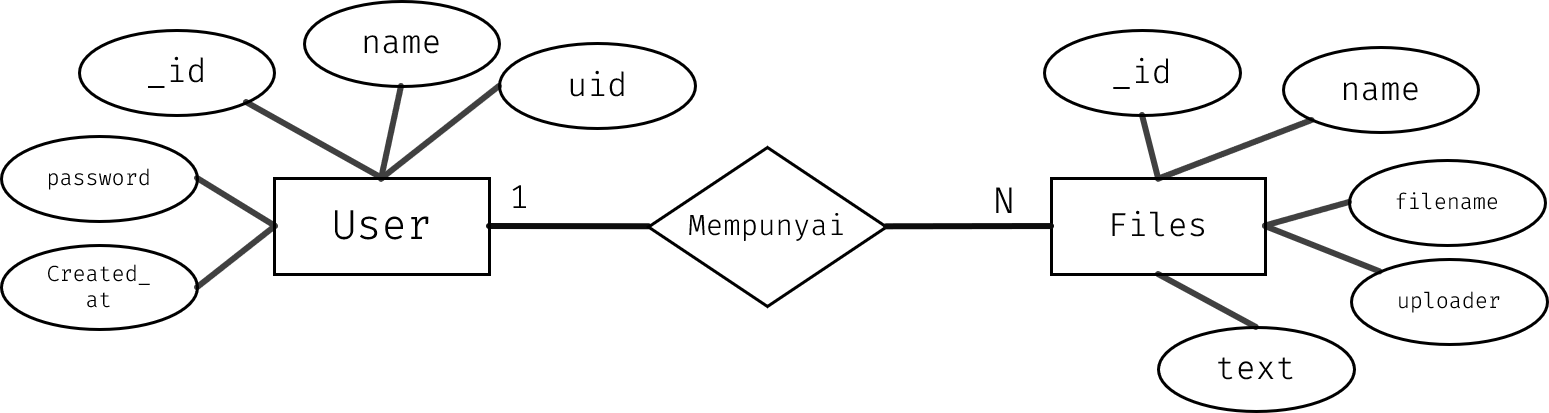


Gambar 3.1 *Use Case Diagram*

Dalam *use case diagram* pada gambar 3.1, terdapat 3 jenis pengakses yang akan menggunakan sistem ini yaitu Admin, User, dan *Guest*. Admin memiliki akses terhadap semua sistem baik manajemen dari user, dan juga manajemen data yang disimpan dalam *database*. Untuk user, hanya memiliki akses terhadap data yang disimpan kedalam *database* nantinya berdasarkan yang dimiliki oleh user. Selain itu, user juga dapat menggunakan sistem pendeteksi kesamaan dokumen yang ada dalam sistem ini. Sedangkan *guest* atau tamu hanya bisa meakukan pendeteksi kesamaan dokumen dan tidak bisa melakukan manajemen data yang ada didalam *database*.

### 3.2.2 *Entity Relationship Diagram* (ERD)

*Entity Relationship Diagram* (ERD) adalah suatu jenis diagram yang digunakan untuk merancang suatu basis data (*database*). Dalam ERD, terdapat beberapa komponen seperti entitas, relasi, atribut, dan garis penghubung dimana digunakan untuk memperlihatkan hubungan atau relasi antar entitas atau objek yang terlihat beserta atributnya. Berikut ERD dari Sistem Pendeteksi Kesamaan Teks Menggunakan Metode Dice Similarity

.

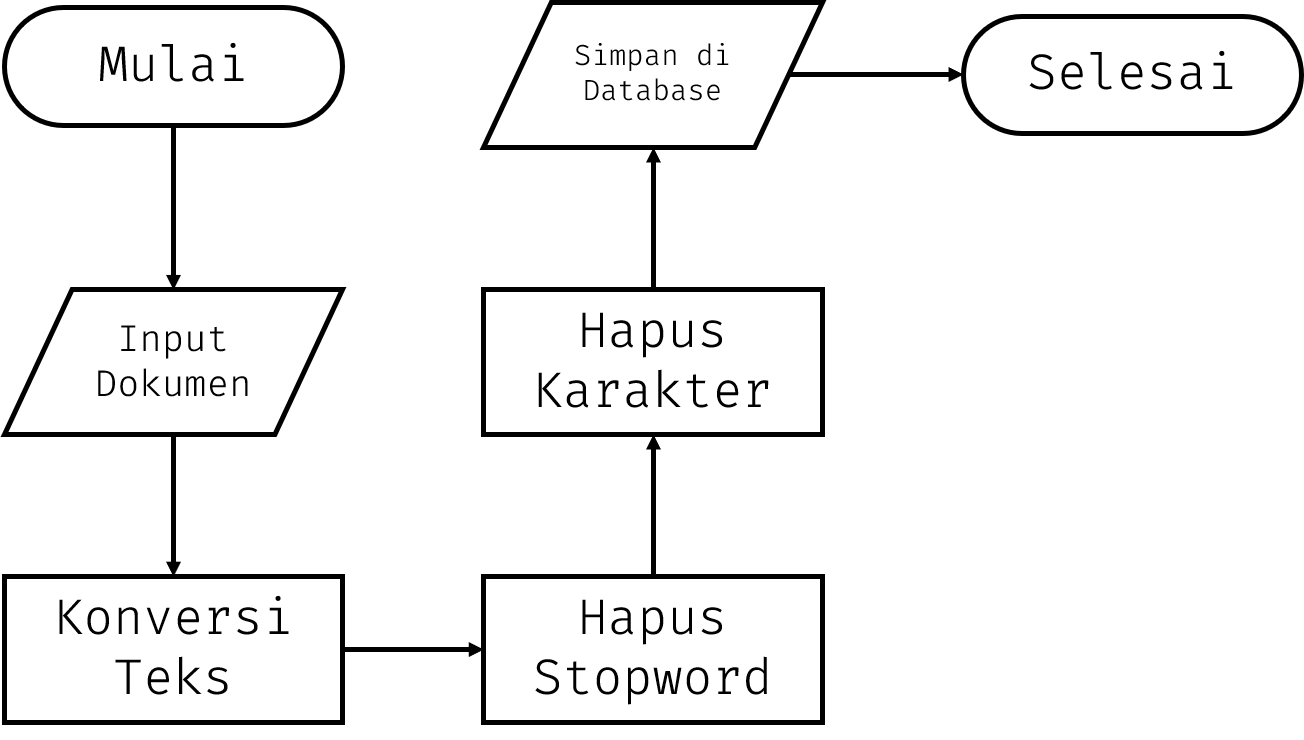
Gambar 3.2 *Entity Relationship Diagram* (ERD)

Berdasarkan ERD pada gambar 3.2, terdapat dua entitas yaitu User dan Files, dimana User memiliki beberapa atribut yaitu \_id, *name*, *password*, *uid*, *password*, dan *created\_at*. Sedangkan Files memiliki beberapa atribut juga sepeerti *\_id*, *name*, *filename*, *uploader*, dan *text*. Tabel User mempunya relasi *One to Many* terhadap Tabel Files, sedangakan sebaliknya yaitu Tabel *Files* memiliki relasi *One to One* terhadap Tabel User.

### 3.2.3 Desain Sistem

Pada bagian ini, akan menjelaskan bagaimana desain suatu sistem yang diimplemetasikan. Dengan adanya desain sistem ini, dapat memberikan gambaran rancang bangun yang lengkap terhadap pengguna dari sistem ini.

#### 3.2.3.1 Algoritma Input Dokumen ke Dalam *Database*



Gambar 3.3 *Flowchart* Input Dokumen ke Dalam Database

Keterangan :

1. Input Dokumen

Sebelum membandingkan teks yang akan ditentukan kesamaannya, terlebih dahulu dibuat sebuah repositori atau database teks yang akan digunakan sebagai pembanding nantinya. Teks yang diinput pada penelitian ini berupa file yang berformat pdf (Portable Document Format).

1. Konversi Teks

Teks yang telah diinput, kemudian diubah kedalam bentuk tipe data *String* agar dapat diolah nantinya didalam sistem.

1. Hapus *Stopword*

Stopword merupakan kata yang diabaikan karena memeiliki frekuensi kemunculan yang sangat tinggi dan tidak mempunyai arti. Beberapa contoh Stopw word seperti, "atau", "dan", dan "tapi". Dengan menghapus Stopword, sistem dapat bekerja lebih cepat karena *Stopword* akan menghapus kata yang abaikan.

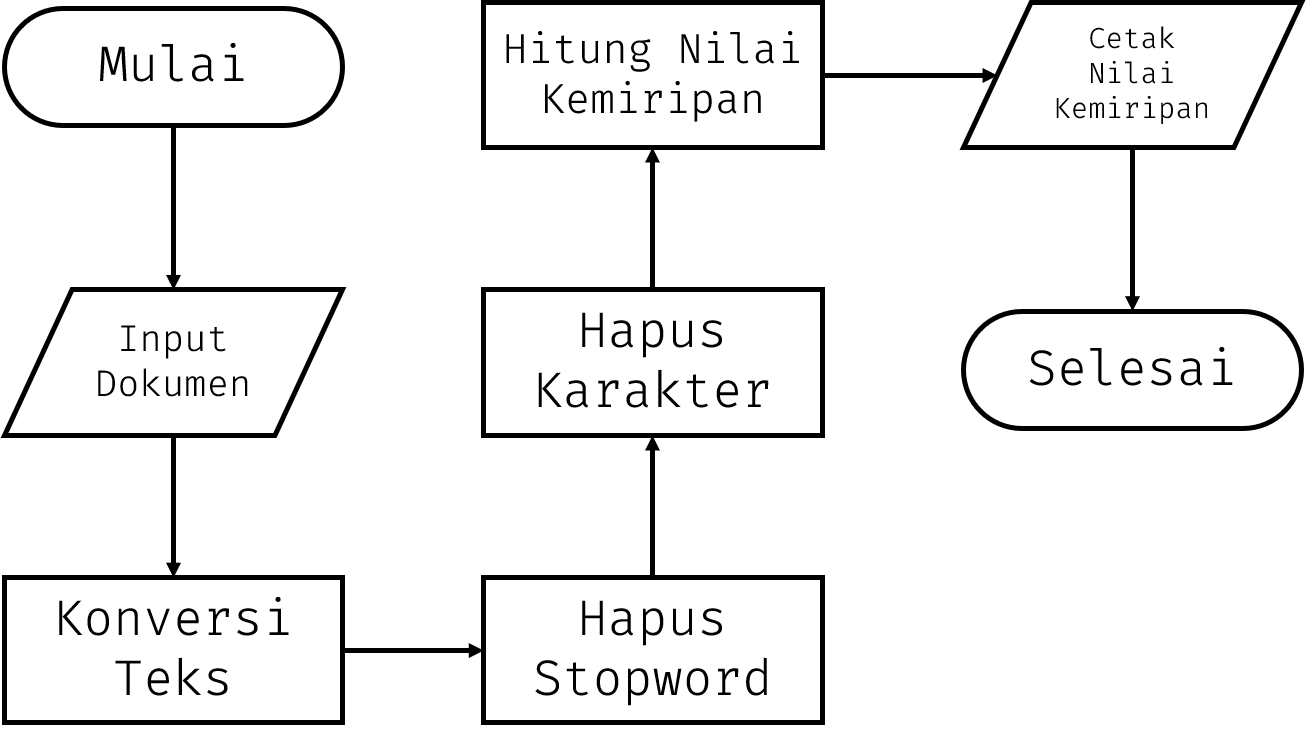
1. Hapus Karakter

Selain *Stopword*, karakter seperti ",", ".", "?", dll yang hanya berupa karakter yang tidak memiliki arti dan fungsi karena sistem hanya memeriksa kata.

1. Simpan di Database

Setelah teks telah melewati tahap sebelumnya, maka teks akan dimasukkan kedalam database yang nantinya akan digunakan saat ingin mendapatkan persentase kemiripan dari teks yang akan dibandingkan nantinya.

#### 3.2.3.2 Algoritma Perbandingan Dokumen



Gambar 3.4 *Flowchart* Perbandingan Dokumen

Keterangan :

1. Input Dokumen

Input teks yang akan dibandingkan kedalam sistem yang nantinya akan dibandingkan terhadap teks yang terdapat dalam database atau repositori. Teks yang diinput berupa teks yang berformat pdf (Portable Document Format).

1. Konversi Teks

Teks yang telah diinput, kemudian diubah kedalam bentuk tipe data *String* agar dapat diolah nantinya didalam sistem.

1. Hapus *Stopword*

*Stopword* merupakan kata yang diabaikan karena memeiliki frekuensi kemunculan yang sangat tinggi dan tidak mempunyai arti. Beberapa contoh Stopw word seperti, "atau", "dan", dan "tapi". Dengan menghapus Stopword, sistem dapat bekerja lebih cepat karena Stopword akan menghapus kata yang abaikan.

1. Hapus Karakter

Selain *Stopword*, karakter seperti ",", ".", "?", dll yang hanya berupa karakter yang tidak memiliki arti dan fungsi karena sistem hanya memeriksa kata.

1. Hitung Nilai Kemiripan

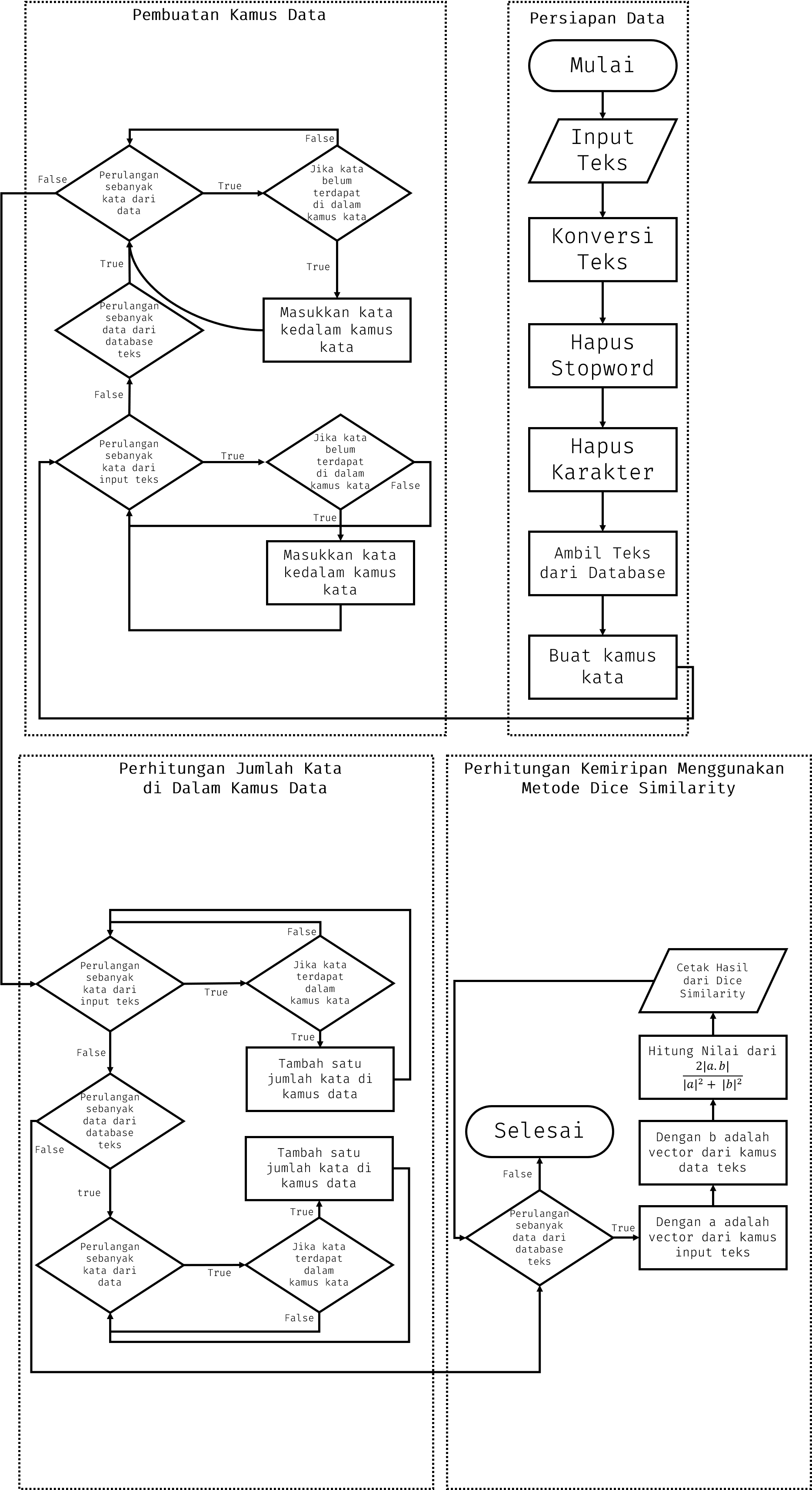
Setelah teks melewati tahap sebelumnya, maka teks akan dibandingan dengan semua teks yang terdapat didalam database atau repositori. Teks akan dibandingkan dengan semua data teks dengan menggunakan metode *Dice Similarity*. Kemudian hasil kemiripan antar teks akan muncul sebanyak data yang terdapat dalam *database* atau repositori teks.

## 3.3 Metode Penelitian

Dalam sistem ini, metode perhitungan yang digunakan adalam metode *Dice Similarity* dengan membandingkan data teks yang diinput pengguna dengan teks yang terdapat didalam database atau repositori teks.

### 3.3.1 Algoritma *Dice Similarity*

Algoritma perhitungan persentase kemiripan antar teks sebagai berikut :



Gambar 3.5 *Flowchart* Implementasi *Dice Similarity* ke Dalam Sistem

1. Persiapan Data.

Sebelum dilakukan perhitungan, data dipsiapkan terlebih dahulu, dimana data input teks merupakan data yang akan dibandingkan dengan data teks yang terdapat didalam *database* atau repositori.

1. Pembuatan Kamus Data.

Untuk mempermudah perhitungan *Dice Similarity*, dilakukan pembuatan kamus data yang berisi kata yang terdapat didalam input teks maupun data teks yang berasal dari *database* atau repositori. Pada awal tahap ini, dilakukan perulangan sebanyak kata yang terdapat didalam input teks kemudian dilakukan penngkondisian jika didalam kamus kata tidak terdapat index yang sama dengan kata, maka kata tersebut ditambahkan kedalam kamus. Setelah itu, dilakukan perulangan terhadap data dalam database atau repositori kemudian dilakukan perulangan kembali sebanyak kata yang terdapat didalam teks dan dilakukan hal yang sama dengan input teks diawal.

1. Perhitungan Jumlah Kata di Dalam Kamus Data.

Sebelum dilakukan perhitungan, terlebih dahulu ditentukan jumlah kata yang terdapat didalam teks untuk data teks input dan data teks dari *database*. Pada awal tahap ini, dilakukan perulangan sebanyak kata yang terdapat didalam input teks kemudian dilakukan penngkondisian jika didalam kamus kata terdapat kata yang sama dengan kata yang terdapat di kamus kata, maka kata untuk indek input teks akan bertambah satu. dilakukan perulangan terhadap data dalam *database* atau repositori kemudian dilakukan perulangan kembali sebanyak kata yang terdapat didalam teks dan dilakukan hal yang sama dengan input teks diawal.

1. Perhitungan Kemiripan Menggunakan Metode *Dice Similarity*.

Setelah memperoleh kamus data, maka akan dilakukan perhitungan untuk memberikan persentase kemiripan antara input teks dan data teks yang berasal dari *database*. Ditahap ini dilakukan perulangan sebanyak data dalam *database*. Kemudian dilakukan perhitungan sesuai dengan rumus dari *Dice Similarity* kemudian hasilnya akan ditampilkan.

## 3.4 Instrumen Penelitian

Instrumen yang digunakan dalam penelitian ini adalah terbagi dua yaitu *hardware* dan *software*. Untuk *hardware,* menggunakan laptop dengan processor AMD Athlon 300U with Radeon Vega Mobile Gfx 2.40 GHz dan RAM sebanyak 8 GB serta sistem operasi windows 10 Home 64 bit. Sedangkan software yang digunakan yaitu *Visual Studio Code* dan *Microsoft Edge.*

# DAFTAR PUSTAKA

Andaru, A. (2018). Pengertian Database Secara Umum. *Fakultas Komputer Umitra*.

Anistyasari, Y., & Hariadi, E. (2019). Algoritma Baru Pembentukan Kata Dasar Pada Proses Stemming Bahasa Indonesia. *Prosiding SNRT (Seminar Nasional Riset Terapan) Politeknik Negeri Banjarmasin*.

Anonymous. (2022, July 15). *Express JS*. Retrieved from Express - Node.js web application framework: https://expressjs.com/

Anonymous. (2022, July 15). *NodeJS*. Retrieved from https://nodejs.dev/

Anonymous. (2022, July 15). *What Is MongoDB ?* Retrieved from MongoDb: https://www.mongodb.com/what-is-mongodb

Arista, R. F., & Listyani, R. H. (2015). Plagiarisme di Kalangan Mahasiswa. *Paradigma.Volume 03 Nomor 02*.

Bianto, M. A., Rahayu, S., Huda, M., & Kusrini. (2018). Perancangan SIstem Pendeteksi Plagiarisme Terhadap Topik Penelitian Mengganakan Metode K-Means Clustering dan Model Bayesian. *Seminar Nasional Teknologi Informasi dan Multimedia*.

Dhamayanti, & Sari, L. P. (2019). Aplikasi Pendeteksi Plagiasi pada Universitas Indo Global Mandiri Berbasis Web. *JURNAL ILMIAH INFORMATIKA GLOBAL VOLUME 10 No. 02*.

Hasanah, U., & Mutiara, D. A. (2019). Perbandingan Metode Cosine Similarity dan Jaccard Similarity Untuk Penilaian Otomatis Jawaban Pendek. *Seminar Nasional Sistem Informasi dan Teknik Informatika SENSITif*.

Pratama, R. P. (2018). Aplikasi Pendeteksi Plagiarisme Menggunakan Cosine Similarity.

Pratama, R. P., Faisal, M., & Hanani, A. (2019). Deteksi Plagiarisme Pada Artikel Jurnal Menggunakan Metode Cosine Similarity . *SMARTICS Journal, Vol.5 No. 1* .

Priambodo, J. (2018). Pendeteksian Plagiarisme Menggunakan Algoritma Rabin - Karp . *JURNAL INFORMATIKA UNIVERSITAS PAMULANG Vol. 3, No. 1*.

S.Kom, E. P. (2015, September 22). *12 Pengertian Komputer Menurut Para Ahli*. Retrieved from DosenIT.com: https://dosenit.com/ilmu-komputer/komputer-dasar/pengertian-komputer-menurut-para-ahli

Sasongko, J., & Diartono, D. A. (2009). Rancang Bangun Sistem Informasi Manajemen Surat. *Jurnal Teknologi Informasi Dinamik*.

Stefanovi, P., Kurasova, O., & Štrimaitis, R. (2019). The N-Grams Based Text Similarity Detection Approach Using Self-Organizing Maps and Similarity Measures. *Applied Science*.

Sunardi, Yudhana, A., & Mukaromah, I. A. (2018). Implementasi Deteksi Plagiarisme Menggunakan Metode N-Gram dan Jaccard Similarity Terhadap Algoritma Winnowing. *TRANSMISI, 20, (3)*.

Susanto, A. (2009). Pengenalan Komputer. In A. Susanto, *Pengenalan Komputer* (p. 2). Jakarta: IlmuKomputer.com.

Wibowo, A. (2012). Mencegah dan Menaggulangi Plagiarisme di Dunia Pendidikan. *Jurnal Kesehatan Masyarakat Nasional Vol. 6, No. 5*.