

**PERBANDINGAN ALGORITMA SEQUENTIAL SEARCH
DAN ALGORITMA BINARY SEARCH PADA APLIKASI
KAMUS BAHASA INDONESIA MENGGUNAKAN
PHP DAN JQUERY**

SKRIPSI

Karya Tulis sebagai syarat memperoleh
Gelar Sarjana Komputer dari Fakultas Teknologi Informasi
Universitas Bale Bandung

Disusun oleh:

MOCH ILHAM BAHARI

NPM. C1A160028



**PROGRAM STRATA 1
PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI
UNIVERSITAS BALE BANDUNG
BANDUNG**

2020

LEMBAR PERSETUJUAN PEMBIMBING

**PERBANDINGAN ALGORITMA SEQUENTIAL SEARCH DAN
ALGORITMA BINARY SEARCH PADA APLIKASI KAMUS BAHASA
INDONESIA MENGGUNAKAN PHP DAN JQUERY**

Disusun oleh :

MOCH ILHAM BAHARI

NPM. C1A160028

Telah diterima dan disetujui untuk memenuhi persyaratan mencapai gelar

SARJANA KOMPUTER

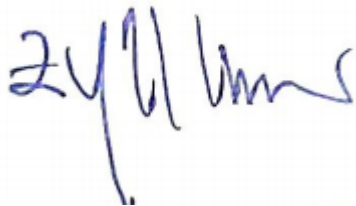
Pada

**PROGRAM STUDI INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI
UNIVERSITAS BALE BANDUNG**

Baleendah, Juli 2020

Disetujui oleh:

Pembimbing 1



Zen Munawar, S.Kom., M.Kom.

NIDN. 0422037002

Pembimbing 2



Nurul Imamah, S.T., M.T.

NIDN. 0412027905

LEMBAR PERSETUJUAN PENGUJI

PERBANDINGAN ALGORITMA SEQUENTIAL SEARCH DAN ALGORITMA BINARY SEARCH PADA APLIKASI KAMUS BAHASA INDONESIA MENGGUNAKAN PHP DAN JQUERY

Disusun oleh :

MOCH ILHAM BAHARI

NPM. C1A160028

Telah diterima dan disetujui untuk memenuhi persyaratan mencapai gelar

SARJANA KOMPUTER

Pada

**PROGRAM STUDI INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI
UNIVERSITAS BALE BANDUNG**

Baleendah, Juli 2020

Disetujui oleh:

Penguji 1



Rosmalinda, S.T., M.Kom.

NIDN. 0425038203

Penguji 2



Iim Abdurohim, S.T., M.T.

NIDN. 0413107002

LEMBAR PERSETUJUAN PROGRAM STUDI

PERBANDINGAN ALGORITMA SEQUENTIAL SEARCH DAN ALGORITMA BINARY SEARCH PADA APLIKASI KAMUS BAHASA INDONESIA MENGGUNAKAN PHP DAN JQUERY

Disusun oleh :

MOCH ILHAM BAHARI

NPM. C1A160028

Telah diterima dan disetujui untuk memenuhi persyaratan mencapai gelar

SARJANA KOMPUTER

Pada

**PROGRAM STUDI INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI
UNIVERSITAS BALE BANDUNG**

Baleendah, Juli 2020

Disetujui oleh:

Mengetahui,

Dekan



Yudi Herdiana, ST., MT.

NIK. 04104808008

Mengesahkan,

Ketua Program Studi



Yaya Suharya, S.Kom., MT.

NIK. 01043170007

LEMBAR PERNYATAAN

Saya yang bertanda tangan dibawah ini:

Nama : MOCH ILHAM BAHARI

NPM : C1A160028

Judul Skripsi : PERBANDINGAN ALGORITMA SEQUENTIAL SEARCH
DAN ALGORITMA BINARY SEARCH PADA APLIKASI
KAMUS BAHASA INDONESIA MENGGUNAKAN PHP DAN
JQUERY

Menyatakan bahwa penulisan skripsi ini berdasarkan hasil penelitian, pemikiran dan pemaparan asli dari saya sendiri. Jika terdapat karya orang lain, saya mencantumkan sumber yang jelas. Pernyataan ini saya buat dengan sesungguhnya dan apabila di kemudian hari terdapat penyimpangan dan ketidakbenaran dalam pernyataan ini, maka saya bersedia menerima sanksi akademik yang berlaku di Fakultas Teknologi Informasi Universitas Bale Bandung. Demikian surat pernyataan ini saya buat dalam keadaan sadar tanpa paksaan dari pihak manapun.

Baleendah, Juli 2020



ABSTRACT

Scientific language is a variety of languages that functions to create precise and effective communication which is usually related to a particular professional or scientific field. Scientific language is starting to be widely used in several media such as books, blogs, journals, news, social media and others. It takes an Indonesian dictionary book to make it easier for people to define a scientific language.

However, searching for a dictionary word in the form of a book can take a long time because the search process is carried out conventionally. So it is necessary to make an application, one of which is a digital dictionary application that can simplify word searches and does not need to pay extra to buy a dictionary. For the search process, there is a search algorithm that can be used to apply to digital dictionary applications so that the search for words in the dictionary can be faster. Searching algorithm is a process to find certain data in a set of data of the same type. Of the various existing search algorithms, in this study the author aims to provide an overview of the two types of search algorithms, namely Sequential Search and Binary Search as well as the search method that is often used, namely SQL search on word searches in making Indonesian dictionary applications.

This study shows a performance analysis in the form of search speed and memory used, of the three search methods in searching a word. Tests carried out are looking for words at the beginning of the data, in the middle of the data, the final data and a lot of data taken from the uploaded pdf file. The results of the testers that have performed the search speed of the Binary Search algorithm are faster in performing word searches, while the memory used the Sequential Search algorithm has less memory usage.

Keywords: scientific words, comparison, dictionary, Sequential Search, Binary Search

ABSTRAK

Bahasa ilmiah adalah ragam bahasa yang berfungsi menciptakan komunikasi yang tepat dan efektif yang biasanya berkaitan dengan bidang profesional atau ilmiah tertentu. Bahasa ilmiah mulai banyak digunakan di beberapa media seperti buku, blog, jurnal, berita, media sosial dan lain-lain. Dibutuhkan sebuah buku kamus Bahasa Indonesia untuk memudahkan masyarakat dalam mendefinisikan sebuah Bahasa ilmiah.

Akan tetapi pencarian kata kamus berupa buku dapat memakan waktu yang cukup lama karena proses pencarian dilakukan secara konvensional. Sehingga perlu dibuat suatu aplikasi, salah satunya yaitu aplikasi kamus digital yang dapat mempermudah dalam pencarian kata dan tidak perlu mengeluarkan biaya tambahan untuk membeli kamus. Untuk proses pencarian terdapat algoritma *searching* yang bisa dimanfaatkan untuk diterapkan pada aplikasi kamus digital agar pencarian kata pada kamus bisa lebih cepat. Algoritma *Searching* merupakan sebuah proses untuk menemukan data tertentu dalam sekumpulan data yang bertipe sama. Dari berbagai macam algoritma *Searching* yang ada, dalam penelitian ini penulis bertujuan untuk memberikan gambaran perbandingan antara dua jenis algoritma *Searching* yaitu *Sequential Search* dan *Binary Search* serta metode pencarian yang sering digunakan yaitu *SQL search* pada pencarian kata didalam pembuatan aplikasi kamus Bahasa Indonesia.

Penelitian ini menunjukkan analisis kinerja berupa kecepatan pencarian dan *memory* yang digunakan, dari ketiga metode pencarian tersebut dalam pencarian sebuah kata. Pengujian yang dilakukan yaitu mencari kata pada awal data, tengah data, akhir data dan banyak data yang diambil dari file pdf yang diupload. Hasil dari pengujian yang sudah dilakukan yaitu kecepatan pencarian algoritma *Binary Search* lebih cepat melakukan pencarian kata, sedangkan untuk *memory* yang digunakan, algoritma *Sequential Search* lebih kecil.

Kata kunci: *kata ilmiah, perbandingan, kamus, Sequential Search, Binary Search*

KATA PENGANTAR

Puji dan Syukur kami panjatkan ke Hadirat Allah SWT, karena berkat limpahan Rahmat dan Karunia-Nya sehingga penulis dapat menyelesaikan skripsi ini dengan baik dan tepat pada waktunya. Skripsi ini merupakan salah satu syarat untuk menyelesaikan program studi Teknik Informatika jenjang Strata-1 Universitas Bale Bandung. Dalam skripsi ini penulis membahas mengenai Perbandingan Algoritma *Sequential Search* dan Algoritma *Binary Search* pada Aplikasi Kamus Bahasa Indonesia menggunakan PHP dan jQuery.

Skripsi ini dibuat dengan berbagai observasi dan beberapa bantuan dari berbagai pihak untuk membantu menyelesaikan tantangan dan hambatan selama mengerjakan skripsi ini. Oleh karena itu, penulis mengucapkan terima kasih yang sebesar-besarnya kepada :

1. Bapak Yudi Herdiana, S.T., M.T., selaku Dekan Fakultas Teknologi Informasi Universitas Bale Bandung.
2. Bapak Yaya Suharya, S.Kom., M.T., selaku Ketua Program Studi Teknik Informatika Fakultas Teknologi Informasi Universitas Bale Bandung.
3. Bapak Zen Munawar, S.Kom., M.Kom., selaku Pembimbing 1.
4. Ibu Nurul Imamah, S.T., M.T., selaku Pembimbing 2.
5. Ibu Rosmalinda, S.T., M.Kom selaku Penguji 1.
6. Bapak Iim Abdurrohman, S.T, M.T., selaku Penguji 2.
7. Bapak Mochamad Ridwan, S.T., yang ikut membantu membimbing.
8. Kedua orang tua yang telah memberi berbagai macam bantuan baik secara dorongan doa, motivasi, moral dan materi.
9. Teman seperjuangan selama di kelas belajar.
10. Semua pihak yang telah membantu dan memberikan dukungan kepada penulis untuk menyelesaikan skripsi ini.

Dalam penulisan skripsi ini, penulis telah berusaha semaksimal mungkin untuk menghasilkan yang terbaik, penulis juga menyadari bahwa masih banyak

kekurangan dalam skripsi ini. Oleh karena itu, segala kritik dan saran yang membangun akan penulis terima dengan baik. Semoga skripsi ini bermanfaat bagi kita semua.

Bandung, Juli 2020

Moch Ilham Bahari

NIM: C1A160028

DAFTAR ISI

LEMBAR PERSETUJUAN PEMBIMBING	i
LEMBAR PERSETUJUAN PENGUJI.....	ii
LEMBAR PERSETUJUAN PROGRAM STUDI.....	iii
LEMBAR PERNYATAAN	iv
<i>ABSTRACT</i>	v
ABSTRAK	vi
KATA PENGANTAR	vii
DAFTAR ISI.....	ix
DAFTAR GAMBAR	xi
DAFTAR TABEL.....	xv
DAFTAR LAMPIRAN	xvi
BAB 1 PENDAHULUAN	1
1.1. Latar Belakang.....	1
1.2. Rumusan Masalah	3
1.3. Batasan Masalah.....	3
1.4. Tujuan.....	3
1.5. Metodologi Penelitian	4
1.6. Sistematika Penulisan.....	6
BAB II TINJAUAN PUSTAKA.....	9
2.1. Landasan Teori	9
2.2. Dasar Teori	11
BAB III METODOLOGI PENELITIAN.....	29
3.1 Kerangka Pikir.....	29
3.2 Deskripsi.....	30
BAB IV ANALISIS DAN PERANCANGAN	34
4.1 Analisis	34
4.2 Perancangan.....	42
BAB V IMPLEMENTASI DAN PENGUJIAN	60

5.1	Implementasi	60
5.2	Pengujian	72
BAB VI KESIMPULAN DAN SARAN		96
6.1	Kesimpulan.....	96
6.2	Saran	97
DAFTAR PUSTAKA		98
LAMPIRAN		99

DAFTAR GAMBAR

Gambar 1.1 Metode Waterfall.....	4
Gambar 2.1 Tahapan pelaksanaan program oleh komputer.....	11
Gambar 2.2 Proses Pencarian dengan algoritma Sequential Search.....	20
Gambar 3.1 Kerangka Pikir.....	29
Gambar 4.1 Flowchart Sistem.....	36
Gambar 4.2 Flowchart Algoritma Sequential Search	36
Gambar 4.3 Flowchart Algoritma Binary Search	38
Gambar 4.4 Usecase Diagram Kamus Bahasa Indonesia	43
Gambar 4.5 Activity Diagram Pencarian Kata	45
Gambar 4.6 Activity Diagram Pencarian kata lewat file	45
Gambar 4.7 Activity Diagram Pengujian.....	46
Gambar 4.8 Activity Diagram Analisis.....	47
Gambar 4.9 Sequence Diagram user pencarian kata.....	48
Gambar 4.10 Sequence Diagram user pencarian kata lewat file.....	48
Gambar 4.11 Sequence Diagram analisis.....	49
Gambar 4.12 Class Diagram Aplikasi Kamus	50
Gambar 4.13 User Interface pencarian kata	51
Gambar 4.14 User Interface pencarian kata lewat file	52
Gambar 4.15 User Interface analysis area chart.....	52
Gambar 4.16 User Interface analisis bar chart	53
Gambar 4.17 User Interface Analisis line chart	53
Gambar 4.18 User Interface analisis kontribusi area chart	54
Gambar 4.19 User Interface analisis kontribusi bar chart.....	54
Gambar 4.20 User Interface pengisian data diri kontribusi	55
Gambar 4.21 User Interface pengujian 1 pada kontribusi.....	55
Gambar 4.22 User Interface pengujian 2 pada kontribusi.....	56
Gambar 4.23 User Interface pengujian 3 pada kontribusi.....	56
Gambar 4.24 User Interface pengujian 4 pada kontribusi.....	57
Gambar 4.25 User Interface fitur dark mode	57
Gambar 5.1 Tampilan halaman utama / Pencarian Kata.....	60

Gambar 5.2 Tampilan hasil pencarian kata.....	61
Gambar 5.3 Tampilan definisi kata yang ditemukan	61
Gambar 5.4 Tampilan pencarian kata lewat file	62
Gambar 5.5 Tampilan hasil pencarian kata lewat file.....	62
Gambar 5.6 Tampilan grafik kecepatan dan table pengujian.....	63
Gambar 5.7 Tampilan grafik memory dan table pengujian	63
Gambar 5.8 Tampilan detail proses pencarian.....	64
Gambar 5.9 Tampilan form data kontribusi	65
Gambar 5.10 Tampilan pengujian 1 kontribusi.....	65
Gambar 5.11 Tampilan pengujian 2 kontribusi.....	66
Gambar 5.12 Tampilan pengujian 3 kontribusi.....	66
Gambar 5.13 Tampilan pengujian 4 kontribusi.....	66
Gambar 5.14 Tampilan selesai pengujian kontribusi	67
Gambar 5.15 Tampilan kesalahan pencarian kata.....	68
Gambar 5.16 Tampilan kesalahan upload file.....	68
Gambar 5.17 Koding Algoritma Sequential Search.....	69
Gambar 5.18 Koding Algoritma Binary Search.....	70
Gambar 5.19 Koding pencarian SQL.....	71
Gambar 5.20 Line chart pengujian kecepatan algoritma Sequential Search pada awal data.....	73
Gambar 5.21 Line chart pengujian memory algoritma Sequential Search pada awal data	74
Gambar 5.22 Line chart pengujian kecepatan algoritma Binary Search pada awal data	75
Gambar 5.23 Line chart pengujian memory algoritma Binary Search pada awal data	75
Gambar 5.24 Line chart pengujian kecepatan algoritma sql search pada awal data	76
Gambar 5.25 Bar chart hasil akhir kecepatan pada awal data.....	76
Gambar 5.26 Bar chart hasil akhir memory pada awal data	77
Gambar 5.27 Line chart pengujian kecepatan algoritma Sequential Search pada awal data.....	78

Gambar 5.28 Line chart pengujian memory algoritma Sequential Search pada awal data	79
Gambar 5.29 Line chart pengujian kecepatan algoritma Binary Search pada tengah data	80
Gambar 5.30 Line chart pengujian memory algoritma Binary Search pada tengah data	80
Gambar 5.31 Line chart pengujian kecepatan algoritma sql search pada tengah data	81
Gambar 5.32 Line chart pengujian memory algoritma sql search pada tengah data	81
Gambar 5.33 Bar chart hasil akhir kecepatan pada tengah data	82
Gambar 5.34 Bar chart hasil akhir memory pada tengah data	82
Gambar 5.35 Line chart pengujian kecepatan algoritma sequence search pada akhir data	84
Gambar 5.36 Line chart pengujian kecepatan algoritma Binary Search pada akhir data	85
Gambar 5.37 Line chart pengujian memory algoritma Binary Search pada akhir data	85
Gambar 5.38 Line chart pengujian kecepatan algoritma sql search pada akhir data	86
Gambar 5.39 Line chart pengujian memory sql search pada akhir data	86
Gambar 5.40 Bar chart hasil akhir kecepatan pada akhir data	87
Gambar 5.41 Bar chart hasil akhir memory pada akhir data	87
Gambar 5.42 Line chart pengujian algoritma Sequential Search pada banyak data	89
Gambar 5.43 Line chart pengujian memory Sequential Search pada banyak data	89
Gambar 5.44 Line chart pengujian kecepatan algoritma Binary Search pada banyak data	90
Gambar 5.45 Line chart pengujian memory algoritma Binary Search pada banyak data	90
Gambar 5.46 Line chart pengujian kecepatan algoritma sql search pada banyak data	91

Gambar 5.47 Line chart pengujian memory algoritma sql search pada banyak data	92
Gambar 5.48 Bar chart hasil akhir kecepatan pada banyak data.....	92
Gambar 5.49 Bar chart hasil akhir memory pada banyak data	92
Gambar 5.50 Hasil akhir pengujian pada kecepatan pencarian	93
Gambar 5.51 Hasil akhir pengujian pada memory pencarian	94
Gambar 5.52 Hasil akhir pengujian oleh mahasiswa pada kecepatan pencarian..	95
Gambar 5.53 Hasil akhir pengujian oleh mahasiswa pada memory pencarian.....	95

DAFTAR TABEL

Table 1.1 Simbol Use Case Diagram	23
Table 1.2 Simbol Activity Diagram	25
Table 2.3 Simbol Sequence Diagram	26
Table 4.1 Instrumen Penelitian Perangkat Keras	35
Table 5.1 Table pengujian algoritma Sequential Search pada awal data	73
Table 5.2 Table pengujian algoritma Binary Search pada awal data	74
Tabel 5.3 Tabel pengujian algoritma Binary Search pada awal data	75
Tabel 5.4 Tabel pengujian algoritma Binary Search pada tengah data	78
Table 5.5 Tabel pengujian algoritma Binary Search pada tengah data	79
Table 5.6 Tabel pengujian algoritma sql search pada tengah data	80
Table 5.7 Tabel pengujian algoritma sequence search pada akhir data	83
Table 5.8 Tabel pengujian algoritma sql search pada tengah data	84
Table 5.9 Tabel pengujian algoritma sql search pada tengah data	86
Table 5.10 Tabel pengujian algoritma sql search pada banyak data	88
Table 5.11 Tabel pengujian algoritma Binary Search pada banyak data	89
Table 5.12 Tabel pengujian algoritma sql search pada banyak data	91

DAFTAR LAMPIRAN

Lampiran 1 Data Pengujian.....	99
Lampiran 2 <i>Source Code</i> Aplikasi	102

BAB 1

PENDAHULUAN

1.1. Latar Belakang

Bahasa, dalam perkembangannya di dunia ilmu pengetahuan serta teknologi, memiliki peran yang sangat krusial bagi pemahaman manusia akan segala segi kehidupan di era global saat ini. Bagi dunia pendidikan dan akademik di Indonesia terutama, bahasa menempati posisi penting sebagai mediator antara satu disiplin ilmu dengan disiplin ilmu lainnya. Salah satu bahasa yang cukup berperan besar dalam menjembatani para akademisi di Indonesia untuk memahami ilmu-ilmu dari luar tersebut adalah bahasa Ilmiah.

Terdapat beragam kata dan istilah dalam Bahasa Indonesia. Istilah-istilah ini merupakan kata serapan maupun bahasa baku asli dari Bahasa Indonesia. Istilah-istilah ini biasanya disebut sebagai istilah bahasa intelektual atau bahasa ilmiah. Di kalangan kaum intelektual, penggunaan bahasa ilmiah sudah sangat menjalar di kehidupan sehari-hari, baik secara lisan maupun tulisan. Namun bagaimana dengan masyarakat awam yang tidak terbiasa mendengar atau membaca bahasa ilmiah. Itu akan menjadi masalah tersendiri ketika masyarakat tidak bisa mendefinisikan atau tidak mengerti bahasa ilmiah.

Masyarakat awam ketika membaca bahasa ilmiah mereka biasanya akan bersikap apatis dibandingkan mencari tahu definisi atau makna dari sebuah bahasa ilmiah tersebut. Dibutuhkan sebuah Kamus Bahasa Indonesia untuk memudahkan masyarakat dalam mendefinisikan sebuah Bahasa ilmiah. Dalam pencarian sebuah kata, tentu kecepatan mencari menjadi salah satu aspek yang cukup penting dalam Aplikasi Kamus Bahasa Indonesia. Beberapa algoritma pencarian seperti Algoritma *Sequential Search* dan *Binary Search* bisa menjadi opsi untuk diterapkan di aplikasi kamus. Dengan ini penulis ingin membandingkan kedua algoritma tersebut untuk

mencari mana yang lebih baik untuk diterapkan pada Aplikasi Kamus Bahasa Indonesia.

Proses pencarian kata kamus berupa buku dapat memakan waktu yang cukup lama karena proses pencariannya secara manual. Sehingga perlu dibuat suatu aplikasi, salah satunya yaitu aplikasi kamus digital yang dapat mempermudah dalam pencarian kata dan tidak perlu mengeluarkan biaya tambahan untuk membeli kamus. Dalam pembuatan aplikasi kamus digital ini perlu metode yang efektif, karena dalam proses pencarian data merupakan suatu bagian yang penting. Metode yang diterapkan pada aplikasi kamus digital ini yaitu *Sequential Searching*. (Gunawan, 2016). Pencarian kata pada algoritma *Binary Search* lebih cepat dalam proses pencarian dan efisien waktu. Dapat juga digunakan untuk pencarian karakter, angka maupun simbol. (Rusydi Umar, 2017).

Ada beberapa faktor yang mempengaruhi sikap apatis terhadap pencarian makna pada sebuah bahasa ilmiah. Salah satunya adalah rasa malas untuk membuka kamus Bahasa Indonesia dalam bentuk buku. Dikarenakan membutuhkan waktu lebih untuk menemukan definisi atau makna pada sebuah bahasa ilmiah yang mereka tidak ketahui. Padahal mendefinisikan sebuah bahasa ilmiah pada sebuah media tulis merupakan salah satu faktor penting dalam penyerapan informasi yang ada. Karena membaca bukan hanya sekedar membaca. Kita perlu menyerap dan mempelajari informasi agar bermanfaat bagi diri kita kedepannya. Alasan utama penggunaan bahasa ilmiah dalam semua bidang ilmu sebagai sarana identifikasi dan referensi. Dengan adanya kamus digital dapat memudahkan pencarian kata pada kamus sehingga proses pencarian pun bisa dilakukan secara cepat. Kecepatan pencarian pun menjadi aspek penting pada proses pencarian kata pada kamus agar pengalaman pencarian kata yang ingin didefinisikan semakin cepat didapat oleh karena itu, penelitian ini dimaksudkan untuk membandingkan algoritma *Sequential Search* dan *Binary Search* pada aplikasi kamus Bahasa Indonesia menggunakan php dan jquery dengan menawarkan kemudahan pendefinisian sebuah bahasa ilmiah dan mengetahui masing-masing kecepatan algoritma.

1.2. Rumusan Masalah

Perumusan masalah dalam penelitian ini yaitu :

1. Bagaimana membandingkan antara algoritma *Sequential Search*, *Binary Search* dan *SQL search* pada aplikasi kamus Bahasa Indonesia?
2. Bagaimana mempercepat pencarian kata pada aplikasi kamus Bahasa Indonesia?

1.3. Batasan Masalah

Pembatasan suatu masalah digunakan untuk menghindari adanya penyimpangan maupun pelebaran pokok masalah supaya penelitian tersebut lebih terarah dan memudahkan dalam pembahasan sehingga tujuan penelitian akan tercapai. Beberapa batasan masalah dalam penelitian ini adalah sebagai berikut:

- 1) Aplikasi hanya menggunakan teknologi web yaitu HTML, CSS, JavaScript, Bootstrap, PHP, JQuery dan AJAX.
- 2) Aplikasi ini terdapat 24.460 kata sebagai bahan penelitian.
- 3) Aplikasi ini menggunakan algoritma pencarian *Sequential Search*, *Binary Search* dan *SQL search*.
- 4) Hanya membandingkan kecepatan pencarian dan memory yang digunakan.
- 5) Hanya menggunakan 5 file pdf untuk pengujian kata.

1.4. Tujuan

Tujuan dari penelitian ini adalah :

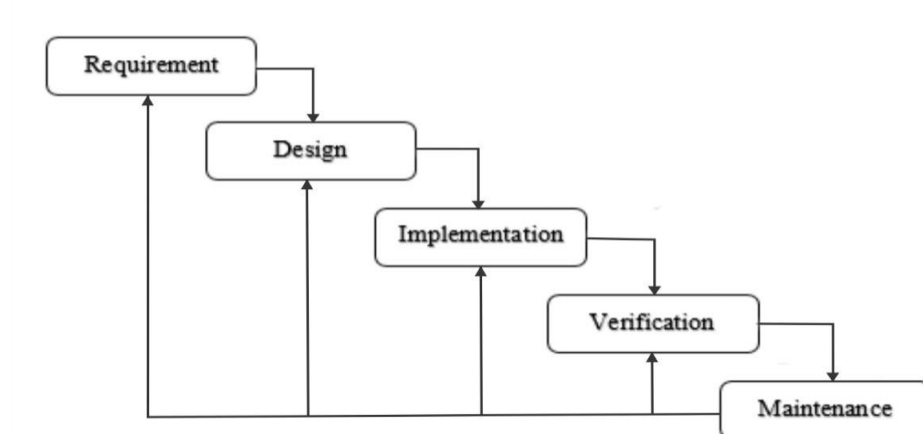
1. Membandingkan algoritma *Sequential Search*, algoritma *Binary Search* dan *SQL seacrh*.
2. Mempercepat pencarian kata pada aplikasi kamus Bahasa Indonesia

1.5. Metodologi Penelitian

1.5.1. Metode *Waterfall*

Metode yang digunakan dalam pembuatan aplikasi kamus Bahasa Indonesia ini adalah metode *waterfall*. Alasan menggunakan metode ini adalah karena metode *waterfall* melakukan pendekatan secara sistematis dan berurutan dalam membangun suatu sistem. Proses metode *waterfall* yaitu pada pengerjaan dari suatu sistem dilakukan secara berurutan. Sistem yang dihasilkan akan berkualitas baik, dikarenakan pelaksanaannya secara bertahap sehingga tidak terfokus pada tahapan tertentu.

Tahapan dari metode *waterfall* adalah:



Gambar 1.1 Metode *Waterfall*

1) *Requirement*

Tahap ini pengembang sistem diperlukan komunikasi yang bertujuan untuk memahami perangkat lunak yang diharapkan oleh pengguna dan batasan perangkat lunak tersebut. Informasi ini biasanya dapat diperoleh melalui wawancara, diskusi atau survei langsung. Informasi dianalisis untuk mendapatkan data yang dibutuhkan oleh pengguna.

2) **Desain Sistem**

Spesifikasi kebutuhan dari tahap sebelumnya akan dipelajari dalam fase ini dan desain sistem disiapkan. Desain Sistem membantu dalam menentukan perangkat keras(*hardware*) dan sistem persyaratan dan juga membantu dalam mendefinisikan arsitektur sistem secara keseluruhan.

3) ***Implementation***

Pada tahap ini, sistem pertama kali dikembangkan di program kecil yang disebut unit, yang terintegrasi dalam tahap selanjutnya. Setiap unit dikembangkan dan diuji untuk fungsionalitas yang disebut sebagai unit testing.

4) ***Verification***

Seluruh unit yang dikembangkan dalam tahap implementasi diintegrasikan ke dalam sistem setelah pengujian yang dilakukan masing-masing unit. Setelah integrasi seluruh sistem diuji untuk mengecek setiap kegagalan maupun kesalahan.

5) ***Maintenance***

Tahap akhir dalam model *waterfall*. Perangkat lunak yang sudah jadi, dijalankan serta dilakukan pemeliharaan. Pemeliharaan termasuk dalam memperbaiki kesalahan yang tidak ditemukan pada langkah sebelumnya. Perbaikan implementasi unit sistem dan peningkatan jasa sistem sebagai kebutuhan baru.

1.5.2. Metode Pengumpulan data

Metode pengumpulan data adalah sebuah metode tentang bagaimana dalam mengumpulkan data yang ada. Adapun metode pengumpulan yang digunakan adalah observasi dan studi pustaka.

1. Observasi

Tahap observasi yang dilakukan penulis yakni mencari dan mengamati penelitian sebelumnya pada topik yang serupa dengan penelitian yang sedang dibuat dan mengamati alur pada masing-masing algoritma yang ingin diteliti serta mencari sumber kata kamus untuk dapat diterapkan pada Aplikasi kamus Bahasa Indonesia.

2. Studi Pustaka

Metode Studi pustaka adalah merupakan metode pengumpulan data dengan cara mempelajari dan mengamati serta menganalisis berkas-berkas atau dokumen-dokumen yang sudah ada yang berhubungan dengan masalah tersebut, disini penulis membaca 3 jurnal yang berelevansi dengan topik penelitian dan mengambil 3 jurnal dan merangkum jurnal tersebut.

1.5.3. Metode Perancangan

Tahap perancangan merupakan pengembangan dari gambaran umum sistem. Dalam tahap perancangan dijelaskan lebih detail tentang isi dari aplikasi yang dibuat yaitu dengan membuat *Flow chart* dan diagram *Unified Modelling Language* (UML) yang meliputi *use case*, *activity diagram* dan *sequence diagram* serta membuat desain *input* dan *output*. Setelah tahap perancangan selesai maka dilakukan tahap implementasi yaitu menerjemahkan desain ke dalam *source code* berbasis web.

1.6. Sistematika Penulisan

Gambaran mengenai keseluruhan skripsi dan pembahasannya dapat dijelaskan dalam sistematika penulisan sebagai berikut :

a. Bab 1 Pendahuluan

- Bagian pendahuluan menjelaskan mengenai latar belakang masalah,

- Rumusan masalah, batasan masalah, tujuan penelitian, metodologi penelitian dan sistematika penulisan,

b. Bab II Tinjauan Pustaka

Bagian ini membahas mengenai kajian teoritis yang meliputi :

- Landasan Teori
- Dasar Teori

c. Bab III Metodologi Penelitian

Bagian ini membahas mengenai komponen dari metode Penelitian yaitu :

- Kerangka Pikir
- Dasar Teori

d. Bab IV Analisis dan Perancangan

Bagian ini membahas mengenai deskripsi hasil dan temuan penelitian sesuai dengan rumusan masalah atau pertanyaan penelitian berupa :

- Analisis
- Instrumen Penelitian
- Analisis Sistem
- Analisis Kebutuhan
- Hasil Analisis
- Perancangan

e. Bab V Implementasi dan Pengujian

Bagian ini membahas mengenai implementasi dan pengujian pada aplikasi dari hasil analisis dan perancangan yang dibuat berupa :

- Implementasi
- Pengujian

f. Bab VI Kesimpulan dan Saran

Bagian ini membahas mengenai makna penelitian terhadap hasil analisis temuan penelitian berupa :

- Kesimpulan
- Saran

BAB II

TINJAUAN PUSTAKA

2.1. Landasan Teori

Dalam penelitian ini akan digunakan tiga tinjauan studi yang nantinya mendukung penelitian yang akan dilakukan, dimana tinjauan studi yang diambil adalah :

1. Oleh Imaddudin Aziz, Hani Harfani (2016) dari program studi teknik informatika, STMIK Nusa Mandiri dengan judul aplikasi kamus bahasa betawi berbasis android menggunakan metode *sequential search*. Pada penelitian yang dilakukan mengangkat masalah menurunnya penggunaan Bahasa daerah dalam komunikasi sehari-hari. Hal ini terlihat dari kebiasaan generasi muda yang lebih menyukai berkomunikasi dengan bahasa gaul daripada bahasa daerah yang digunakan. Penelitian ini membuat aplikasi kamus bahasa Betawi agar bisa menjadi solusi untuk mengenalkan Bahasa daerah dengan mudah. Untuk pencarian kata menggunakan algoritma *Sequential Search*. Pencarian kata dilakukan dengan menelusuri kata satu persatu, kemudian di cocokan dengan kata yang dicari. Jika kata yang dicari sama dengan kata yang di cocokan, maka penelusuran dihentikan, sebaliknya jika kata yang dicari belum sama dengan kata yang dicocokan maka penelusuran dilanjutkan hingga kata yang dicari ketemu. Hasil dari penelitian ini yaitu algoritma *Sequential Search* dapat diterapkan pada aplikasi kamus bahasa Betawi dan dapat digunakan sebagai media yang efisien dan efektif menerjemahkan bahasa betawi.
2. Oleh Muhamad Fatkhur Rohim, Retno Hapsari, Yoga Religia, Dwi Prasetyo dari Fakultas Ilmu Komputer, Universitas Dian Nuswantoro,

Semarang dengan judul analisis algoritma *Sequential Search* dan *Binary Search* pada big data. Pada penelitian ini menjelaskan ada beberapa algoritma yang bisa digunakan untuk melakukan pencarian, yaitu algoritma *Sequential Search* dan *Binary Search*. Algoritma *Sequential Search* yaitu algoritma yang dasar dan simpel dari pencarian, dimana menggunakan metode pencarian data dari paling awal hingga paling akhir dari sebuah list sampai data ditemukan. *Binary Search* yaitu sebuah list yang sudah terurut kemudian dibagi menjadi dua bagian. Awalnya adalah membandingkan inputan dengan nilai tengah, selanjutnya dibandingkan ke kanan atau ke kiri sesuai dengan urutan listnya. Pada penelitian ini menjelaskan masalah dengan adanya data yang besar, di bandingkanlah kedua algoritma dengan menguji kecepatan waktu proses dan kompleksitas algoritma pencarian. Hasil dari penelitian ini yaitu dari percobaan dengan menggunakan bahasa python pada *big data*, dapat disimpulkan algoritma *Sequential Search* memiliki kompleksitas waktu lebih besar dibanding dengan *Binary Search*.

3. Oleh Rusydi Umar, Imam Riadi, Bashor dan Fauzan Muthohirin dari Program Studi Teknik Informatika dan Sistem Informasi, Universitas Ahmad Dahlan, Yogyakarta dengan judul implementasi algoritma *Binary Search* pada aplikasi konkordansi al-qur'an berbasis android. Pada penelitian dijelaskan salah satu teknologi yang banyak di kembangkan adalah konkordansi. Konkordansi merupakan pencarian akar kata. Salah satu aplikasi nya adalah konkordansi Al-Quran. Pencarian Al-Quran untuk keperluan tertentu sangat dibutuhkan oleh sebagian masyarakat. Sebagian orang mengetahui bunyi atau ayat alquran sebagiannya saja dan tidak ingat ayat seutuhnya atau hafal ayat sepenuhnya. Akan tetapi untuk mencarinya didalam Al-Qur'an akan kesulitan. Hasil penelitian yaitu implementasi algoritma *Binary Search* untuk mempercepat pencarian akar kata di dalam Al-Qur'an pada aplikasi Android.

2.2. Dasar Teori

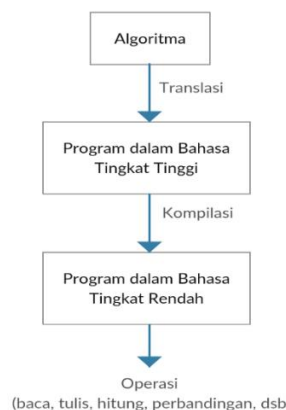
2.2.1. Algoritma

Mesin menjalankan prosedur pengurutan untuk memberikan solusi dari setiap instansiasi persoalan. Langkah-langkah mengurutkan itu sebagai penyelesaian atau pemecahan masalah. Prosedur yang berisi langkah-langkah penyelesaian masalah disebut algoritma. Algoritma adalah urutan langkah-langkah untuk memecahkan suatu masalah. (Munir & Leony, 2016).

Algoritma berisi urutan langkah-langkah untuk menyelesaikan persoalan yang diberikan. Pada dasarnya sebuah algoritma menerima beberapa masukan (*input*), memprosesnya dalam urutan langkah-langkah tadi, dan menghasilkan luaran (*output*).

Ketika algoritma "dijalankan" (oleh manusia atau oleh komputer), maka langkah-langkah tersebut dikerjakan dari awal sampai akhirnya berhenti dan kita memperoleh solusi persoalan. jika algoritmanya benar, Maka hasilnya (solusinya) pasti benar, sebaliknya jika algoritmanya salah Maka hasilnya juga salah.

Ada 2 pesan penting tentang algoritma. pertama, sebuah algoritma harus benar. kedua, algoritma harus berhenti dan setelah berhenti algoritma memberikan hasil yang benar.



Gambar 2.1 Tahapan pelaksanaan program oleh komputer

2.2.2. Pencarian

Pencarian (*searching*) merupakan proses yang fundamental di dalam pengolahan data. Pencarian adalah proses menemukan nilai (data) tertentu di dalam sekumpulan data yang terstruktur. sebagai contoh, untuk mengubah (*update*) data tertentu di dalam sebuah larik, maka aktivitas awal yang harus dilakukan adalah mencari keberadaan data tersebut di dalam larik. Jika data yang dicari ditemukan maka data tersebut dapat diubah nilainya dengan nilai yang baru. Aktivitas awal yang sama juga dilakukan pada proses penambahan (*insert*) data baru ke dalam larik (tidak boleh ada duplikasi). proses penambahan data dimulai dengan mencari apakah data baru sudah terdapat di dalam larik. jika sudah ada, maka data tersebut tidak perlu dimasukkan, tetapi jika belum ada, maka masukkan data baru tersebut ke dalam larik. (Munir & Leony, 2016).

2.2.3. Kamus

Buku yang berisi daftar kosa kata suatu bahasa yg disusun secara alfabetis dengan disertai penjelasan makna dan keterangan lain yang diperlukan serta dilengkapi dengan contoh pemakaian entri dalam kalimat. (Kamus Bahasa Indonesia).

Kamus memuat khazanah kosakata bahasa yang dapat menjadi lambang atau indikator kemajuan peradaban masyarakat pendukungnya. Demikian pula, bahasa Indonesia memiliki kekayaan kosakata yang memadai sebagai sarana pikir, ekspresi, dan komunikasi di berbagai bidang kehidupan. Kamus Bahasa Indonesia ini merupakan buku rujukan yang memuat khazanah kata bahasa Indonesia. Selain kosakata umum bahasa Indonesia, kamus ini memuat berbagai istilah dari bidang ilmu yang pasti akan sangat bermanfaat bagi pelajar dan mahasiswa.

2.2.4. Bahasa Ilmiah

Bahasa ilmiah ialah bahasa yang mendefinisikan secara tepat istilah dan pengertian yang berkaitan dengan suatu penelitian, agar tidak menimbulkan kerancuan.

2.2.5. HTML

HTML adalah bahasa standar yang digunakan untuk menampilkan halaman web. Yang bisa dilakukan dengan html yaitu :

- Mengatur tampilan dari halaman web dan isinya
- Membuat tabel dalam halaman web
- Mempublikasikan halaman web secara online
- Membuat form yang bisa digunakan untuk menangani registrasi dan transaksi via web
- Menambahkan objek-objek seperti citra, audio, video, animasi, Java applet dalam halaman web
- Menampilkan area gambar (canvas) di browser

(Hidayatullah & Kawistara, 2017:15).

2.2.6. CSS

CSS (Cascading Style Sheet) adalah salah satu bahasa desain web (style sheet language) yang mengontrol format tampilan sebuah halaman web yang ditulis dengan menggunakan penanda(markup language. Biasanya CSS digunakan untuk mendesain sebuah halaman HTML dan XHTML, tetapi sekarang CSS bisa diaplikasikan untuk segala dokumen XML, termasuk SVG dan XUL bahkan ANDROID.

CSS dibuat untuk memisahkan konten utama dengan tampilan dokumen yang meliputi layout, warna dan font. Pemisahan ini dapat meningkatkan daya akses konten pada web, menyediakan lebih banyak fleksibilitas dan kontrol dalam spesifikasi dari sebuah karakteristik dari

sebuah tampilan, memungkinkan untuk membagi halaman untuk sebuah formatting dan mengurangi kerumitan dalam penulisan kode dan struktur dari konten, contohnya teknik *tableless* pada desain web.

2.2.7. JavaScript

JavaScript (JS) sangat ringan, terinterpretasi, bahasa pemrograman dengan *first-class functions*. Umum dikenal sebagai bahasa scripting untuk halaman web, Banyak lingkungan non-browser juga menggunakan javascript, seperti node.js dan Apache CouchDB. JS merupakan *prototype-based*, *multi-paradigm*, bahasa scripting dinamis, mendukung *object-oriented*, diperlukan, dan declarative (mis. functional programming) styles. (developer.mozilla.org).

JavaScript adalah bahasa pemrograman web yang bersifat *Client Side Programming Language*. *Client Side Programming Language* adalah tipe bahasa pemrograman yang pemrosesannya dilakukan oleh *client*. Aplikasi *client* yang dimaksud merujuk kepada web browser seperti Google Chrome dan Mozilla Firefox.

Bahasa pemrograman *Client Side* berbeda dengan bahasa pemrograman *Server Side* seperti PHP, dimana untuk *server side* seluruh kode program dijalankan di sisi server.

Untuk menjalankan JavaScript, kita hanya membutuhkan aplikasi text editor dan web browser. JavaScript memiliki fitur: high-level programming language, client-side, loosely typed dan berorientasi objek.

2.2.8. JQuery

JQuery merupakan sebuah javascript library atau bisa disebut juga perpustakaan atau kumpulan kode-kode program javascript yang siap pakai. dalam arti sederhana JQuery dapat digunakan untuk meringkas sebuah kode program javascript yang panjang dalam sebuah proyek pembuatan website.

Sehingga anda sebagai developer web akan diberi kemudahan dalam menghadapi bagian yang mengandung JavaScript. JQuery merupakan program yang berjalan pada sisi server dan akan ditampilkan pada browser web dapat berjalan di dalam html atau bahasa pemrograman berbasis web lainnya seperti PHP atau JSP (Adi, 2019:3).

JQuery memiliki beberapa keunggulan diantaranya :

- Mudah dioperasikan karena hanya mengikuti kan beberapa baris kode saja sehingga tidak perlu menuliskan program yang panjang
- JQuery menyediakan fasilitas plugins yang beragam. Plugins 2 ini sangat berguna dalam aksesoris maupun teknologi suatu website yang Anda bangun menggunakan jquery
- JQuery dapat menyesuaikan style CSS dalam semua browser web sehingga pengunjung dapat menikmati tampilan web Anda yang bagus dari semua browser
- Menyediakan fasilitas untuk tampilan animasi seperti pada pembuatan animasi menggunakan flash
- Didukung oleh komunitas dan pengembang JQuery yang tersebar di seluruh dunia
- JQuery merupakan open source sehingga bebas dan gratis digunakan oleh siapa saja

(Adi, 2019:6).

2.2.9. AJAX

Ajax (*Asynchronous JavaScript and XML*) merupakan teknik yang memanfaatkan *XMLHttpRequest* untuk berkomunikasi dengan server *side* script. Ajax ini bisa menerima dan mengirim data/informasi dalam format JSON, XML, HTML dan TEXT. Keuntungannya adalah Ajax ini bisa mengirim dan menerima data/informasi tanpa perlu melakukan *refresh* halaman web (Adi, 2019:146).

2.2.10. PHP

PHP (*hypertext preprocessor*) atau disingkat dengan PHP ini adalah suatu bahasa *scripting* khususnya digunakan untuk *web development*. Karena sifatnya yang *server-side-scripting*, maka untuk menjalankan PHP harus menggunakan web server. (Hidayatullah & Kawistara, 2017:223).

PHP sudah menjadi bahasa *scripting* umum yang banyak digunakan di kalangan *developer* web. Mempunyai banyak kelebihan menjadi alasan utama kenapa PHP lebih dipilih sebagai basis umum dalam membuat sebuah web.

PHP juga sudah banyak komunitasnya, sehingga jika ada pengguna yang kesulitan dalam menyelesaikan error, akan lebih mudah menemukan solusinya. Di Indonesia sendiri Bahasa PHP masih banyak digunakan karena lebih mudah, lebih murah dan masih banyak lowongan kerja yang membutuhkan programmer PHP, sehingga pengguna baru tidak usah takut akan masa depan PHP di Indonesia.

2.2.11. MySQL

MySQL adalah salah satu aplikasi DBMS yang sudah sangat banyak digunakan oleh para pemrogram aplikasi. Kelebihan dari MySQL adalah gratis, handal, selalu di-update dan banyak forum yang memfasilitasi para pengguna jika memiliki kendala. MySQL juga menjadi DBMBS yang sering dibundling dengan server sehingga proses instalasinya jadi lebih mudah (Hidayatullah dan Kawistara, 2017:175).

2.2.12. Pdf-to-text

Pdf-to-text adalah library PHP yang berfungsi untuk memudahkan merubah isi file pdf menjadi teks.

2.2.13. Morris.js

Morris.js adalah sebuah *library* javascript untuk membuat grafik. *Simple* API yang dapat digunakan untuk membuat line, bar, area dan donat chart. (Morrisjs.github).

Pembuatan chart pada morris.js dibantu oleh library javascript lainnya seperti jQuery dan raphael.js.

2.2.14. Raphael.js

Raphael.js adalah *library* javascript untuk memungkinkan menggambar vektor grafis di browser. Raphael.js digunakan untuk syarat aset yang harus ditambahkan untuk menampilkan grafik chart pada morris.js.

2.2.15. XAMPP

Xampp adalah sebuah perangkat lunak yang berfungsi sebagai server lokal. Biasanya digunakan pada saat membuat website untuk menguji fitur dan menampilkan konten pada sebuah website tanpa terkoneksi internet. Karena berfungsi sebagai server lokal, alhasil website yang ditampilkan pun hanya bisa di komputer lokal, tidak bisa diakses oleh semua orang.

Web server ini adalah tempat untuk menyimpan aplikasi web, kemudian mengaksesnya melalui internet. setiap perubahan, kecil maupun besar yang di upload ke server baru setelah itu, baru diperiksa apakah scriptnya sudah sesuai dengan keinginan atau belum” (Hidayatullah dan Kawistara, 2017:123).

2.2.16. Bootstrap

Bootstrap merupakan sebuah *framework* CSS yang paling banyak diminati oleh para developer website. Class-class CSS dalam bootstrap

sudah dibakukan sehingga pengerjaan sebuah project berbasis web menjadi semakin mudah dilakukan secara bersama-sama dalam sebuah tim. kita dengan mudah dapat mendesain tampilan website yang responsif dengan menggunakan bootstrap. responsif maksudnya adalah lebar halaman website akan disesuaikan secara otomatis berdasarkan perangkat yang digunakan untuk mengakses nya baik itu ketika diakses menggunakan PC laptop tablet ataupun smartphone sehingga website akan menyesuaikan dengan lebar perangkat yang digunakan pengunjung (Kaban, 2019:1).

Bootstrap memudahkan seorang *developer* web untuk membuat tampilan website tanpa harus membuatnya dari awal. Karena bootstrap menyediakan beberapa *component* yang siap pakai dengan cara menambahkan sebuah *class* di dalam tag HTML. Dengan begitu, untuk membuat tampilan website akan menjadi sangat cepat. Untuk dapat menggunakan bootstrap, setidaknya pengguna sudah bisa dasar HTML dan CSS.

2.2.17. Sublime Text

Sublime text adalah text editor yang kini cukup banyak peminatnya, dan penggunaan software ini bisa digunakan juga oleh berbagai macam platform OS (Operating System). Sublime text juga banyak sekali mendukung banyak bahasa pemrograman dan bahasa markup. Sublime Text mendukung berbagai bahasa pemrograman dan mampu menyajikan fitur syntax highlight hampir di semua bahasa pemrograman yang didukung ataupun dikembangkan oleh komunitas seperti; C, C++, C#, CSS, D, Dylan, Erlang, HTML, Groovy, Haskell, Java, JavaScript, LaTeX, Lisp, Lua, Markdown, MATLAB, OCaml, Perl, PHP, Python, R, Ruby, SQL, TCL, Textile dan XML

2.2.18. Web Browser

Web browser adalah perangkat utama yang akan kita gunakan untuk menampilkan halaman web dengan memanfaatkan koneksi internet. Di setiap komputer telah terinstall Web Browser bawaan seperti Internet Explorer (Windows), Safari (Mac) dan Firefox (Linux Ubuntu).

Setiap browser memiliki perbedaan dalam hal menampilkan halaman web dan fitur-fitur yang didukung dalam HTML dan CSS. Boleh jadi halaman web yang anda buat ditampilkan benar pada salah satu browser namun acak-acakan pada browser lainnya. Untuk itu perlu beberapa browser yang terinstall dalam komputer anda untuk menguji penampilan website yang dibuat.

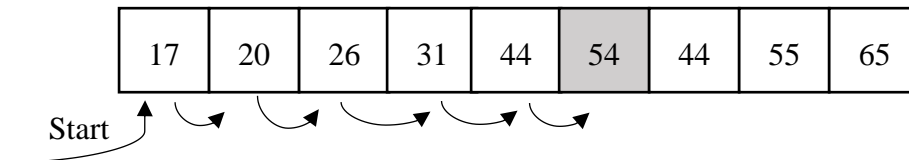
2.2.19. Sequential Search

Algoritma pencarian yang paling sederhana yaitu metode pencarian beruntun (*Sequential Search*) nama lain algoritma pencarian beruntun adalah pencarian lurus (linear search). Algoritma pencarian beruntun adalah proses membandingkan setiap elemen larik satu persatu secara berurutan, mulai dari elemen pertama, sampai elemen yang dicari ditemukan atau seluruh elemen sudah habis diperiksa. (Munir dan Lidya, 2016:444).

Terdapat dua versi algoritma pencarian beruntun. Pada algoritma versi pertama, aksi pembandingan dilakukan di awal pengulangan, tepatnya pada kondisi pengulangan, sedangkan algoritma versi kedua, aksi pembandingan dilakukan didalam badan pengulangan. Versi pertama tidak menggunakan peubah Boolean dalam proses pencarian, sedangkan versi kedua menggunakan peubah boolean.

Secara umum, metode pencarian beruntun berjalan lambat. Waktu pencarian sebanding dengan jumlah elemen larik. Misalkan larik berukuran n elemen. Maka, pada kasus di mana x tidak terdapat di dalam larik atau x ditemukan pada elemen yang terakhir, kita harus melakukan perbandingan

dengan seluruh elemen larik, yang berarti jumlah perbandingan yang terjadi sebanyak n kali. Kita katakan bahwa waktu pencarian dengan algoritma pencarian beruntun sebanding dengan n . Bayangkan bila larik berukuran 100.000 buah elemen, maka kita harus melakukan perbandingan sebanyak 100.000 buah elemen. Andaikan satu operasi perbandingan elemen larik membutuhkan waktu 0.01 detik, maka untuk 100.000 buah perbandingan diperlukan waktu sebesar 1000 detik atau 16,7 menit. Semakin banyak elemen larik, semakin lama pula waktu pencariannya. Karena itulah metode pencarian beruntun tidak bagus untuk volume data yang besar.



Gambar 2.2 Proses Pencarian dengan algoritma *Sequential Search*

Adapun Proses Algoritma *Sequential Searching* adalah sebagai berikut:

- a) Pertama data melakukan perbandingan satu per satu secara berurutan dalam kumpulan data dengan data yang dicari sampai data tersebut ditemukan atau tidak ditemukan.
- b) Pada dasarnya, pencarian ini hanya melakukan pengulangan data dari 1 sampai dengan jumlah data (n).
- c) Setiap pengulangan, dibandingkan data ke- i dengan data yang sedang dicari.
- d) Apabila data sama dengan yang dicari, berarti data telah berhasil di temukan. Sebaliknya apabila sampai akhir melakukan pengulangan tidak ada data yang sama dengan yang dicari, berarti data tidak ada yang ditemukan. Urutan Algoritma *Sequential Searching*:

1. $i \leftarrow 0$
2. $Ketemu \leftarrow \text{false}$
3. Selama (tidak ketemu) dan ($i < N$) kerjakan baris 4

4. Jika ($\text{Data}[i] = \text{key}$) maka ketemu $\leftarrow \text{true}$ Jika tidak $i \leftarrow i + 1$
5. Jika (Ketemu) maka i adalah indeks dari data yang dicari

2.2.20. Binary Search

Algoritma ini digunakan untuk kebutuhan pencarian dengan waktu yang cepat. Sebenarnya, dalam kehidupan sehari-hari kita sering menerapkan pencarian bagi dua. Untuk mencari kata tertentu di dalam kamus kita tidak membuka kamus itu dari halaman awal sampai halaman akhir satu persatu, namun kita mencarinya dengan cara membelah atau membagi dua buku itu jika kata yang dicari tidak terletak di halaman pertemuan itu kita mencari lagi di belahan bagian kiri atau belahan bagian kanan dengan cara membagi dua belahan yang dimaksud begitu seterusnya sampai kata yang dicari ditemukan hal ini hanya bisa dilakukan jika kata-kata di dalam kamus sudah terurut. (Munir dan Lidya, 2016:456).

Algoritma *Binary Search* sangat efektif pada pencarian data yang cukup banyak, beda dengan algoritma *Sequential Search* yang mencari tiap elemen pada array, algoritma *Binary Search* mencari dengan cara membagi sebuah data menjadi dua bagian lalu memeriksa tiap datanya, dengan begitu pencarian pun menjadi lebih cepat langkahnya.

Pencarian Biner (*Binary Search*) dilakukan untuk :

- Memperkecil jumlah operasi perbandingan yang harus dilakukan antara data yang dicari dengan data yang ada di dalam tabel, khususnya untuk jumlah data yang sangat besar ukurannya.
- Prinsip dasarnya adalah melakukan proses pembagian ruang pencarian secara berulang-ulang sampai data ditemukan atau sampai ruang pencarian tidak dapat dibagi lagi (berarti ada kemungkinan data tidak ditemukan).
- Syarat utama untuk pencarian biner adalah data di dalam tabel harus sudah terurut, misalkan terurut menaik.

Setiap kali pencarian, larik dibagi dua menjadi dua bagian yang berukuran hampir sama. Pada setiap pembagian, elemen tengah dibandingkan apakah sama dengan x (if $(L[k] = x)$). Pada kasus terburuk, yaitu pada kasus x tidak terdapat di dalam larik atau x ditemukan setelah ukuran larik tinggal 1 elemen, larik akan dibagi sebanyak $2\log(n)$ kali, sehingga jumlah perbandingan yang dibutuhkan adalah sebanyak $2\log(n)$ kali. Kita katakan bahwa waktu pencarian sebanding dengan $2\log(n)$ [WIR76]. Untuk $n = 256$ elemen misalnya, kasus terburuk menghasilkan pembagian larik sebanyak $2\log(256) = 8$ kali,

Proses yang terjadi pada pencarian dengan metode ini adalah sebagai berikut :

- Membaca Array data
- Apabila Array belum terurut maka array diurutkan terlebih dahulu.
- Menentukan data yang akan dicari
- Menentukan elemen tengah dari array
- Jika nilai elemen tengah sama dengan data yang dicari, maka pencarian berhenti.
- Jika elemen tengah tidak sama dengan data yang dicari maka :
- Jika nilai elemen tengah $>$ data yang dicari maka pencarian dilakukan pada setengah array pertama.
- Jika nilai elemen tengah lebih kecil dari pada data yang dicari maka pencarian dilakukan pada setengah array berikutnya.

2.2.21. Unified Modeling Language (UML)


Unified Modeling Language (UML) adalah salah satu standar bahasa yang banyak digunakan di dunia industri untuk mendefinisikan requirement, membuat analisis dan desain, serta menggambarkan arsitektur dalam pemrograman berorientasi objek (Rosa dan Shalahuddin, 2018:133).




a. *Use Case Diagram*

Use case diagram merupakan pemodelan untuk melakukan (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat secara kasar yang digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi itu. (Rosa dan Shalahuddin, 2018:155).

Simbol Simbol yang digunakan pada use case diagram bisa dilihat pada tabel 1.1.

Table 1.1 Simbol Use Case Diagram

NO	Simbol	Nama	Keterangan
1		<i>Actor</i>	Menspesifikasikan himpunan peran yang pengguna mainkan ketika berinteraksi dengan use case.
2		<i>Use Case</i>	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu actor
3		<i>Association</i>	Apa yang menghubungkan antara objek satu dengan objek lainnya.





4		<i>Extend</i>	Menspesifikasikan bahwa use case target memperluas perilaku dari use case sumber pada suatu titik yang diberikan
5		<i>Generalization</i>	Hubungan dimana objek anak (<i>descendent</i>) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (<i>ancestor</i>).
6		<i>Include</i>	Menspesifikasikan bahwa use case sumber secara eksplisit.


b. Activity Diagram

Activity Diagram menggambarkan workflow atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak yang perlu diperhatikan disini adalah bahwa diagram aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan aktor jadi aktivitas yang dapat dilakukan oleh sistem. (Rosa dan Shalahuddin, 2018:161).

Simbol Simbol yang digunakan pada *Activity diagram* bisa dilihat pada tabel 1.2.

Table 1.2 Simbol Activity Diagram

NO	Simbol	Nama	Keterangan
1		Status Awal	Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja
2		Aktivitas	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu actor
3		Percabangan	Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu
4		Penggabungan	Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu


5		Status Akhir	Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir
---	---	--------------	---


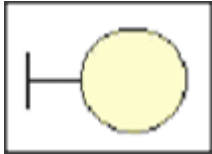


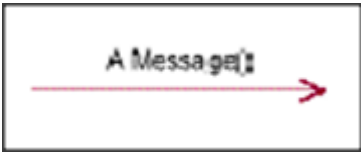
c. *Sequence Diagram*

Sequence Diagram menggambarkan kelakuan objek pada use case dengan mendeskripsikan waktu hidup objek dan message yang dikirimkan dan diterima antar objek. Oleh karena itu untuk menggambar diagram sekuen maka harus diketahui objek-objek yang terlibat dalam sebuah use case beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek itu. membuat diagram sekuen juga dibutuhkan untuk melihat skenario yang ada pada use case. (Rosa dan Shalahuddin, 2018:165).

Simbol Simbol yang digunakan pada use case diagram bisa dilihat pada tabel 2.3.

Table 2.3 Simbol Sequence Diagram

Simbol	Deskripsi
Actor 	Menggambarkan orang yang berinteraksi dengan sistem.

Entity Class		Menggambarkan hubungan kegiatan yang akan dilakukan.
Boundary Class		Menggambarkan sebuah penggambaran dari form.
Control Class		Menggambarkan penghubung antara boundary dengan tabel.
Lifeline		Menggambarkan tempat mulai dan berakhirnya sebuah pesan.
Line Message		Menggambarkan pengiriman pesan.

2.2.22. Creately

Creately merupakan aplikasi yang memungkinkan pengguna untuk membuat diagram secara online serta menyediakan fasilitas bagi para penggunanya untuk bekerjasama secara kolaboratif. Creately merupakan aplikasi asal Australia yang dikembangkan oleh Cinergix.

Creately menyediakan layanan, dimana pengguna bisa membuat diagram secara online, Anda bisa membuat *flowcharts*, *wireframes*, *mind*

maps, UML serta berbagai diagram lain. Selain itu pengguna juga bisa melakukan kerja kolaborasi dengan mengundang pengguna lain, dengan fasilitas satu klik pengguna juga bisa mengajak tim kerja atau klien berada pada satu halaman kerja yang sama.

2.2.23. Hosting

Hosting adalah layanan web yang berfungsi untuk menyimpan file website dan database pada server agar sebuah website bisa diakses. Setelah website dan database sudah berada di dalam server, website bisa diakses dimana saja secara *online*.

2.2.24. Domain

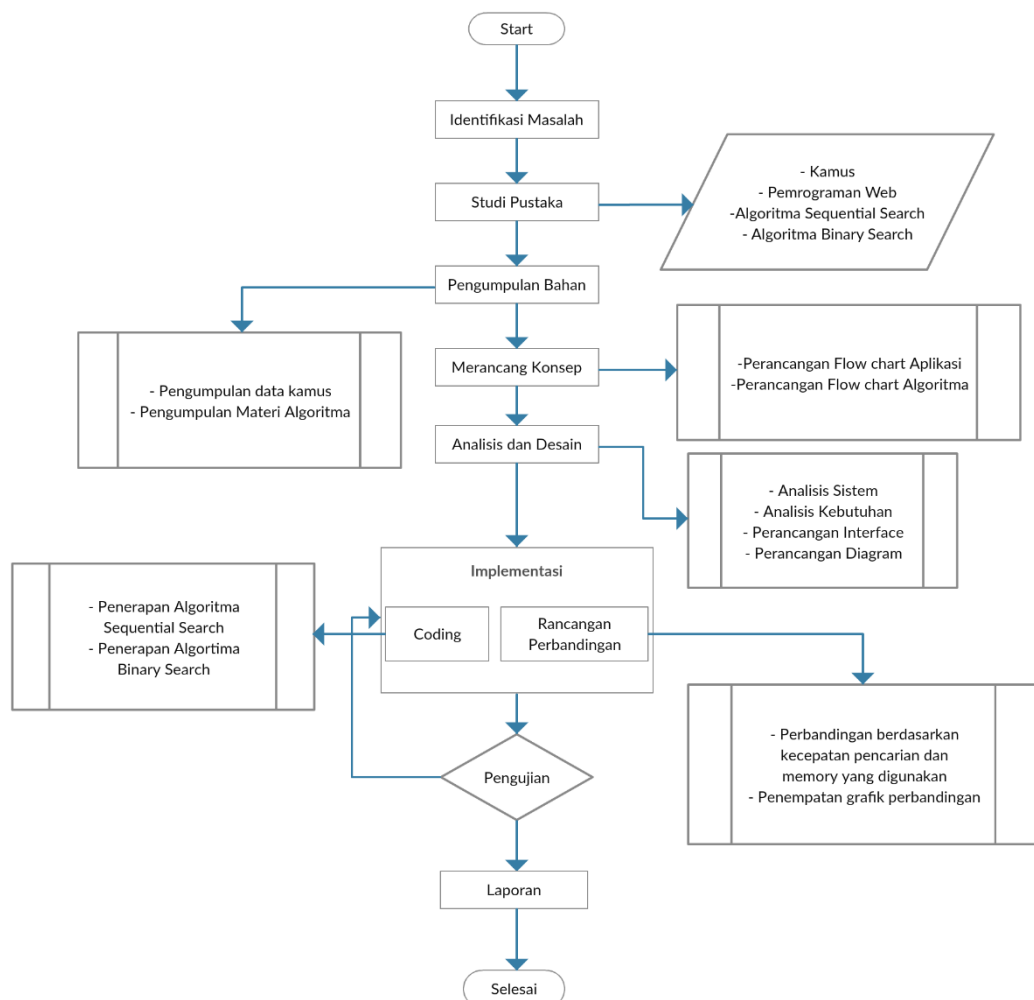
Domain adalah sebuah nama unik yang diberikan pada suatu server untuk mengidentifikasi nama server tersebut seperti halnya pada web server atau mail server.

BAB III

METODOLOGI PENELITIAN

3.1 Kerangka Pikir

Pada pembuatan prototype Aplikasi Kamus Bahasa Indonesia ini dibutuhkan beberapa tahapan yang harus dilakukan untuk dapat membandingkan algoritma *Sequential Search* dan *Binary Search* secara tepat. Untuk penjelasan kerangka pikir terdapat pada halaman selanjutnya.



Gambar 3.1 Kerangka Pikir

3.2 Deskripsi

3.2.1. Identifikasi Masalah

Pada setiap Masalah apabila telah ditangani dengan cara merumuskannya dengan lebih baik lagi, maka hal tersebut bukan hanya bisa membantu pikiran kita jadi lebih fokus, melainkan bisa lebih menggiring pola pikir kita terhadap suatu masalah yang akan dipecahkan. Dengan itu identifikasi masalah sangat lah penting pada sebuah penelitian.

Pada tahapan ini pun beberapa masalah yang diambil yaitu :

1. Membandingkan algoritma *Sequential Search* dan *Binary Search* pada aplikasi kamus Bahasa Indonesia.
2. Memahami masing-masing cara kerja algoritma.
3. Mencari tau algoritma mana yang secara penerapannya ke dalam aplikasi kamus Bahasa Indonesia lebih baik.

Jika tahapan ini sudah tepat, dilanjutkan ke tahapan selanjutnya yaitu studi pustaka.

3.2.2. Studi Pustaka

Studi pustaka dilakukan untuk mendapatkan informasi yang relevan dengan topik atau masalah yang akan diteliti. Informasi itu dapat diperoleh dari buku, karangan-karangan ilmiah, dan jurnal dari penelitian sebelumnya. Penulis menggunakan 5 buku yang terdiri dari buku pemrograman web, buku membuat website cantik & menarik dengan jQuery, buku algoritma dan pemrograman dalam Bahasa pascal, C dan C++, buku bootstrap CSS *Framework* dan buku rekayasa perangkat lunak terstruktur dan berorientasi objek serta menggunakan 3 jurnal yang terdiri dari jurnal aplikasi kamus bahasa betawi berbasis android menggunakan metode *Sequential Search*, jurnal analisis algoritma *Sequential Search* dan *Binary Search* pada big data dan jurnal implementasi algoritma *Binary Search* pada aplikasi konkordansi al-qur'an berbasis android sebagai referensi untuk penelitian yang dilakukan.

3.2.3. Pengumpulan Bahan

Pengumpulan bahan dilakukan untuk memperoleh informasi yang dibutuhkan dalam rangka mencapai tujuan penelitian. Bahan penelitian yang akan dikumpulkan antara lain data kamus dan materi algoritma. Kamus adalah buku yang berisi kumpulan kata beserta penjelasannya, Untuk pembuatan aplikasi kamus Bahasa Indonesia maka dibutuhkan data kata Bahasa Indonesia sebagai bahan untuk diterapkan ke dalam aplikasi. Setelah data kata pada kamus, dibutuhkan juga bahan materi algoritma untuk mengetahui masing-masing alur pencarian algoritma dalam menemukan sebuah kata yang akan diterapkan pada aplikasi.

3.2.4. Merancang Konsep

Setelah mendapat informasi dari tahapan sebelumnya, selanjutnya membuat rancangan konsep dimulai dari menyusun alur sistem dalam bentuk *Flowchart*. *Flowchart* bertujuan untuk memecah dan menganalisis langkah-langkah yang akan dilakukan selanjutnya dalam prosedur suatu sistem. *Flowchart* yang akan dibuat antara lain *flowchart* sistem, *flowchart* algoritma *Sequential Search* dan *flow chart* algoritma *Binary Search*.

3.2.5. Desain

Setelah mengetahui alur aplikasi hasil dari pembuatan *Flowchart*, selanjutnya adalah tahapan pembuatan desain aplikasi. Desain bertujuan untuk merancang bagaimana suatu website atau aplikasi yang dibuat terlihat seperti apa dan menerapkan hasil alur *flowchart* yang sudah dibuat. Desain yang dibuat antara lain desain *interface* pencarian kata pada kamus, desain pencarian kata didalam file pada kamus, desain perbandingan dan desain diagram. Desain *interface* adalah gambaran dari sebuah website atau aplikasi untuk memastikan bagaimana seorang user berinteraksi dengan aplikasi atau website tersebut serta bagaimana informasi ditampilkan di dalam sebuah website atau aplikasinya. Sedangkan desain diagram adalah gambaran alur kerja sebuah sistem yang akan di buat, dengan menggunakan

UML maka gambaran secara garis besar sebuah sistem yang akan dibuat dapat direncanakan. Dalam penelitian ini pembuatan desain *interface* dan diagram menggunakan *creately*.

3.2.6. Implementasi

Tahapan selanjutnya adalah implementasi, tahapan ini dilakukan setelah mengetahui alur jalannya aplikasi dari pembuatan *flowchart* dan desain. Pada tahap ini terdapat dua bagian, yang pertama adalah *coding* aplikasi dan yang kedua adalah melakukan perbandingan algoritma. Pada bagian *coding* yaitu mentransformasikan hasil *flowchart* dan pembuatan desain ke dalam Bahasa pemrograman agar dimengerti oleh mesin (komputer). Pada bagian rancangan perbandingan yaitu membuat fitur perbandingan didalam aplikasi agar aplikasi bisa menganalisis dan membandingkan hasil dari uji coba algoritma yang sudah diterjemahkan kedalam Bahasa pemrograman. Pada tahap ini dilakukan pembuatan *prototype* sebagai bahan uji coba untuk memastikan alur benar-benar bekerja sesuai dengan rancangan. Hasil coding dan rancangan perbandingan akan dihubungkan pada aplikasi agar perbandingan algoritma bisa ditampilkan dalam bentuk *chart*, sehingga lebih mudah untuk melihat perbedaan dari masing-masing algoritma dari segi kecepatan dan *memory* yang digunakan. Setelah itu aplikasi disimpan ke dalam suatu hosting agar bisa diakses secara online. Tahap ini juga dapat disebut tahap evaluasi untuk pengembangan aplikasi yang sudah jadi supaya menjadi lebih baik. Hasil evaluasi ini dapat digunakan sebagai masukan untuk tahap konsep pada aplikasi selanjutnya.

3.2.7. Pengujian

Pada tahapan ini dilakukan pengujian aplikasi yang telah dibuat untuk memastikan apakah hasil aplikasi sudah sesuai dengan rancangan yang diharapkan. Jika masih ada kekurangan atau kesalahan maka kembali

ke tahap pembuatan aplikasi untuk diperbaiki sampai benar-benar sesuai rancangan.

3.2.8. Pembuatan Laporan

Tahapan terakhir adalah pembuatan laporan sebagai salah satu persyaratan kelulusan. Laporan disusun sesuai dengan ketentuan yang tercantum dalam Pedoman Penulisan Skripsi Fakultas Teknologi Informasi Universitas Bale Bandung.

BAB IV

ANALISIS DAN PERANCANGAN

4.1. Analisis

Analisis dilakukan sebagai langkah awal peneliti untuk mengetahui kebutuhan-kebutuhan yang diperlukan pengguna. Dalam analisis ini penulis mempunyai instrumen penelitian dan melakukan beberapa tahapan yaitu, analisis sistem, analisis kebutuhan, *user interface*, fitur-fitur dan hasil analisis.

4.1.1. Instrumen Penelitian

Instrumen penelitian pada penelitian ini terdiri dari perangkat lunak, perangkat keras dan observasi pada studi pustaka.

A. Perangkat Lunak

Penulis menggunakan beberapa perangkat lunak pada penelitian kali ini, yaitu :

1. Menggunakan Microsoft Windows 10 Home Single Language 64-bit
2. Menggunakan XAMPP sebagai web server yang berdiri sendiri (localhost)
3. *Code editor* menggunakan Sublime Text versi 3
4. Google Chrome untuk melihat hasil *compile* pembuatan web

B. Perangkat Keras

Penulis menggunakan perangkat keras dengan spesifikasi berikut :

Table 4.1 Instrumen Penelitian Perangkat Keras

Perangkat Keras Laptop

Spesifikasi	Deskripsi
Tipe Laptop	Lenovo Ideapad 100
Prosesor	Intel Core i3-5005U 2 GHz
RAM	4GB
HDD	500GB

C. Server

Penulis menggunakan server hosting pada penelitian kali ini, dengan spesifikasi berikut :

Table 4.2 Server Hosting

Server Hosting

Spesifikasi	Deskripsi
Core	1
RAM	768 MB
SSD	2 GB
OS	Linux
Apache Version	2.4.43
PHP Version	7.3.17

D. Observasi

Observasi yang dilakukan oleh penulis dalam penelitian ini yaitu observasi pada penelitian sebelumnya yang membahas topik serupa dengan penelitian ini yang mencakup tentang aplikasi kamus, serta membahas tentang implementasi menggunakan algoritma *Sequential Search* dan *Binary Search*.

4.1.2. Analisis Sistem

Analisis sistem adalah tahapan yang memberikan gambaran tentang sistem yang akan berjalan nantinya. Dalam perbandingan algoritma *Sequential Search* dan *Binary Search* pada aplikasi kamus Bahasa Indonesia menggunakan PHP dan jQuery, untuk mempermudah melihat perbandingan dari kedua algoritma tersebut, dibutuhkan kebutuhan sistem yang meliputi:

a. *Input*

User membuka aplikasi, selanjutnya aplikasi menampilkan halaman awal aplikasi

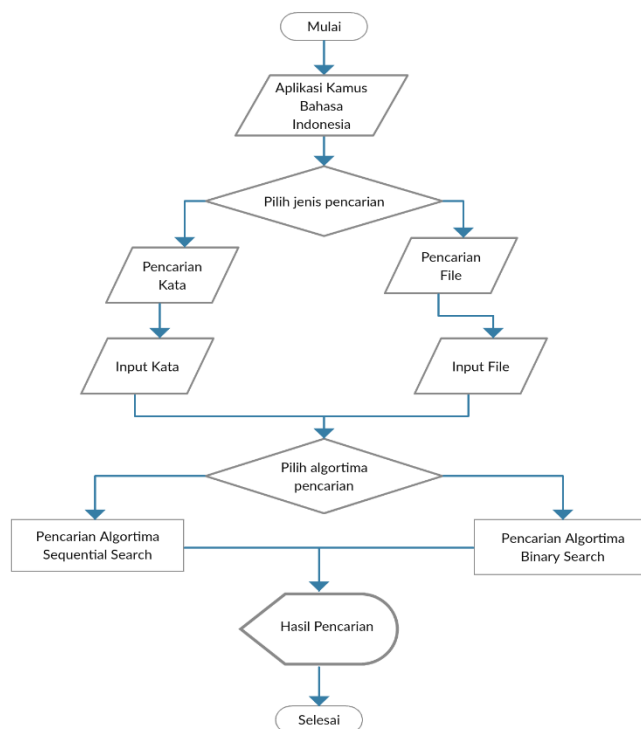
b. *Proses*

User memilih algoritma *Sequential Search* atau *Binary Search* untuk melakukan pencarian kata

c. *Output*

Menampilkan hasil pencarian kata, beserta *memory* yang digunakan pada saat pencarian dan kecepatan pencarian

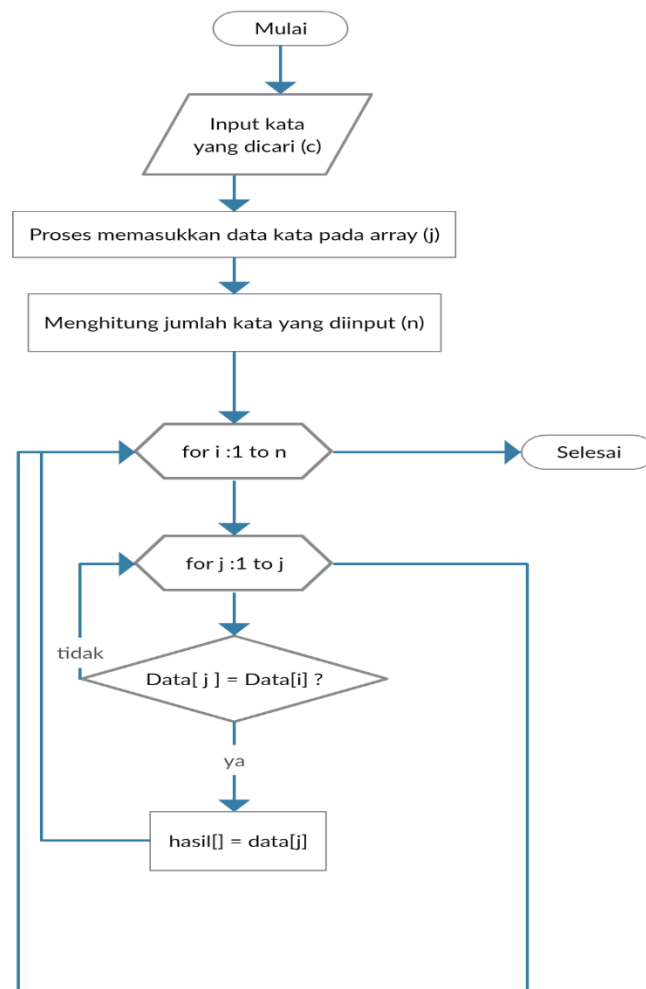
Berikut adalah *flow chart* sistem yang akan dibuat :



Gambar 4.2 *Flowchart* Sistem

Pada gambar 4.1 menjelaskan tentang alur kerja sistem yang nantinya akan diterapkan pada Aplikasi Kamus Bahasa Indonesia untuk membandingkan kedua algoritma.

Berikut adalah *Flowchart* dari algoritma *Sequential Search*



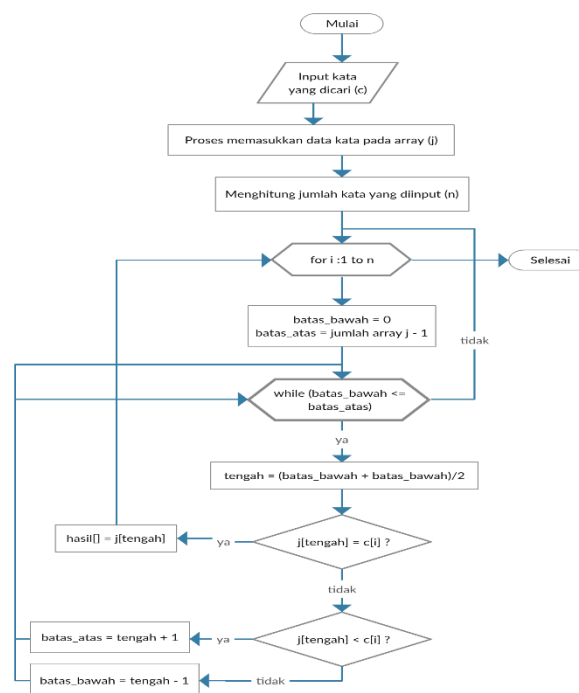
Gambar 4.2 *Flowchart* algoritma *Sequential Search*

Cara kerja algoritma *Sequential Search* yaitu menelusuri elemen-elemen array dengan membandingkan satu per satu secara berurutan dengan data yang dicari sampai data tersebut ditemukan atau tidak ditemukan, dimana data-data tersebut tidak perlu diurutkan terlebih dahulu.

Penjelasan dari gambar 4.2 :

1. Memasukkan semua kata yang terdapat didalam database kedalam sebuah array j
2. Memasukkan kata yang diinput kedalam array c
3. Menghitung jumlah kata yang ada didalam array c
4. Mulai perulangan i berdasarkan jumlah array c
5. Mulai perulangan j berdasarkan jumlah array j
6. Didalam perulangan pertama sistem melakukan pengecekan apakah kata yang didalam array c sama dengan yang kata ada didalam array j ($kata[i] = kata[j]$)
7. Jika ada kata yang ditemukan maka perulangan dan proses pencarian berhenti
8. Jika tidak ada, dilanjutkan ke perulangan kedua, prosesnya sama seperti no 6
9. Jika setelah melakukan pengecekan tidak ada yang ditemukan maka perulangan dan proses pencarian berhenti

Berikut adalah *Flow Chart* dari algoritma *Binary Search*



Gambar 4.3 *Flowchart* algoritma *Binary Search*

Cara kerja algoritma *Binary Search* yaitu dengan membagi dua elemen penampung nilai dan membandingkan nilainya. Proses pencarian *Binary Search* hanya dapat dilakukan pada data yang berurutan.

Penjelasan pada gambar 4.3 :

1. Memasukkan semua kata yang terdapat didalam database kedalam sebuah array j
2. Memasukkan kata yang diinput kedalam array c
3. Menghitung jumlah kata yang ada didalam array c
4. Mulai perulangan i berdasarkan jumlah array c
5. Menentukan batas atas dan batas bawah
6. Mulai perulangan jika batas bawah lebih kecil atau sama dengan batas atas
7. Hitung posisi tengah $(\text{batas bawah} + \text{batas atas}) / 2$
8. Bandingkan posisi tengah dengan kata yang dicari
9. Jika kata yang dicari sama maka perulangan dan proses pencarian berhenti
10. Jika lebih besar maka akan dilakukan pencarian kembali ke bagian kiri dengan nilai batas bawah = posisi tengah + 1 dan batas atas tetap kemudian ulangi mulai poin 7
11. Jika nilai datanya lebih kecil maka akan dilakukan pencarian kembali ke bagian kiri dengan nilai posisi awal tetap dan nilai posisi akhir = posisi tengah – 1 kemudian ulangi mulai poin 7

4.1.3. Analisis Kebutuhan

Pada tahap ini yaitu menyiapkan kebutuhan-kebutuhan dari semua elemen sistem perangkat lunak yang akan di bangun. Pada tahap ini dibentuk kebutuhan perangkat lunak dan fungsi perangkat lunak yang dibutuhkan.

1. Kebutuhan Software

Berikut *software* yang digunakan dalam pembuatan aplikasi , antara lain :

- a) XAMPP digunakan sebagai web server yang berdiri sendiri (*localhost*)
- b) *Framework* Bootstrap digunakan untuk mempercepat dan mempermudah pembuatan tampilan website agar bisa di buka secara *responsive* sehingga dapat mendukung untuk segala jenis resolusi, baik itu tablet, smartphone ataupun juga PC dan laptop
- c) *Library* jQuery digunakan untuk memudahkan penulisan kode javascript
- d) Morris.js digunakan untuk menampilkan sebuah grafik perbandingan algoritma

2. Kebutuhan Antar Muka

Kebutuhan antar muka pada pembuatan aplikasi ini sebagai berikut :

- a) Aplikasi harus mampu mencari sebuah input kata dan kata yang didalam file pada proses pencarian menggunakan algoritma *Sequential Search* dan *Binary Search*
- b) Aplikasi harus mampu menampilkan hasil dari sebuah kata beserta definisinya
- c) Aplikasi harus mampu menampilkan sebuah kecepatan pencarian dan *memory* yang digunakan pencarian dan sebagai bahan yang akan dibandingkan dari kedua algoritma
- d) Aplikasi dapat menyimpan *history* pencarian
- e) Aplikasi dapat menampilkan analisis perbandingan kedua algoritma

3. Kebutuhan Data

Data yang diolah pada aplikasi ini antara lain :

- a) Data kamus pada table kbbs
- b) Data perbandingan pencarian yang berupa hasil kata yang dicari dan definisinya, beserta waktu pencarian dan *memory* yang digunakan pencarian pada table perbandingan

4. Kebutuhan Fungsional

Penjelasan secara rinci dari setiap fungsi pada aplikasi. Fungsi-fungsi yang dimiliki aplikasi adalah :

- a) Saat user memilih pencarian kata, selanjutnya user memilih algoritma pencarian yang digunakan untuk proses pencarian, selanjutnya proses algoritma berjalan untuk mencari kata dan menghasilkan kata yang ditemukan beserta waktu pencarian dan *memory* yang digunakan pencarian
- b) Saat user memilih pencarian file, selanjutnya user memilih algoritma pencarian yang digunakan untuk proses pencarian, selanjutnya proses algoritma berjalan untuk mencari kata dan menghasilkan kata yang ditemukan beserta jumlah kata yang ditemukan, waktu pencarian dan, *memory* yang digunakan pencarian
- c) Jika kata ditemukan maka kata tersebut ketika di klik akan menampilkan definisinya
- d) Ketika user sudah mencoba kedua algoritma, user bisa klik tombol analisis untuk menampilkan perbandingan dari kedua algoritma tersebut

5. Kebutuhan Non Fungsional

Adapun kebutuhan non fungsional yang terdapat didalam aplikasi sebagai berikut :

- a) Sistem dapat dijalankan oleh beberapa *software* web browser diantaranya Microsoft Edge, Google Chrome dan Mozilla Firefox.
- b) Sistem dapat menggunakan fitur *dark mode* agar mata nyaman saat mengakses aplikasi
- c) Besarnya program dari sistem maksimal sebesar 100 MB
- d) Sistem memiliki tampilan (antar muka) yang mudah dipahami.

4.1.4. Hasil Analisis

Hasil analisis yang didapat dalam penelitian ini yaitu pencarian kata pada aplikasi kamus Bahasa Indonesia yang dapat membandingkan performa dari algoritma *Sequential Search* dan *Binary Search* dalam bentuk analisis yang terdapat pada aplikasi. Pada aplikasi kamus Bahasa Indonesia ini dapat mencari sebuah kata yang diinput dan sebuah kata yang terdapat didalam file dalam bentuk pdf, sebagai uji coba pada kedua algoritma. Proses pencarian kata dan file dilakukan secara *realtime* agar proses pencarian dan hasil yang didapat langsung diterima secara langsung oleh user tanpa reload halaman website. Untuk menunjang tercapainya penelitian ini maka pembuatan aplikasi ini menggunakan beberapa *library* dan *framework* yang terdiri dari Bootstrap, jQuery dan Morris.js.

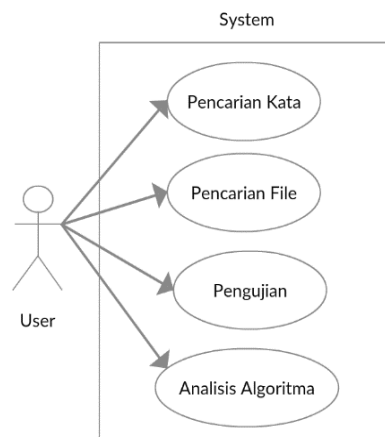
4.2. Perancangan

Sebelum masuk dalam pembuatan aplikasi dibutuhkan perancangan yaitu merancang *software* dalam bentuk UML yang terdiri dari *use case*, *activity diagram*, *sequence diagram* dan *class diagram*, desain keseluruhan dan struktur tabel aplikasi .

4.2.1. UML

1. *Usecase diagram*

Usecase diagram menggambarkan sebuah interaksi antara satu atau lebih aktor dengan sistem yang akan dibuat untuk mengetahui fungsi apa saja yang ada didalam sistem dan siapa yang menggunakan fungsi tersebut. Berikut adalah usecase pada aplikasi Kamus Bahasa Indonesia



Gambar 4.4 *Usecase Diagram* Kamus Bahasa Indonesia

Table 4.3 Aktor Usecase

Aktor	Deskripsi
User	Pengguna disini merupakan orang yang melakukan pencarian pada aplikasi kamus menggunakan algoritma dan dapat melihat hasil perbandingan dari kedua algoritma tersebut

Penjelasan usecase kamus Bahasa Indonesia sebagai berikut :

Table 4.4 Deskripsi Usecase Kamus Bahasa Indonesia

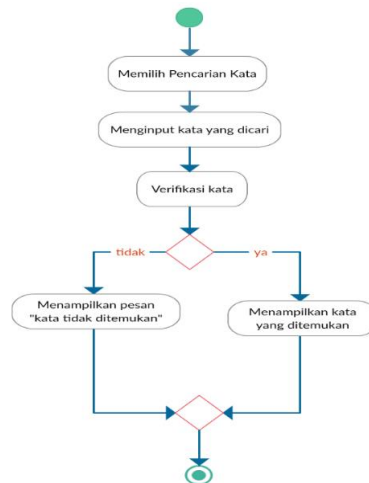
No	Usecase	Deskripsi
1	Pencarian Kata	Proses dimana pengguna melakukan pencarian kata dengan memilih salah satu algoritma sebagai algoritma pencariannya
2	Pencarian File	Proses dimana pengguna melakukan pencarian kata didalam file dengan memilih salah satu algoritma sebagai algoritma pencariannya
3	Pengujian	Proses pengujian pada kedua algoritma dengan melakukan beberapa kali percobaan pencarian
4	Analisis Algoritma	Proses menampilkan analisis perbandingan algoritma dari hasil pencarian yang sudah dilakukan oleh pengguna

2. Activity Diagram

Activity Diagram menggambarkan urutan aktifitas proses pada sebuah sistem. Berikut adalah *activity* diagram pada aplikasi Kamus Bahasa Indonesia :

A. Activity Pencarian Kata

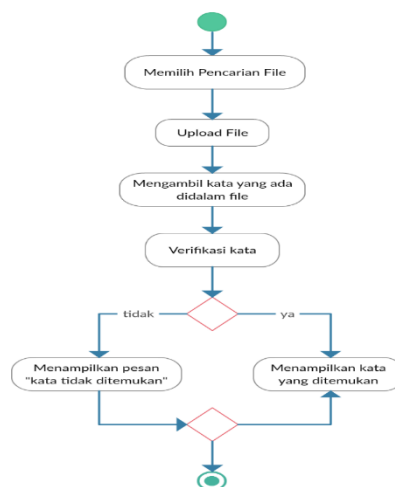
Pertama pengguna memilih algoritma pencarian yang akan digunakan. Selanjutnya pengguna menginput sebuah kata yang ingin dicari lalu sistem akan mengecek apakah data yang dicari ada didalam database, jika ada maka sistem akan menampilkan kata dan definisinya beserta waktu pencarian, memory yang digunakan dan langkah pencarian.



Gambar 4.5 *Activity Diagram* Pencarian Kata

B. *Activity Diagram* Pencarian kata lewat file

Pertama pengguna memilih algoritma pencarian yang akan digunakan. Selanjutnya pengguna mengupload sebuah file berformat pdf, setelah file diupload lalu sistem akan mengambil sebuah kata yang didalam file kemudian sistem melakukan pengecekan apakah kata yang ada didalam database sama dengan yang ada didalam file, jika ada maka sistem akan menampilkan kata dan definisinya beserta waktu pencarian dan memory yang digunakan.



Gambar 4.6 *Activity Diagram* Pencarian kata lewat file

C. Activity Diagram Pengujian

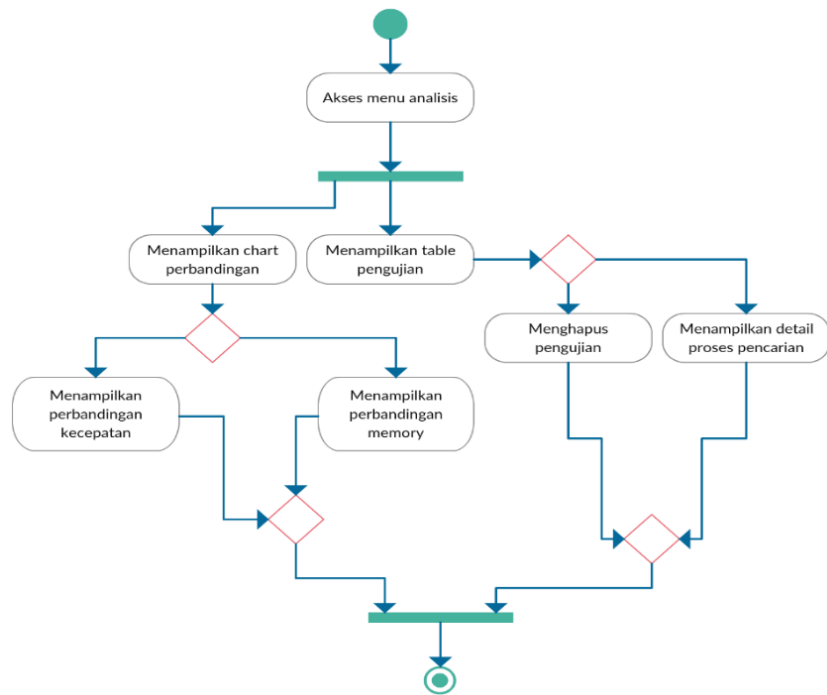
Untuk menguji kedua algoritma, user lain bisa ikut berkontribusi pada pengujian algoritma yang terdapat pada aplikasi dengan masuk ke halaman kontribusi, lalu mengisi data diri setelah itu mengikuti setiap pengujian sampai selesai.



Gambar 4.7 Activity Diagram Pengujian

D. Activity Diagram Analisis

Setelah melakukan pencarian kata atau file dan melakukan pengujian maka didapat data-data hasil pencarian sebelumnya yang dapat dianalisis untuk membandingkan kedua algoritma dari kecepatan dan memory yang digunakan pada pencarian, Pada saat akses halaman analisis, akan langsung menampilkan chart pada semua pengujian. User juga bisa melihat analisis berdasarkan kecepatan atau memory yang digunakan



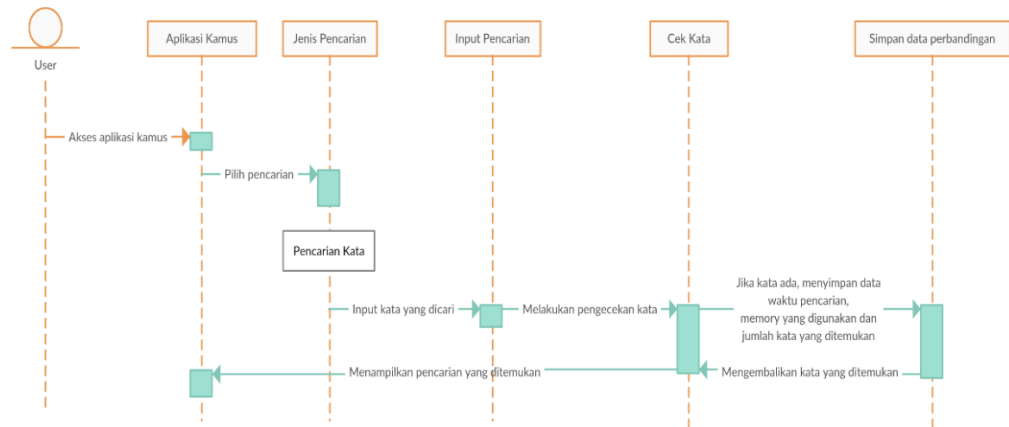
Gambar 4.8 Activity Diagram Analisis

3. *Sequence Diagram*

Sequence diagram digunakan untuk menggambarkan perilaku pada sebuah skenario dan mendeskripsikan bagaimana entitas dan sistem berinteraksi, termasuk pesan yang digunakan saat interaksi.

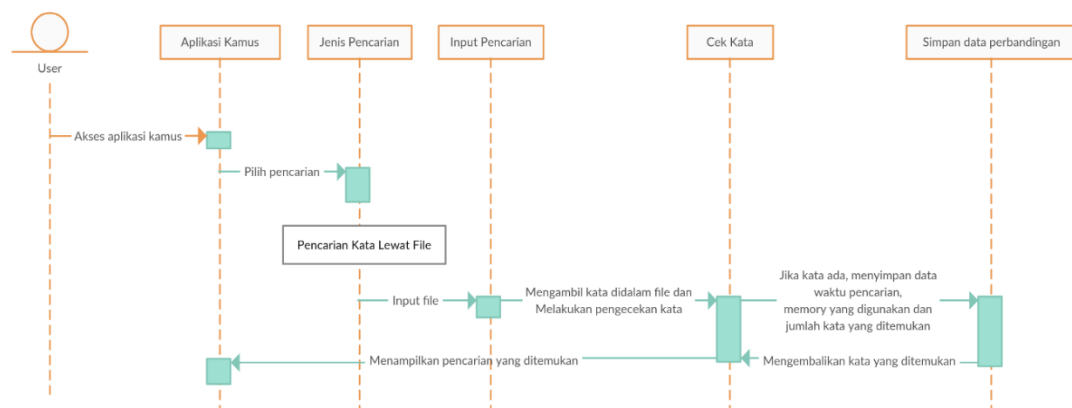
A. *Sequence Diagram User*

Pertama user mengakses aplikasi kamus, lalu memilih jenis pencarian, jika memilih jenis pencarian kata lalu tampil form untuk mencari kata, setelah itu sistem akan melakukan pengecekan apakah kata yang dicari ada atau tidak, jika kata ditemukan maka sistem menyimpan riwayat kata yang dicari berupa kata yang dicari, kecepatan pencarian, memory yang digunakan dan jumlah kata yang ditemukan, setelah itu sistem menampilkan hasil pencarian yang ditemukan beserta kecepatan pencarian dan memory yang digunakan.



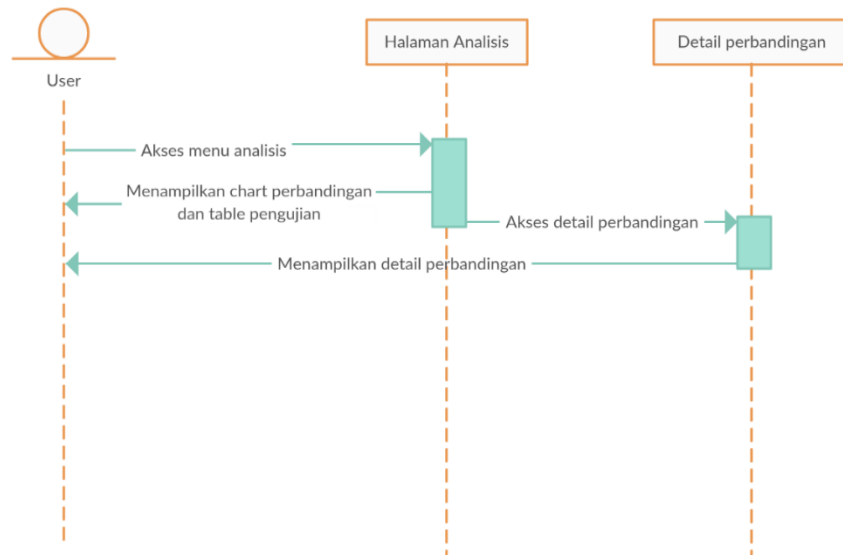
Gambar 4.9 Sequence Diagram user pencarian kata

Jika memilih pencarian kata lewat file, sistem mengambil kata yang terdapat didalam file dan menghilangkan simbol yang terdapat pada file untuk memudahkan pencarian kata. Setelah kata didalam file didapat, sistem melakukan pengecekan apakah kata yang terdapat didalam file ada atau tidak, jika kata ditemukan maka sistem menyimpan riwayat kata yang dicari berupa kata yang dicari, kecepatan pencarian, memory yang digunakan dan jumlah kata yang ditemukan, setelah itu sistem menampilkan hasil pencarian yang ditemukan beserta kecepatan pencarian dan memory yang digunakan.



Gambar 4.10 Sequence Diagram user pencarian kata lewat file

Pada menu analisis, user akan melihat chart perbandingan, table pengujian dan detail proses pencarian jika memang ingin ditampilkan



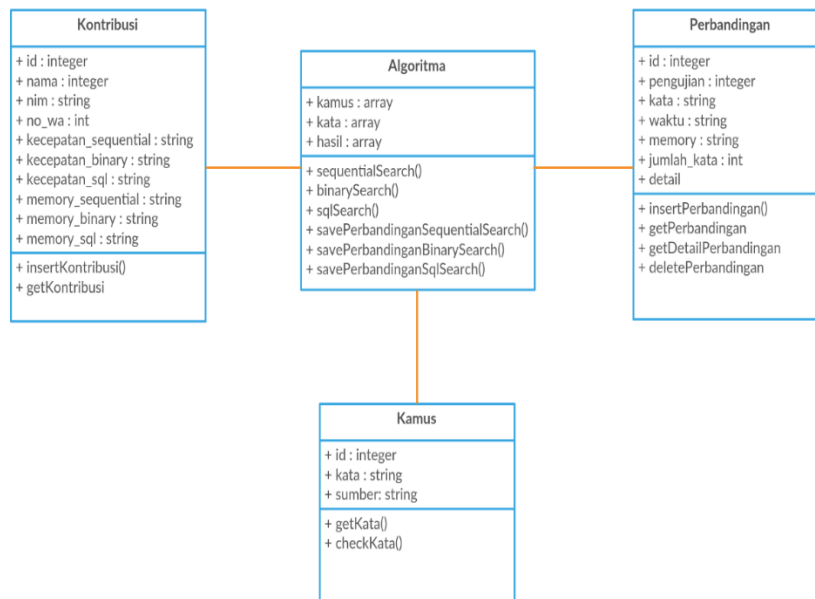
Gambar 4.11 Sequence Diagram analisis

4. Class Diagram

Class diagram adalah diagram yang menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem.

Pada aplikasi kamus terdapat 4 elemen penting, yaitu kamus, algoritma dan kontribusi dan perbandingan. Pencarian kata pada kamus menggunakan 2 algoritma dan hasil pencarian dengan menggunakan algoritma disimpan di table perbandingan untuk melihat perbedaan dari masing-masing algoritma.

Berikut adalah *class* diagram pada aplikasi kamus



Gambar 4.12 Class Diagram Aplikasi Kamus

4.2.2. Perancangan User Interface

Perancangan untuk aplikasi yaitu terdapat beberapa fungsi-fungsi link yang ada di dalamnya, adapun link-link tersebut adalah sebagai berikut;

1. Home

Tampilan utama aplikasi yang berisi pencarian kamus. Terdapat pencarian kata dan pencarian kata didalam file dengan mengupload file berformat pdf.

2. Analisis Penyusun

Tampilan hasil pengujian pencarian kata oleh penulis berupa grafik pengujian dan tabel pengujian.

3. Analisis Kontribusi

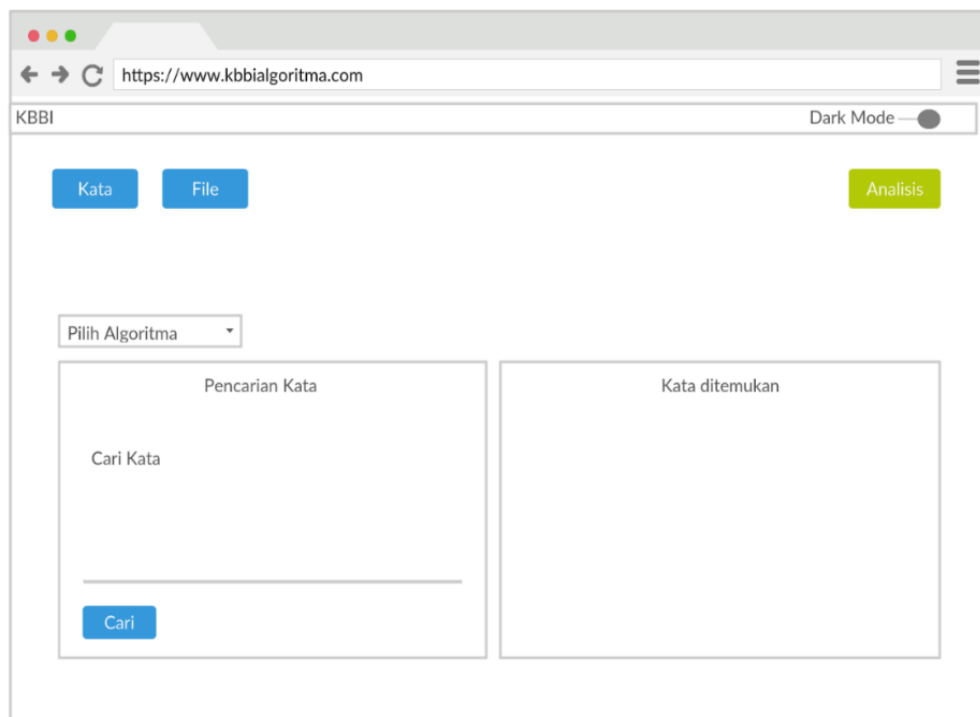
Tampilan hasil pengujian pencarian kata oleh mahasiswa yang ikut berkontribusi pada penelitian ini berupa grafik pengujian dan tabel pengujian.

4. Kontribusi Penelitian

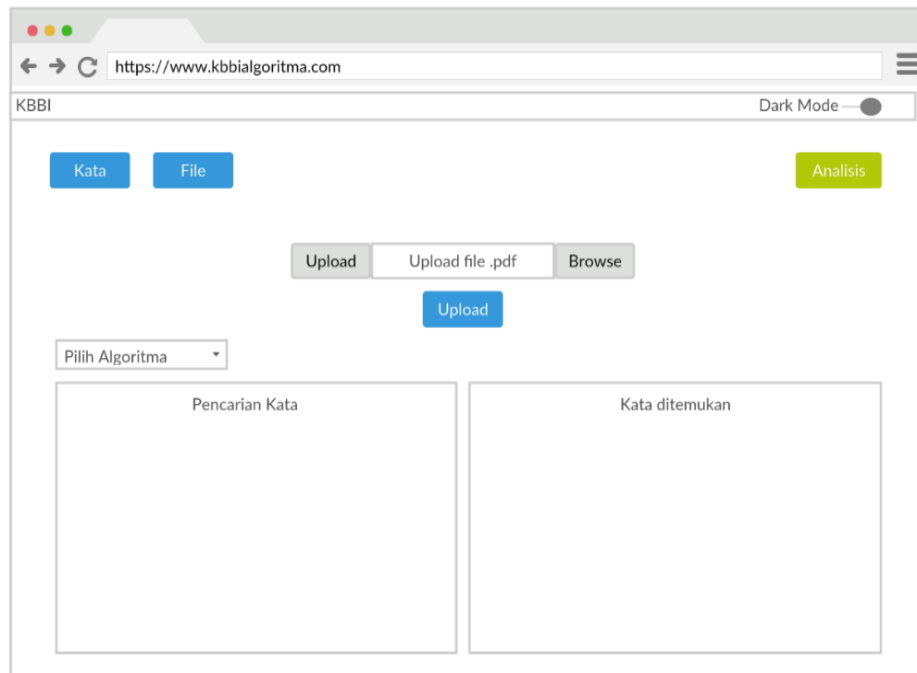
Tampilan untuk mahasiswa yang ikut berkontribusi pada penelitian ini dengan mengisi form data mahasiswa dan mengikuti pengujian pada aplikasi.

4.2.3. *User Interface*

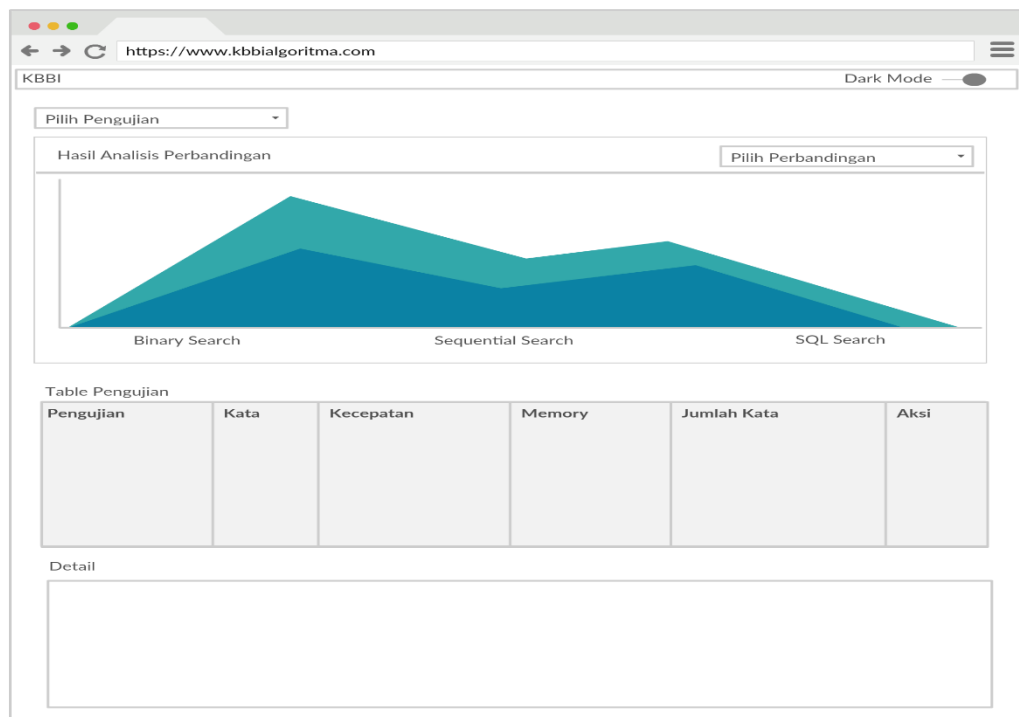
User Interface dari aplikasi kamus Bahasa Indonesia ini adalah tampilan dari input sebuah kata sampai output kata yang ditemukan dan tampilan analisis untuk melihat perbandingan kedua algoritma. *User Interface* disesuaikan dari kebutuhan dalam penelitian.



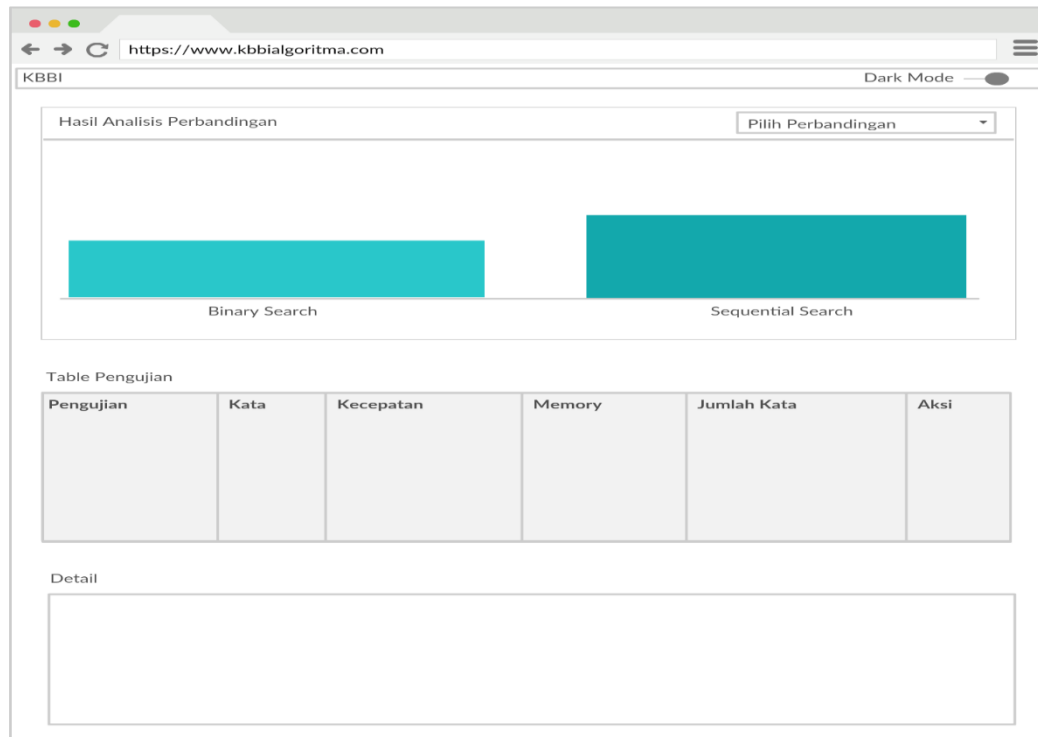
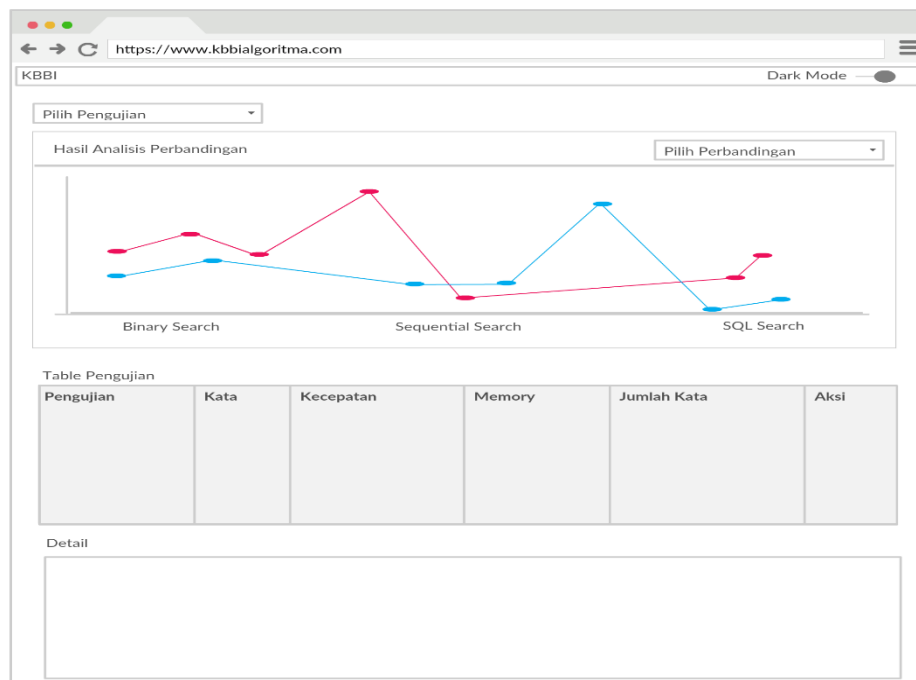
Gambar 4.13 User Interface pencarian kata

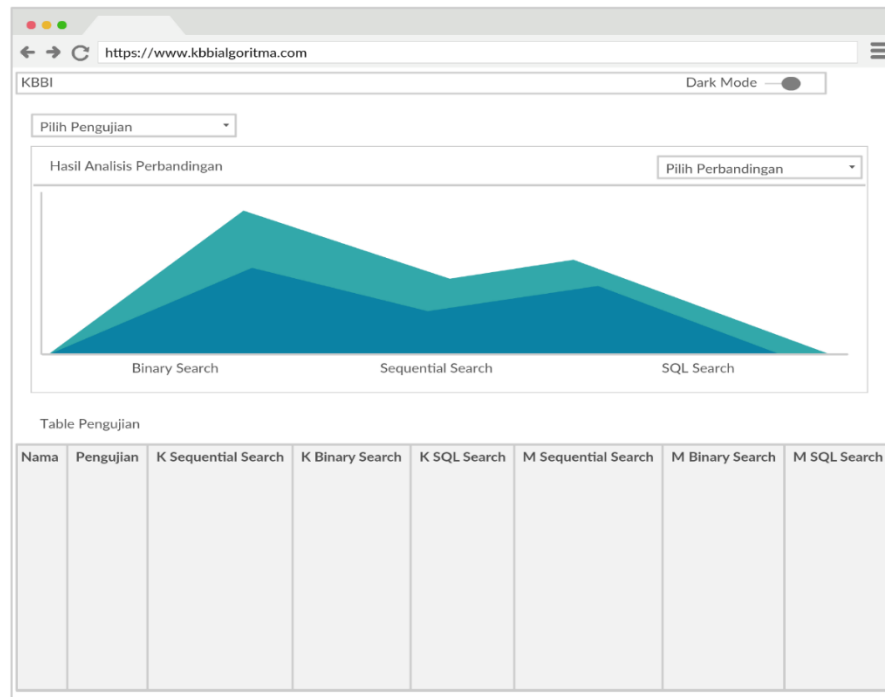


Gambar 4.14 User Interface pencarian kata lewat file

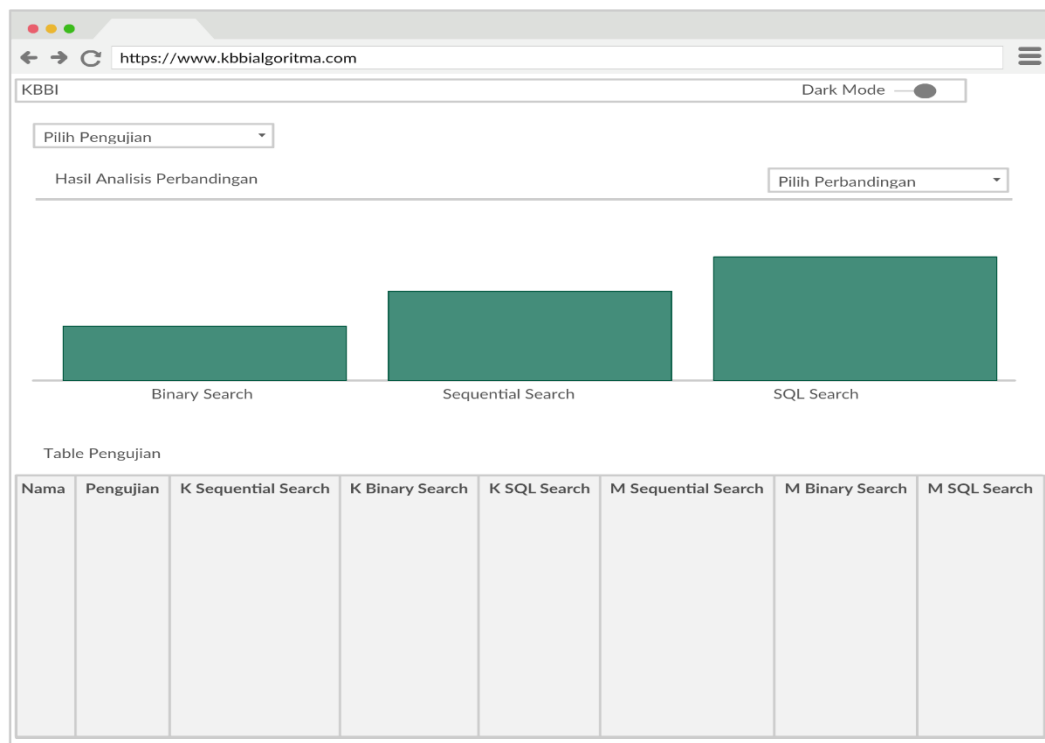


Gambar 4.15 User Interface analysis *area chart*

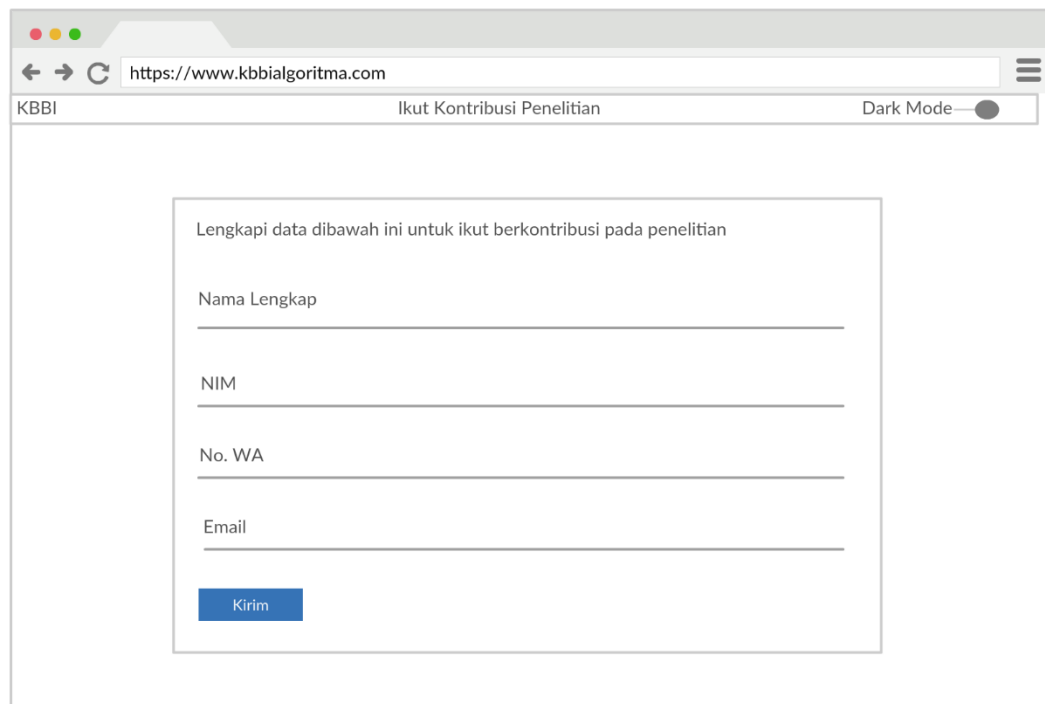
Gambar 4.16 User Interface analisis *bar chart*Gambar 4.17 User Interface Analisis *line chart*



Gambar 4.18 User Interface analisis kontribusi *area chart*



Gambar 4.19 User Interface analisis kontribusi *bar chart*



https://www.kbbialgoritma.com

KBBI Ikut Kontribusi Penelitian Dark Mode

Lengkapi data dibawah ini untuk ikut berkontribusi pada penelitian

Nama Lengkap

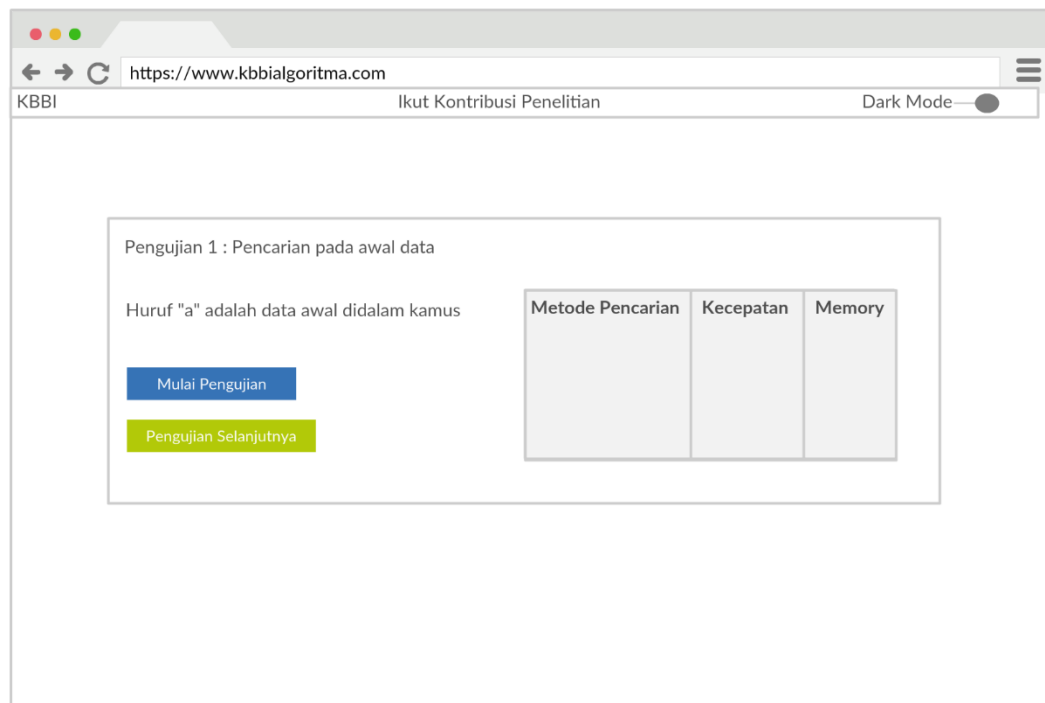
NIM

No. WA

Email

Kirim

Gambar 4.20 User Interface pengisian data diri kontribusi



https://www.kbbialgoritma.com

KBBI Ikut Kontribusi Penelitian Dark Mode

Pengujian 1 : Pencarian pada awal data

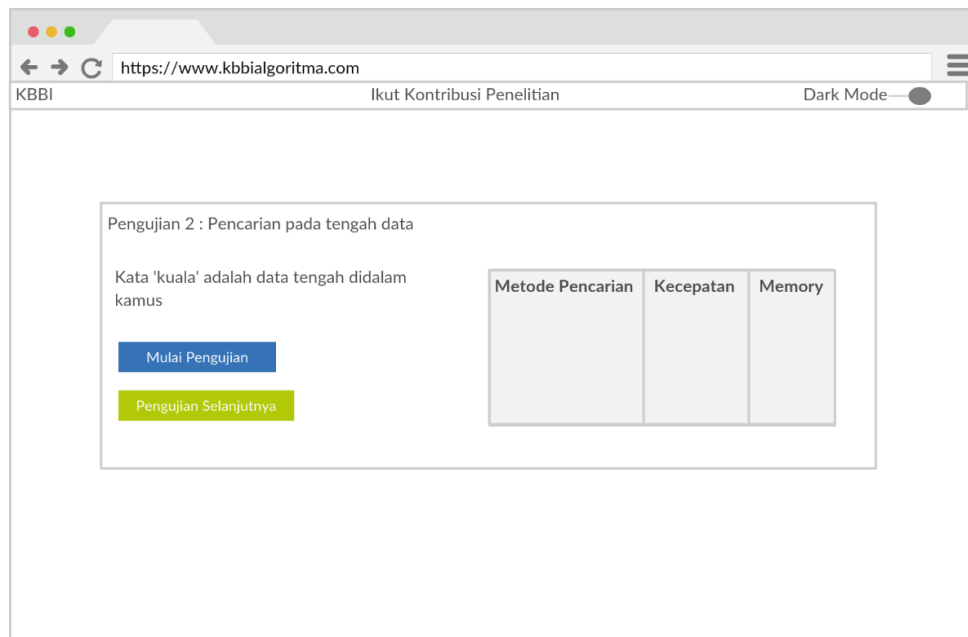
Huruf "a" adalah data awal didalam kamus

Mulai Pengujian

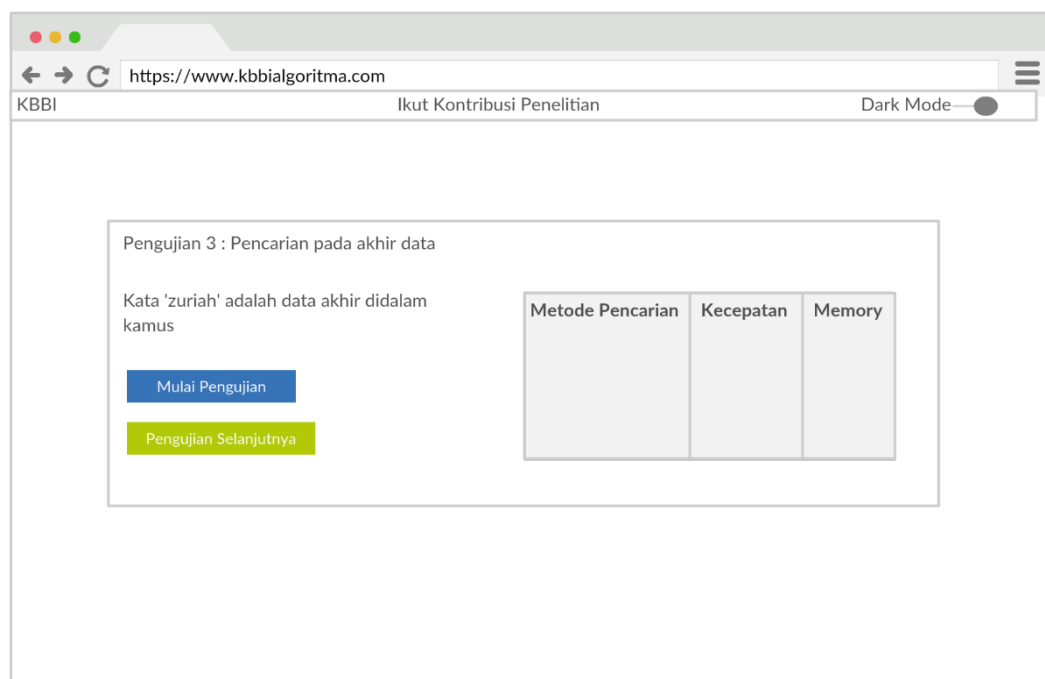
Pengujian Selanjutnya

Metode Pencarian	Kecepatan	Memory

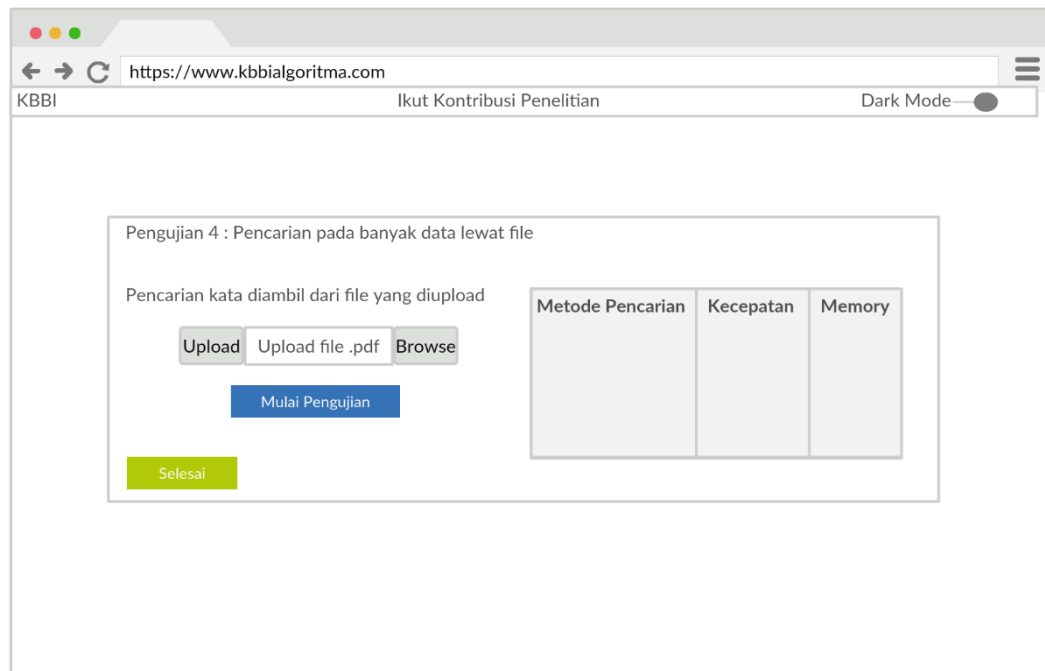
Gambar 4.21 User Interface pengujian 1 pada kontribusi



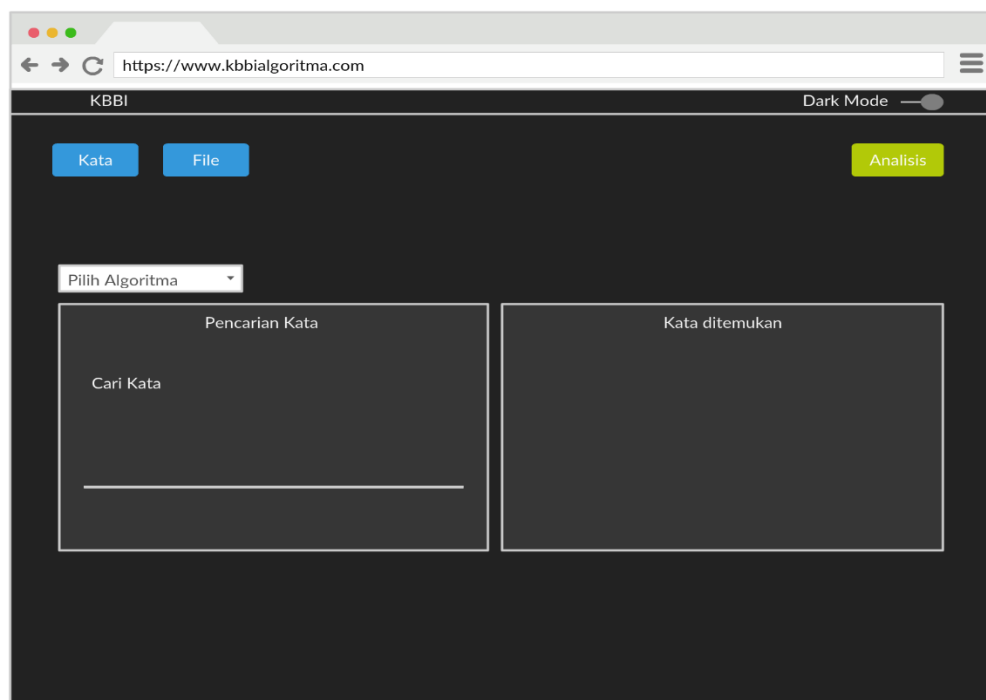
Gambar 4.22 User Interface pengujian 2 pada kontribusi



Gambar 4.23 User Interface pengujian 3 pada kontribusi



Gambar 4.24 User Interface pengujian 4 pada kontribusi



Gambar 4.25 User Interface fitur *dark mode*

4.2.4. Struktur Tabel

Berikut adalah struktur table basis data pembuatan aplikasi kamus Bahasa Indonesia dengan nama database kbbs dengan rincian table sebagai berikut :

Table 4.4 Struktur Tabel : kamus

No	Nama Field	Tipe Data	Length	Keterangan
1	id	Integer	11	<i>Primary Key</i>
2	Kata	Varchar	50	Kata yang dicari
3	Definisi	Text		Definisi kata
4	Sumber	Varchar	128	Sumber kata

Table 4.5 Struktur Tabel : perbandingan

No	Nama Field	Tipe Data	Length	Keterangan
1	Id	Integer	11	<i>Primary Key</i>
2	Pengujian	Integer	1	Jenis Pengujian
3	Kata	Text		Kata yang dicari
4	Algoritma	Varchar	50	Jenis Algoritma
5	Waktu	Varchar	128	Kecepatan pencarian
6	Memory	Varchar	128	Memori pencarian
7	Jumlah_kata	Int	12	Jumlah kata ditemukan
8	Detail	text		Detail proses pencarian

Table 4.6 Struktur Tabel : kontribusi

No	Nama Field	Tipe Data	Length	Keterangan
1	Id	Integer	11	<i>Primary Key</i>
2	Nama	Varchar	128	Nama mahasiswa
3	Nim	Varchar	20	Nim mahasiswa
4	No_wa	Varchar	15	Nomor <i>Whatsapp</i>
5	Email	Varchar	128	Email mahasiswa
6	Pengujian	Integer	1	Jenis pengujian
7	Kecepatan_sequential	Varchar	50	Kecepatan pencarian <i>Sequential Search</i>
8	Kecepatan_binary	Varchar	50	Kecepatan pencarian <i>Binary Search</i>
9	Kecepatan_sql	Varchar	50	Kecepatan pencarian sql search
10	Memory_sequential	Varchar	50	Memory pencarian <i>Sequential Search</i>
11	Memory_binary	Varchar	50	Memory pencarian <i>Binary Search</i>
12	Memory_sql	Varchar	50	Memory pencarian sql search

BAB V

IMPLEMENTASI DAN PENGUJIAN

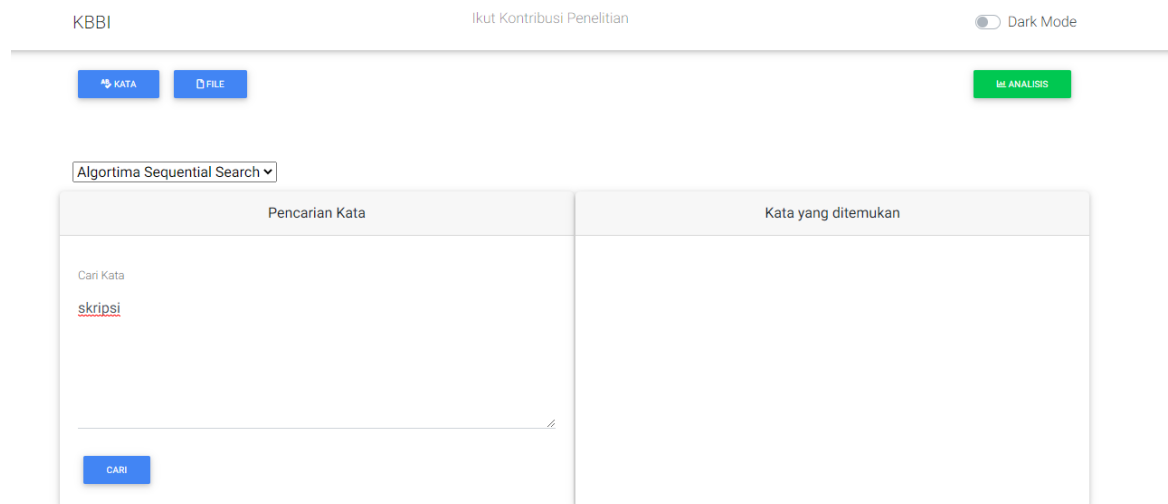
5.1 Implementasi

Implementasi adalah tahap penerapan sistem berdasarkan hasil analisis dan perancangan yang dilakukan pada bab IV. Pada bab V ini merupakan implementasi hasil rancangan menjadi Aplikasi Kamus Bahasa Indonesia Menggunakan PHP dan JQuery untuk membandingkan Algoritma *Sequential Search* dan *Binary Search*.

5.1.1. Implementasi User Interface

Berikut adalah beberapa gambar hasil implementasi user interface :

1. Tampilan halaman utama dan pencarian kata

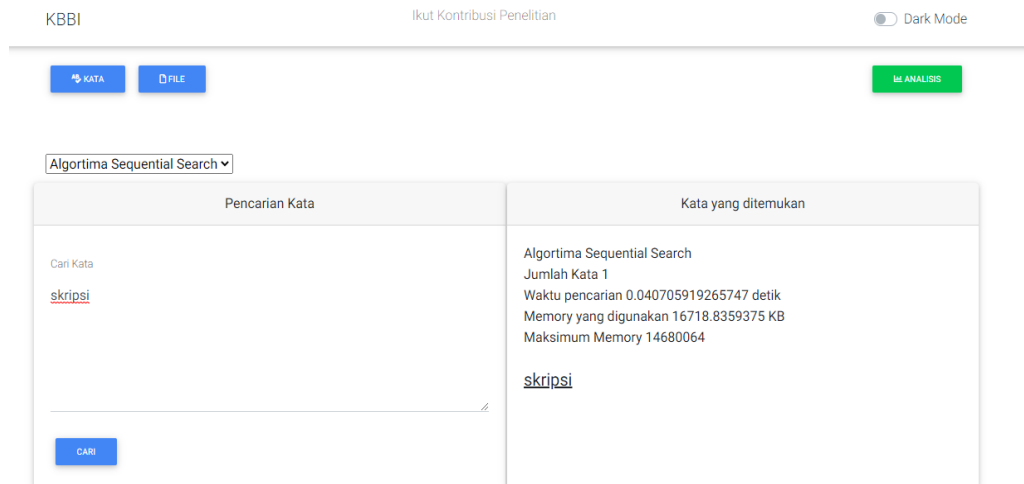


Gambar 5.1 Tampilan halaman utama / Pencarian Kata

Pada gambar diatas adalah halaman utama dan pencarian kata pada aplikasi kamus Bahasa Indonesia. Dibagian atas aplikasi terdapat tombol kata, file dan analisis. Dibagian tengah aplikasi 2

bagian, bagian kiri yaitu form pencarian kata dan bagian kanan hasil yang ditemukan.

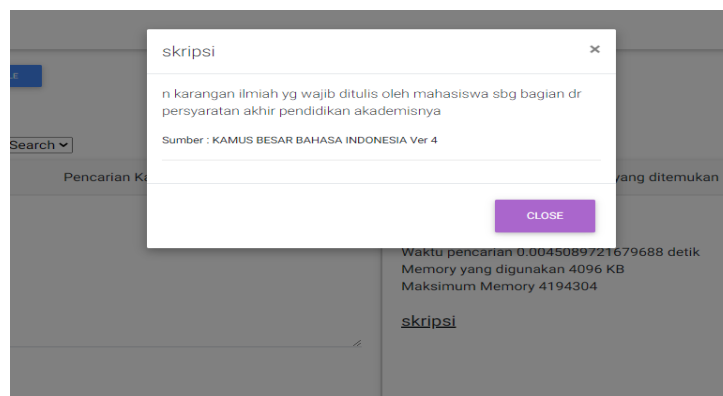
2. Tampilan hasil pencarian kata



Gambar 5.2 Tampilan hasil pencarian kata

Pada gambar diatas adalah contoh pencarian kata ‘skripsi’ menggunakan algoritma *Sequential Search* dan menampilkan kata yang ditemukan, beserta jumlah kata, waktu pencarian dan memory yang digunakan untuk bahan perbandingan.

3. Tampilan definisi kata yang ditemukan



Gambar 5.3 Tampilan definisi kata yang ditemukan

Pada gambar diatas menampilkan definisi kata yang ditemukan dengan mengklik kata tersebut.

4. Tampilan pencarian kata lewat file

Pencarian Kata	Kata yang ditemukan

Gambar 5.4 Tampilan pencarian kata lewat file

Pada gambar diatas adalah tampilan pencarian kata lewat file. Dibawah tombol pencarian kata, file dan analisis terdapat form upload file pdf. Sama seperti pencarian kata terdapat 2 bagian yaitu kata yang dicari dan kata yang ditemukan.

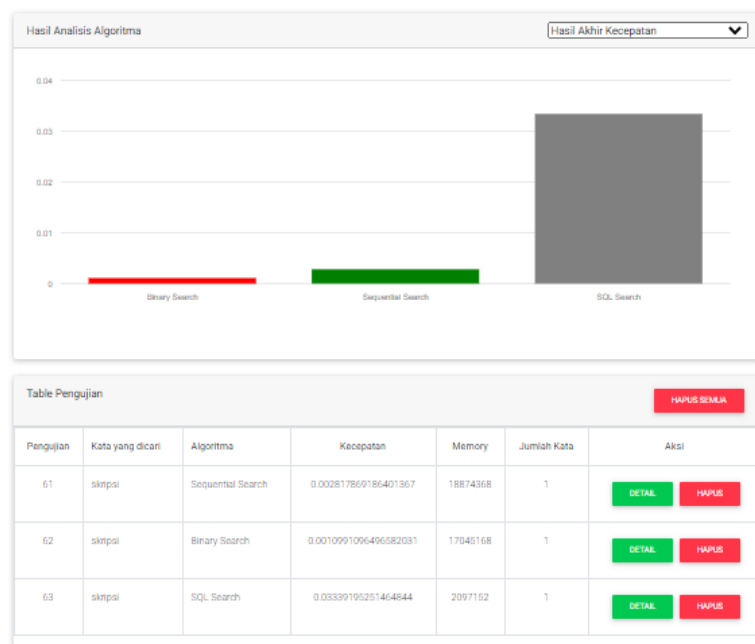
5. Tampilan hasil pencarian kata lewat file

Pencarian Kata	Kata yang ditemukan
<p>dhieka avrilia lantana analisis dan desain algoritma g651120381 1 analisis algoritma binary search metode binary search binary search merupakan salah satu algoritma untuk melakukan pencarian pada array yang sudah terurut jika kita tidak mengetahui informasi bagaimana integer dalam array maka penggunaan binary search akan menjadi tidak efisien kita harus melakukan sorting terlebih dahulu atau menggunakan metode lain yaitu linear search namun jika kita telah mengetahui integer dalam array terorganisasi baik secara menaik atau menurun maka bisa dengan cepat menggunakan algoritma binar search adanun ide dasar binarv search yaitu</p>	<p>Algoritma Binary Search Jumlah Kata 322 Waktu pencarian 0.051905155181885 detik Memory yang digunakan 16866.578125 KB Maksimum Memory 8388608</p> <p><u>analisis dan desain metode salah satu untuk sudah kita tidak informasi bagaimana dalam maka akan efisien harus dahulu atau lain yaitu namun telah baik bisa dengan cepat adapun ide dasar dua ruang a ingin lokasi dari spesifik target kondisi data langsung di indeks elemen dapat hanya sisi kiri seluruh kanan lebih saja cari nilai seperti ini kamus</u></p>

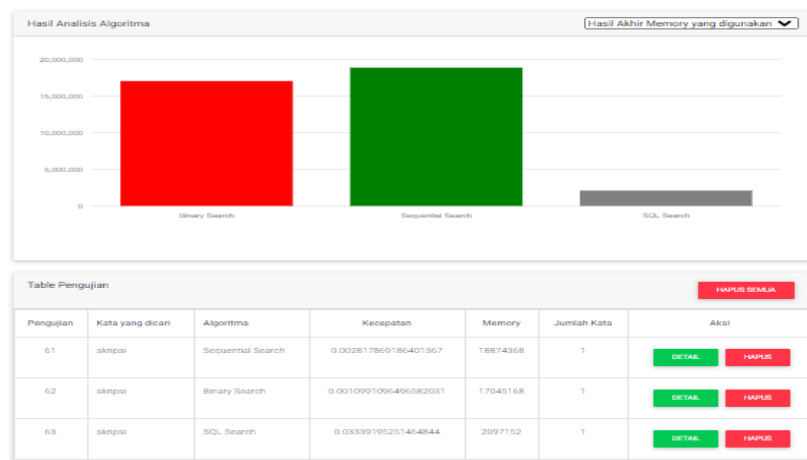
Gambar 5.5 Tampilan hasil pencarian kata lewat file

Pada gambar diatas adalah contoh pencarian lewat file dengan nama file 'binary-search-analysis.pdf' menggunakan algoritma *Binary Search* dan menampilkan kata yang ditemukan, beserta jumlah kata, waktu pencarian dan memory yang digunakan untuk bahan perbandingan.

6. Tampilan hasil analisis pencarian



Gambar 5.6 Tampilan grafik kecepatan dan table pengujian



Gambar 5.7 Tampilan grafik memory dan table pengujian

Selain membandingkan algoritma *Sequential Search* dan *Binary Search*, pencarian kata menggunakan juga SQL search sebagai perbandingan lainnya.

Pada Gambar 5.6 dan Gambar 5.7 adalah contoh hasil analisis pada kata yang sudah ditemukan ketika proses pencarian. Terdapat grafik dan table pengujian yang menampilkan data kata yang ditemukan pada pencarian sebelumnya. Pada Gambar 5.6 grafik menampilkan perbedaan kecepatan pencarian dari 3 metode pencarian. Pada Gambar 5.7 grafik menampilkan perbedaan memory yang digunakan dari 3 metode pencarian.

7. Tampilan detail proses pencarian

Detail			
Tutup			
<p>Apakah 0 <= 24459? Nilai Tengah = $(0 + 24459) / 2 = 12229$ apakah kuala = skripsi Tidak, kata lebih kecil dari yang dicari Batas atas = $12229 + 1 = 12230$</p>	<p>Apakah 12230 <= 24459? Nilai Tengah = $(12230 + 24459) / 2 = 18344$ apakah ruat = skripsi Tidak, kata lebih kecil dari yang dicari Batas atas = $18344 + 1 = 18345$</p>	<p>Apakah 18345 <= 24459? Nilai Tengah = $(18345 + 24459) / 2 = 21402$ apakah tahlil = skripsi Tidak, kata lebih besar dari yang dicari Batas bawah = $21402 - 1 = 21401$</p>	<p>Apakah 18345 <= 21401? Nilai Tengah = $(18345 + 21401) / 2 = 19873$ apakah sentrum = skripsi Tidak, kata lebih kecil dari yang dicari Batas atas = $19873 + 1 = 19874$</p>
<p>Apakah 19874 <= 21401? Nilai Tengah = $(19874 + 21401) / 2 = 20637$ apakah slendro = skripsi Tidak, kata lebih besar dari yang dicari Batas bawah = $20637 - 1 = 20636$</p>	<p>Apakah 19874 <= 20636? Nilai Tengah = $(19874 + 20636) / 2 = 20255$ apakah setrip = skripsi Tidak, kata lebih kecil dari yang dicari Batas atas = $20255 + 1 = 20256$</p>	<p>Apakah 20256 <= 20636? Nilai Tengah = $(20256 + 20636) / 2 = 20446$ apakah sinemaskop = skripsi Tidak, kata lebih kecil dari yang dicari Batas atas = $20446 + 1 = 20447$</p>	<p>Apakah 20447 <= 20636? Nilai Tengah = $(20447 + 20636) / 2 = 20541$ apakah sirah = skripsi Tidak, kata lebih kecil dari yang dicari Batas atas = $20541 + 1 = 20542$</p>
<p>Apakah 20542 <= 20636? Nilai Tengah = $(20542 + 20636) / 2 = 20589$ apakah siswa = skripsi Tidak, kata lebih kecil dari yang dicari Batas atas = $20589 + 1 = 20590$</p>	<p>Apakah 20590 <= 20636? Nilai Tengah = $(20590 + 20636) / 2 = 20613$ apakah siur = skripsi Tidak, kata lebih kecil dari yang dicari Batas atas = $20613 + 1 = 20614$</p>	<p>Apakah 20614 <= 20636? Nilai Tengah = $(20614 + 20636) / 2 = 20625$ apakah skeptis = skripsi Tidak, kata lebih kecil dari yang dicari Batas atas = $20625 + 1 = 20626$</p>	<p>Apakah 20626 <= 20636? Nilai Tengah = $(20626 + 20636) / 2 = 20631$ apakah skor = skripsi Tidak, kata lebih kecil dari yang dicari Batas atas = $20631 + 1 = 20632$</p>
<p>Apakah 20632 <= 20636? Nilai Tengah = $(20632 + 20636) / 2 = 20634$ apakah skuter = skripsi Tidak, kata lebih besar dari yang dicari Batas bawah = $20634 - 1 = 20633$</p>	<p>Apakah 20632 <= 20633? Nilai Tengah = $(20632 + 20633) / 2 = 20632$ apakah skors = skripsi Tidak, kata lebih kecil dari yang dicari Batas atas = $20632 + 1 = 20633$</p>	<p>Apakah 20633 <= 20633? Nilai Tengah = $(20633 + 20633) / 2 = 20633$ apakah skripsi = skripsi Ya, kata sama Kata ditemukan: skripsi (pada langkah 15)</p>	

Gambar 5.8 Tampilan detail proses pencarian

Pada Gambar 5.8 adalah contoh proses pencarian kata ‘skripsi’ menggunakan algoritma *Binary Search* dari awal pencarian sampai kata ditemukan pada langkah ke-15.

8. Tampilan form data kontribusi

KBBI Ikut Kontribusi Penelitian Dark Mode

Lengkapi data dibawah ini untuk ikut berkontribusi pada penelitian

Nama Lengkap

NIM

No. WA

Email

KIRIM

Gambar 5.9 Tampilan form data kontribusi

Pada gambar 5.9 terdapat form data mahasiswa yang ikut terlibat dalam pengujian algoritma.

9. Tampilan pada pengujian 1 kontribusi

KBBI Ikut Kontribusi Penelitian Dark Mode

Pengujian 1 : Pencarian pada awal data

Huruf "a" adalah data awal didalam kamus

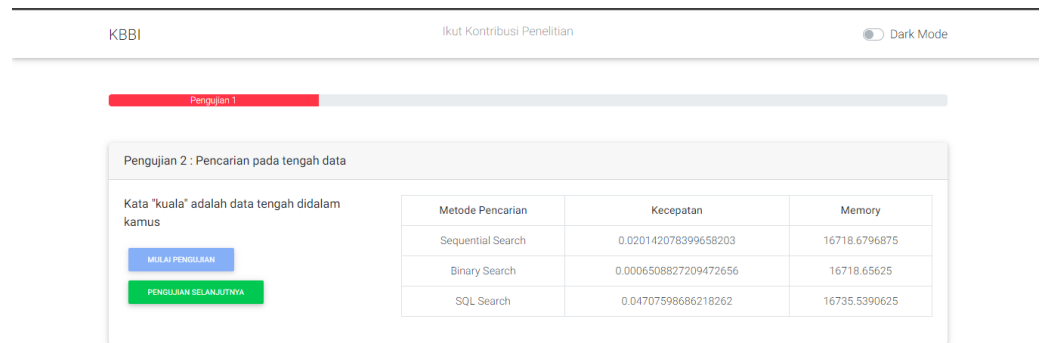
MULAI PENGUJIAN

PENGUJIAN SELANJUTNYA

Metode Pencarian	Kecepatan	Memory
Sequential Search	0.0010149478912353516	16718.609375
Binary Search	0.00044083595275878906	16718.5859375
SQL Search	0.027239084243774414	16735.734375

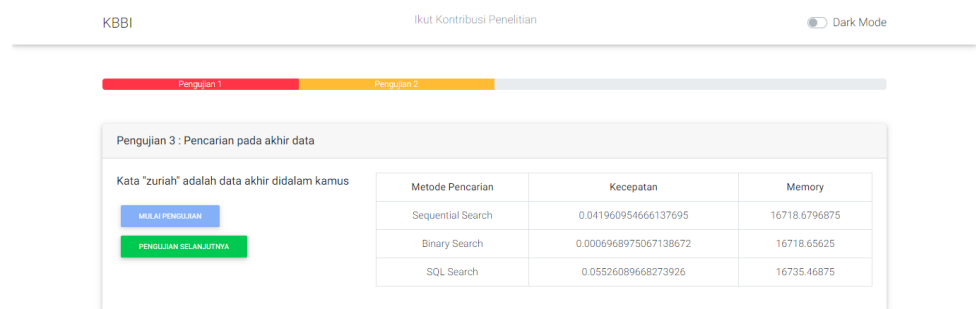
Gambar 5.10 Tampilan pengujian 1 kontribusi

10. Tampilan pada pengujian 2 kontribusi



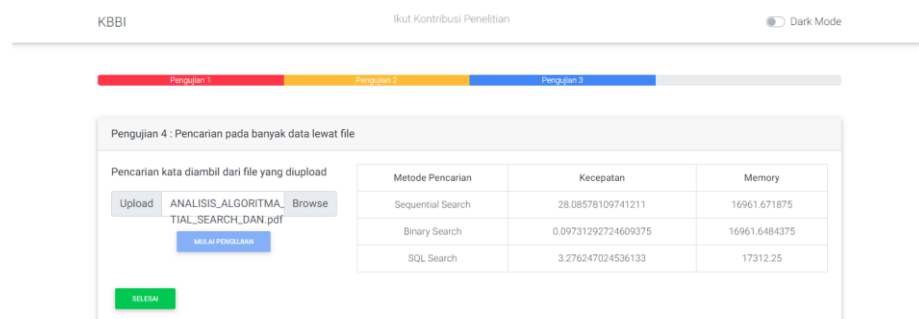
Gambar 5.11 Tampilan pengujian 2 kontribusi

11. Tampilan pada pengujian 3 kontribusi



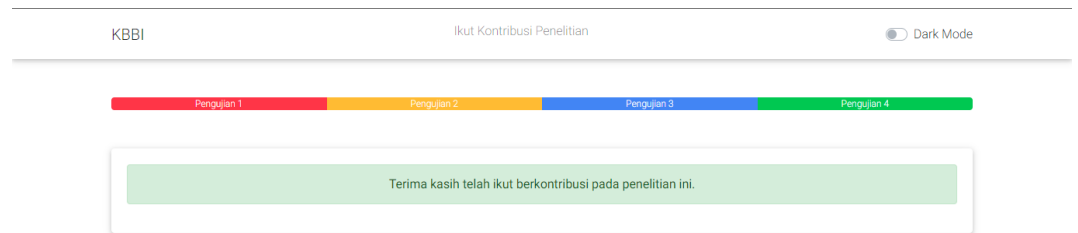
Gambar 5.12 Tampilan pengujian 3 kontribusi

12. Tampilan pada pengujian 4 kontribusi



Gambar 5.13 Tampilan pengujian 4 kontribusi

13. Tampilan selesai pengujian



Gambar 5.14 Tampilan selesai pengujian kontribusi

Pada gambar 5.10, gambar 5.11, gambar 5.12 dan gambar 5.13 terdapat pengujian 4 pengujian yaitu pencarian pada awal data, tengah data, akhir data dan banyak data yang diambil didalam file pdf. Didalam data kamus data paling awal yaitu huruf a, data tengah yaitu kuala dan data akhir yaitu zuriah, serta pencarian banyak data yang diambil dari kata didalam file pdf yang diupload yang akan diuji kecepatan dan memory yang digunakan pada masing-masing metode pencarian. Mahasiswa yang ikut kontribusi pada pengujian hanya mengklik tombol mulai pengujian setelah itu proses pencarian pada masing-masing metode pencarian berjalan. Setelah pengujian 1 selesai, akan tampil tombol pengujian selanjutnya untuk melanjutkan pengujian lainnya. Hasil proses pencarian pada masing-masing metode pencarian adalah kecepatan dan memory yang ditampilkan didalam tabel. Dengan ditampilkannya data kecepatan dan memory didalam tabel, mahasiswa yang ikut terlibat bisa melihat perbedaan dari masing-masing metode pencarian. Setelah semua pengujian selesai akan tampil tombol selesai untuk mengakhiri pengujian.

14. Tampilan kesalahan pencarian kamus

Algoritma Sequential Search ▼

Pencarian Kata	Kata yang ditemukan
<p>Cari Kata</p> <p><u>skripsi</u></p> <hr/> <p>CARI</p>	<p>Kata tidak ditemukan</p>

Gambar 5.15 Tampilan kesalahan pencarian kata

Tampilan kesalahan ketika user mencari kata dan pencarian kata tidak ditemukan.

15. Tampilan kesalahan upload file

Upload sistem.png Browse

UPLOAD

Algoritma Sequential Search ▼

Pencarian Kata	Kata yang ditemukan
undefined	<p>undefined</p> <p>Kata tidak ditemukan</p>

Gambar 5.16 Tampilan kesalahan upload file

Tampilan kesalahan ketika user salah mengupload file atau file tidak dapat dibaca.

5.1.2. Implementasi Metode Pencarian

Berikut adalah 3 metode pencarian yang digunakan di aplikasi kamus Bahasa Indonesia untuk dibandingkan, sebagai berikut :

1. Algoritma *Sequential Search*

```

1 function sequentialSearch($array, $cari)
2 {
3     $hasil = [];
4     for($j=0; $j<count($cari); $j++){
5         for($i=0; $i<=count($array); $i++){
6             if( (isset($array[$i])) && ($array[$i] == $cari[$j]) ){
7                 $hasil[] = $array[$i];
8                 break;
9             }
10        }
11    }
12
13    return $hasil;
14 }

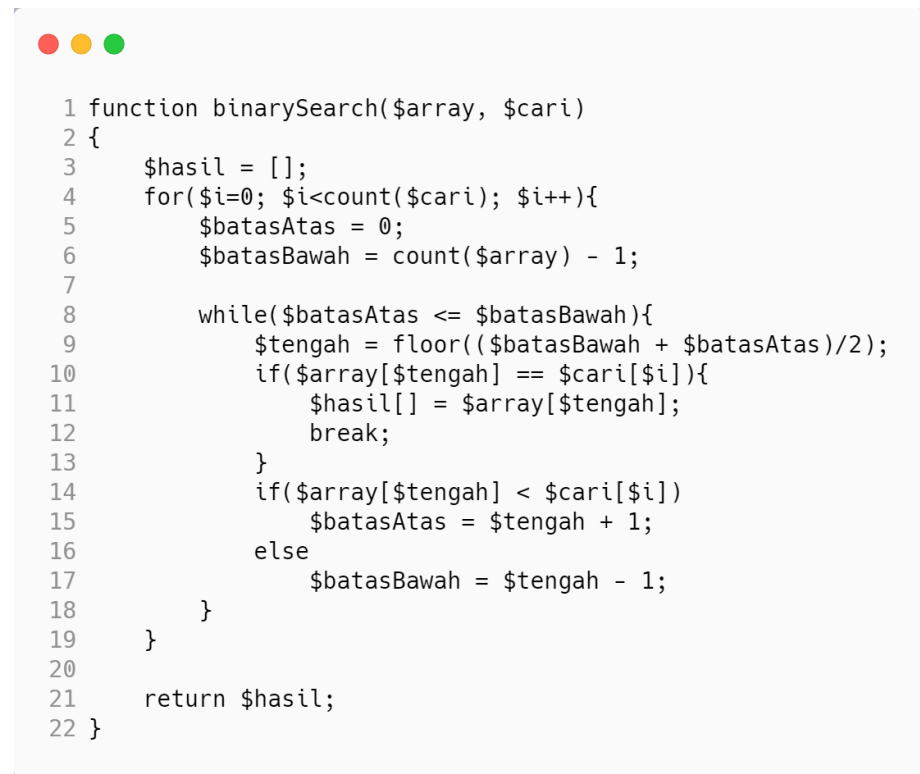
```

Gambar 5.17 Koding Algoritma *Sequential Search*

Pada Gambar 5.17 adalah function *Sequential Search* yang mempunyai parameter variable array dan cari. Variable array menampung data kamus dan variable cari menampung data kata yang dicari.

Baris tiga adalah inisialisai variable hasil yang berisi array kosong. Baris 4 sampai 9 terdapat perulangan jumlah kata yang dicari. Didalam perulangan jumlah kata pada baris 5 sampai 8, terdapat perulangan jumlah data kamus. Didalam perulangan jumlah data kamus, pada baris 6 sampai 7 terdapat pengecekan apakah kata yang dicari sama dengan kata yang ada di kamus, jika kata sama, masukkan kata pada kamus kedalam array hasil. Pada baris 11 terdapat variable hasil yang akan dikembalikan (*return*) ke fungsi tersebut.

2. Algoritma *Binary Search*



```

1 function binarySearch($array, $cari)
2 {
3     $hasil = [];
4     for($i=0; $i<count($cari); $i++){
5         $batasAtas = 0;
6         $batasBawah = count($array) - 1;
7
8         while($batasAtas <= $batasBawah){
9             $tengah = floor(($batasBawah + $batasAtas)/2);
10            if($array[$tengah] == $cari[$i]){
11                $hasil[] = $array[$tengah];
12                break;
13            }
14            if($array[$tengah] < $cari[$i])
15                $batasAtas = $tengah + 1;
16            else
17                $batasBawah = $tengah - 1;
18        }
19    }
20
21    return $hasil;
22 }

```

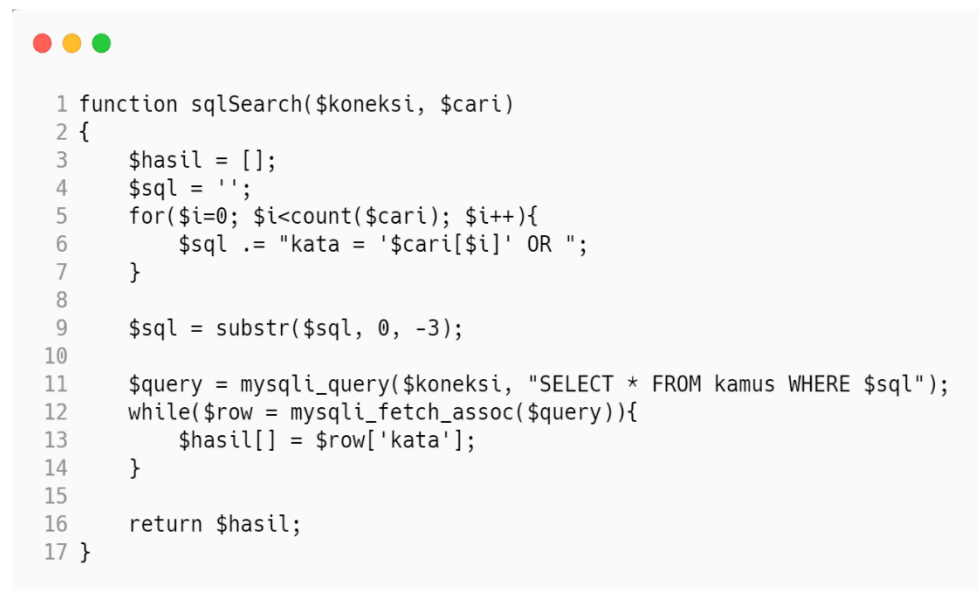
Gambar 5.18 Koding Algoritma *Binary Search*

Pada Gambar 5.18 adalah function *Binary Search* yang mempunyai parameter variable array dan cari. Sama seperti algoritma *Sequential Search*, variable array menampung data kamus dan variable cari menampung data kata yang dicari.

Baris tiga adalah inisialisai variable hasil yang berisi array kosong. Baris 4 sampai 17 terdapat perulangan jumlah kata yang dicari. Didalam perulangan jumlah kata pada baris 5 inisialisasi variable batas atas berisi 0. Pada baris 6 inisialisasi variable batas atas dengan isi jumlah kata pada kamus – 1. Pada baris 8 sampai 16 terdapat perulangan bersyarat dengan kondisi apakah batas atas kurang dari sama dengan batas bawah. Pada baris 9 variable tengah mencari nilai tengah dengan perhitungan (batas bawah + batas atas) / 2 dengan hasil yang dibulatkan jika hasilnya bilangan decimal.

Pada baris 10 terdapat pengecekan apakah data tengah didalam data kamus sama dengan kata yang dicari. Jika kata sama maka pada baris 11, kata yang ditemukan ditampung pada variable array hasil. Jika tidak sama pada baris 12, terdapat pengecekan apakah data tengah didalam data kamu lebih kecil dari kata yang dicari. Jika lebih kecil, pada baris 13, nilai tengah + 1 dan ditampung pada variable batas atas. Jika kata lebih besar pada baris 15, nilai tengah – 1 dan ditampung pada variable batas bawah. Pada baris 19 terdapat variable hasil yang akan dikembalikan (*return*) ke fungsi tersebut.

3. Pencarian SQL



```

1 function sqlSearch($koneksi, $cari)
2 {
3     $hasil = [];
4     $sql = '';
5     for($i=0; $i<count($cari); $i++){
6         $sql .= "kata = '$cari[$i]' OR ";
7     }
8
9     $sql = substr($sql, 0, -3);
10
11     $query = mysqli_query($koneksi, "SELECT * FROM kamus WHERE $sql");
12     while($row = mysqli_fetch_assoc($query)){
13         $hasil[] = $row['kata'];
14     }
15
16     return $hasil;
17 }

```

Gambar 5.19 Koding pencarian SQL

Pada gambar 5.19 adalah function pencarian menggunakan sql yang mempunyai parameter variable koneksi dan cari. Variable koneksi menampung koneksi database agar bisa menjalankan query, variable cari menampung data yang dicari.

Baris 3 adalah inisialisasi variable hasil yang berisi array kosong. Baris 4 adalah inisialisasi variable sql yang berisi string

kosong. Baris 5 adalah perulangan jumlah kata yang dicari. didalam perulangan pada baris 6 mengisi variable sql dengan mengisi kata sama dengan kata yang dicari dengan kata or jika kata yang dimasukkan lebih dari satu. Pada baris 9 variable sql memotong 3 kata dari belakang untuk menghilangkan kata or. Pada baris 11 variable query menjalankan query cari data berdasarkan variable sql. Pada baris 12 perulangan untuk mengeluarkan data hasil query berdasarkan jumlah kata yang ditemukan. Pada baris 16 terdapat variable hasil yang akan dikembalikan (*return*) ke fungsi tersebut

5.2 Pengujian

Pengujian perbandingan algoritma pencarian pada penelitian ini, membandingkan kedua algoritma dari segi kecepatan, memory yang digunakan dan Langkah pencarian yang didapat dari masing-masing algoritma. Sebagai perbandingan lainnya, dibandingkan juga metode pencarian yang sering digunakan yaitu SQL search. Proses pencarian dan perbandingan menggunakan browser Google Chrome dengan mode samaran, agar file dan cache tidak tersimpan dibrowser, sehingga proses pencarian yang didapat pun hasil dari pencarian saat itu juga.

Penulis melakukan pencarian sebanyak lima kali untuk masing-masing pengujian metode pencarian. Pengujian yang dilakukan antara lain :

- A. Pengujian pencarian data pada awal data
- B. Pengujian pencarian data pada tengah data
- C. Pengujian pencarian data pada akhir data
- D. Pengujian banyak kata lewat pencarian kata lewat file

5.2.1. Pengujian pencarian data pada awal data

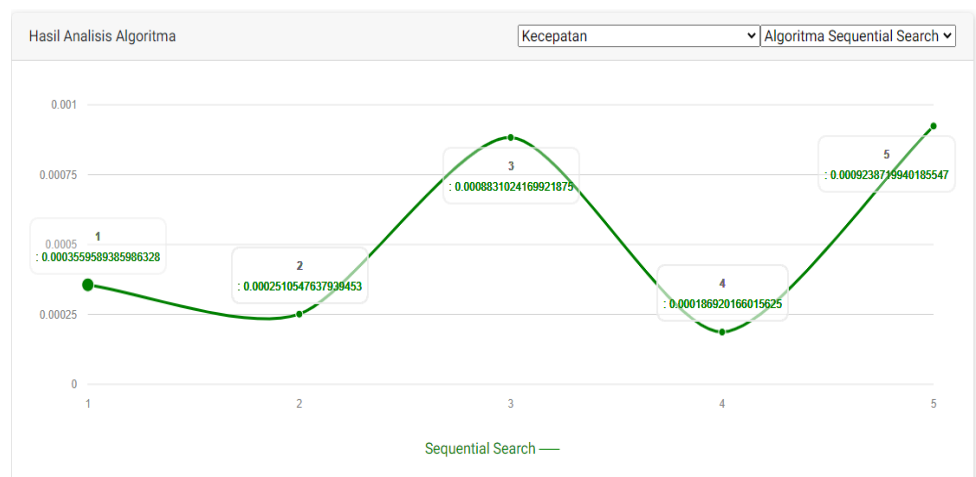
Data yang dicari terletak pada index array yang paling depan. Ini adalah kemungkinan terbaik (*best case*) pada algoritma *Sequential Search*

yang hanya melakukan sekali perbandingan dengan notasi $f(n) = O(1)$. Penulis akan mengujinya pada 3 metode pencarian. Berikut adalah hasil pengujiannya :

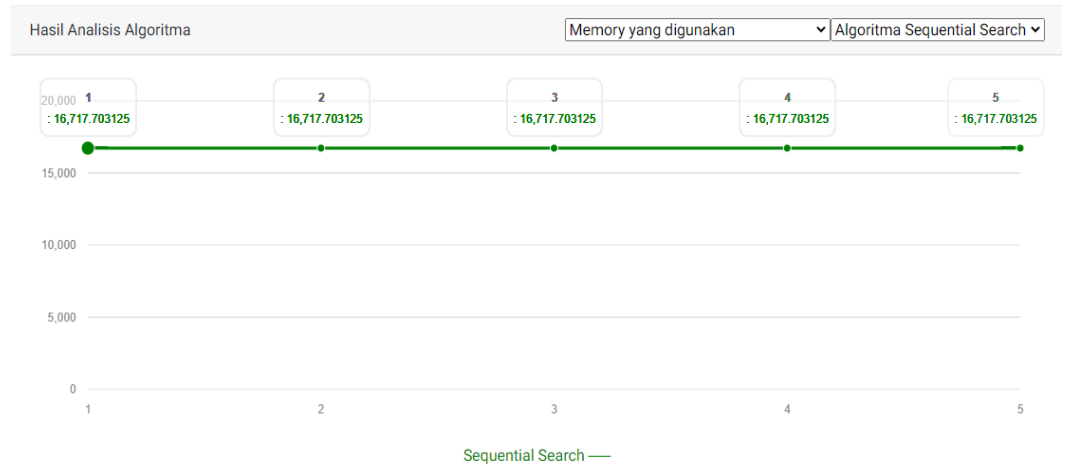
1. Algoritma *Sequential Search*

Table 5.1 Table pengujian algoritma *Sequential Search* pada awal data

No	Kata yang dicari	Kecepatan	Memory
1	a	0.0003559589385986328 s	16717.703125 KB
2	a	0.0002510547637939453 s	16717.703125 KB
3	a	0.0008831024169921875 s	16717.703125 KB
4	a	0.000186920166015625 s	16717.703125 KB
5	a	0.0009238719940185547 s	16717.703125 KB



Gambar 5.20 Line chart pengujian kecepatan algoritma *Sequential Search* pada awal data

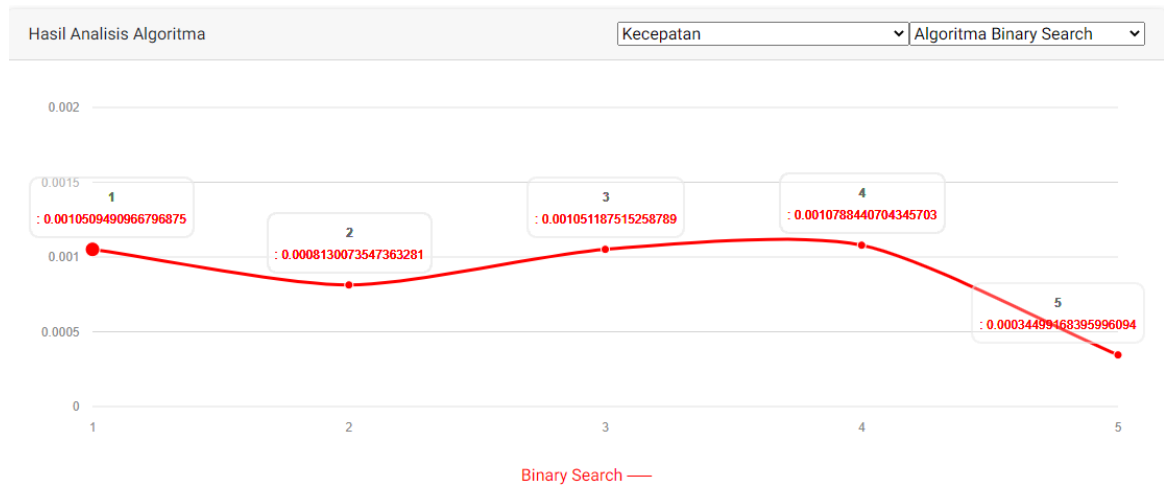


Gambar 5.21 Line chart pengujian memory algoritma *Sequential Search* pada awal data

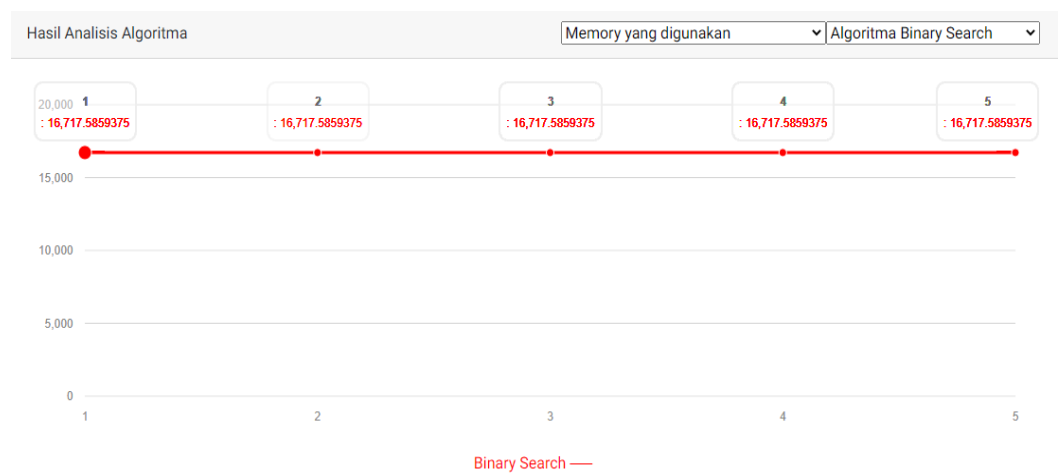
2. Algoritma *Binary Search*

Table 5.2 Table pengujian algoritma *Binary Search* pada awal data

No	Kata yang dicari	Kecepatan	Memory
1	a	0.0010509490966796875 s	16717.5859375 KB
2	a	0.0008130073547363281 s	16717.5859375 KB
3	a	0.001051187515258789 s	16717.5859375 KB
4	a	0.0010788440704345703 s	16717.5859375 KB
5	a	0.0009238719940185547 s	16717.703125 KB



Gambar 5.22 Line chart pengujian kecepatan algoritma *Binary Search* pada awal data



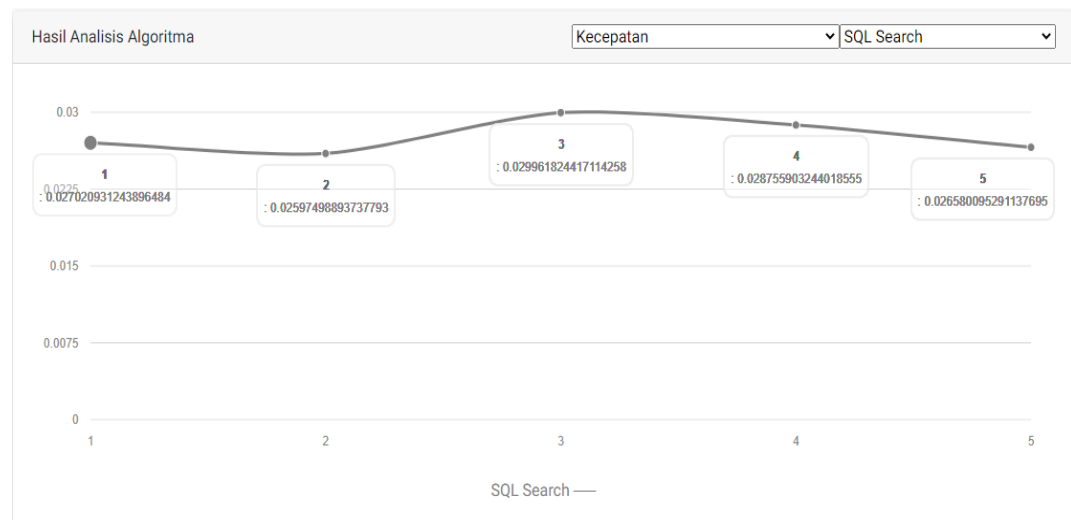
Gambar 5.23 Line chart pengujian memory algoritma *Binary Search* pada awal data

3. SQL Search

Tabel 5.3 Tabel pengujian algoritma *Binary Search* pada awal data

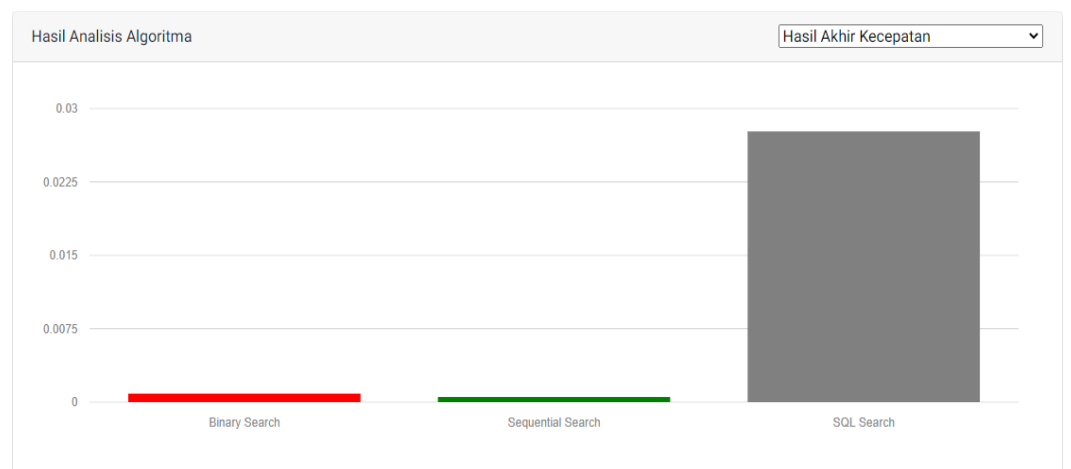
No	Kata yang dicari	Kecepatan	Memory
1	a	0.027020931243896484 s	16734.46875 KB

2	a	0.02597498893737793 s	16734.46875 KB
3	a	0.029961824417114258 s	16734.46875 KB
4	a	0.028755903244018555 s	16734.46875 KB
5	a	0.026580095291137695 s	16734.46875 KB



Gambar 5.24 Line chart pengujian kecepatan algoritma sql search pada awal data

4. Hasil akhir



Gambar 5.25 Bar chart hasil akhir kecepatan pada awal data



Gambar 5.26 Bar chart hasil akhir memory pada awal data

Dari hasil pengujian yang telah dilakukan, masing-masing metode pencarian mendapatkan hasil yang berbeda dari segi kecepatan, dan *memory* yang digunakan. Dari segi kecepatan algoritma *Sequential Search* mendapatkan kecepatan dengan rata-rata sebesar 0.0005201816558837891, sedangkan algoritma *Binary Search* dengan kecepatan rata-rata sebesar 0.0008677959442138672 dan SQL search dengan kecepatan rata-rata sebesar 0.027658748626708984. Dari segi kecepatan, *Sequential Search* lebih cepat dari kedua metode pencarian lainnya karena ini adalah kemungkinan terbaik (*best case*) dari algoritma *Sequential Search*.

Dari segi *memory* yang digunakan, algoritma *Sequential Search* mendapatkan *memory* dengan rata-rata sebesar 16,717.703125, sedangkan algoritma *Binary Search* dengan *memory* rata-rata sebesar 16,717.5859375 dan SQL search dengan *memory* rata-rata sebesar 16,734.46875. Dari segi *memory* yang digunakan, SQL search lebih besar menggunakan *memory* daripada 2 metode pencarian lainnya.

5.2.2. Pengujian pencarian data pada tengah data

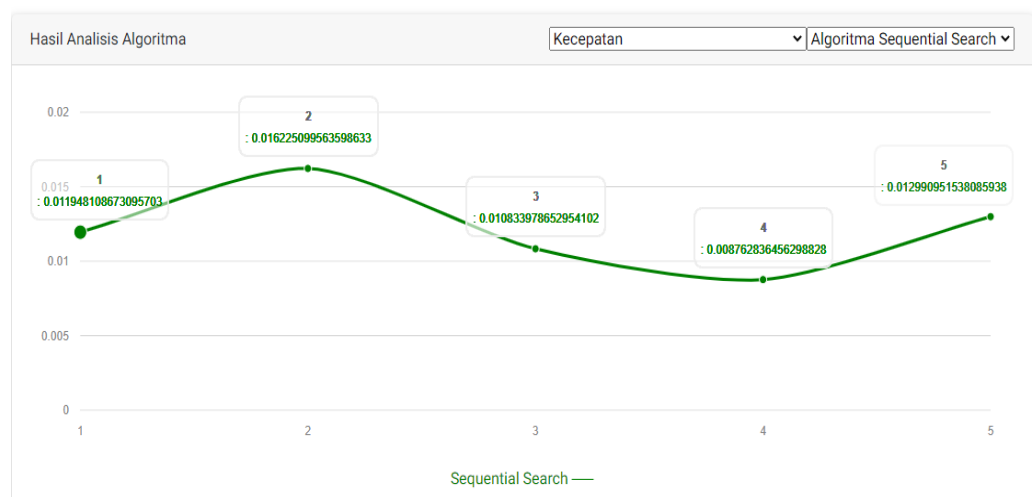
Data yang dicari terletak pada index array tengah data. Ini adalah kemungkinan terbaik (*best case*) pada algoritma *Binary Search* yang hanya

melakukan sekali perbandingan dengan notasi $f(n) = O(1)$. Penulis akan mengujinya pada 3 metode pencarian. Berikut adalah hasil pengujiannya :

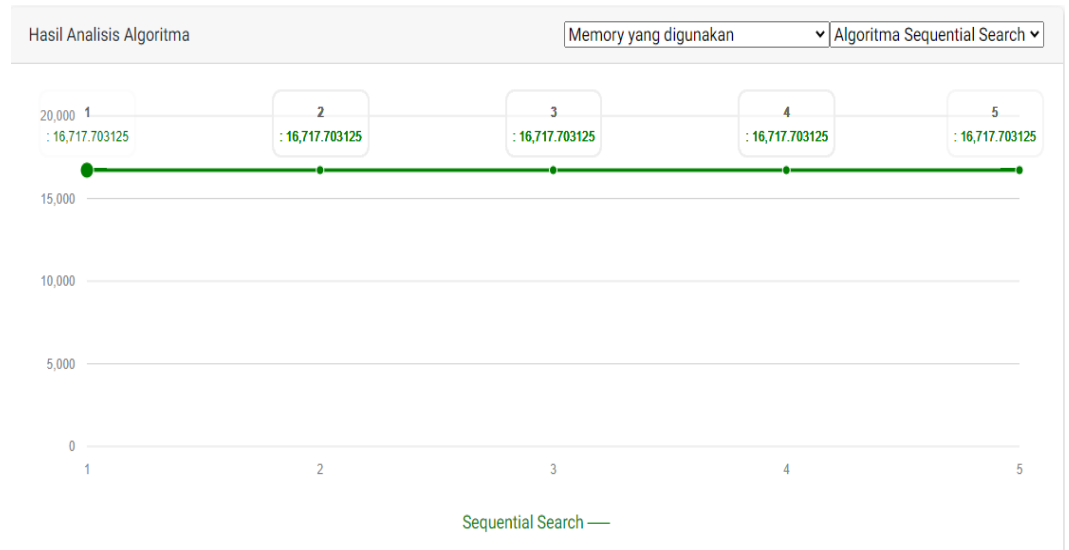
1. Algoritma *Sequential Search*

Tabel 5.4 Tabel pengujian algoritma *Binary Search* pada tengah data

No	Kata yang dicari	Kecepatan	Memory
1	kuala	0.011948108673095703 s	16717.703125 KB
2	kuala	0.016225099563598633 s	16717.703125 KB
3	kuala	0.010833978652954102 s	16717.703125 KB
4	kuala	0.008762836456298828 s	16717.703125 KB
5	kuala	0.012990951538085938 s	16717.703125 KB



Gambar 5.27 Line chart pengujian kecepatan algoritma *Sequential Search* pada awal data

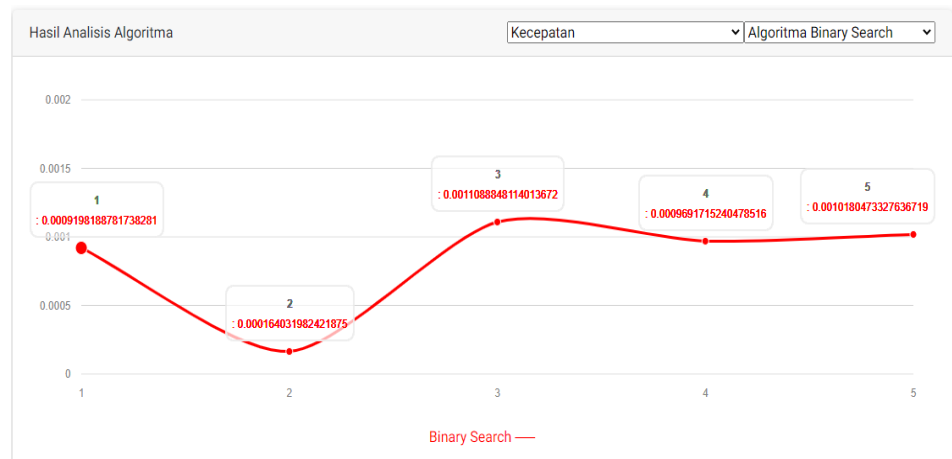


Gambar 5.28 Line chart pengujian memory algoritma *Sequential Search* pada awal data

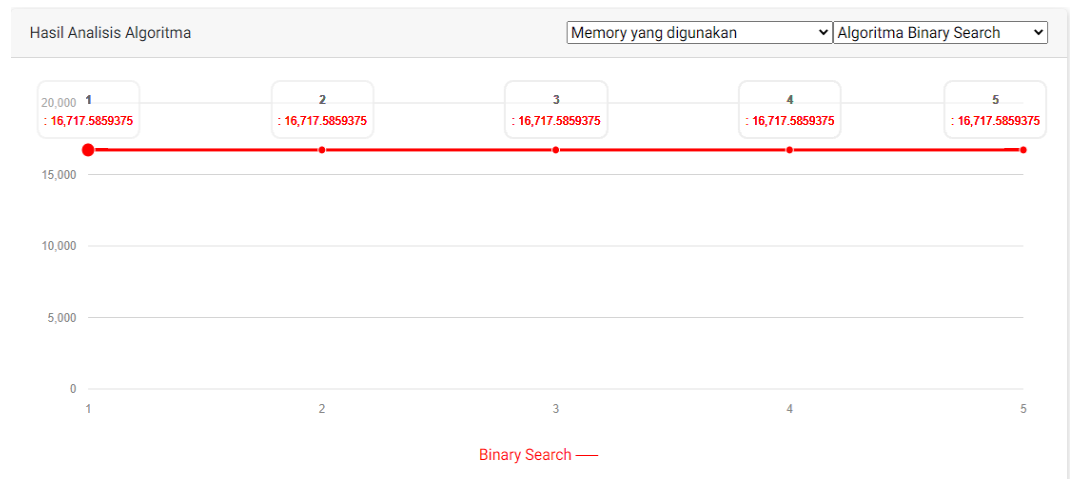
2. Algoritma *Binary Search*

Table 5.5 Tabel pengujian algoritma *Binary Search* pada tengah data

No	Kata yang dicari	Kecepatan	Memory
1	kuala	0.0009198188781738281 s	16717.5859375 KB
2	kuala	0.000164031982421875 s	16717.5859375 KB
3	kuala	0.0011088848114013672 s	16717.5859375 KB
4	kuala	0.0009691715240478516 s	16717.5859375 KB
5	kuala	0.0010180473327636719 s	16717.5859375 KB



Gambar 5.29 Line chart pengujian kecepatan algoritma *Binary Search* pada tengah data



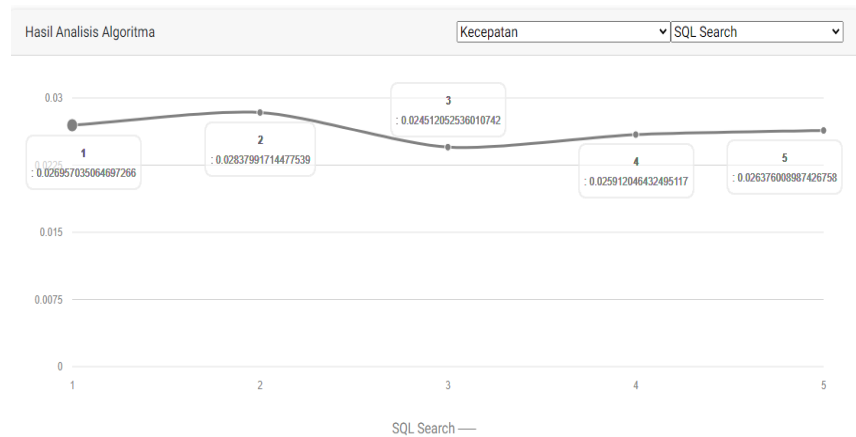
Gambar 5.30 Line chart pengujian memory algoritma *Binary Search* pada tengah data

3. SQL Search

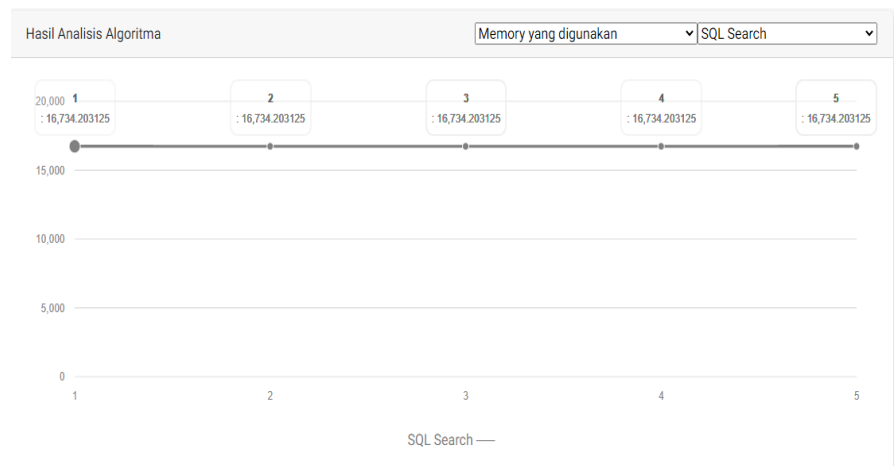
Table 5.6 Tabel pengujian algoritma sql search pada tengah data

No	Kata yang dicari	Kecepatan	Memory
1	kuala	0.026957035064697266 s	16734.203125 KB
2	kuala	0.02837991714477539 s	16734.203125 KB

3	kuala	0.024512052536010742 s	16734.203125 KB
4	kuala	0.025912046432495117 s	16734.203125 KB
5	kuala	0.026376008987426758 s	16734.203125 KB

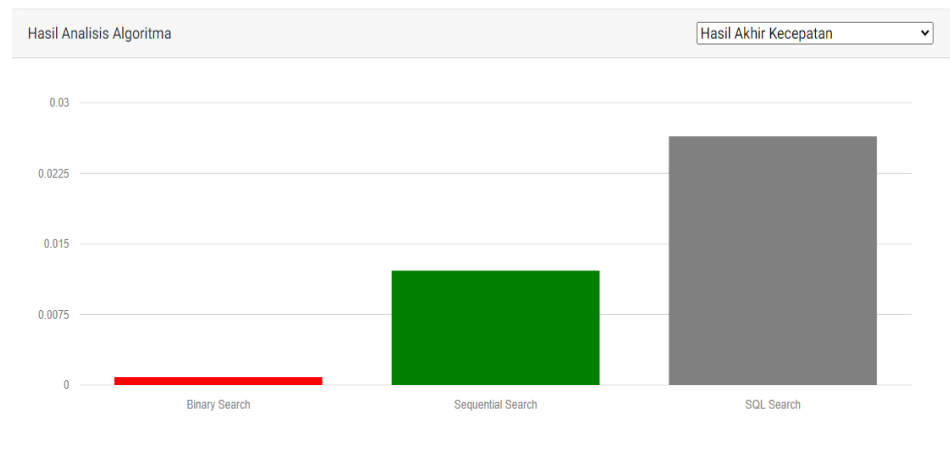


Gambar 5.31 Line chart pengujian kecepatan algoritma sql search pada tengah data



Gambar 5.32 Line chart pengujian memory algoritma sql search pada tengah data

4. Hasil Akhir



Gambar 5.33 Bar chart hasil akhir kecepatan pada tengah data



Gambar 5.34 Bar chart hasil akhir memory pada tengah data

Dari hasil pengujian yang telah dilakukan, masing-masing metode pencarian mendapatkan hasil yang berbeda dari segi kecepatan dan *memory* yang digunakan. Dari segi kecepatan algoritma *Sequential Search* mendapatkan kecepatan dengan rata-rata sebesar 0.01215219497680664, sedangkan algoritma *Binary Search* dengan kecepatan rata-rata sebesar 0.0008359909057617188 dan SQL search dengan kecepatan rata-rata sebesar 0.026427412033081056. Dari segi kecepatan, *Binary Search* lebih

cepat dari kedua metode pencarian lainnya karena ini adalah kemungkinan terbaik (*best case*) dari algoritma *Binary Search*.

Dari segi *memory* yang digunakan, algoritma *Sequential Search* mendapatkan *memory* dengan rata-rata sebesar 16,717.703125, sedangkan algoritma *Binary Search* dengan *memory* rata-rata sebesar 16,717.5859375 dan SQL search dengan *memory* rata-rata sebesar 16,734.203125. Dari segi *memory* yang digunakan, sql search lebih besar menggunakan *memory* daripada 2 metode pencarian lainnya.

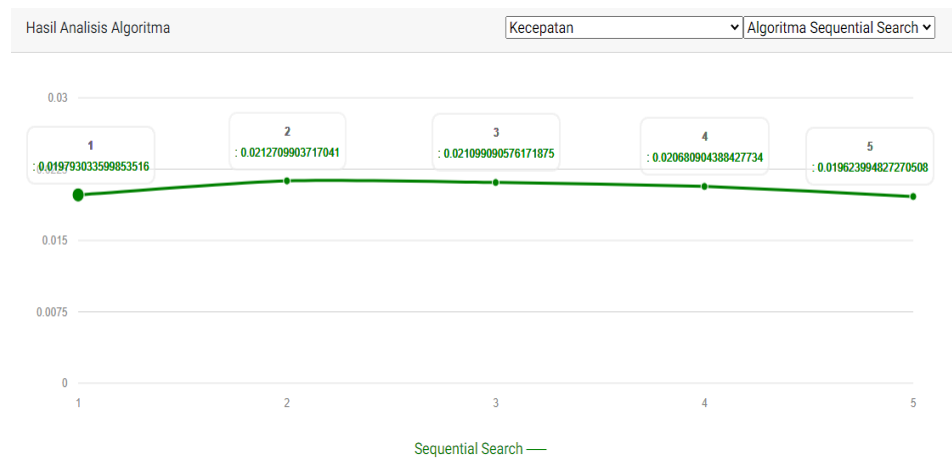
5.2.3. Pengujian pencarian data pada akhir data

Data yang dicari terletak pada index array akhir data. Kemungkinan terburuk (*worst case*) adalah melakukan perbandingan sampai akhir data atau data tidak ditemukan. Pada *Sequential Search* dengan notasi $f(n) = O(n)$ sedangkan binay search dengan notasi $f(n) = O(\log n)$. Penulis akan mengujinya pada 3 metode pencarian. Berikut adalah hasil pengujiannya :

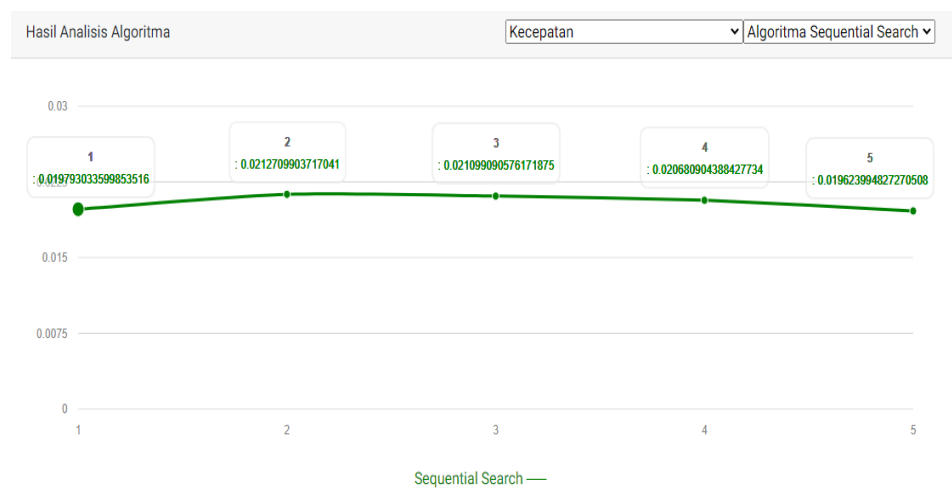
1. Algoritma *Sequential Search*

Table 5.7 Tabel pengujian algoritma sequence search pada akhir data

No	Kata yang dicari	Kecepatan	Memory
1	zuriah	0.019793033599853516 s	16717.703125 KB
2	zuriah	0.0212709903717041 s	16717.703125 KB
3	zuriah	0.021099090576171875 s	16717.703125 KB
4	zuriah	0.020680904388427734 s	16717.703125 KB
5	zuriah	0.019623994827270508 s	16717.703125 KB



Gambar 5.35 Line chart pengujian kecepatan algoritma sequence search pada akhir data

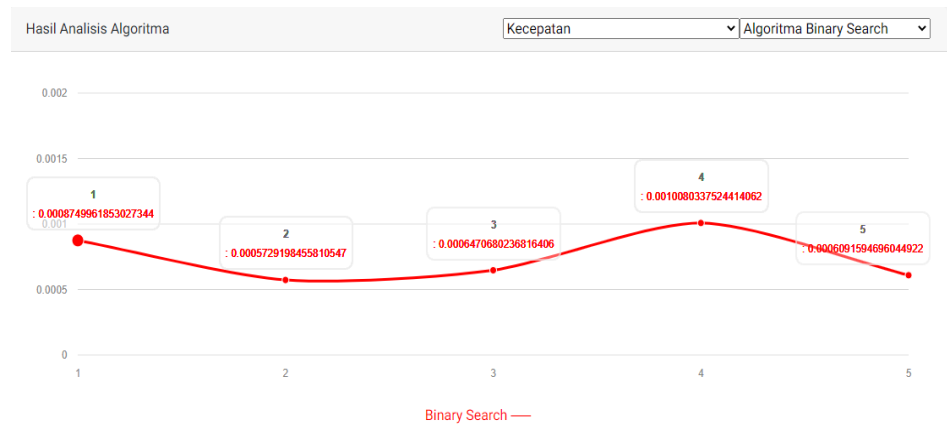


2. Binary Search

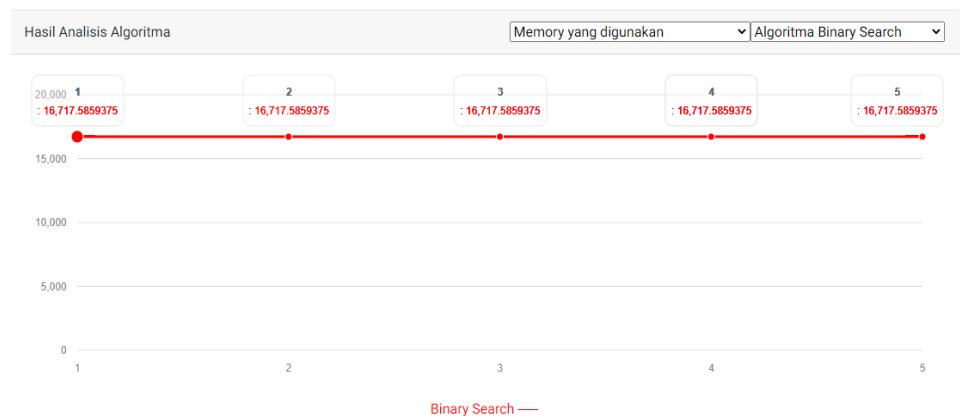
Table 5.8 Tabel pengujian algoritma sql search pada tengah data

No	Kata yang dicari	Kecepatan	Memory
1	zuriah	0.0008749961853027344 s	16717.5859375 KB
2	zuriah	0.0005729198455810547 s	16717.5859375 KB

3	zuriah	0.0006470680236816406 s	16717.5859375 KB
4	zuriah	0.0010080337524414062 s	16717.5859375 KB
5	zuriah	0.0006091594696044922 s	16717.5859375 KB



Gambar 5.36 Line chart pengujian kecepatan algoritma *Binary Search* pada akhir data

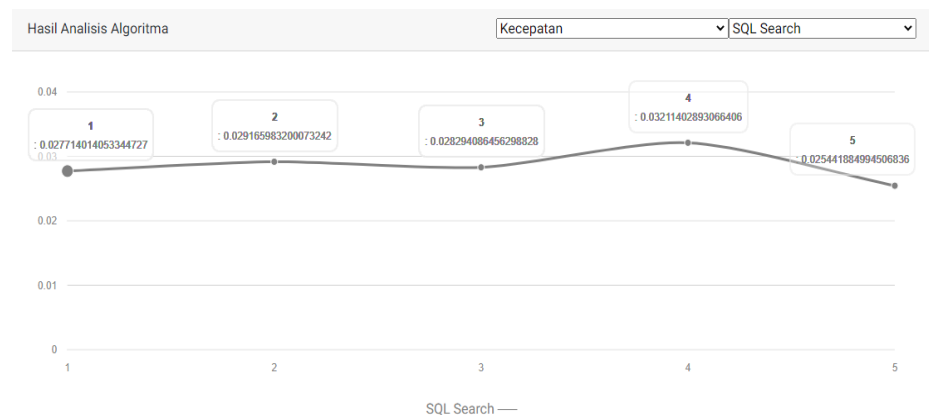


Gambar 5.37 Line chart pengujian memory algoritma *Binary Search* pada akhir data

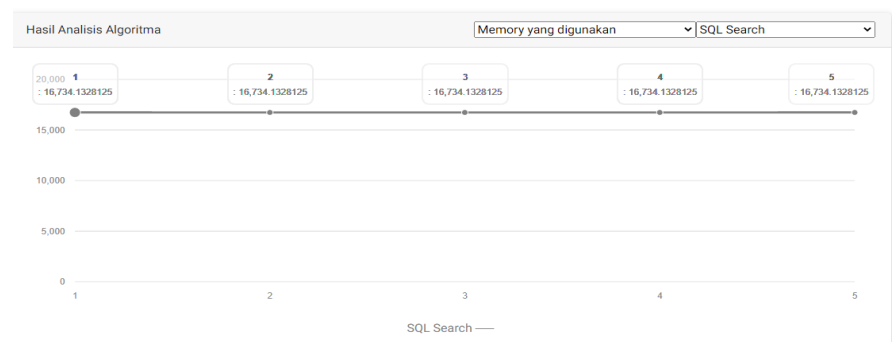
3. SQL Search

Table 5.9 Tabel pengujian algoritma sql search pada tengah data

No	Kata yang dicari	Kecepatan	Memory
1	zuriah	0.027714014053344727 s	16734.1328125 KB
2	zuriah	0.029165983200073242 s	16734.1328125 KB
3	zuriah	0.028294086456298828 s	16734.1328125 KB
4	zuriah	0.03211402893066406 s	16734.1328125 KB
5	zuriah	0.025441884994506836 s	16734.1328125 KB

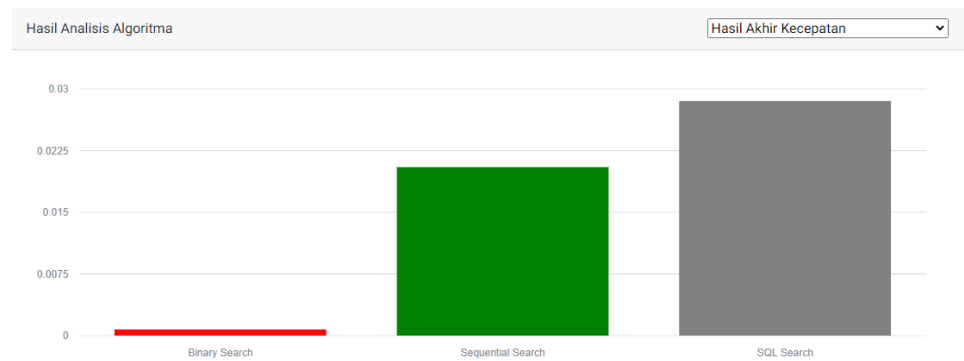


Gambar 5.38 Line chart pengujian kecepatan algoritma sql search pada akhir data



Gambar 5.39 Line chart pengujian memory sql search pada akhir data

4. Hasil Akhir



Gambar 5.40 Bar chart hasil akhir kecepatan pada akhir data



Gambar 5.41 Bar chart hasil akhir memory pada akhir data

Dari hasil pengujian yang telah dilakukan, masing-masing metode pencarian mendapatkan hasil yang berbeda dari segi kecepatan, *memory* dan langkah pencarian. Dari segi kecepatan algoritma *Sequential Search* mendapatkan kecepatan dengan rata-rata sebesar 0.020493602752685545, sedangkan algoritma *Binary Search* dengan kecepatan rata-rata sebesar 0.0007424354553222656 dan SQL search dengan kecepatan rata-rata sebesar 0.02854599952697754. Dari segi kecepatan, *Binary Search* lebih cepat dari kedua metode pencarian lainnya.

Dari segi *memory* yang digunakan, algoritma *Sequential Search* mendapatkan *memory* dengan rata-rata sebesar 16,717.703125, sedangkan algoritma *Binary Search* dengan *memory* rata-rata sebesar 16,717.5859375

dan SQL search dengan *memory* rata-rata sebesar 16,734.1328125. Dari segi *memory* yang digunakan, sql search lebih besar menggunakan *memory* daripada 2 metode pencarian lainnya.

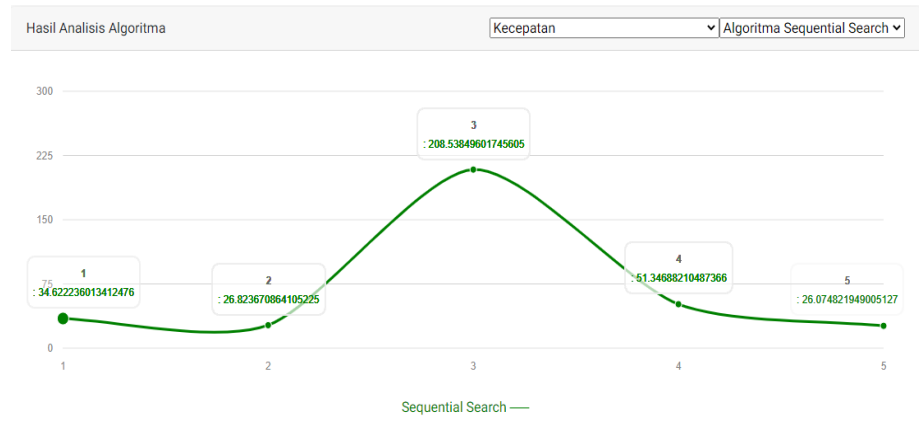
5.3.1. Pengujian banyak kata lewat pencarian kata lewat file

Data yang dicari adalah banyak kata lewat file pdf yang diupload. File yang dijadikan pengujian adalah file jurnal yang penulis gunakan sebagai referensi penelitian. Jumlah kata yang dicari sesuai semua kata yang terdapat didalam file tersebut Penulis akan mengujinya pada 3 metode pencarian. Berikut adalah hasil pengujiannya :

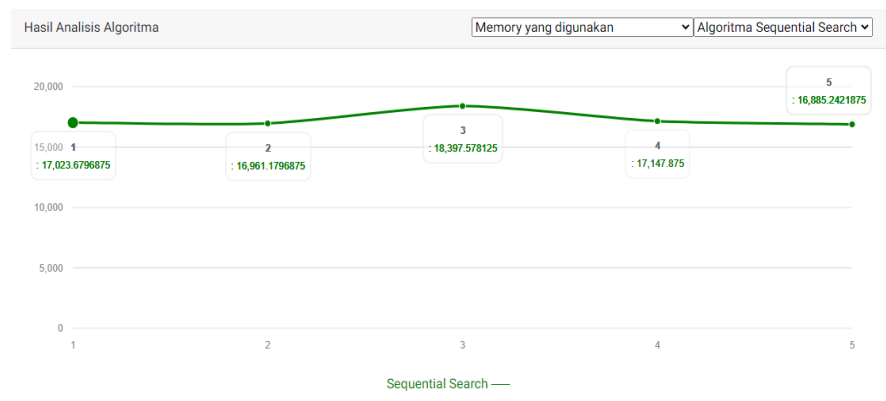
1. Algoritma *Sequential Search*

Table 5.10 Tabel pengujian algoritma sql search pada banyak data

No	Nama File	Kecepatan	Memory
1	Aplikasi Kamus Bahasa Betawi Berbasis Android Menggunakan.pdf	34.622236013412476 s	17023.6796875 KB
2	Analisis_Algoritma_Sequential_Search_Dan.pdf	26.823670864105225 s	16734.1328125 KB
3	Pedoman Skripsi Fti Unibba 2019 02 Desember 2019.Pdf	208.53849601745605 s	18397.578125 KB
4	Analisa Perbandingan Query Pencarian.pdf	51.34688210487366 s	17147.875 KB
5	Binary-search-analysis.pdf	26.074821949005127 s	16885.2421875 KB



Gambar 5.42 Line chart pengujian algoritma *Sequential Search* pada banyak data



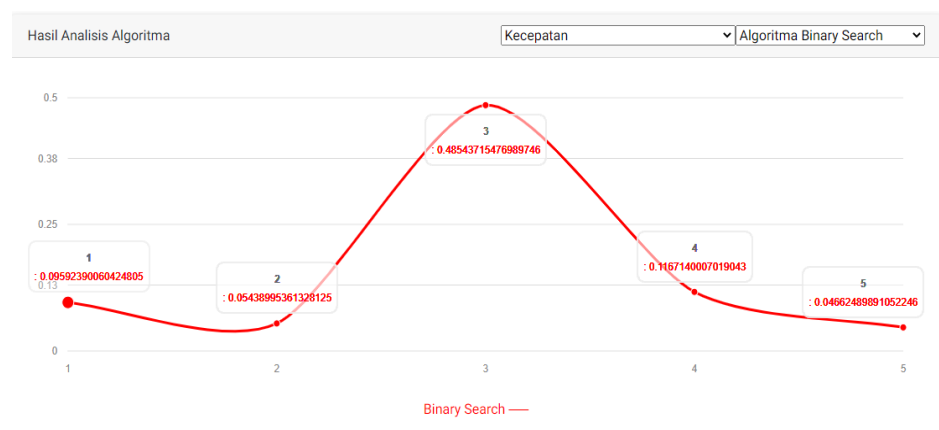
Gambar 5.43 Line chart pengujian memory *Sequential Search* pada banyak data

2. Algoritma *Binary Search*

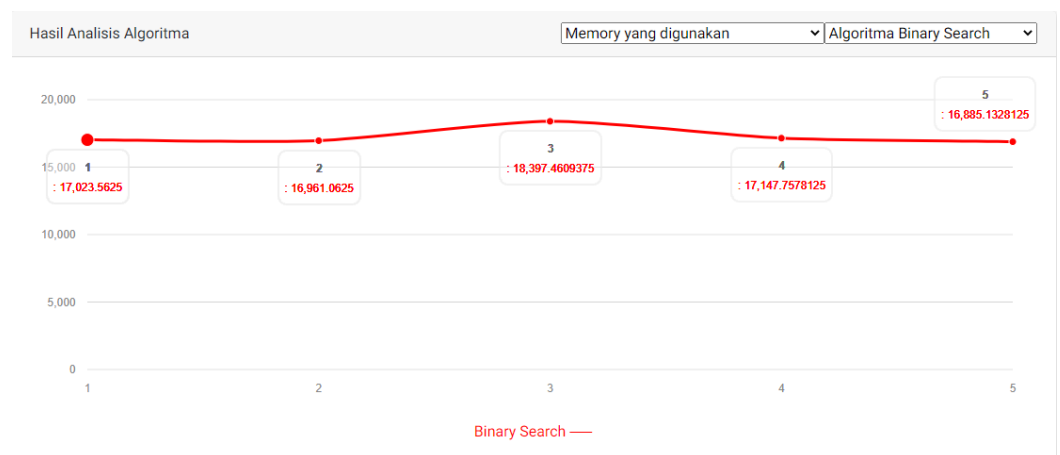
Table 5.11 Tabel pengujian algoritma *Binary Search* pada banyak data

No	Nama File	Kecepatan	Memory
1	Aplikasi Kamus Bahasa Betawi Berbasis Android Menggunakan.pdf	0.09592390060424805 s	17023.5625 KB
2	Analisis_Algoritma_	0.05438995361328125 s	16961.0625 KB

	Sequential_Search_Dan.pdf		
3	Pedoman Skripsi Fti Unibba 2019 02 Desember 2019.pdf	0.48543715476989746 s	18397.4609375 KB
4	Analisa Perbandingan Query Pencarian.pdf	0.1167140007019043 s	17147.7578125 KB
5	Binary-search-analysis.pdf	0.04662489891052246 s	16885.1328125 KB



Gambar 5.44 Line chart pengujian kecepatan algoritma *Binary Search* pada banyak data

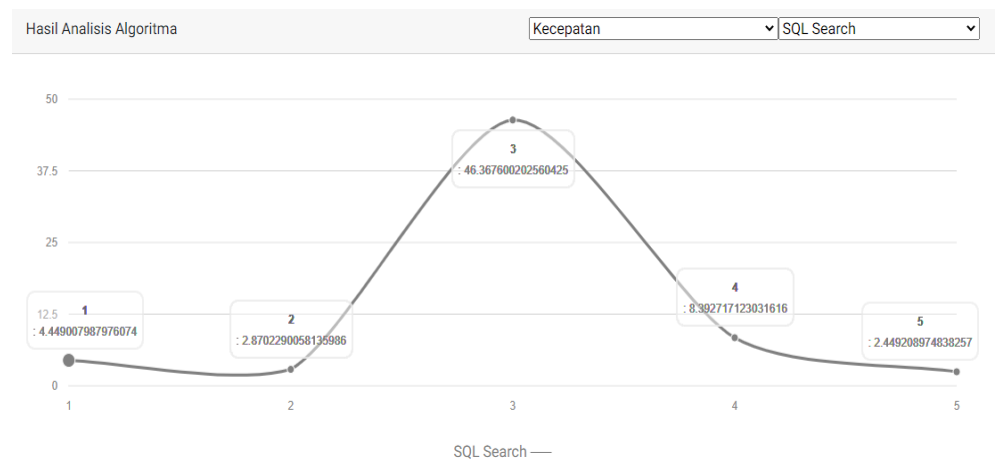


Gambar 5.45 Line chart pengujian memory algoritma *Binary Search* pada banyak data

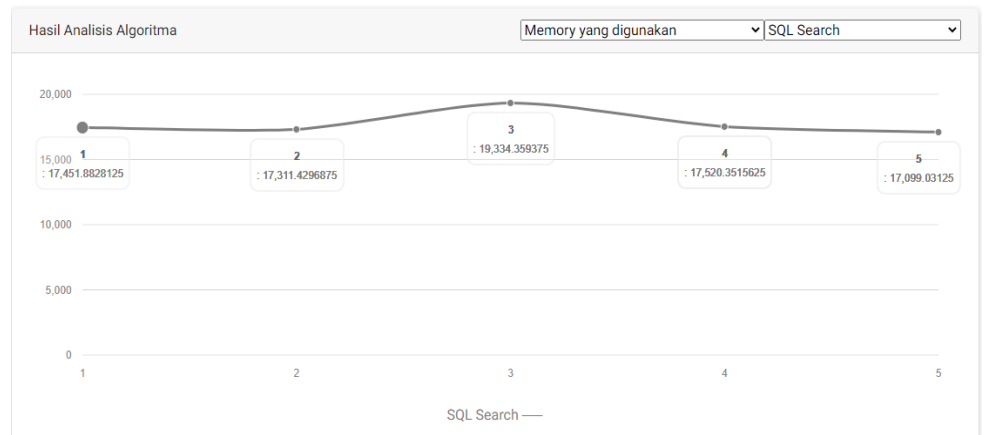
3. SQL Search

Table 5.12 Tabel pengujian algoritma sql search pada banyak data

No	Nama File	Kecepatan	Memory
1	Aplikasi Kamus Bahasa Betawi Berbasis Android Menggunakan.pdf	4.449007987976074 s	17451.8828125 KB
2	Analisis_Algoritma_Sequential_Search_Dan.pdf	2.8702290058135986 s	17311.4296875 KB
3	Pedoman Skripsi Fti Unibba 2019 02 Desember 2019.pdf	46.367600202560425 s	19334.359375 KB
4	Analisa Perbandingan Query Pencarian.pdf	8.392717123031616 s	17520.3515625 KB
5	Binary-search-analysis.pdf	2.449208974838257 s	17099.03125 KB

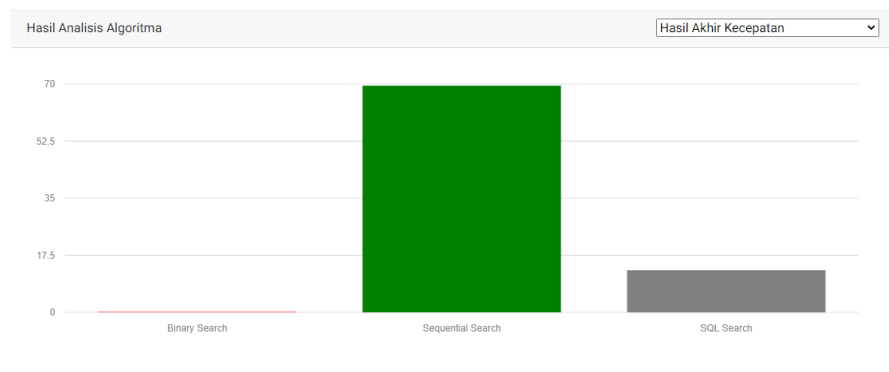


Gambar 5.46 Line chart pengujian kecepatan algoritma sql search pada banyak data



Gambar 5.47 Line chart pengujian memory algoritma sql search pada banyak data

4. Hasil Akhir



Gambar 5.48 Bar chart hasil akhir kecepatan pada banyak data



Gambar 5.49 Bar chart hasil akhir memory pada banyak data

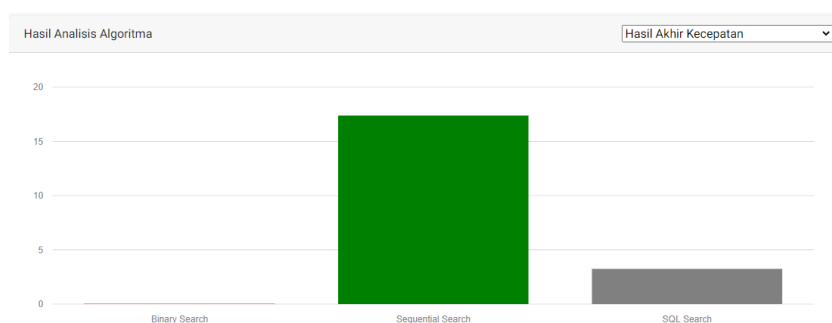
Dari hasil pengujian yang telah dilakukan, masing-masing metode pencarian mendapatkan hasil yang berbeda dari segi kecepatan dan *memory*

yang digunakan. Dari segi kecepatan algoritma *Sequential Search* mendapatkan kecepatan dengan rata-rata sebesar 69.48122138977051, sedangkan algoritma *Binary Search* dengan kecepatan rata-rata sebesar 0.1598179817199707 dan SQL search dengan kecepatan rata-rata sebesar 12.905752658843994. Dari segi kecepatan, *Binary Search* lebih cepat dari kedua metode pencarian lainnya.

Dari segi *memory* yang digunakan, algoritma *Sequential Search* mendapatkan *memory* dengan rata-rata sebesar 17,283.1109375, sedangkan algoritma *Binary Search* dengan *memory* rata-rata sebesar 17,282.9953125 dan SQL search dengan *memory* rata-rata sebesar 17,743.4109375. Dari segi *memory* yang digunakan, sql search lebih besar menggunakan *memory* daripada 2 metode pencarian lainnya.

5.3.2. Hasil Pengujian

Setelah melakukan pengujian terhadap 3 metode pencarian, didapat hasil analisis kecepatan dan *memory* yang dilakukan masing-masing metode pencarian. Masing-masing metode pencarian melakukan 5 kali pencarian pada 4 pengujian. Berikut adalah hasil akhir pengujian :



Gambar 5.50 Hasil akhir pengujian pada kecepatan pencarian

Pada kecepatan pencarian didapat algoritma *Binary Search* dengan kecepatan pencarian tercepat dengan rata-rata kecepatan 0.04056605100631714, pencarian tercepat kedua yaitu sql search dengan

rata-rata kecepatan 3.2470962047576903 dan yang ketiga yaitu algoritma *Sequential Search* dengan rata-rata kecepatan 17.37859684228897.

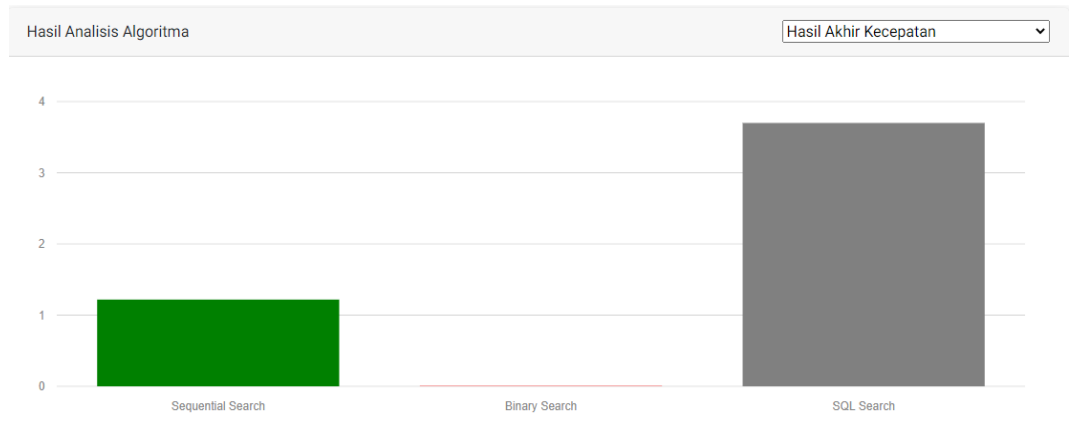


Gambar 5.51 Hasil akhir pengujian pada memory pencarian

Pada memory pencarian didapat algoritma *Binary Search* dengan penggunaan memory paling sedikit dengan rata-rata memory 16,858.93828125, sedangkan memory paling sedikit kedua yaitu sequential seacrh dengan rata-rata memory 16,859.055078125 dan yang ketiga yaitu sql search dengan rata-rata memory 16,986.55390625.

5.3.3. Hasil Pengujian Oleh Mahasiswa

Selain penulis yang menguji ketiga metode pencarian, penulis membuat fitur untuk mahasiswa bisa ikut kontribusi pada pengujian kecepatan dan *memory* yang digunakan terhadap 3 metode pencarian. Berikut adalah hasil akhir pengujian :



Gambar 5.52 Hasil akhir pengujian oleh mahasiswa pada kecepatan pencarian



Gambar 5.53 Hasil akhir pengujian oleh mahasiswa pada memory pencarian

BAB VI

KESIMPULAN DAN SARAN

6.1. Kesimpulan

Berdasarkan penelitian yang dilakukan penulis melalui beberapa tahapan yang dilakukan pada bab-bab sebelumnya, maka penulis dapat menyimpulkan bahwa:

1. Dengan membandingkan kedua algoritma dan metode pencarian yang sering digunakan yaitu SQL search, penulis jadi lebih mudah untuk menentukan metode pencarian yang akan digunakan di aplikasi kamus Bahasa Indonesia, karena mendapatkan perbandingan dari segi kecepatan dan *memory* yang digunakan. Hasil perbandingan dari 4 pengujian pada 3 metode pencarian, pada kecepatan pencarian didapat algoritma *Binary Search* dengan kecepatan pencarian tercepat dengan rata-rata kecepatan 0.04056605100631714 s, pencarian tercepat kedua yaitu sql search dengan rata-rata kecepatan 3.2470962047576903 s dan yang ketiga yaitu algoritma *Sequential Search* dengan rata-rata kecepatan 17.37859684228897 s. Sedangkan pada memory pencarian didapat algoritma *Binary Search* dengan penggunaan memory paling sedikit dengan rata-rata memory 16,858.93828125 KB, sedangkan memory paling sedikit kedua yaitu sequential seacrh dengan rata-rata memory 16,859.055078125 KB dan yang ketiga yaitu sql search dengan rata-rata memory 16,986.55390625 KB.
2. Dari perbandingan yang sudah dilakukan ternyata penggunaan algoritma pada proses pencarian data sangat berpengaruh pada kecepatan pencarian yang dilakukan. Hal ini terlihat pada penggunaan algoritma *Binary Search* yang mempercepat pencarian kata pada aplikasi kamus Bahasa Indonesia.

6.2. Saran

Berdasarkan hasil penelitian, maka peneliti merekomendasikan atau menyarankan beberapa hal, yaitu :

1. Membandingkan dengan algoritma atau metode pencarian lainnya, karena masih banyak algoritma pencarian yang dapat dibandingkan.
2. Menambahkan fitur terjemahan Bahasa Inggris pada aplikasi kamus sehingga user bisa mencari kata dengan Bahasa Inggris dengan definisi Bahasa Indonesia.
3. Aplikasi ini bisa dikembangkan menjadi aplikasi kamus menggunakan algoritma hasil dari penelitian atau menjadi media pembelajaran algoritma pada aplikasi kamus pada user yang ingin melihat perbandingan dari berbagai algoritma pencarian.

DAFTAR PUSTAKA

- Adi, A. P. (2019). *Membuat Website Cantik & Menarik dengan jQuery*.
- Aziz, I., & Harafani, H. (2016). Aplikasi Kamus Bahasa Betawi Berbasis Android Menggunakan Metode Sequential Search. *Penelitian Ilmu Komputer Sistem Embedded Dan Logic*, 4(1), 27–35. <https://doi.org/10.33558/piksel>
- Hidayatullah, P., & Kawistara, J. K. (2017). *Pemrograman Web Edisi Revisi. JavaScript*. (2019). <https://developer.mozilla.org/id/docs/Web/JavaScript>
- Kaban, R. (2019). *Bootstrap CSS Framework*.
- Kamus Bahasa Indonesia*. (2008). Pusat Bahasa Departemen Pendidikan Nasional Jakarta.
- Munir, R., & Leony, L. (2016). *Algoritma Dan Pemrograman Dalam Bahasa Pascal, C, Dan C++*.
- Religia, Y., Rohim, M. F., & Hapsari, R. (2019). Analisis Algoritma Sequential Search Dan Binary Search Pada Big Data. *Jurnal Ilmiah Informatika, Arsitektur Dan Lingkungan*, 14(1), 74–79.
- Umar, R., Riadi, I., & Muthohirin, B. F. (2017). Implementasi Algoritma Binary Search Pada Aplikasi Konkordansi Al-Qur'an Berbasis Android. *Seminar Nasional Teknologi Informasi Dan Komunikasi - SEMANTIKOM*, 49–54.

LAMPIRAN

Lampiran 1 Data Pengujian

1. Nama : Sabda Alam
NIM : C1A160028
Email : insabda@gmail.com

Pengujian	Kecepatan Sequence	Kecepatan Binary	Kecepatan SQL	Memory Sequence	Memory Binary	Memory SQL
1	0.0002579 68902587 8906 s	0.000218 1529998 779297 s	0.0235648 15521240 234 s	2268.41 40625 KB	2241.1 71875 KB	2323.703 125 KB
2	0.0106768 60809326 172 s	0.002682 9242706 29883 s	0.0221660 13717651 367 s	2240.00 78125 KB	2240.0 078125 KB	2240.171 875 KB
3	0.0016191 00570678 711 s	0.000220 0603485 107422 s	0.0190210 34240722 656 s	2240.00 78125 KB	2240.0 078125 KB	2240.179 6875 KB
4	3.9843068 12286377 s	0.014483 2134246 82617 s	5.2845439 91088867 s	2669.13 28125 KB	2669.1 328125 KB	2653.617 1875 KB

2. Nama : Muhamad Hanif Ridwannulloh
NIM : C1A160053
Email : hanifridwan18@gmail.com

Pengujian	Kecepatan Sequence	Kecepatan Binary	Kecepatan SQL	Memory Sequence	Memory Binary	Memory SQL
1	0.00032615 6616210937 5 s	0.000212 9077911 376953 s	0.000437 02125549 316406 s	2239.796 875 KB	2239.79 6875 KB	2240.0 234375 KB

2	0.00162100 7919311523 4 s	0.000183 8207244 873047 s	0.000451 08795166 015625 s	2239.867 1875 KB	2239.86 71875 KB	2240.0 3125 KB
3	0.00278496 7422485351 6 s	0.000231 0276031 4941406 s	0.000451 08795166 015625 s	2239.867 1875 KB	2239.86 71875 KB	2240.0 390625 KB
4	1.90460801 12457275 s	0.006625 8907318 115234 s	2.119887 11357116 7 s	2482.5 KB	2482.5 KB	2485.5 15625 KB

3. Nama : Galih Raxy Hakiki
 NIM : C1A160009
 Email : galihrexyhakiki@gmail.com

Pengujian	Kecepatan Sequence	Kecepatan Binary	Kecepatan SQL	Memory Sequence	Memory Binary	Memory SQL
1	0.000227 9281616 2109375 s	0.000243 9022064 2089844 s	0.00106382 369995117 19 s	2239.796 875 KB	2239.79 6875 KB	2240.02 34375 KB
2	0.001022 8157043 457031 s	0.000292 0627593 9941406 s	0.00187301 635742187 5 s	2239.804 6875 KB	2239.80 46875 KB	2239.96 875 KB
3	0.002016 0675048 828125 s	0.000302 0763397 216797 s	0.00040483 474731445 31 s	2239.804 6875 KB	2239.80 46875 KB	2239.97 65625 KB
4	2.379453 1822204 59 s	0.007033 1096649 16992 s	3.09999895 0958252 s	2608.937 5 KB	2608.93 75 KB	2557.37 5 KB

4. Nama : Muhammad Jamil Zainu Noor
 NIM : C1A160029
 Email : wangisagara178@gmail.com

Pengujian	Kecepatan Sequence	Kecepatan Binary	Kecepatan SQL	Memory Sequence	Memory Binary	Memory SQL
1	0.000201 2252807 6171875 s	0.000196 9337463 3789062 s	0.01134300 231933593 8 s	2239.79 6875 KB	2239.79 6875 KB	2240.02 34375 KB
2	0.001470 8042144 77539 s	0.000233 8886260 986328 s	0.00051403 045654296 88 s	2239.86 71875 KB	2239.86 71875 KB	2240.03 125 KB
3	0.001760 9596252 441406 s	0.001546 8597412 109375 s	0.00052309 036254882 81 s	2239.86 71875 KB	2239.86 71875 KB	2240.03 90625 KB
4	1.466041 0881042 48 s	0.005129 8141479 49219 s	1.89088821 41113281 s	2438.57 03125 KB	2438.57 03125 KB	2418.50 78125 KB

5. Nama : Ahmad Kamal Fasya
NIM : C1A160013
Email : fasyaaoi9@gmail.com

Pengujian	Kecepatan Sequence	Kecepatan Binary	Kecepatan SQL	Memory Sequence	Memory Binary	Memory SQL
1	0.000200 0331878 6621094 s	0.000217 1993255 6152344 s	0.02401614 189147949 2 s	2239.79 6875 KB	2239.7 96875 KB	2240.023 4375 KB
2	0.006189 8231506 34766 s	0.000774 8603820 800781 s	0.08438110 3515625 s	2239.80 46875 Kb	2239.8 046875 Kb	2239.968 75 Kb
3	0.001654 1481018 066406 s	0.000222 2061157 2265625 s	0.02352404 594421386 7 s	2239.80 46875 KB	2239.8 046875 KB	2239.976 5625 KB
4	14.59964 7998809 814 s	0.093561 1724853 5156 s	61.3817250 7286072 s	3918.83 59375 KB	3918.8 359375 KB	3775.289 0625 KB

Lampiran 2 *Source code* aplikasi**Source code index**

```

<?php require_once 'header.php'; ?>
<div class="container mt-3">
    <div class="my-2">
        <button class="btn btn-primary btn-sm" id="pilih-kata"><i class="fas fa-
spell-check"></i> Kata</button>
        <button class="btn btn-primary btn-sm" id="pilih-file"><i class="far fa-
file"></i></i> File</button>
        <a href="analisis" class="btn btn-success btn-sm float-right"><i class="far fa-
fa-chart-bar"></i> Analisis</a>
    </div>

    <div class="row justify-content-center mt-5">
        <div id="upload-file" class="col-md-6" style="display: none">
            <form method="POST" enctype="multipart/form-data" id="import-file">

                <div class="input-group mb-3">
                    <div class="input-group-prepend">
                        <span class="input-group-text">Upload</span>
                    </div>
                    <div class="custom-file">
                        <input type="file" class="custom-file-input" name="pdf" id="nama-
file" aria-describedby="inputGroupFileAddon01">
                        <label class="custom-file-label" for="inputGroupFile01">Pilih file
.pdf</label>
                    </div>
                </div>
                <div class="text-center">
                    <button type="submit" name="upload" class="btn btn-primary btn-
sm" id="submit-file">Upload</button>
                </div>
            </form>
        </div>
    </div>

    <div class="row mt-3">
        <div class="col-md-6">
            <select id="pilih-algoritma">
                <option value="" disabled selected>--- Pilih Metode Pencarian ---
</option>
                <option value="ss">Algoritma Sequential Search</option>
                <option value="bs">Algoritma Binary Search</option>
                <option value="sql">SQL Search</option>
            </select>

```

```

    </div>
  </div>

  <div class="row mt-2">
    <div class="col-md-6 mb-3" style="margin-right: 0 !important; padding:
1px !important;">
      <div class="card">
        <div class="card-header text-center">Pencarian Kata</div>
        <div class="card-body" id="tempat-cari" style="min-height: 300px;
display: none;">

          </div>
          <div class="card-body" id="card-cari">
            <form method="POST" id="form-cari">
              <div class="md-form">
                <textarea name="kata" class="md-textarea form-control"
id="textarea-kata" rows="5"></textarea>
                <label for="form7">Cari Kata</label>
              </div>
              <button type="submit" id="tombol-cari" class="btn btn-primary
btn-sm">Cari</button>
            </form>
          </div>
        </div>
      </div>
      <div class="col-md-6" style="margin-left: 0 !important; padding: 1px
!important;">
        <div class="card">
          <div class="card-header text-center">Kata yang ditemukan</div>
          <div class="card-body" style="min-height: 300px">
            <div>
              <div id="hasil_waktu"></div><br>
              <div id="hasil_kata"></div>
            </div>
          </div>
        </div>
      </div>
    </div>
  </div>
  <!-- Modal -->
  <div class="modal fade" id="exampleModal" tabindex="-1" role="dialog" aria-
labelledby="exampleModalLabel" aria-hidden="true">
    <div class="modal-dialog" role="document">
      <div class="modal-content">
        <div class="modal-header">
          <h5 class="modal-title" id="judul"></h5>
          <button type="button" class="close" data-dismiss="modal" aria-
label="Close">

```



```

        <span aria-hidden="true">&times;</span>
    </button>
</div>
<div class="modal-body" id="data-kata">
    <p id="judul-kata"></p>
    <p id="definisi-kata"></p>
    <p id="sumber-kata"></p>
</div>
<div class="modal-footer">
    <button type="button" class="btn btn-secondary" data-
dismiss="modal">Close</button>
</div>
</div>
</div>
</div>
<script src="assets/js/jquery.min.js"></script>
<script src="assets/js/popper.min.js"></script>
<script src="assets/js/bootstrap.min.js"></script>
<script src="assets/js/mdb.min.js"></script>
<script src="assets/js/dark-mode-switch.min.js"></script>
<script>
function getClassNama(namakelas)
{
    if(namakelas){
        let klik = namakelas;
        $.ajax({
            url: 'proses_ajax.php',
            type: 'post',
            data: {kata: klik},
            dataType: 'json',
            cache: false,
            success:function(response){
                $('#data-kata').html("");
                $.each(response, function(i, data){
                    $('#judul').html(data.kata);
                    $('#data-kata').append(
                        '<p>'+data.definisi+'</p>'+
                        '<small> Sumber : '+data.sumber+'</small> <hr>'
                    );
                });
                $('#exampleModal').modal('show');
            }
        });
    }
}
function simpanDataPerbandingan(tipe, data)
{
    let hasil = {hasil: data, tipe:tipe};

```

```

$.ajax({
    url: 'simpan_data_perbandingan.php',
    type: 'post',
    data: hasil,
    cache: false,
    success: function(response){
    }
});
}

$(document).ready(function(){
    getClassName();
    $('#pilih-kata').click(function(){
        $('#cari-kata').show();
        $('#upload-file').hide();
        $('#tempat-cari').html("");
        $('#hasil_kata').html("");
        $('#card-cari').show();
        $('#tempat-cari').hide();
        // $('#textarea-kata').val("");
    });
    $('#pilih-file').click(function(){
        $('#cari-kata').hide();
        $('#upload-file').show();
        $('#tempat-cari').html("");
        $('#hasil_kata').html("");
        $('#tempat-cari').show();
        $('#card-cari').hide();
    });
    let link;
    let upload;
    $('#tombol-cari').hide();
    $('#submit-file').hide();
    $('#pilih-algoritma').change(function(){
        if( $(this).val() == 'ss' ){
            link = 'cari-kata/cariSequential.php';
            upload = 'upload.php?algoritma=ss';
            $('#tombol-cari').show();
            $('#submit-file').show();
        }else if( $(this).val() == 'bs' ){
            link = 'cari-kata/cariBinary.php';
            upload = 'upload.php?algoritma=bs';
            $('#tombol-cari').show();
            $('#submit-file').show();
        }else{
            link = 'cari-kata/cariSql.php';
            upload = 'upload.php?algoritma=sql';
            $('#tombol-cari').show();
        }
    });

```

```

        $('#submit-file').show();
    }
});
$('#form-cari').on('submit', function(e){
    e.preventDefault();
    $.ajax({
        url: link,
        method: 'POST',
        contentType: false,
        cache:false,
        processData:false,
        dataType: 'json',
        data: new FormData(this),
        beforeSend: function(){
            $('#hasil_kata').html("<h5>Mencari...</h5>");
        },
        success: function(response){
            if(response.status == 0){
                $('#hasil_kata').html(response.hasil_unik);
            }else{
                $('#hasil_waktu').html(response.hasil_waktu);
                $('#hasil_kata').html("<h5>" + response.hasil_unik + "</h5>");
                simpanDataPerbandingan('kata', response.data_perbandingan);
            }
        }
    });
});
let kata = $('#hasil_kata').text();
let array_kata = kata.split(" ");
$('#import-file').on('submit', function(e){
    e.preventDefault();
    $.ajax({
        url: link,
        method: 'POST',
        contentType: false,
        cache:false,
        processData:false,
        dataType: 'json',
        data: new FormData(this),
        beforeSend: function(){
            $('#submit-file').html('Loading...');
            $('#submit-file').attr('disabled', 'disabled');
        },
        success: function(response){
            $('#custom-file-label').text('Pilih File');
            $('#submit-file').html('Upload');
            $('#submit-file').attr('disabled', false);
            $('#tempat-cari').html("<h5>" + response.cari + "</h5>");
        }
    });
});

```

```

        $('#hasil_waktu').html("<h6>" + response.hasil_waktu + "</h6>");
        $('#hasil_kata').html("<h5>" + response.hasil_unik + "</h5>");
        simpanDataPerbandingan('file', response.data_perbandingan);
    }
    });
});
});
</script>
</body>
</html>

```

Source code header cari kata

```

<?php
    ini_set('max_execution_time', 1000);
    include('../class.pdf2text.php');
    require_once "../algoritma.php";
    require_once "../koneksi.php";
    $cari = "";
    $kata = [];
    $data = [];
    $hasil = [];
    $sql = "SELECT * FROM kamus ORDER BY kata ASC";
    $query = mysqli_query($koneksi, $sql);
    while($row = mysqli_fetch_assoc($query)){
        $kata[] = $row['kata'];
    }
    if(isset($_FILES['pdf']['name'])){
        $nama_file = $_FILES['pdf']['name'];
        $target = basename($_FILES['pdf']['name']);
        move_uploaded_file($_FILES['pdf']['tmp_name'], $target);
        chmod($_FILES['pdf']['name'], 0777);
        $var = new PDF2Text();
        $var->setFilename($_FILES['pdf']['name']);
        $var->decodePDF();
        $cari = preg_replace('/[^A-Za-z0-9 ]/', " ", trim(strtolower($var-
>output())));
        $data = explode(" ", $cari);
        unlink($nama_file);
    }else{
        $cari = preg_replace('/[^A-Za-z0-9 ]/', " ",
trim(strtolower($_POST['kata'])));
        $data = explode(" ", $cari);
    } ?>

```

Source code footer cari kata

```

<?php
    if(count($hasil) == 0){
        $tidak_ditemukan = '<span>Kata tidak ditemukan</span>';
        $output = [
            'status' => 0,
            'hasil_unik' => $tidak_ditemukan,
        ];
        echo json_encode($output);
    }else{
        $final_kata_ditemukan = [];
        $kata_ditemukan = array_unique($hasil);
        $jumlah_kata = count($hasil);
        foreach($kata_ditemukan as $ditemukan) $final_kata_ditemukan[]
= '<span style="cursor:pointer;"
onclick="getClassName('.$ditemukan.'')"><u>'.$ditemukan.'</u></span>';
        $final_hasil_unik = implode(" ", $final_kata_ditemukan);
        $hasil_waktu = "Algoritma ".$algoritma." <br> Jumlah Kata
".$jumlah_kata." <br> Waktu pencarian ".$kecepatan." detik <br> Memory yang
digunakan ".$memory." KB <br> Maksimum Memory
".memory_get_peak_usage(true);
        if(isset($_FILES['pdf']['name'])){
            $data_perbandingan = [
                'algoritma' => $algoritma,
                'memory' => $memory,
                'waktu' => $kecepatan,
                'file' => $nama_file,
                'jumlah_kata' => count($data)
            ];

            $output = [
                'status' => 1,
                'data_perbandingan' => $data_perbandingan,
                'cari' => $cari,
                'hasil_unik' => $final_hasil_unik,
                'hasil_waktu' => $hasil_waktu,
            ];
        }else{
            $data_perbandingan = [
                'algoritma' => $algoritma,
                'array_hasil' => $hasil,
                'memory' => $memory,
                'waktu' => $kecepatan,
                'cari' => $data,
            ];

            $output = [
                'status' => 1,

```

```

        'data_perbandingan' => $data_perbandingan,
        'hasil_unik' => $final_hasil_unik,
        'hasil_waktu' => $hasil_waktu,
    ];
    }
    echo json_encode($output);
}
?>

```

Source code cari kata *Sequential Search*

```

<?php
    require 'header.php';
    $start = microtime(true);
    usleep(100);
    $hasil = sequentialSearch($kata, $data);
    $end = microtime(true);
    $kecepatan = $end - $start;
    $memory = memorySequentialSearch($kata, $data);
    $algoritma = 'Sequential Search';
    require 'footer.php';
?>

```

Source code cari kata *Binary Search*

```

<?php
    require 'header.php';
    $start = microtime(true);
    usleep(100);
    $hasil = binarySearch($kata, $data);
    $end = microtime(true);
    $kecepatan = $end - $start;
    $memory = memoryBinarySearch($kata, $data);
    $algoritma = 'Binary Search';
    require 'footer.php';
?>

```

Source code cari kata sql search

```

<?php
    require 'header.php';
    $start = microtime(true);
    usleep(100);
    $hasil = sqlSearch($koneksi, $data);
    $end = microtime(true);
    $kecepatan = $end - $start;
    $memory = memorySqlSearch($koneksi, $data);
    $algoritma = 'SQL Search';
    require 'footer.php';
?>

```

Souce code simpan data perbandingan

```

<?php
    require_once "koneksi.php";
    require_once "algoritma.php";
    if($_POST['tipe'] == 'kata'){
        $hasil = $_POST['hasil'];
        $kata_ditemukan = array_unique($hasil['array_hasil']);
        $kata_dicari = implode(",", $kata_ditemukan);
        $jumlah_kata = count($hasil['array_hasil']);
        $algoritma = $hasil['algoritma'];
        $waktu = $hasil['waktu'];
        $memory = $hasil['memory'];
        $sql = "SELECT * FROM kamus";
        $query = mysqli_query($koneksi, $sql);
        while($row = mysqli_fetch_assoc($query)){
            $kata[] = $row['kata'];
        }
        if($algoritma == 'Sequential Search'){
            $output = simpanDataSequentialSearch($kata,
$hasil['cari']);
        }elseif($algoritma == 'Binary Search'){
            $output = simpanDataBinarySearch($kata, $hasil['cari']);
        }else{
            $output = simpanDataSqlSearch($koneksi, $hasil['cari']);
        }
        $jumlah_perbandingan =
mysqli_num_rows(mysqli_query($koneksi, "SELECT * FROM perbandingan"));
        $sql_perbandingan = "INSERT INTO perbandingan(pengujian,
kata, algoritma, waktu, memory, jumlah_kata, detail)
VALUES($jumlah_perbandingan+1, '$kata_dicari', '$algoritma', '$waktu',
'$memory', '$jumlah_kata', '$output')";
        mysqli_query($koneksi, $sql_perbandingan);
    }else{
        $hasil = $_POST['hasil'];
        $kata_dicari = $hasil['file'];
        $jumlah_kata = $hasil['jumlah_kata'];
        $algoritma = $hasil['algoritma'];
        $waktu = $hasil['waktu'];
        $memory = $hasil['memory'];
        $output = "";
        $jumlah_perbandingan =
mysqli_num_rows(mysqli_query($koneksi, "SELECT * FROM perbandingan"));
        $sql_perbandingan = "INSERT INTO perbandingan(pengujian,
kata, algoritma, waktu, memory, jumlah_kata, detail)
VALUES($jumlah_perbandingan+1, '$kata_dicari', '$algoritma', '$waktu',
'$memory', '$jumlah_kata', '$output')";
        mysqli_query($koneksi, $sql_perbandingan);
    }
}

```

```
    }
?>
```

Source code header pengujian algoritma

```
<?php
    ini_set('max_execution_time', 1000);
    include('../class.pdf2text.php');
    require_once "../koneksi.php";
    require_once "../algoritma.php";
    $sql = "SELECT * FROM kamus ORDER BY kata ASC";
    $query = mysqli_query($koneksi, $sql);
    while($row = mysqli_fetch_assoc($query)){
        $kata[] = $row['kata'];
    }
    if(isset($_FILES['pdf']['name'])){
        $nama_file = $_FILES['pdf']['name'];
        $target = basename($_FILES['pdf']['name']);
        move_uploaded_file($_FILES['pdf']['tmp_name'], $target);
        chmod($_FILES['pdf']['name'], 0777);
        $var = new PDF2Text();
        $var->setFilename($_FILES['pdf']['name']);
        $var->decodePDF();
        $cari = preg_replace('/[^A-Za-z0-9 ]/', " ", trim(strtolower($var-
>output())));
        $data = explode(" ", $cari);
    }else{
        $cari = preg_replace('/[^A-Za-z0-9 ]/', " ",
trim(strtolower($_POST['kata'])));
        $data = explode(" ", $cari);
    }
?>
```

Source code pengujian algoritma *Sequential Search*

```
<?php
    require 'header.php';
    $start = microtime(true);
    usleep(100);
    $hasil = sequentialSearch($kata, $data);
    $end = microtime(true);
    $kecepatan_sequential = $end - $start;
    $memory_sequential = memorySequentialSearch($kata, $data);
    $json = [
        'kecepatan_sequential' => $kecepatan_sequential,
        'memory_sequential' => $memory_sequential,
    ];
    echo json_encode($json);
?>
```


Source code pengujian algoritma *Binary Search*

```
<?php
require 'header.php';
$start = microtime(true);
usleep(100);
$hasil = binarySearch($kata, $data);
$end = microtime(true);
$kecepatan_binary = $end - $start;
$memory_binary = memoryBinarySearch($kata, $data);
$json = [
    'kecepatan_binary' => $kecepatan_binary,
    'memory_binary' => $memory_binary,
];
echo json_encode($json);
?>
```

Source code pengujian algoritma sql search

```
<?php
require 'header.php';
$start = microtime(true);
usleep(100);
$hasil = sqlSearch($koneksi, $data);
$end = microtime(true);
$kecepatan_sql = $end - $start;
$memory_sql = memorySqlSearch($koneksi, $data);
$json = [
    'kecepatan_sql' => $kecepatan_sql,
    'memory_sql' => $memory_sql,
];
if(isset($_FILES['pdf']['name'])) unlink($nama_file);
echo json_encode($json);
?>
```

Source code proses ajax definisi kata

```
<?php
require_once "koneksi.php";
$cari = "";
$kata = [];
$data = [];
$sql = "SELECT * FROM kamus WHERE kata = '$_POST[kata]'";
$query = mysqli_query($koneksi, $sql);
while($row = mysqli_fetch_assoc($query)){
    $hasil[] = [
        'id' => $row['id'],
        'kata' => $row['kata'],
        'definisi' => $row['definisi'],
        'sumber' => $row['sumber'],
    ];
}
```

```

        ];
    }
    echo json_encode($hasil);
?>

```

Source code get analisis

```

<?php
    require_once "koneksi.php";
    $table = "";
    $pengujian = isset($_POST['pengujian']) ? $_POST['pengujian'] : "";
    if($pengujian == 'awal')
        $sql = "WHERE pengujian BETWEEN '1' AND '15'";
    elseif($pengujian == 'tengah')
        $sql = "WHERE pengujian BETWEEN '16' AND '30'";
    elseif($pengujian == 'akhir')
        $sql = "WHERE pengujian BETWEEN '31' AND '45'";
    elseif($pengujian == 'banyak')
        $sql = "WHERE pengujian BETWEEN '46' AND '60'";
    else
        $sql = "";

    $query_table = mysqli_query($koneksi, "SELECT * FROM perbandingan
    $sql ORDER BY pengujian ASC");
    if(mysqli_num_rows($query_table) == 0){
        $table .= '<td colspan="7" class="text-center">Tidak ada data</td>';
    }

    while($row = mysqli_fetch_assoc($query_table)) {
        $table .= '<tr>
            <td class="text-center">'.$row['pengujian'].'</td>
            <td>'.substr($row['kata'], 0, 50).</td>
            <td>'.$row['algoritma'].'</td>
            <td class="text-center">'.$row['waktu'].'</td>
            <td class="text-center">'.$row['memory'].'</td>
            <td class="text-center">'.$row['jumlah_kata'].'</td>
            <td class="text-center">
                <button class="btn btn-success btn-sm btn-detail"
data-detail="'. $row['pengujian'].'">Detail</button>
                <a onclick="return confirm('."Apa anda yakin?"')"
href="hapus_perbandingan.php?id='. $row['id'].'" class="btn btn-danger btn-
sm">Hapus</a>
            </td>
        </tr>';
    }

    echo $table;
?>

```

Source code analisis

```

<?php
    require_once "header.php";
?>

<!-- BAR CHART -->
<div class="container mt-5 mb-5">
    <select id="pilih_pengujian" class="mb-3">
        <option value="" disabled selected>Pilih
Pengujian</option>
        <option value="semua">Semua Pengujian</option>
        <option value="awal">Pengujian Awal Kata</option>
        <option value="tengah">Pengujian Tengah Kata</option>
        <option value="akhir">Pengujian Akhir Kata</option>
        <option value="banyak">Pengujian Banyak Kata</option>
    </select>

    <div class="card">
        <div class="card-header">
            Hasil Analisis Algoritma
            <select id="pilih-algoritma" class="float-right"
style="display: none">
                <option value="" disabled selected>Pilih Metode
Pencarian</option>
                <option value="semua">Semua Metode</option>
                <option value="ss">Algoritma Sequential Search</option>
                <option value="bs">Algoritma Binary Search</option>
                <option value="sql">SQL Search</option>
            </select>

            <select id="pilih-perbandingan"
name="perbandingan" class="float-right" style="display: none">
                <option value="" disabled selected>Pilih perbandingan</option>
                <option value="kecepatan">Kecepatan</option>
                <option value="memory">Memory yang digunakan</option>
                <option value="hasil_kecepatan">Hasil Akhir
Kecepatan</option>
                <option value="hasil_memory">Hasil Akhir Memory yang
digunakan</option>
            </select>
        </div>
        <div class="card-body chart-responsive">
            <div class="chart" id="bar-chart" style="height:
350px;"></div>
            <div id="legend" class="text-center"></div>
        </div>
    </div><br>

```

```

<div class="card">
  <div class="card-header">
    Table Pengujian
    <a onclick="return confirm('Apa anda yakin ingin
menghapus semua data?')" href="hapus_perbandingan.php" class="btn btn-danger
btn-sm float-right">Hapus Semua</a>
  </div>
  <table class="table table-bordered">
    <thead>
      <tr>
        <th class="text-
center">Pengujian</th>
        <th>Kata yang dicari</th>
        <th>Algoritma</th>
        <th class="text-
center">Kecepatan</th>
        <th class="text-
center">Memory</th>
        <th class="text-center">Jumlah
Kata</th>
        <th class="text-center">Aksi</th>
      </tr>
    </thead>
    <tbody id="table_analisis">

    </tbody>
  </table>
</div>

<div id="detail" class="mt-4 card" style="display: none;">
  <div class="card-header">
    Detail
    <button id="clear" class="btn btn-danger btn-sm
float-right">Tutup</button>
  </div>
  <div class="card-body">
    <div class="row">

    </div>
  </div>
</div>
</div>

<script src="assets/js/jquery.min.js"></script>
<script src="assets/js/popper.min.js"></script>
<script src="assets/js/bootstrap.min.js"></script>
<script src="assets/js/mdb.min.js"></script>

```

```

<script src="assets/js/dark-mode-switch.min.js"></script>
<script src="assets/vendor/raphael.js/raphael.min.js"></script>
<script src="assets/vendor/morris.js/morris.js"></script>
<script>

    function moveTo(el, options, x, y)
    {
        options.append(el);
        var hoverHeight, hoverWidth, left, parentHeight, parentWidth,
top;

        parentWidth = options.innerWidth();
        parentHeight = options.innerHeight();
        hoverWidth = el.outerWidth();
        hoverHeight = el.outerHeight();

        left = Math.min(Math.max(0, x - hoverWidth / 2), parentWidth -
hoverWidth);
        if (y != null) {
            top = y - hoverHeight - 10;
            if (top < 0) {
                top = y + 10;
                if (top + hoverHeight > parentHeight) {
                    top = parentHeight / 2 - hoverHeight / 2;
                }
            }
        } else {
            top = parentHeight / 2 - hoverHeight / 2;
        }

        el.css({
            left: left + "px",
            top: parseInt(top) + "px"
        });

        return;
    }

    function chart_line_semua(data)
    {
        $('#bar-chart').html("");
        $('#legend').html("");
        let chart = new Morris.Area({
            element: 'bar-chart',
            resize: true,
            data: data,
            lineColors: ['green', 'red', 'gray'],
            xkey: 'y',
            ykeys: ['ss', 'bs', 'sql'],

```

```

        labels: [",", ","],
        fillOpacity: 0.5,
        hideHover: false,
        parseTime: false,
        hoverCallback: function (index, options, content, row, x , y) {
default_html = "<div class='morris-hover morris-default-style'>
</div>";

        default_html = $(default_html).append(content);
        moveTo($(default_html), $('#bar-chart'), x, y);
        return "";
    }

    });

    chart.options.labels.forEach(function(label, i){
        var legendItem =
        $('<span></span>').text(label).css('color', chart.options.lineColors[i])
        $('#legend').append(legendItem)
    });
}

function chart_line(json)
{
    $('#bar-chart').html("");
    $('#legend').html("");
    let default_html = "";
    let label = json.perbandingan;
    let data = json.data;
    let chart = new Morris.Line({
        element: 'bar-chart',
        resize: true,
        data: data,
        lineColors: [json.color],
        xkey: 'y',
        ykeys: ['a'],
        labels: [""],
        hideHover: false,
        parseTime: false,
        hoverCallback: function (index, options, content, row, x , y) {
default_html = "<div class='morris-hover morris-default-style'>
</div>";

        default_html = $(default_html).append(content);
        moveTo($(default_html), $('#bar-chart'), x, y);
        return "";
    }

    });

    chart.options.labels.forEach(function(label, i){

```

```

        var legendItem =
$(<span></span>').text(json.algoritma).css('color', chart.options.lineColors[i])
    $('#legend').append(legendItem)
    });
}

function chart_bar(perbandingan, data)
{
    $('#bar-chart').html("");
    $('#legend').html("");
    let label;
    if(perbandingan == 'hasil_kecepatan')
        label = ['Hasil Akhir Kecepatan'];
    else if(perbandingan == 'hasil_memory')
        label = ['Hasil Akhir Memory yang digunakan'];

    let chart = new Morris.Bar({
    element: 'bar-chart',
    resize: true,
    data: data,
    // barColors: ['green', 'red', 'gray'],
    barColors: function (row, series, type) {
        if(row.label == "Sequential Search") return "green";
        else if(row.label == "Binary Search") return "red";
        else if(row.label == "SQL Search") return "gray";
    },
    xkey: 'y',
    ykeys: ['a'],
    labels: label,
    hideHover: 'auto',
    });

    // let barColors = ['red', 'green', 'gray'];
    // chart.options.labels.forEach(function(label, i){
    //     var legendItem =
$(<span></span>').text(label).css('color', barColors[i])
    //     $('#legend').append(legendItem)
    // });
}

function getAnalisis(pengujian){
    $.ajax({
        url: 'get_analisis.php',
        data: {pengujian: pengujian},
        type: 'post',
        success: function(response){
            $('#table_analisis').html(response);
        }
    })
}

```

```

    });
}

$(document).ready(function(){

    getAnalisis();
    let pengujian;
    $('#pilih_pengujian').change(function(){
        $('#bar-chart').html("");
        $('#pilih-perbandingan').val("");
        $('#pilih-algoritma').val("");
        $('#pilih-perbandingan').show();
        pengujian = $(this).val();
        if(pengujian == 'semua'){
            $('#select[name*="perbandingan"]
option[value="kecepatan"]').hide();
            $('#select[name*="perbandingan"]
option[value="memory"]').hide();
        }else{
            $('#select[name*="perbandingan"]
option[value="kecepatan"]').show();
            $('#select[name*="perbandingan"]
option[value="memory"]').show();
        }
        getAnalisis(pengujian);
    });

    $(document).on('click', '.btn-detail', function(){
        let id = $(this).data('detail');
        $.ajax({
            url: 'get_detail.php',
            type: 'post',
            data: {id: id},
            cache: false,
            success:function(response){
                $('#detail').show();
                $('#detail .row').html(response);
            }
        });
    });

    $('#clear').click(function(){
        $('#detail .row').html("");
        $('#detail').hide();
    });

    $('#pilih-perbandingan').change(function(){
        $('#bar-chart').html("");

```



```

        $('#pilih-algoritma').val("");
        pengujian = $('#pilih_pengujian').val();
        let perbandingan = $(this).val();
        if(perbandingan == 'kecepatan' || perbandingan == 'memory'){
            $('#pilih-algoritma').show();
            $('#pilih-algoritma').change(function(){
                let algoritma = $(this).val();
                $.ajax({
                    url:"proses_analisis.php",
                    method:"POST",
                    data:{pengujian:                pengujian,
perbandingan:perbandingan, algoritma:algoritma},
                    dataType:"JSON",
                    cache: false,
                    success:function(response)
                    {
                        if(response.metode == 'algoritma')
                            chart_line(response);
                        else

chart_line_semua(response.data);
                    }
                });
            });
        }else{
            $('#pilih-algoritma').val("");
            $('#pilih-algoritma').hide();
            $.ajax({
                url:"proses_analisis.php",
                method:"POST",
                data:{pengujian:                pengujian,
perbandingan:perbandingan},
                dataType:"JSON",
                cache: false,
                success:function(response)
                {
                    chart_bar(perbandingan, response.data);
                }
            });
        }
    });

});

});

</script>
</body>
</html>

```

