ISSN 2301-8984 (Print), 2809-1884 (Online)
Published by Institute of Computer Science

Implementation of binary search on website-based traffic data

Novi Muhimmah Lailatun Najah[1], Walidini Syaihul Huda[2]
[1,2]Department of Informatics Engineering, Faculty of Engineering, Universitas Yudharta Pasuruan, Pasuruan, Indonesia

ARTICLE INFO

Article history:

Received Aug 3, 2025
Revised Aug 7, 2025
Accepted Aug 24, 2025

Keywords:

Binary Search;
Data Search;
Efficiency;
Website.

ABSTRACT

Fast and efficient data retrieval is a fundamental requirement in the development of web-based systems, especially in environments with large volumes of data. The goal of this study is to implement the Binary Search algorithm on the PT Jasamarga Pandaan Tol website to enhance the efficiency of the data retrieval process. System development was carried out using the waterfall method. The Binary Search algorithm was applied to a systematically sorted dataset. Then, the average time required for the search process was measured through testing. After conducting 100 searches on a dataset of 1,000 entries, we found that the average search time was 24.135 milliseconds. This finding indicates that the algorithm operates optimally and efficiently, making it suitable for implementation in large-scale information systems. The efficiency of the search algorithm is crucial to enhancing the system's overall performance.

Corresponding Author:

Novi Muhimmah Lailatun Najah,
Department of Informatics Engineering,
Faculty of Engineering,
Universitas Yudharta Pasuruan,
Jl. Yudharta No. 07 (Pesantren Ngalah) Sengonagung Purwosari Pasuruan, Jawa Timur, 67162, Indonesia
Email: noviml n99@gmail.com

1. INTRODUCTION

A website is a digital medium that serves to convey information and promote services or companies to the general public (Arafat et al., 2022). In today's digital age, websites play an increasingly important role in supporting the rapid dissemination of information. However, not all websites are able to optimize their search features, especially those that handle large amounts of data (Tambunan & Siagian, 2022). Some platforms still use linear search methods, which are inefficient because they cause search times to increase significantly as the amount of data grows (Elizabeth, 2024).

Data search is a fundamental process in data processing, aimed at finding specific values from a homogeneous data set (Sitompul et al., 2022). Binary Search emerges as one of the more efficient search solutions compared to linear methods. The Binary Search algorithm works by recursively dividing the sorted data into two parts until the sought value is found (Toyib et al., 2021) In previous studies, the Binary Search algorithm has demonstrated superior performance in terms of speed and efficiency. For example, (Nasution & Siddik, n.d.) tested the Binary Search algorithm on an Android-based Indonesian-English dictionary application, achieving an average search time of only 0.23 seconds. Research by (Imamah & Bahari, n.d.) also showed that the Binary Search algorithm is faster (0.0406 s) and more memory-efficient compared to SQL and Sequential Search.

Furthermore, research by (Khamdani et al., 2021) examined the effectiveness of the Brute force algorithm in searching for land ownership data. In this study, it was found that the Brute Force algorithm had an average search time of 0.06335 seconds. Although relatively fast for searches with short keywords, the search time increases significantly as the character length increases, resulting in the algorithm having a complexity of O(n), which is inefficient for large datasets. Binary Search is more efficient for large and sorted datasets due to its complexity of O(log n), making it superior for structured data searches compared to Brute Force, which is more flexible for free-text searches.

Several recent studies also reinforce the relevance of Binary Search in modern data processing. A study by (Avilla et al., 2024) on an e-tracking website demonstrated an empirical comparison between the Sequential Search algorithm and Binary Search in the context of small-scale web searches. (Nurrahmi, 2024) developed a web-based digital archive search system using the Binary Search algorithm, which was proven to improve search efficiency. (Istiono, 2023) analyzed the comparison between Binary Search and Interpolation Search in National ID Number (NIK) searches, where the Binary Search algorithm was proven to be 12.43% faster. (Lin, 2024) introduced hybrid-based Interpolation Once Binary Search variants and binary optimization, respectively for code search and computational time efficiency.

Complementing these findings, (Henneberg & Schuhknecht, 2025) demonstrated that the application of the Binary Search algorithm on GPU systems outperforms B-Tree indexes in database indexing. (Rorong et al., 2025) developed a more efficient hybrid Jump-Binary Search approach for non-uniform data distributions. Additionally, (Amato et al., 2023) demonstrated that variations of the Binary Search algorithm in modern indexing can significantly accelerate the search process. (Al-Hashimi & Aljabri, 2022) also emphasized the efficiency of power consumption and search time for large datasets through this approach.

The O (log n) complexity of the Binary Search algorithm is directly related to the performance requirements of web-based information systems, especially in environments that handle large volumes of data. With logarithmic time complexity, the search process remains efficient even as the dataset grows significantly. This characteristic makes Binary Search highly relevant for web-based platforms, where rapid response times are crucial to ensure user satisfaction and system scalability.

The limitations of linear search methods in previous web-based systems significantly affect the efficiency of large-scale data searches. Because linear search operates with O(n) time complexity, the search duration increases proportionally with the size of the dataset. As a result, web applications that relied on this method often experienced longer response times, reduced scalability, and decreased overall system performance when processing large volumes of data.

Previous studies have focused on the application of the Binary Search algorithm in mobile applications and text editors. However, there are still limitations in studies that implement this algorithm in web-based information systems, particularly those related to traffic data management or public services. Therefore, this study was conducted to address this gap by implementing the Binary Search algorithm on the PT Jasamarga Pandaan Toll Road website. The primary objective of this study is to enhance the efficiency and speed of data search processes through the application of the Binary Search algorithm.

2. RESEARCH METHOD

This research is an experimental study that aims to implement the Binary Search algorithm into a website-based information system to improve data search efficiency. This approach was chosen because it allows direct measurement of algorithm performance through testing the search process time on a dataset.

Data Collection

Data collection was conducted through interviews and observations of the traffic data management system. The data used in this study is historical traffic data from 2021 to 2023. Each attribute indicates the type of information contained in each data entry, as well as the data type used in the database and the range or number of values.

Data Preprocessing

The data preprocessing stage is carried out to ensure that the data meets the main requirements for implementing the Binary Search algorithm, namely that it is sorted. The data is sorted by date and type using the ORDER BY function in an SQL query, then stored in a two-dimensional array structure on the server side using the PHP programming language.

System Development Methodology

System development is carried out using the Waterfall model, which is a step-by-step approach to software development, with each stage being completed sequentially from planning to design, resembling a waterfall flowing from top to bottom (Mone & Pekuwali, 2023).

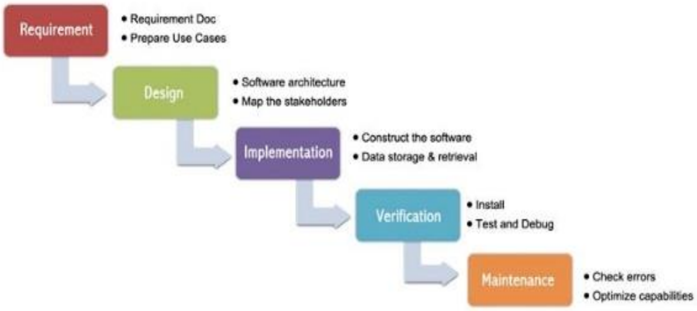

Figure 1. Waterfall methodology

Binary Search Algorithm Flowchart

The Binary Search Algorithm is used to speed up data searches in sorted datasets. The Binary Search Algorithm is an algorithm that uses a different approach to solve data search problems in programming (Markuci & Prianto, 2022). The Binary Search Algorithm can only be applied if the data is already sorted (Wijaya et al., 2021). The search process begins by dividing the array into two parts and comparing the middle element with the value being searched for. If the value is smaller than the middle element, the search continues on the left part of the array, which is then repeatedly divided into two parts (Aviantika et al., 2021).

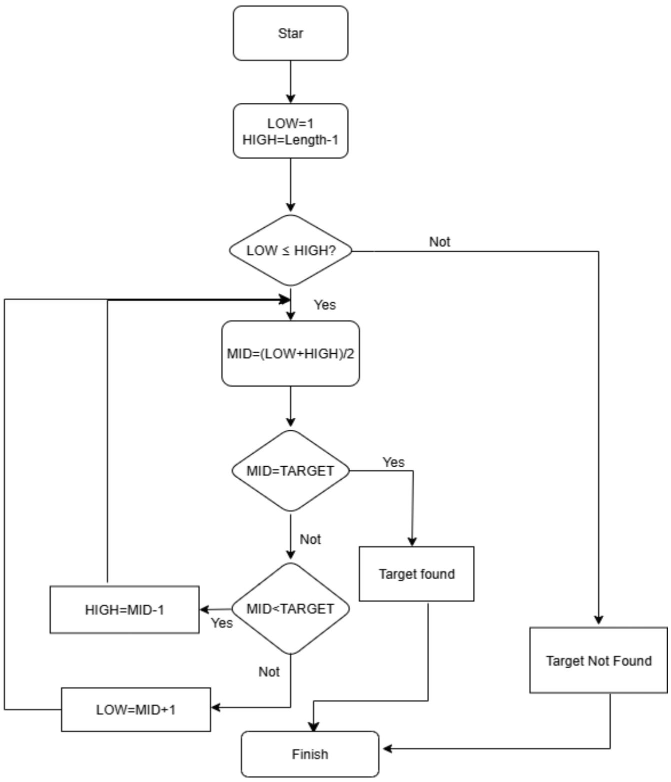

Figure 2. Binary search algorithm flowchart

Figure 2 shows the Binary Search algorithm, which illustrates the process of searching for data in a sorted array by repeatedly dividing the search space into two parts. The process begins with the initialization of the lower bound index (LOW) and upper bound index (HIGH). While LOW ≤

HIGH, the algorithm calculates the middle index (MID) and compares the value at that position with the search target. If the values match, the target is found; if not, the search continues on the left or right side of the array depending on the comparison result.

System Design

System design is carried out to describe the functional flow and interaction between users and the website-based search system that implements the Binary Search algorithm. The purpose of this stage is to provide a conceptual framework that explains how the system is built.

a. Use Case Diagram, are used to explain how people (actors) interact with system capabilities, as well as to show system limitations based on functional requirements (Avilla et al., 2024).

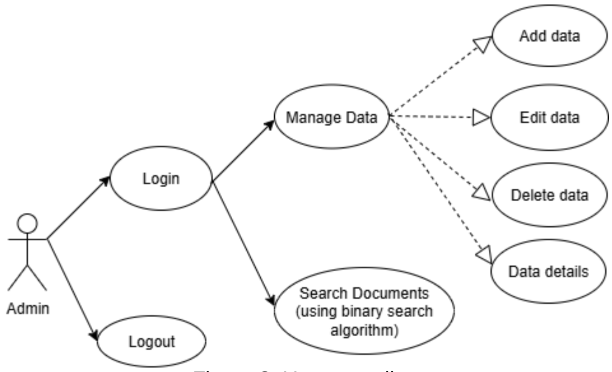

Figure 3. Use case diagram

Figure 3 shows a Use Case Diagram that illustrates the interactions of the main actor, namely the administrator. The administrator has access to log in, manage data (including adding, editing, deleting, and viewing data), and search for documents using the Binary Search algorithm.

b. Activity Diagram, an activity diagram is a process that explains the running of an activity in a system that is being designed (Khamdani et al., 2021).

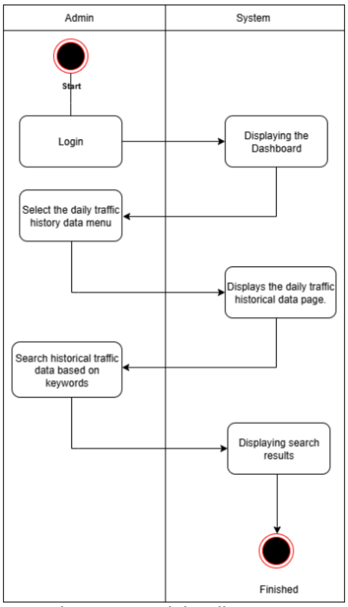

Figure 4. Activity diagram

Figure 4 illustrates the flow of activities performed by administrators when using the system. The process begins with logging in, followed by accessing the traffic data menu. Once the data page is displayed, administrators can perform searches based on specific keywords. The system then processes the search and displays the search results.

Binary Search Algorithm Logic

The Binary Search Algorithm is used to search for data in a sorted data set. The search process is carried out by comparing the target value with the middle element of the data set, then recursively narrowing the search space until the data is found or all elements have been checked.

```
// Binary Search
$startBinary = microtime(true);

$low = 0;
$high = count($data) - 1;
$firstIndex = -1;
$lastIndex = -1;

// Cari index pertama
while ($low <= $high) {
    $mid = floor(($low + $high) / 2);
    if ($data[$mid]['tanggal'] < $tanggalDicari) {
        $low = $mid + 1;
    } elseif ($data[$mid]['tanggal'] > $tanggalDicari) {
        $high = $mid - 1;
    } else {
        $firstIndex = $mid;
        $high = $mid - 1;
    }
}
```

Figure 5. Binary search algorithm logic

Figure 5 illustrates a basic logic code snippet for determining the initial position of data that matches the input parameters. This strategy utilizes the principle of repeatedly dividing the search space, where the middle index is compared with the value being searched for, and the search limits are adjusted until the correct index is found.

3. RESULTS AND DISCUSSIONS

The results and discussion section presents the results of the implementation and testing of the Binary Search algorithm applied to the traffic data search system on the PT Jasamarga Pandaan Tol website. The implementation stage is a process of incorporating the design results into the program code and carrying out the implementation (Alvianda & Sumaryana, 2023).

The workflow of the Binary Search algorithm in the system

For example, the workflow of the Binary Search algorithm is known to have 1000 data points, searching for data based on the target date of 2022-10-09, which is located at index 647:

Iteration 1
Low = 0 (first index)
High = 999 (last index)
Calculate MID:
MID = (LOW+HIGH)/2
MID = (0+999)/2= 499
Fill in the data at index 499 = 2022-05-14

Iteration 2:
LOW = 500
HIGH = 999
Calculate MID:
MID = (LOW+HIGH)/2
MID = (500+999)/ 2 = 749
Fill data at index 749 = 2023-01-09

Iteration 3:
LOW = 500
HIGH = 748
Calculate MID:
MID = (LOW+HIGH)/2
MID = (500 + 748)/2 = 624
Fill in the data at index 624 = 2022-09-16

Iteration 4:

LOW = 625
HIGH = 748
Calculate MID:
MID = (LOW + HIGH)/2
MID = (625 + 748)/2 = 686
Fill in the data at index 686 = 2022-11-17

Iteration 5:
LOW = 625
HIGH = 685
Calculate MID:
MID = (LOW+HIGH)/2
MID = (625+685)/2 = 655
Fill in the data at index 655 = 2022-10-27

Iteration 6:
LOW = 625
HIGH = 654
Calculate MID:
MID = (LOW+HIGH)/2
MID = (625 + 654)/2 = 639
Enter data at index 639 = 2022-10-01

Iteration 7:
LOW = 640
HIGH = 654
Calculate MID:
MID = (LOW+HIGH)/2
MID = (640+654)/2 = 647
Fill in the data at index 647 = 2022-10-09

From the search results on 2022-10-09, it can be concluded that the Binary Search algorithm only needs 7 iterations to find data from 1000 data points. The midpoint (MID) value is calculated by adding the lower limit (LOW) and upper limit (HIGH) values and then dividing them by two.

Binary Search Algorithm Testing

System testing was conducted to evaluate the performance of the Binary Search algorithm implemented on the website. At this stage, 100 tests were conducted using 1,000 pieces of traffic data.

Table 1. Microtime testing

ID	Date	Type_Class	Binary Search
1	2021-01-01	Class 1	0.02408 ms
2	2023-02-02	Empty	0.02193 ms
3	2022-12-03	Empty	0.02885 ms
4	2022-05-20	Empty	0.02694 ms
5	2021-02-02	Empty	0.0391 ms
-------	--------------------------	------------------------	--------------------
96	2022-11-29	Empty	0.01216 ms
97	2021-09-25	Empty	0.00906 ms
98	2021-07-07	Empty	0.00917 ms
99	2023-08-18	Invalid	0.01001 ms
100	2022-03-27	Invalid	0.01296 ms
		Total Search Time	2.413,48 ms
		Average Search Time	24,135 ms

Based on the test results in Table 1, it can be seen that the Binary Search algorithm can perform searches in real time. After conducting 100 tests with varying data, it was found that the Binary Search algorithm has efficient performance. The total time required by the Binary Search algorithm was 2,413.48 ms, with an average search time of 24.135 ms.

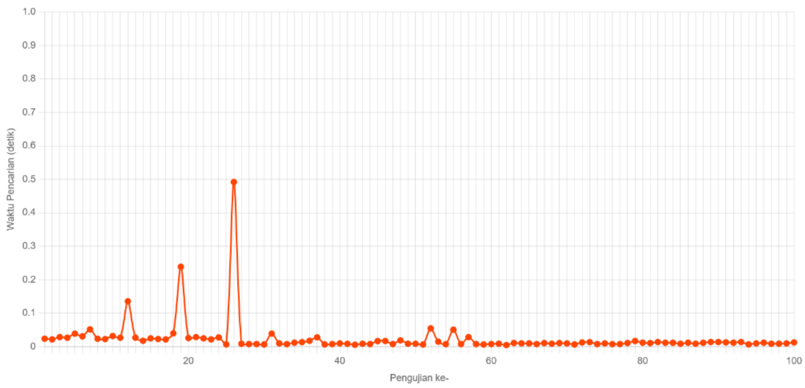

Figure 6. Microtime testing graph

Figure 6 shows the results of 100 tests, indicating that the Binary Search algorithm has relatively stable and efficient search times, with the majority below 0.1 seconds (100 ms). Although there are some spikes at certain points, the search times remain within acceptable limits. These spikes may be influenced by factors such as temporary server load, memory allocation, or background processes during execution. In practical web-based implementations, such issues can be minimized through optimization strategies such as database indexing, caching, and efficient server resource management. This finding confirms that the Binary Search algorithm is highly effective when applied to large, sorted datasets, as it has a time complexity of O(log n), which supports fast and consistent search operations in web-based information systems.

Black Box Testing

Table 2. Black box testing

No	Feature	Description	Result
1	Login	The system displays the dashboard	Successful
2	Dashboard	The system displays the main page after login	Successful
3	Traffic data	Access the traffic data page through the navigation menu	Successful
4	Add data	The form is filled out and saved, and the data is displayed in the list	Successful
5	Delete data	Data is deleted via the icon, data is removed from the list	Successful
6	Edit data	Data is edited via the form and updated according to input	Successful
7	View data	Data is displayed when the row is clicked	Successful
8	Empty search	Only the date is entered, results matching the date appear	Successful
9	Valid search	Valid date and type input, matching results appear	Successful
10	No search	Invalid input, system displays notification	Successful
11	Logout	User logs out of the system, returns to the index page	Successful

Table 2 explains the tests conducted to evaluate the main functions of the website-based data search system using the Black Box Testing method. This method focuses on testing the system's functionality from the user's perspective without knowing the internal structure of the program code. The system is able to respond correctly to both valid and invalid inputs, and it can also handle invalid inputs by providing informative notifications to the user. This indicates that the system has good input validity. The logout feature was also tested and successfully redirected the user back to the index page, indicating that the session had been closed properly.

Overall, the Black Box Testing demonstrated that the system meets the previously designed functional specifications. The success of all tests indicates that the system can be reliably used to perform data search and management processes efficiently.

4. CONCLUSION

The results of this study indicate that the implementation of the Binary Search algorithm in a traffic data search information system can significantly improve the efficiency of the search process. With a time complexity of O(log n), this algorithm enables data searches to be performed more quickly and stably, even with large amounts of data.

Thus, it fulfills the primary objective of the research to optimize data search in a web-based system. The application of the Waterfall system development model supports the systematic and structured implementation of the algorithm.

For further development, effective strategies to expand system functionality include combining Binary Search with indexing techniques for keyword-based searches, and adopting range query optimizations (such as segment trees or B+ trees) for handling interval data efficiently. These approaches enable the system to support more complex search types while maintaining efficiency and scalability. In addition, future work can explore the integration of adaptive or hybrid search methods to further enhance flexibility and robustness in large-scale environments.

Integrating Binary Search with adaptive search techniques or dynamic data management preserves logarithmic lookup costs while reducing tail latency as data and workloads evolve. By leveraging workload-aware indexing, adaptive caching, dynamic partitioning/sharding, and incremental index maintenance, the system can scale horizontally, sustain high concurrency, and keep response times consistent under large-scale, real-world conditions.

ACKNOWLEDGEMENTS

The author would like to thank all parties who contributed to the completion of this research. Special appreciation is extended to the supervising lecturer, colleagues, and related institutions for their support, input, and facilities provided. These contributions were very meaningful in supporting the smooth running of the research process and the writing of this article.

REFERENCES

Al-Hashimi, M., & Aljabri, N. (2022). Exploring Power Advantage of Binary Search: An Experimental Study. *International Journal of Advanced Computer Science and Applications*, *13*(10). https://doi.org/10.14569/IJACSA.2022.0131094

Alvianda, F. F., & Sumaryana, Y. (2023). *PerbandinganAlgoritma Brute Force Dengan Boyer- Moore Pada Aplikasi Pencarian Kerja Berbasis Web*. *12*.

Amato, D., Giancarlo, R., & Lo Bosco, G. (2023). Learned Sorted Table Search and Static Indexes in Small-Space Data Models. *Data*, *8*(3), 56. https://doi.org/10.3390/data8030056

Arafat, M., Trimarsiah, Y., & Susantho, H. (2022). Rancang Bangun Sistem Informasi Pemesanan Online Percetakan Sriwijaya Multi Grafika Berbasis Website. *INTECH*, *3*(2), 58–63. https://doi.org/10.54895/intech.v3i2.1691

Abidah, D. Y., Wijoyo, S. H., & Rahman, K. (n.d.). *Pengaruh Platform Visual Studio Code Terhadap Hasil Belajar Siswa pada Mata pelajaran Pemrograman Dasar Kelas X Jurusan Teknik Komputer dan Jaringan SMKN 3 Malang*.

Aviantika, R. D. A., Kustanto, K., & Hasbi, M. (2021). Pencarian Data Barang Produk Atribut Sekolah Menggunakan Algoritma Binary Search. *Jurnal Teknologi Informasi dan Komunikasi (TIKomSiN)*, *9*(1), 75. https://doi.org/10.30646/tikomsin.v9i1.546

Avilla, B., Fauziah, F., & Winarsih, W. (2024a). Perbandingan Algoritma Sequential Search dan Binary Search Pada Website E-Tracking Pengajuan Surat. *Jurnal Informatika: Jurnal Pengembangan IT*, *9*(1), 18–24. https://doi.org/10.30591/jpit.v9i1.5383

Avilla, B., Fauziah, F., & Winarsih, W. (2024b). Perbandingan Algoritma Sequential Search dan Binary Search Pada Website E-Tracking Pengajuan Surat. *Jurnal Informatika: Jurnal Pengembangan IT*, *9*(1), 18–24. https://doi.org/10.30591/jpit.v9i1.5383

Damayanti, S. E., & Permana, F. (2024). SISTEM INFORMASI PADA USAHA WANGI PROJECT DI LIMBANGAN TENGAH DENGAN MENGGUNAKAN ALGORITMA LINEAR SEARCH DAN AES 256. *Naratif : Jurnal Nasional Riset, Aplikasi dan Teknik Informatika*, *6*(1), 79–88. https://doi.org/10.53580/naratif.v6i1.288

Dinitz, M., Im, S., & Lavastida, T. (n.d.). *Binary Search with Distributional Predictions*.

Elizabeth, T. (2024). *Implementasi Algoritma Sequential Search Dan Binary Search Dalam Pencarian Data Faktur*. *11*(2).

Henneberg, J., & Schuhknecht, F. (2025). *All You Need Is Binary Search! A Practical View on Lightweight Database Indexing on GPUs* (No. arXiv:2506.01576). arXiv. https://doi.org/10.48550/arXiv.2506.01576

Imamah, N., & Bahari, M. I. (n.d.). PERBANDINGAN ALGORITMA SEQUENTIAL SEARCH DAN ALGORITMA BINARY SEARCH PADA APLIKASI KAMUS BAHASA INDONESIA MENGGUNAKAN PHP DAN JQUERY. *Jurnal Informatika*, *08*.

Istiono, W. (2023). Speed Analysis of Binary Search and Interpolation Search for Searching Identification Numbers on National Identity Cards. *Asian Journal of Research in Computer Science*, *15*(4), 34–41. https://doi.org/10.9734/ajrcos/2023/v15i4328

Khamdani, R. R., Astuti, I. P., & Masykur, F. (2021). IMPLEMENTASI ALGORITMA BRUTE FORCE PADA PENCARIAN DATA KEPEMILIKAN TANAH. *KOMPUTEK*, *5*(1), 101. https://doi.org/10.24269/jkt.v5i1.687

Lin, J.-L. (2024). Interpolation Once Binary Search over a Sorted List. *Mathematics*, *12*(9), 1394. https://doi.org/10.3390/math12091394

Markuci, D., & Prianto, C. (2022). ANALISIS PERBANDINGAN PENGGUNAAN ALGORITMA SEQUENTIAL SEARCH DAN BINARY SEARCH PADA APLIKASI SURAT PERJALANAN DINAS. *JATI (Jurnal Mahasiswa Teknik Informatika)*, *6*(1), 110–119. https://doi.org/10.36040/jati.v6i1.4569

Mone, L. U. B., & Pekuwali, A. A. (2023). *Application Of The Binary Search Algorithm In The Library Information System At SMP Negeri 4 Mauliru*.

Nasution, A., & Siddik, M. (n.d.). *IMPLEMENTASI ALGORITMA BINARY SEARCH PADA APLIKASI KAMUS INDONESIA-INGGRIS BERBASIS ANDROID*.

Nurrahmi, H. (2024). Development of a Web-Based Digital Archive Search System Using The Binary Search Algorithm. *International Journal of Engineering Business and Social Science*, *3*(1), 14–20. https://doi.org/10.58451/ijebss.v3i1.188

Rorong, G. Y., Pane, S. F., & Siregar, M. A. H. (2025). Optimizing Search Efficiency in Ordered Data: A Hybrid Approach Using Jump Binary Search. *Journal of Dinda : Data Science, Information Technology, and Data Analytics*, *5*(1), 36–44. https://doi.org/10.20895/dinda.v5i1.1764

Sitompul, B. J. D., Yusupa, A., & Tuturoong, N. J. (2022). IMPLEMENTASI ALGORITMA BINARY SEARCH PADA PENCARIAN DATA JEMAAT GEREJA HKBP MANADO. *Jurnal Informatika Polinema*, *9*(1), 17–24. https://doi.org/10.33795/jip.v9i1.1123

Tambunan, M. A., & Siagian, P. (2022). *PENGEMBANGAN MEDIA PEMBELAJARAN INTERAKTIF BERBASIS WEBSITE (GOOGLE SITES) PADA MATERI FUNGSI DI SMA NEGERI 15 MEDAN*. *2*(10).

Toyib, R., Darnita, Y., & Deva, A. R. S. (2021). PENERAPAN ALGORITMA BINARY SEARCH PADA APLIKASI E-ORDER. *JURNAL MEDIA INFOTAMA*, *17*(1). https://doi.org/10.37676/jmi.v17i1.1314

Wijaya, A., Maulana, M. F., & Danil, M. (2021). APLIKASI PENCARIAN RESEP MASAKAN KHAS BENGKULU MENGGUNAKAN ALGORITMA BINARY SEARCH BERBASIS ANDROID. *JURNAL MEDIA INFOTAMA*, *17*(1). https://doi.org/10.37676/jmi.v17i1.1310