Teknika 16 (01): 149 – 159

JURNAL TEKNIKA
ISSN: 0854-3143
e-ISSN: 2622-3481
Journal homepage: http://jurnal.polsri.ac.id/index.php/teknika
Journal Email: teknika@polsri.ac.id

Perbandingan Efisiensi dengan Algoritma *Sorting* dalam Penentuan Jarak (Studi Kasus: *Pet Shop* di Bandar Lampung)

Yuni Puspita Sari[1]. Rionaldi Ali[2]. Arya Rajasa[3]
[1]Fakultas Ilmu Komputer, Informatics & Business Institute Darmajaya
Jl. Z.A. Pagar Alam No. 93, Bandar Lampung - Indonesia 35142
Telp. (0721) 787214 Fax. (0721) 700261
e-mail : yunipuspita@darmajaya.ac.id, rionaldi@darmajaya.ac.id, aryaa.rajasa@gmail.com

Abstract

Pet Shop is a place to sell pets along with equipment and supplies for animal care. In Bandar Lampung *there are approximately* 100 *Pet Shops which are widely spread, both small and large Pet Shops. However, not all people know the Pet Shop locations, starting from the closest one or having complete facilities and services from the existing Pet Shop. They tend to have difficulty when it comes to grooming their pets or finding their pet's food needs. This application was designed using the prototype development method and in sorting the closest distance at the Pet Shop using the Selection Sort and Insertion Sort algorithms. This application is dynamic because the data can change according to the increase or decrease in the Pet Shop registered in the application. The result of this study was an application that displayed an information medium to determine the distance of the closest and furthest Pet Shop, tables and memory graphs that function to determine the memory used and display a time graph. From the results of the comparative analysis of the efficiency of the Selection Sort and Insertion Sort algorithms, it was found that the Selection Sort algorithm was the best algorithm that was effective and efficient in dealing with the data sorting problems. This application was tested using the black box testing with the test components, namely, loading performance functions, and* 3 *device interfaces.*

Keywords*Pet Shop, Prototype, Selection Sort, Insertion Sort.*

Abstrak

Pet Shop merupakan salah satu tempat untuk menjual hewan peliharaan beserta peralatan dan perlengkapan untuk pemeliharaan hewan. Di Bandar Lampung terdapat kurang lebih 100 *Pet Shop* yang tersebar luas, baik *Pet Shop* kecil maupun besar. Namun tidak semua masyarakat mengetahui lokasi-lokasi *Pet Shop* mulai dari yang terdekat atau memiliki fasilitas yang lengkap dan layanan dari *Pet Shop* yang ada. Mereka cenderung mengalami kesulitan ketika akan melakukan perawatan hewan peliharaannya atau mencari kebutuhan makanan hewan peliharaan

mereka. Aplikasi ini dirancang menggunakan metode pengembangan *prototype* serta dalam pengurutan jarak terdekat pada *Pet Shop* menggunakan algortima *Selection Sort* dan *Insertion Sort*. Aplikasi ini bersifat dinamis dikarenakan data dapat berubah sesuai dengan pertambahan atau pengurangan *Pet Shop* yang terdaftar pada aplikasi. Hasil dari penelitian ini berupa aplikasi yang menampilkan media informasi untuk menentukan jarak *Pet Shop* terdekat dan terjauh, tabel dan grafik memori yang berfungsi untuk menentukan memori yang terpakai serta menampilkan grafik waktu. Dari hasil analisa perbandingan efisiensi algoritma *Selection Sort* dan *Insertion Sort* didapatkan bahwa algoritma *Selection Sort* adalah algortima terbaik yang efektif dan efisien dalam menangani masalah pengurutan data. Aplikasi ini telah di uji dengan menggunakan *black box testing* dengan komponen uji yaitu, fungsi kinerja *loading*, dan *interface* sebanyak 3 *device.*

Kata kunci : *Pet Shop, Prototype, Selection Sort, Insertion Sort.*

1. PENDAHULUAN

Dewasa ini aktivitas memelihara hewan yang kian digemari juga dapat dilihat dari banyaknya jumlah pemilik hewan peliharaan di Indonesia terbukti dari banayaknya juga *petshop* yang tersebar diseluruh penjuru daerah. *Pet Shop* merupakan tempat untuk menjual hewan peliharaan beserta peralatan dan perlengkapan untuk pemeliharaan hewan., namun saat ini banyak pemilik hewan peliharaan yang belum banyak mengetahui lokasi-lokasi *petshop* yang tersebar di Bandar Lampung, dari mulai yang terdekat atau yang lengkap fasilitas yang dimiliki oleh *petshop,* mereka cenderung kesulitan ketika akan melakukan perawatan hewan peliharaannya atau mencari kebutuhan makanan hewan peliharaan mereka.

Berdasarkan hasil observasi Bandar Lampung memiliki kurang lebih 100 *Pet Shop* baik kecil maupun besar. Namun tidak semua masyarakat mengetahui lokasi dan layanan dari *Pet Shop* yang ada. Pada penelitian sebelumnya dilakukan tentang pencarian suatu tempat, namun belum membahas perbandinagn efisiensi waktu dan memori saat pencarian dan pengurutan jarak yang dilakukan dalam aplikasi. Dalam penelitian ini maka dilakukan penerapan efisiensi pengurutan dalam data yang banyak diperlukan untuk mengoptimalkan kecepatan saat pemrosesan. Jika algoritma memiliki efisiensi yang tinggi maka proses eksekusi akan menggunakan lebih sedikit memori dengan waktu yang lebih cepat. Metode yang digunakan dalam pengurutan jarak menggunakan metode *Selection Sort* dan *Insertion Sort* [1]. Aplikasi ini bersifat dinamis dikarenakan data dapat berubah sesuai dengan pertambahan atau pengurangan *Pet Shop* yang terdaftar pada aplikasi[2].

2. METODE PENELITIAN

2.1 Metode Pengumpulan Data

Dalam penelitian ini tahap mengumpulkan data dan teknik yang digunakan antara lain sebagai berikut:

a. Wawancara

Dalam tahap ini peneliti melakukan wawancara kepada pihak yang berwenang di seluruh *petshop* yang tersebar di Bandar Lampung, proses interaksi yang dilakukan adalah dengan melakukan *interview* sebagai sumber informasi atau orang yang di wawancarai melalui komunikasi langsung..

b. Observasi

Observasi yang dilakukan dalam tahap peneltian ini dengan melihat langsung di lapangan dari aktifitas transaksi pada tempat penelitian untuk mendapatkan data sebagi acuan dalam menganalisa permasalahan yang ada dan untuk melanjuktkan ke tahap penelitian berikutnya.

c. Dokumentasi

Metode dokumentasi dalam penelitian merupakan pelengkap dari penggunaan metode observasi dan wawancara. Studi dokumentasi yaitu mengumpulkan dokumen dan data-data yang diperlukan di tempat penelitian sehingga

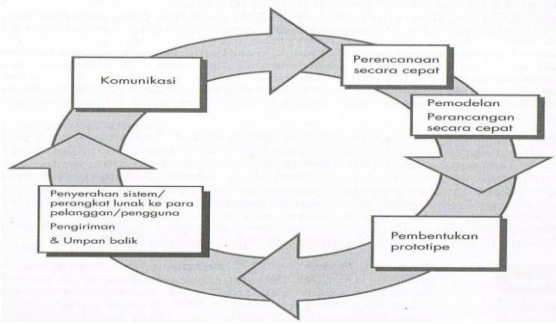
dapat mendukung dan menambah kepercayaan dan pembuktian suatu kejadian.

d. Perancangan Secara Cepat

Pada saat membangun aplikasi, Perancangan secara cepat merupakan tahapan dimana peneliti menetapkan bagaimana perangkat lunak tersebut dapat dioperasikan. Hal ini berkaitan dalam menentukan spesifikasi perangkat keras, spesifikasi perangkat lunak tampilan aplikasi dan *form-form* yang akan digunakan.

2.2 Metode Pengembangan Perangkat Lunak

Gambar 1 menjelaskan Metode pengembangan perangkat lunak yang diperlukan untuk memudahkan peneliti dalam merancang dan menerapkan metode *selection Sort dan Insertion Sort* [3] untuk Algoritma perbandingan [4] untuk penentuan jarak *Pet Shop* yang tersebar di Bandar Lampung. Metode yang digunakan adalah model *prototype* [5].



Gambar 1. Metode *Prototype*

2.3 Algoritma *Selection Sort*

Selection Sort adalah suatu algoritma pengurutan yang membandingkan elemen yang sekarang dengan elemen berikutnya sampai ke elemen yang terakhir. Jika ditemukan elemen lain yang lebih kecil dari elemen sekarang maka dicatat posisisnya dan langsung ditukar. Metode pengurutan data [6] ini adalah dengan cara memilih suatu data pada urutan tertentu, kemudian membandingkannya dengan data-data lainnya mulai dari posisi (posisi data+1) sampai dengan data pada posisi ke-n, untuk mencari data terkecil pada rentang posisi tersebut. Jika data terkecil ditemukan, maka pindahkan data terkecil tersebut ke posisi (posisi data), dan data yang semula berada di posisi [posisi data] dipindahkan ke posisi dimana data terkecil tadi ditemukan. Demikian seterusnya hingga data terakhir**.** Tabel 1 menjelaskan simulasi dari cara kerja metode selection sort [7]

Tabel 1. Simulasi Cara kerja Algoritma *Selection Sort*

22	**2**	**90**	**25**	**20**	**30**	**6**	**3**	**Data awal**
22	2	90	25	20	30	6	3	Data ke 8 di tukar dengan data ke 3 (Terbesar)
22	2	3	25	20	30	6	90	Data ke 7 ditukar dengan data ke 6
22	2	3	25	20	6	30	90	Data ke 6 di tukar dengan data ke 4
22	2	3	6	20	25	30	90	Data ke 5 ditukar dengan data ke 1
20	2	3	6	22	25	30	90	Data 4 ditukar dengan data ke 1
6	2	3	20	22	25	30	90	Data ke 3 ditukar dengan data ke 1
3	2	6	20	22	25	30	90	Data ke 2 ditukar dengan data ke 1
2	3	6	20	22	25	30	90	Maka didapatkan data sebagai berikut
2	**3**	**6**	**20**	**22**	**25**	**30**	**90**	**Data setelah terurut**

2.4 Algoritma *Insertion Sort*

Insertion Sort adalah adalah sebuah algoritma pengurutan yang membandingkan dua elemen data pertama, mengurutkannya, kemudian mengecek elemen data berikutnya satu persatu dan membandingkannya dengan elemen data elemen, menemukan yang terkecil atau yang terbesar sesuai yang telah diurutkan. Algoritma pengurutan data *insertion sort* akan memeriksa setiap kebutuhan jenis pengurutan, dan menyisipkan dalam tempat yang tepat. Cara kerja algoritma ini yaitu pengurutan dengan penyisipan bekerja dengan cara menyisipkan masing-masing nilai di tempat yang sesuai di antara elemen yang lebih kecil atau sama dengan nilai tersebut. Untuk menghemat memori, implementasinya menggunakan pengurutan di tempat yang membandingkan elemen saat itu dengan elemen sebelumnya yang sudah diurut, lalu menukarnya terus sampai posisinya tepat. Tabel 2 menjelaskan simulasi dari cara kerja metode *selection sort*

Tabel 2.Simulasi Cara kerja Algoritma *Insertion Sort*

72	**12**	**59**	**45**	**51**	**Data Awal**
72	12	59	45	51	Data ke 2 di tukar dengan data ke 1 (Terkecil)
12	72	59	45	51	Data ke 2 di tukar dengan data ke 3 (Terbesar)
12	59	72	45	51	Data ke 3 ditukar dengan data ke 4
12	59	45	72	51	Data ke 4 ditukar dengan data ke 5
12	59	45	51	72	Data masih belum urut , Data ke 2 di tukar dengan data ke 3
12	45	59	51	72	Data ke 3 ditukar dengan data ke 4
12	**45**	**51**	**59**	**72**	**Data setelah terurut**

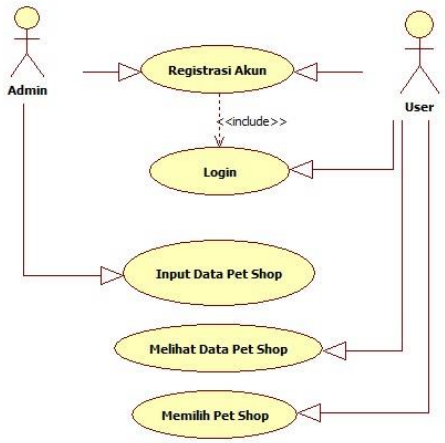
3. HASIL DAN PEMBAHASAN

3.1 Pemodelan Perancangan Secara Cepat

Pada tahap desain perancangan "Perbandingan Efisiensi Algoritma *Sorting* Dalam Penentuan Jarak Terdekat (Studi Kasus : *Pet Shop* Di Bandar Lampung)" ini dimulai dari desain perancangan UML (*Unified Modeling Language*) yaitu untuk menentukan desain *Use Case* Diagram, *Activity Diagram* dan *Class Diagram* [8] sebagai berikut :

1. Use Case Diagram

Gambar 2 berikut adalah *use case diagram* dari Perbandingan Efisiensi Algoritma *Sorting* Dalam Penentuan Jarak Terdekat (Studi Kasus : *Pet Shop* di Bandar Lampung).



Gambar 2. *Use Case* Diagram dari Perangkat Lunak yang Diajukan.

Gambar *use case diagram* di atas menjelaskan bahwa *admin* bertindak sebagai orang yang dapat mengelola semua data *Pet Shop* serta melakukan penginputan data *Pet Shop*.

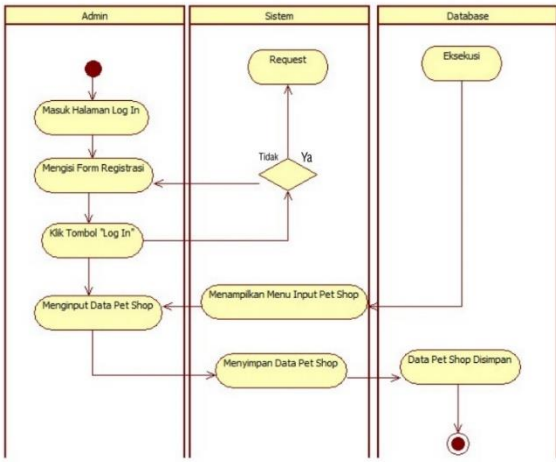
3.2. Activity Diagram

Activity diagram berfungsi untuk memberikan visualisasi alur tindakan dalam sistem, percabangan yang mungkin terjadi, dan alur sistem yang dimulai dari awal hingga akhir. Yang akan menampilkan beberapa menu pilihan dimana dalam pilihan menu terdapat penjelasan yang akan di bahas pada

masing – masing menu tersebut. Berikut adalah *activity diagram* dari Perbandingan Efisiensi Algoritma *Sorting* Dalam Penentuan Jarak Terdekat (Studi Kasus : *Pet Shop* di Bandar Lampung) :

a. Activity Diagram Admin

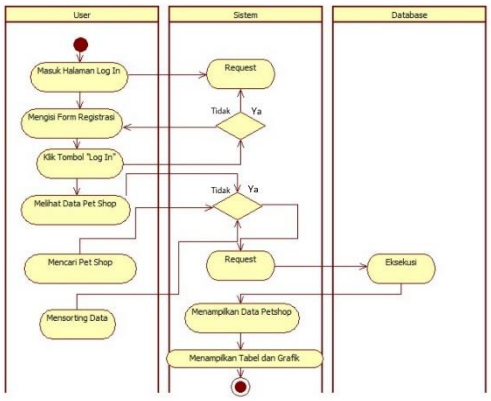
Gambar 3 berikut adalah *Activity diagram* penggambaran alur aktivitas *admin* yang memiliki akses penuh di dalam Aplikasi, dan Admin dapat menginput data *Pet Shop* serta mengubah atau menghapus data.



Gambar 3. *Activity Diagram* Admin dari Perangkat Lunak yang Diajukan.

b. Activity Diagram User

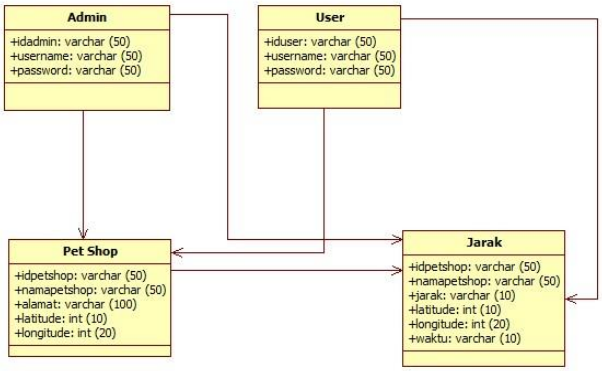
Gambar 4 berikut adalah *Activity diagram* penggambaran alur aktivitas yang dapat dilakukan oleh *user* pada aplikasi, seperti melakukan pencarian lokasi *Pet Shop* dan membandingkan jarak menggunakan algoritma *sorting.*



Gambar 4. *Activity* Diagram *User* dari Perangkat Lunak yang Diajukan.

3.3 Class Diagram

Gambar 5 berikut adalah *Class diagram* [9] struktur dan deskripsi *class*, *package* dan objek beserta hubungan antara satu sama lain dari Perbandingan Efisiensi Algoritma *Sorting* Dalam Penentuan Jarak Terdekat (Studi Kasus : *Pet Shop* di Bandar Lampung),



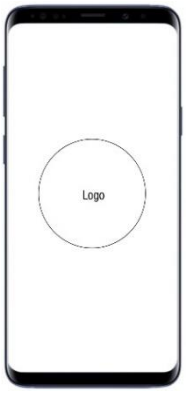
Gambar 5. Class Diagram dari Perangkat Lunak yang Diajukan.

3.4. Rancangan *Interface* Aplikasi

Rancangan *interface* adalah desain awal sebelum membangun suatu perangkat lunak, hasil dari perangkat lunak yang dibangun nantinya tidak akan jauh berbeda dengan perancangan *interface* yang dibuat.

a. Rancangan Interface Admin Halaman Splash Screen

Gambar 6 adalah Halaman *splash screen* akan muncul saat pertama kali Aplikasi dijalankan.



:

Gambar 6. Rancangan Tampilan Halaman *Splash Screen*

b. Rancangan *Interface Halaman Utama User*

Gambar 7 adalah Halaman Utama *User* muncul pada saat *user* pertama kali mengakses aplikasi.

Gambar 7. Rancangan Tampilan Halaman Utama *User*

c. Tampilan rancangan halaman untuk melihat Waktu dan Grafik Memori yang terpakai

Gambar 8 menampilkan hasil grafik *memory* yang terpakai pada saat proses pencarian.

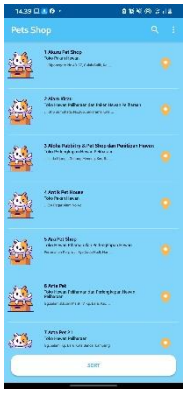
Gambar 8. Halaman Hasil Grafik Memory

3.5. Hasil *Interface Aplikasi*

Berikut tampilan *interface* dari "Perbandingan Efisiensi Algoritma *Sorting* Dalam Penentuan Jarak Terdekat (Studi Kasus : *Pet Shop* Di Bandar Lampung)" yang di bangun :

a. Tampilan *Interface* Halaman *Splash Screen*

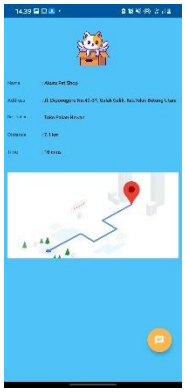
Gambar 9 adalah Halaman *splash screen* akan muncul saat pertama kali aplikasi dijalankan, yang menampilkan informasi seluruh petshop yang tersebar di Bandar Lampung dengan jarak terdekat.



Gambar 9. *Interface* Halaman *Splash Screen*

b. Tampilan Interface Deskripsi petshop

Gambar 10 adalah Halaman ini muncul pada saat user mengakses aplikasi dan memilih petshop dengan cara terdekat dan kemudian akan muncul deskripsi dari *petshop* yang dipilih serta map yang akan menentukan jarak tempuh.



Gambar 10. *Interface* Halaman Deskripsi *Pet Shop* dan *Map View.*

c. Tampilan *Interface* halaman *sorting* dengan menggunakan metode *Insertion Sort* secara *Ascending* dan *Descending*

Gambar 11 adalah Halaman yang menerapkan *sorting* menggunakan metode *Selection Sort*. Terdapat daftar nama-nama *Pet Shop* yang sudah terurut dari jarak terjauh hingga terdekat. Di halaman *sorting*, *user* dapat memilih fitur *Ascending* dan *Descending* [10] untuk menentukan dari jarak terdekat atau jarak terjauh terlebih dahulu,

Gambar 11. *Interface* Halaman *Sorting Selection Sort (Ascending dan Descending)*

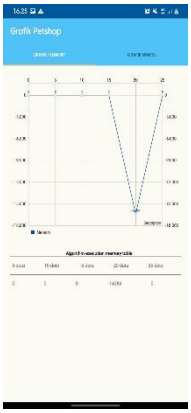
d. Tampilan *Interface* halaman *sorting* dengan menggunakan metode *Selection Sort* secara *Ascending* dan *Descending*

Gambar 12 adalah Halaman yang menerapkan *sorting* menggunakan metode *Insertion Sort*. Terdapat daftar nama-nama *Pet Shop* yang sudah terurut dari jarak terjauh hingga terdekat. Di halaman *sorting*, *user* dapat memilih fitur *Ascending* dan *Descending* untuk menentukan dari jarak terdekat atau jarak terjauh terlebih dahulu,

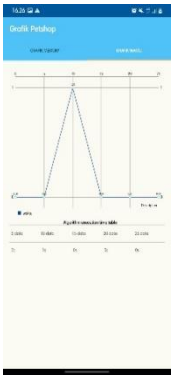
Gambar 12. *Interface* Halaman *Sorting Insertion Sort (Ascending dan Descending)*

e. *Interface* Tampilan *Interface* Halaman Grafik Memori dan Grafik waktu dengan *Selection Sort*

Gambar 13 adalah Halaman yang berguna untuk melihat grafik memori dan waktu yang terpakai pada saat melakukan *sorting* menggunakan *Selection Sort*.



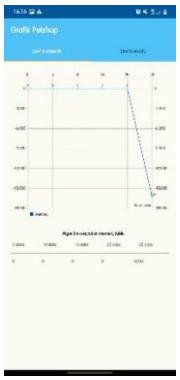
Gambar 13. *Interface* Halaman Grafik Memori *Selection Sort*



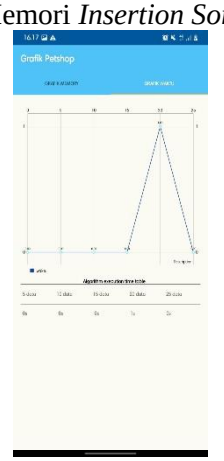
Gambar 13 *Interface* Halaman Grafik Waktu *Selection Sort.*

f. Interface Tampilan dan *Interface* Halaman Grafik Memori dan Grafik waktu dengan *Insertion Sort*

Gambar 14 dan 15 adalah Halaman yang berguna untuk melihat grafik memori dan waktu yang terpakai pada saat melakukan *sorting* menggunakan *Insertion Sort*.



Gambar 14 *Interface* Halaman Grafik Memori *Insertion Sort.*



Gambar 15 *Interface* Halaman Grafik Waktu *Selection Sort.*

g. Pembahasan Hasil Perbandingan Efisiensi Algoritma

Tabel 3 adalah Hasil perbandingan efisiensi algoritma *Selection Sort* dan *Insertion Sort* [10] dengan komponen memori yang terpakai (*memory usage*) dan waktu eksekusi (*execution time*) pada Aplikasi Perbandingan Efisiensi Algoritma *Sorting* Dalam Penentuan Jarak Terdekat (Studi Kasus : *Pet Shop* di Bandar Lampung)

Tabel 3. Hasil Perbandingan Efisiensi Algoritma

Proses	***Memory Usage Selection Sort* (100 Data)**	**Waktu Eksekusi *Selection Sort***	***Memory Usage Insertion Sort* (100 Data)**	**Waktu Eksekusi *Insertion Sort***
Percobaan 1	19451 KB	4 ms / 100 data.	30354 KB	8 ms / 100 data.
Percobaan 2	11663 KB	1 ms / 100 data.	24356 KB	2 ms / 80 data. 3 ms / 100 data.
Percobaan 3	29802 KB	1 ms / 80 data. 1 ms / 100 data.	18791 KB	2 ms / 80 data. 2 ms / 100 data.
Percobaan 4	16359 KB	1 ms / 100 data.	22905 KB	1 ms / 100 data.
Percobaan 5	13345 KB	1 ms / 80 data.	11134 KB	1 ms / 60 data. 1 ms / 80 data. 1 ms / 100 data.
Percobaan 6	22793 KB	1 ms / 100 data.	24356 KB	1 ms / 40 data. 2 ms / 80 data. 3 ms / 100 data.
Percobaan 7	15075 KB	1 ms / 100 data.	17480 KB	1 ms / 20 data.

				2 ms / 60 data. 1 ms / 100 data.
Percobaan 8	13269 KB	1 ms / 60 data.	39470 KB	2 ms / 60 data. 1 ms / 100 data.
Percobaan 9	15179 KB	1 ms / 60 data. 1 ms / 80 data.	30827 KB	1 ms / 40 data. 1 ms / 60 data. 1 ms / 80 data.
Percobaan 10	29688 KB	1 ms / 80 data.	36375 KB	3 ms / 100 data.
Rata-rata	18.662 KB	1,5 ms / 100 data. 1 ms / 80 data. 1 ms / 60 data	25.604 KB	2,56 ms / 100 data. 1,6 ms / 80 data 1,5 ms / 60 data. 1 ms / 40 data.

3.6 Pembahasan

Berdasarkan Tabel 1 hasil perbandingan efisiensi, menunjukkan jumlah data sebanyak 100 data untuk waktu eksekusi paling cepat adalah algoritma *Selection Sort* dengan *running time* rata-rata 1.5 ms / 100 data, 1 ms / 80 data, 1 ms / 60 data dan waktu eksekusi paling lama adalah algoritma *Insertion Sort* dengan *running time* rata-rata 2.56 ms / 100 data, 1.6 ms / 80 data, 1.5 ms / 60 data, dan 1 ms / 40 data. Sedangkan untuk penggunaan memori yang terpakai dari 100 data tersebut, algoritma *Selection Sort* menghasilkan penggunaan memori yang terpakai lebih sedikit yaitu sebesar 18.662 KB dan algoritma *Insertion Sort* menghasilkan penggunaan memori yang terpakai lebih besar yaitu sebesar 25.604 KB. Dari hasil analisa perbandingan efisiensi algoritma *Selection Sort* dan *Insertion Sort* didapatkan bahwa algoritma *Selection Sort* adalah algortima terbaik yang efektif dan efisien dalam menangani masalah pengurutan data.

4. SIMPULAN

4.1 Simpulan

Pada penelitian Perbandingan efisiensi dengan Algoritma sorting dalam penentuan jarak terdekat pada *PetShop* ini dapat diambil kesimpulan sebagai berikut:

1. Berdasarkan jumlah data sebanyak 100 data, hasil perbandingan efisiensi untuk waktu eksekusi paling cepat adalah algoritma *Selection Sort* dengan *running time* rata-rata 1.5 ms / 100 data, 1 ms / 80 data, 1 ms / 60 data dan waktu eksekusi paling lama adalah algoritma *Insertion Sort* dengan *running time* rata-rata 2.56 ms / 100 data, 1.6 ms / 80 data, 1.5 ms / 60 data, dan 1 ms / 40 data. Sedangkan untuk penggunaan memori yang terpakai dari 100 data tersebut, algoritma *Selection Sort* menghasilkan penggunaan memori yang terpakai lebih sedikit yaitu sebesar 18.662 KB dan algoritma *Insertion Sort* menghasilkan penggunaan memori yang terpakai lebih besar yaitu sebesar 25.604 KB. Dari hasil analisa perbandingan efisiensi algoritma *Selection Sort* dan *Insertion Sort* didapatkan bahwa algoritma *Selection Sort* adalah algortima [11] terbaik yang efektif dan efisien dalam menangani masalah pengurutan data
2. Aplikasi ini menampilkan media informasi untuk menentukan jarak *Pet Shop* terdekat dan terjauh, tabel dan grafik memori yang berfungsi untuk menentukan memori yang terpakai serta menampilkan grafik waktu.
3. Aplikasi ini diharapkan dapat dikembangkan dengan menerapakan beberapa metode lainnya untuk mendapatkan hasil perbandingan yang maksimal.

5. SARAN

Saran penelitian Perbandingan efisiensi dengan Algoritma sorting dalam penentuan jarak terdekat pada *PetShop* yang telah dibangun ini adalah sebagai berikut:

1. Algoritma *Selection Sort* diharapkan kedepanya dapat dikembangkan agar tidak

membutuhkan penggunaan beberapa *method* tambahan.

2. Algoritma *Selection Sort* diharapkan dapat lebih stabil dalam menangani masalah pengurutan data.
3. Algoritma *Insertion Sort* diharapkan dapat lebih efektif dalam mencari posisi yang tepat untuk elemen larik agar tidak membutuhkan banyak operasi.
4. Algoritma *Insertion Sort* [12] diharapkan kedepannya dapat dikembangkan agar dapat menangani masalah pengurutan dengan *list* yang terbalik tanpa harus memindai mengganti seluruh bagian.

UCAPAN TERIMA KASIH

Penulis mengucapkan terima kasih kepada Tim Redaksi Jurnal Teknika Politeknik Negeri Sriwijaya yang telah memberi memberi kesempatan, sehingga artikel ilmiah ini dapat diterbitkan

DAFTAR PUSTAKA

[1] E. Retnoningsih, "Algoritma Pengurutan Data (Sorting) Dengan Metode Insertion Sort dan Selection Sort," *Inf. Manag. Educ. Prof.*, vol. 3, no. 1, 2018.

[2] D. Anggreani, A. P. Wibawa, P. Purnawansyah, and H. Herman, "Perbandingan Efisiensi Algoritma Sorting dalam Penggunaan Bandwidth," *Ilk. J. Ilm.*, vol. 12, no. 2, 2020, doi: 10.33096/ilkom.v12i2.538.96-103.

[3] R. R. Basir, "Analisis Kompleksitas Ruang dan Waktu Terhadap Laju Pertumbuhan Algoritma Heap Sort, Insertion Sort dan Merge dengan Pemrograman Java," *STRING (Satuan Tulisan Ris. dan Inov. Teknol.*, vol. 5, no. 2, 2020, doi: 10.30998/string.v5i2.6250.

[4] R. Kurniawan and P. M. Putra, "Implementasi Metode Sequential Searching pada Aplikasi 'RUMAH KUCING PASIFIK' Berbasis Mobile," *J. SIMADA (Sistem Inf. dan Manaj. Basis Data)*, vol. 4, no. 2, 2021, doi: 10.30873/simada.v4i2.3009.

[5] R. Pressman, "Rekayasa Perangkat Lunak: Pendekatan Praktisi Buku I," *J. Inf. Politek. Indonusa Surakarta ISSN*, vol. 4, no. 1, 2015.

[6] A. Ramawan and Y. P. Sari, "Designing Website-Based Mobile Application Using Quick Sort And Insert Sort Methods In Assipa Drug Store, Karang Anyar, South Lampung," *Int. Conf. ...*, 2020.

[7] E. N. Putri, R. Kurniawan, and Y. P. Sari, "Rancang Bangun Aplikasi E-info Produk Halal Majelis Ulama Indonesia Menggunakan Metode Quick Search Algorithm Berbasis Mobile," *Pros. Semin. ...*, 2019.

[8] R. Rachmatullah, D. Kardha, and M. P. Yudha, "Aplikasi E-Commerce Petshop dengan Fitur Petpedia," *Go Infotech J. Ilm. STMIK AUB*, vol. 26, no. 1, 2020, doi: 10.36309/goi.v26i1.120.

[9] Y. P. Sari, I. Agus, and E. P. Sari, "SMART TROLLEY PADA FITRINOPANE SWALAYAN DENGAN MENERAPKAN METODE BRUTE FORCE BERBASIS MOBILE," *J. Inform.*, vol. 21, no. 2, 2022, doi: 10.30873/ji.v21i2.3080.

[10] M. Chaeron, B. S. Wahyuaji, and A. Soepardi, "Development of Direction-Parallel Strategy for Shorting A Tool Path in The Triangular Pocket Machining," *J. Ilm. Tek. Ind.*, vol. 18, no. 1, 2019, doi: 10.23917/jiti.v18i1.7151.

[11] A. Sonita and F. Nurtaneo, "ANALISIS PERBANDINGAN ALGORITMA BUBBLE SORT, MERGE SORT, DAN QUICK SORT DALAM PROSES PENGURUTAN KOMBINASI ANGKA DAN HURUF," *Pseudocode*, vol. 2, no. 2, 2016, doi:

10.33369/pseudocode.2.2.75-80.

[12] A. Kalaivani and K. Swetha, "An Enhanced Bidirectional Insertion Sort over Classical Insertion Sort," *Int. J. Image Graph.*, vol. 21, no. 2, 2021, doi: 10.1142/S0219467821500248.