BAB IV

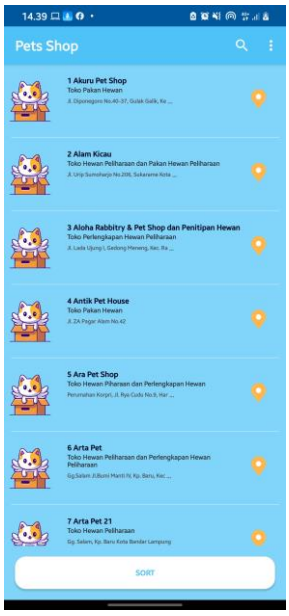
HASIL DAN PEMBAHASAN

4.1 Implementasi Aplikasi

Tahap ini berisi hasil dari analisis dan perancangan yang telah dibahas pada bab sebelumnya, dan untuk mengetahui apakah Aplikasi yang dibangun dapat memenuhi kebutuhan pengguna dan dapat berjalan dengan baik serta dapat menghasilkan *output* sesuai dengan tujuan yang di inginkan oleh pengguna.

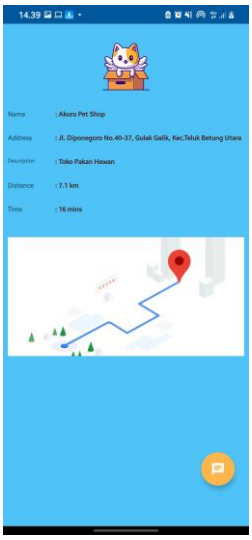
4.1.1 Hasil Penerapan Metode *Seletion Sort* dan *Insertion Sort*

Berikut hasil penerapan metode pada aplikasi Penentuan Jarak Terdekat Pada *Pet Shop :*



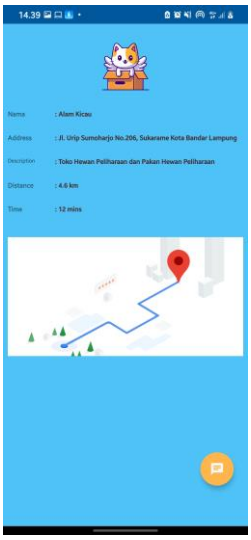
Gambar 4.1 *Interface* Halaman Daftar *Pet Shop* Admin.

Dapat dilihat pada gambar di atas, yaitu terdapat daftar nama- nama *Pet Shop* dan jarak pada setiap *Pet Shop* yang belum terurut dari 1 sampai 100 daftar *Pet Shop* yang tersedia pada aplikasi. Jarak *Pet Shop* dapat dilihat ketika kita masuk ke halaman deskripsi *Pet Shop.*



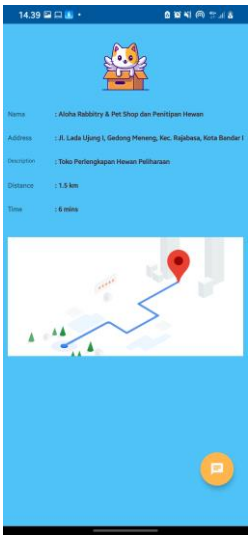
Gambar 4.2 *Interface* Halaman Deskripsi *Pet Shop* dan *Map View.*

Dapat dilihat pada gambar diatas, yaitu *Pet Shop* dengan urutan pertama pada halaman daftar *Pet Shop* pada aplikasi. *Pet Shop* tersebut memiliki jarak 7,1 km.



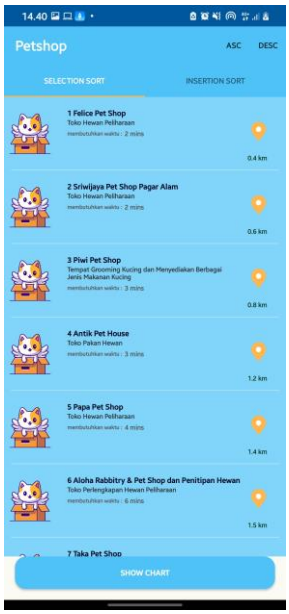
Gambar 4.3 *Interface* Halaman Deskripsi *Pet Shop* dan *Map View.*

Dapat dilihat pada gambar di atas, yaitu *Pet Shop* dengan urutan ke-2 (dua) pada halaman daftar *Pet Shop*. *Pet Shop* tersebut memiliki jarak 4,6 km.

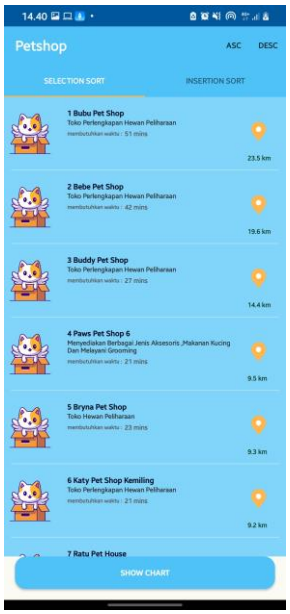


Gambar 4.4 *Interface* Halaman Deskripsi *Pet Shop* dan *Map View.*

Dapat dilihat pada gambar diatas, yaitu *Pet Shop* dengan urutan ke-3 (tiga) pada halaman daftar *Pet Shop*. *Pet Shop* tersebut memiliki jarak 1,5 km. Jarak pada daftar-daftar *Pet Shop* yang tersedia semuanya masih belum terurut sehingga perlu dilakukan *sorting*.

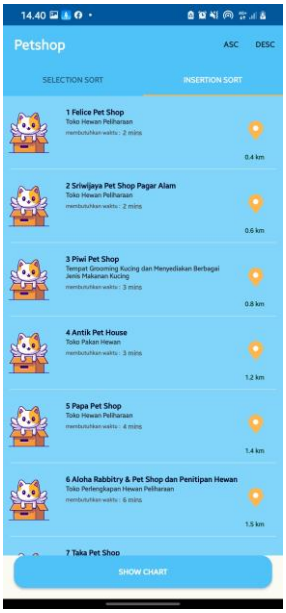


Gambar 4.5 *Interface* Halaman *Sorting Selection Sort (Ascending).*

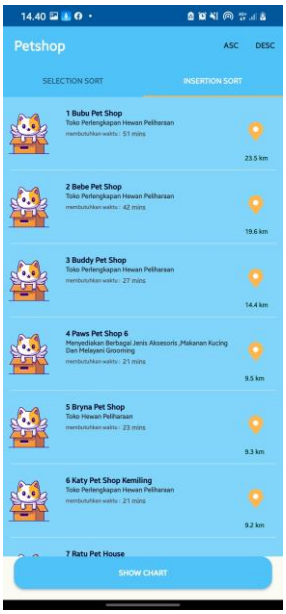


Gambar 4.6 *Interface* Halaman *Sorting Selection Sort (Descending).*

Dapat dilihat pada gambar di atas, yaitu *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* untuk menentukan dari jarak terdekat atau jarak terjauh terlebih dahulu.



Gambar 4.7 *Interface* Halaman *Sorting Insertion Sort (Ascending).*



Gambar 4.8 *Interface* Halaman *Sorting Insertion Sort (Descending).*

Dapat dilihat pada gambar di atas yaitu *sorting* menggunakan metode *Insertion Sort*, terdapat daftar nama-nama *Pet Shop* yang sudah terurut dari jarak terjauh sampai terdekat. Di halaman *sorting user* dapat memilih fitur *Ascending* dan *Descending*, sehingga *user* bisa menentukan dari jarak terdekat atau jarak terjauh terlebih dahulu.

4.2 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 :

4.2.1 Tampilan *Interface* Aplikasi Admin

4.2.1.1 Tampilan *Interface* Halaman *Splash Screen*

Halaman *splash screen* akan muncul saat pertama kali aplikasi dijalankan.

Gambar 4.9 *Interface Halaman Splash Screen*.

4.2.1.2 Tampilan *Interface* Halaman *Login*

Halaman *Login* muncul pada saat admin mengakses aplikasi dan akan muncul *form sign in* yaitu *username* dan *password* serta *Login* untuk masuk ke aplikasi.

Gambar 4.10 *Interface Halaman Login*.

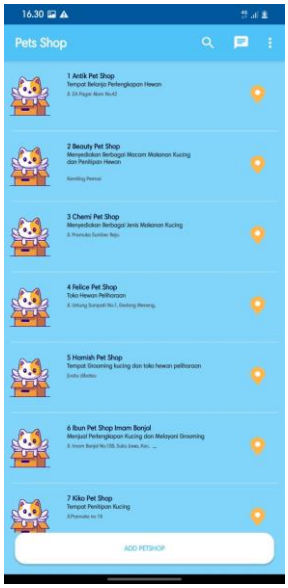
4.2.1.3 Tampilan *Interface* Halaman *Register*

Halaman *Register* berguna bagi pengguna aplikasi untuk melakukan pendaftaran sebagai *admin* atau *user*.

Gambar 4.11 *Interface Halaman Register*.

4.2.1.4 Tampilan *Interface* Halaman Daftar *Pet Shop* Admin

Halaman Daftar *Pet Shop* pada admin berguna untuk melihat *Pet Shop* mana saja yang ada pada aplikasi.



Gambar 4.12 *Interface* Halaman Daftar *Pet Shop* Admin.

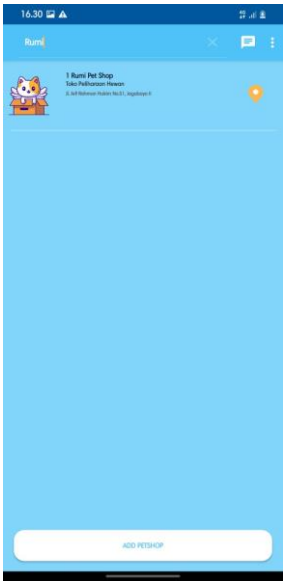
4.2.1.5 Tampilan *Interface* Halaman Tambah *Pet Shop*

Halaman Tambah *Pet Shop* berguna bagi admin untuk menambahkan *Pet Shop* dan produk pada *Pet Shop* tersebut.

Gambar 4.13 *Interface* Halaman Tambah *Pet Shop.*

4.2.1.6 Tampilan *Interface* Halaman Pencarian *Pet*

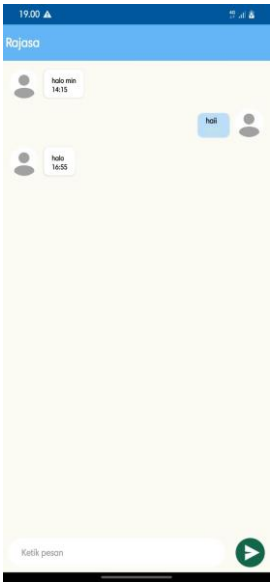
Halaman Pencarian *Pet Shop*,berguna untuk user dan admin menemukan *Pet Shop* yang di tuju.



Gambar 4.14 *Interface* Halaman Pencarian *Pet Shop.*

4.2.1.7 Tampilan *Interface* Halaman *Chat* Admin

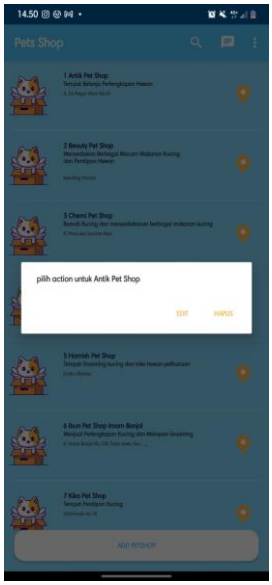
Halaman *Chat* Admin berguna untuk admin dan *user* saling berinteraksi lewat fitur ini.



Gambar 4.15 *Interface* Halaman *Chat* Admin.

4.2.1.8 Tampilan *Interface* Halaman Opsi

Halaman Opsi ini berguna bagi admin untuk mengubah *Pet Shop* jika ada perubahan pada *Pet Shop* dan untuk menghapus *Pet Shop* tersebut.

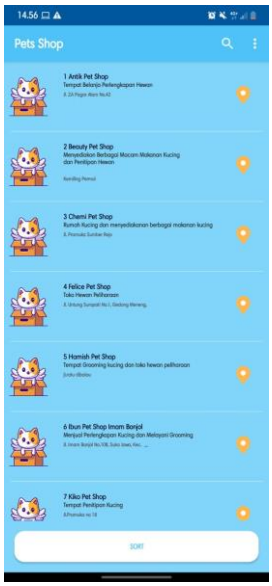


Gambar 4.16 *Interface* Halaman Opsi.

4.2.2 Tampilan *Interface* Aplikasi *User*

4.2.2.1 Tampilan *Interface* Halaman Daftar *Pet Shop User*

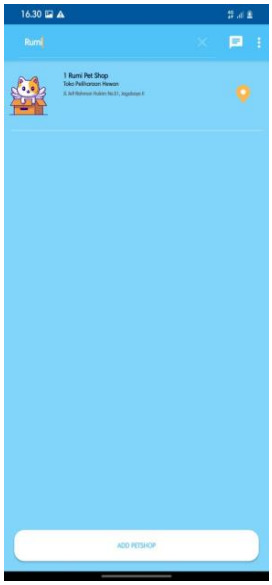
Halaman Daftar *Pet Shop* pada *user* berguna untuk melihat daftar nama-nama *Pet Shop* yang ada pada aplikasi.



Gambar 4.17 *Interface* Halaman Daftar *Pet Shop.*

4.2.2.2 Tampilan *Interface* Halaman Pencarian *Pet Shop*

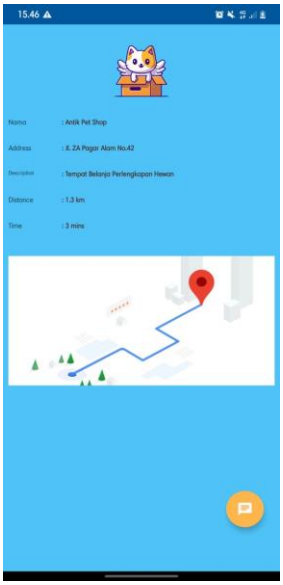
Halaman Pencarian *Pet Shop* pada *user* berfungsi untuk mencari *Pet Shop* yang diinginkan oleh *user*.



Gambar 4.18 *Interface* Halaman Pencarian *Pet Shop.*

4.2.2.3 Tampilan *Interface* Halaman Deskripsi *Pet Shop* dan *Map View*

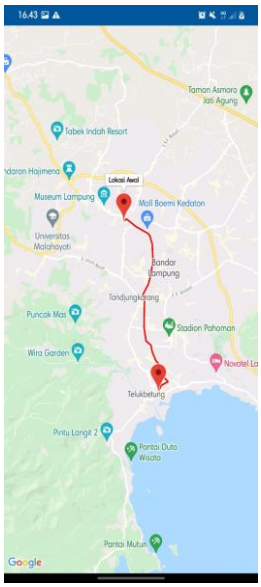
Halaman ini menampilkan deskripsi yang ada pada *Pet Shop* tersebut, serta menampilkan lokasi *Pet Shop* yang akan di tuju oleh *user*.



Gambar 4.19 *Interface* Halaman Deskripsi *Pet Shop* dan *Map View*.

4.2.2.4 Tampilan *Interface* Halaman Rute *Pet Shop*

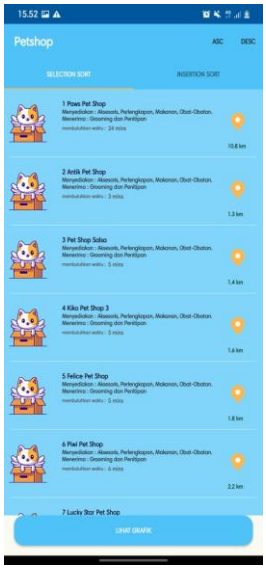
Halaman Rute *Pet Shop* berguna bagi *user* sebagai petunjuk lokasi dari titik *user* menuju titik *Pet Shop* yang akan di tuju.



Gambar 4.20 *Interface* Halaman Rute *Pet Shop*.

4.2.2.5 Tampilan *Interface* Halaman *Sorting Selection Sort (Ascending)*

Halaman ini berguna untuk melihat jarak *Pet Shop* dari yang terdekat hingga terjauh dengan menggunakan *Sorting Selection Sort (Ascending).*



Gambar 4.21 *Interface* Halaman *Sorting Selection Sort (Ascending).*

4.2.2.6 Tampilan *Interface* Halaman *Sorting Selection Sort (Descending)*

Halaman ini berguna untuk melihat jarak *Pet Shop* dari yang terjauh hingga yang terdekat, dengan menggunakan *Sorting Selection Sort (Descending).*

Gambar 4.22 *Interface* Halaman *Sorting Selection Sort (Descending).*

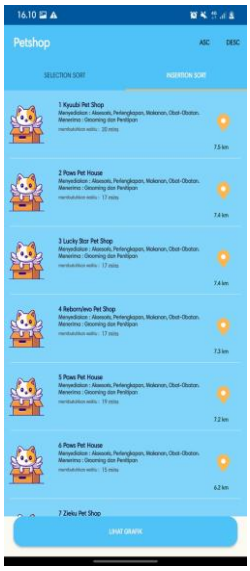
4.2.2.7 Tampilan *Interface* Halaman *Sorting Insertion Sort (Ascending)*

Halaman ini berguna untuk melihat jarak *Pet Shop* dari yang terdekat hingga yang terjauh dengan menggunakan *Sorting Insertion Sort (Ascending)*

Gambar 4.23 *Interface* Halaman *Sorting Insertion Sort (Ascending).*

4.2.2.8 Tampilan *Interface* Halaman *Sorting Insertion Sort (Descending)*

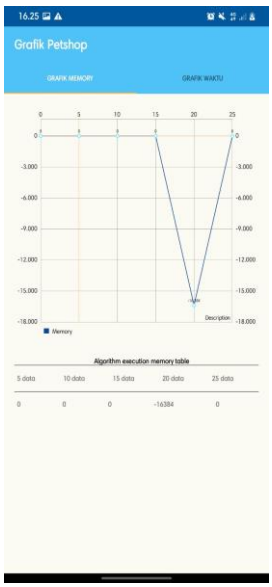
Halaman ini berfungsi untuk melihat jarak *Pet Shop* dari yang terjauh hingga yang terdekat dengan menggunakan *Sorting Insertion Sort (Descending).*



Gambar 4.24 *Interface* Halaman *Sorting Insertion Sort (Descending).*

4.2.2.9 Tampilan *Interface* Halaman Grafik Memori *Selection Sort*

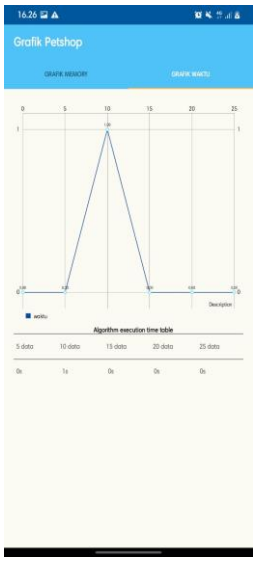
Halaman ini berguna untuk melihat grafik memori yang terpakai pada saat melakukan *sorting* menggunakan *Selection Sort.*



Gambar 4.25 *Interface* Halaman Grafik Memori *Selection Sort.*

4.2.2.10 Tampilan *Interface* Halaman Grafik Waktu *Selection Sort*

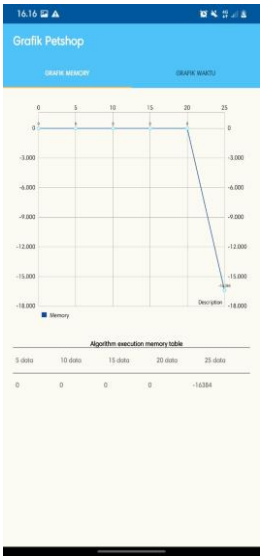
Halaman ini berfungsi untuk melihat grafik waktu yang terpakai pada saat melakukan *sorting* menggunakan *Selection Sort.*



Gambar 4.26 *Interface* Halaman Grafik Waktu *Selection Sort.*

4.2.2.11 Tampilan *Interface* Halaman Grafik Memori *Insertion Sort*

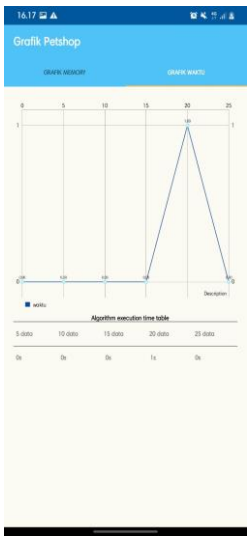
Halaman ini berguna untuk melihat grafik memori yang terpakai pada saat melakukan *sorting* menggunakan *Insertion Sort.*



Gambar 4.27 *Interface* Halaman Grafik Memori *Insertion Sort.*

4.2.1.12 Tampilan *Interface* Halaman Grafik Waktu *Insertion Sort*

Halaman ini berfungsi untuk melihat grafik waktu yang terpakai pada saat melakukan *sorting* menggunakan *Insertion Sort*.



Gambar 4.28 *Interface* Halaman Grafik Waktu *Selection Sort.*

4.3 Pembahasan Hasil Pengujian Aplikasi

Hasil pengujian *(testing)* aplikasi yang telah dibuat menggunakan *Black box testing*. Pengujian ini dimaksudkan untuk mengevalusi hasil. *Black box testing* sendiri memiliki 5 komponen pengujian yaitu uji *interface*, uji fungsi menu dan tombol, uji kinerja *loading* dan tingkah laku, dan uji inisiasi dan terminasi.

4.3.1 Hasil Pengujian Fungsi Kinerja *Loading*

Pengujian ini dilakukan saat aplikasi mulai dijalankan sampai menampilkan halaman pada android yang dipakai dalam pengujian. Berikut hasil perbedaan waktu loading :

Tabel 4.1 Hasil Pengujian Fungsi Kinerja Loading.

Proses	**Waktu Loading (Detik)**		
	Device 1	**Device 2**	**Device 3**
Loading membuka aplikasi	0,5	0,4	0,2

4.3.2 Pembahasan Hasil Pengujian *Interface*

Berdasarkan Tabel 2. Hasil Uji *Interface* pada lampiran hasil pengujian menggunakan metode *Black Box testing*. *Black box testing* terdiri dari 5 komponen yaitu uji fungsi menu dan tombol, uji *interface*, uji kinerja loading dan tingkah laku, uji struktur dan *database*, dan uji inisiasi dan terminasi. Pengujian Aplikasi Perbandingan Efisiensi Algoritma *Sorting* Dalam Penentuan Jarak Terdekat (Studi Kasus : *Pet Shop* di Bandar Lampung) dilakukan dengan 3 perangkat yang spesifikasi dan ukuran layar berbeda.

4.3.3 Pembahasan Hasil Perbandingan Efisiensi Algoritma

Hasil perbandingan efisiensi algoritma *Selection Sort* dan *Insertion Sort* 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) adalah sebagai berikut :

Tabel 4.2 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.

4.4 Pembahasan

Berdasarkan tabel 4.2 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.

Kelebihan algoritma *Selection Sort* pada Perbandingan Efisiensi Algoritma *Sorting* Dalam Penentuan Jarak Terdekat (Studi Kasus : *Pet Shop* di Bandar Lampung) adalah sebagai berikut :

1. Algoritma *Selection Sort* lebih sederhana dibandingkan dengan algoritma *sorting* yang lain.
2. Algoritma *Selection Sort* lebih mudah dalam menentukan data minimum dan maksimum.
3. Algoritma *Selection Sort* lebih cepat dalam pencarian data.
4. Operasi algoritma *Selection Sort* pertukarannya hanya dilakukan sekali saja.
5. Algoritma *Selection Sort* memiliki kompleksitas yang relatif lebih kecil.

Kekurangan algoritma *Selection Sort* pada Perbandingan Efisiensi Algoritma *Sorting* Dalam Penentuan Jarak Terdekat (Studi Kasus : *Pet Shop* di Bandar Lampung) adalah sebagai berikut :

1. Algoritma *Selection Sort* membutuhkan *method* tambahan.
2. Algoritma *Selection Sort* adalah algoritma yang tidak stabil.

Kelebihan algoritma *Insertion Sort* pada Perbandingan Efisiensi Algoritma *Sorting* Dalam Penentuan Jarak Terdekat (Studi Kasus : *Pet Shop* di Bandar Lampung) adalah sebagai berikut :

1. Algoritma *Insertion Sort* sederhana dalam penerapanya.
2. Algoritma *Inserion Sort* Lebih efisien dalam data yang sebagian sudah terurut.
3. *Looping* pada *Insertion Sort* sangat cepat, sehingga membuatnya menjadi salah satu algoritma pengurutan tercepat pada jumlah elemen yang sedikit.
4. Algoritma *Insertion Sort* adalah algoritma yang stabil.

Kekurangan algoritma *Insertion Sort* pada Perbandingan Efisiensi Algoritma *Sorting* Dalam Penentuan Jarak Terdekat (Studi Kasus : *Pet Shop* di Bandar Lampung) adalah sebagai berikut :

1. Banyak operasi yang diperlukan oleh algoritma *Insertion Sort* dalam mencari posisi yang tepat untuk elemen larik.
2. Jika *list* terurut terbalik algoritma *Insertion Sort* harus memindai dan mengganti seluruh bagian.