# BAB II

# TINJAUAN PUSTAKA

## 2.1 Aplikasi

Menurut Syani & Werstantia (2018) didalam jurnalnya : “Aplikasi sering juga disebut sebagai perangkat lunak, merupakan program komputer yang isi instruksinya dapat diubah dengan mudah”. Menurut Amnah, Sulyono, Ifan Reynaldi Utama (2019) Dijelaskan bahwa perangkat lunak aplikasi adalah subkelas perangkat lunak komputer yang memanfaatkan kemampuan komputer untuk melakukan suatu tugas yang diinginkan pengguna. Biasanya dibandingkan dengan perangkat lunak sistem yang mengintegrasikan berbagai kemampuan komputer, tetapi tidak secara langsung menerapkan kemampuan untuk melakukan tugas yang menguntungkan pengguna.

## 2.2 *Pet Shop*

Menurut Purwaningtias, F. (2017), *Pet Shop* merupakan salah satu tempat untuk menjual hewan peliharaan beserta peralatan dan perlengkapan untuk pemeliharaan hewan. Peralatan merupakan barang-barang yang cenderung wajib untuk dimiliki oleh para pemilik hewan peliharaan, sedangkan perlengkapan merupakan barang-barang yang tidak wajib dimiliki oleh pemilik hewan atau hanya berupa barang-barang aksesoris untuk hewan. *Pet Shop* juga menjual berbagai jenis hewan dan menyediakan fasilitas serta pelayanan untuk hewan peliharaan.

## 2.3 *Selection Sort*

Menurut Retnoningsih E. 2018 , *Selection Sort* adalah suatu metode pengurutan data 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 2.1 Cara Kerja *Selection Sort (Maximum).*

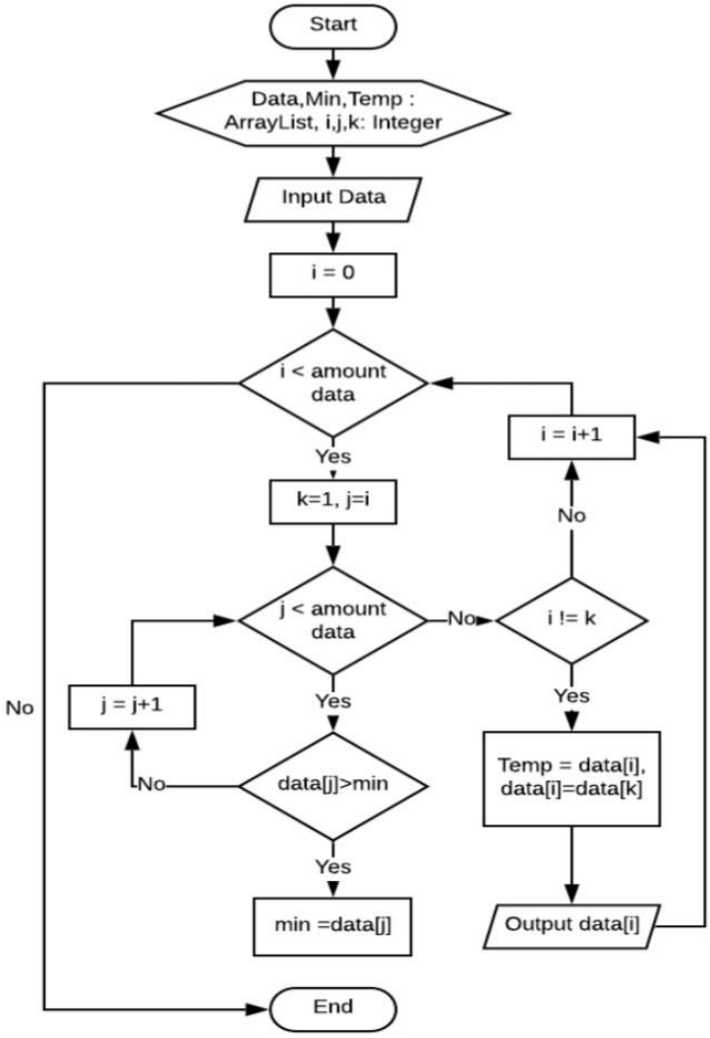
| **22** | **2** | **90** | **25** | **20** | **30** | **6** | **3** | **Data awal** |
|---|---|---|---|---|---|---|---|---|
| 22 | 2 | 90 | 25 | 20 | 30 | 6 | 3 | Data ke 8 di tukar dengan data ke 3 ( Terbesar) |
| 22 | 2 | 3 | 25 | 20 | 30 | 6 | 90 | Data ke 7 ditukar dengan data ke 6 |
| 22 | 2 | 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** |

Tabel 2.2 Cara Kerja *Selection Sort (Minimum Sort)*

| **44** | **55** | **12** | **42** | **94** | **18** | **6** | **67** | **Data Awal** |
|---|---|---|---|---|---|---|---|---|
| 6 | 55 | 12 | 42 | 94 | 18 | 44 | 67 | Tukarkan data ke 1 dengan data ke 7 |
| 6 | 12 | 55 | 42 | 94 | 18 | 44 | 67 | Tukarkan data ke 2 dengan data ke 3 |
| 6 | 12 | 18 | 42 | 94 | 55 | 44 | 67 | Tukarkan data ke 3 dengan data ke 6 |
| 6 | 12 | 18 | 42 | 94 | 55 | 44 | 67 | Data ke 4 tidak ditukarkan |
| 6 | 12 | 18 | 42 | 44 | 55 | 94 | 67 | Tukarkan data ke 5 dengan data ke 7 |
| 6 | 12 | 18 | 42 | 44 | 55 | 94 | 67 | Data ke 6 tidak ditukarkan |
| 6 | 12 | 18 | 42 | 44 | 55 | 67 | 94 | Tukarkan data ke 7 dengan data ke 8 |
| **6** | **12** | **18** | **42** | **44** | **55** | **67** | **94** | **Data setelah terurut** |

## 2.4 *Flowchart 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. Konsep *Selection Sort* adalah mencari atau memilih nilai terkecil ataupun nilai terbesar dan menukarnya dengan elemen paling awal pada paling kiri pada setiap tahap, proses akan terus berjalan hingga data akan mengahasilkan data yang terurut. Algoritma ini melakukan banyak perbandingan namun memiliki jumlah perpindahan data yang sedikit. Adapun *flowchart* algoritma *Selection Sort* dapat dilihat pada gambar 2.1 berikut :




Gambar 2.1 *Flowchart Selection Sort*.

## 2.5 *Insertion Sort*

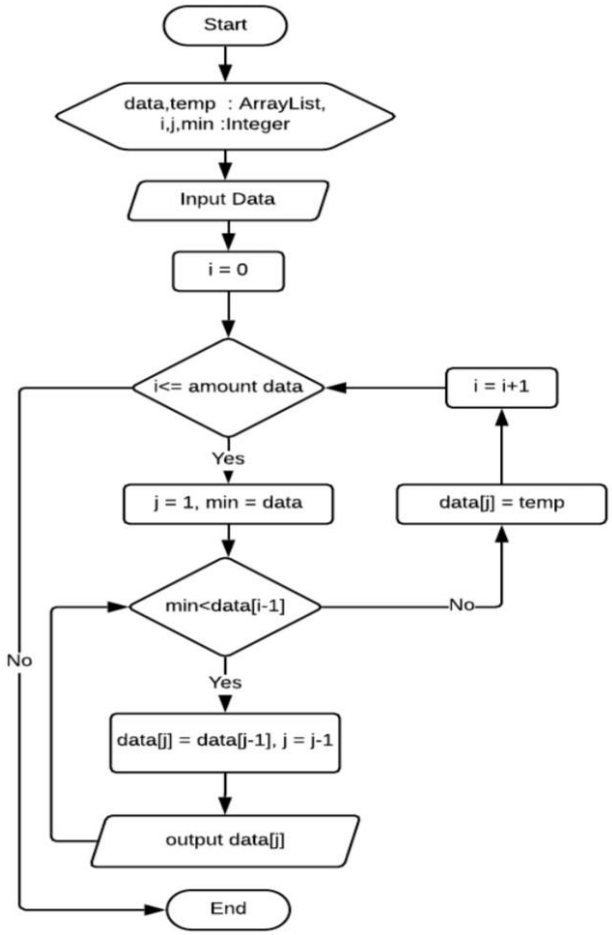
Menurut Retnoningsih E. 2018, cara kerja algoritma ini yaitu pengurutan dengan penyisipan bekerja dengan cara menyisipkan masing-masing nilai di tempat yang sesuai di antara elemen yang lebih kecil atau sama dengan nilai tersebut. Untuk menghemat memori, implementasinya menggunakan pengurutan di tempat yang membandingkan elemen saat itu dengan elemen sebelumnya yang sudah diurut, lalu menukarnya terus sampai posisinya tepat.

Tabel 2.3 Cara Kerja *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** |

## 2.6 *Flowchart 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 yang telah diurutkan. Algoritma pengurutan data insertion sort akan memeriksa setiap elemen, menemukan yang terkecil atau yang terbesar sesuai kebutuhan jenis pengurutan, dan menyisipkan dalam tempat yang tepat. Adapun *flowchart* algoritma *Insertion Sort* dapat dilihat pada gambar 2.2 berikut :




Gambar 2.2 *Flowchart Insertion Sort*.

## 2.7 Aplikasi *Mobile*

Menurut Putri E. N., Kurniawan R, & Sari, Y.P. (2019, August), Aplikasi *Mobile* adalah perangkat lunak yang berjalan pada perangkat *mobile* seperti *smartphone* atau *tablet PC.* Aplikasi *Mobile* juga dikenal sebagai aplikasi yang dapat diunduh dan memiliki fungsi tertentu sehingga menambah fungsionalitas dari perangkat *mobile* itu sendiri. Untuk mendapatkan *mobile application* yang diinginkan, *user* dapat mengunduhnya melalui situs tertentu sesuai dengan sistem operasi yang dimiliki. Google *Play Store* dan *Appstore* merupakan beberapa contoh dari situs yang menyediakan beragam aplikasi bagi pengguna Android dan iOS untuk mengunduh aplikasi yang diinginkan.

## 2.8 Android

Menurut Yuni Puspita Sari & Rionaldi Ali (2019), Android adalah sebuah sistem operasi untuk smartphone dan Tablet. Sistem operasi dapat diilustrasikan sebagai jembatan‟ antara piranti (*device*) dan penggunanya, sehingga pengguna bisa berinteraksi dengan *device*-nya dan menjalankan aplikasi-aplikasi yang tersedia pada *device*. *Mobile phone* adalah salah satu perangkat yang bergerak seperti telepon seluler atau komputer bergerak yang digunakan untuk mengakses jasa jaringannya.

## 2.9 *Firebase Realtime Database*

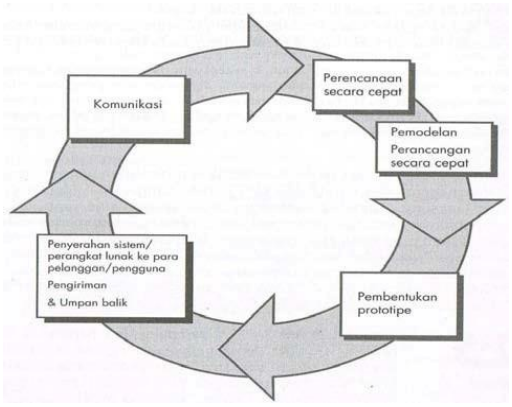
Menurut Ilhami, M. (2018), *Firebase Realtime Database* merupakan *cloud database*. Data disimpan dalam format JSON dan disinkronkan secara realtime ke setiap klien yang terhubung. Ketika membangun aplikasi *hybrid* lintas *platform*, seperti Android dan iOS maka semua klien berbagi satu *instance Realtime Database* dan secara otomatis menerima pembaruan dengan data tertentu. *Firebase Realtime Database* adalah basis data NoSQL dan karena itu memiliki optimalisasi dan fungsionalitas yang berbeda dibandingkan dengan basis data relasional. Membuat *database Firebase* bisa melalui *import file* JSON ke konsol *Firebase*, atau dapat juga dibuat langsung melalui halaman konsol *Realtime Database* secara manual.

## 2.10 Google Maps API

Google Maps API adalah suatu *library* yang berbentuk *JavaScript*. Cara membuat google maps untuk ditampilkan pada situs web atau aplikasi sangat mudah, hanya dengan membutuhkan pengetahuan mengenai HTML serta *JavaScript*, serta koneksi internet yang stabil. Dengan menggunakan google *maps* API, kita dapat menghemat waktu dan biaya untuk membangun aplikasi peta digital yang handal, sehingga kita bisa fokus hanya pada data-data yang akan ditampilkan. Jadi jika kita hanya membuat suatu data sedangkan peta yang akan ditampilkan adalah milik google sehingga kita tidak dipusingkan dengan membuat peta suatu daerah, bahkan dunia.

## 2.11 Metode Perangkat Lunak yang digunakan

Menurut Pressman, Rekayasa Perangkat Lunak (2018). *Prototype* merupakan metode yang efektif dalam merancang perangkat lunak. *Prototype* dimulai dengan mengumpulkan kebutuhan yang akan di rancang. Pengembang mendefinisikan *object* keseluruhan dari perangkat lunak, mengidentifikasi segala aktifitas yang diketahui dan kemudian melakukan "perancangan kilat". Perancangan kilat berfokus pada penyajian dari aspek-aspek perangkat lunak tersebut yang akan nampak bagi pelanggan atau pemakai (contohnya pendekatan *input* dan format *output*), *prototype* memiliki 5 tahapan seperti pada gambar 2.3 berikut :




Gambar 2.3 Metode *Prototype* (sumber : Pressman, Rekayasa Perangkat Lunak, 2018).

## 2.12 *Unified Modeling Language* (UML)

Menurut Yuli Syafitri (2016), *Unified Modeling Language* (UML) adalah sebuah bahasa yang berdasarkan gambar untuk menvisualisasikan menspesifikasikan, membangun dan pendokumentasian dari sebuah sistem pengembangan perangkat lunak berbasis objek. UML bukanlah merupakan bahasa pemrograman tetapi model-model yang tercipta berhubungan langsung dengan berbagai macam bahasa pemrograman, sehingga memungkinkan melakukan pemetaan (*mapping*) langsung dari model-model yang dibuat dengan UML degan bahasa-bahasa pemrograman berorientasi obyek, seperti Java.

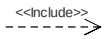
### 2.12.1 *Use Case Diagram*

*Use case diagram* merupakan pemodelan untuk kelakuakn (*behavior*) sistem informasi yang akan dibuat. *Use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut. Simbol-simbol yang digunakan dalam *use case diagram* yaitu:

Tabel 2.4 Simbol-Simbol *Use Case Diagram.*

| **Simbol** | **Deskripsi** |
|---|---|
| *Use Case*<br>  | Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor; dinyatakan dengan menggunakan kata kerja diawal-awal *frase* nama *use case.* |
| Aktor<br>  | Orang, proses atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar itu sendiri. |
| *Asosiasi*<br> | Komunikasi antara aktor dan *use case* yang berpartisipasi pada *use case* atau *use case* memiliki interaksi dengan aktor. |
| Ekstensi<br>  | Relasi *use case* tambahan ke sebuah *use case*, dimana *use case* yang ditambahkan dapat berdiri sendiri walau tanpa *use case* tambahan itu; mirip dengan prinsip *inheritance* pada pemograman berorientasi objek. |
| Generalisasi<br> | Hubungan generalisasi dan spesialisasi (umum-khusus) antara dua buah *use case* dimana fungsi yang satu adalah fungsi yang lebih umum dari lainnya. |

Tabel 2.4 Simbol-Simbol *Use Case Diagram* (Lanjutan).

| Simbol | Deskripsi |
|---|---|
| *Include*<br>  | a. *Include* berarti *use case* yang ditambahkan akan selalu dipanggil saat *use case* tambahan dijalankan<br>b. *Include* berarti *use case* yang tambahan akan selalu melakukan pengecekan apakah *use case* yang ditambahkan telah dijalankan sebelum *use case* tambahan dijalankan. |

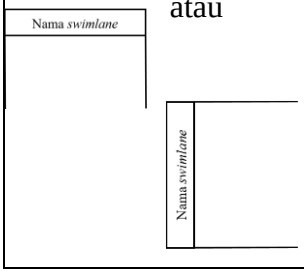
### 2.12.2 *Activity Diagram*

*Activity Diagram* menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak. Perlu diperhatikan bahwa diagram aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan aktor, jadi aktivitas yang dapat dilakukan oleh sistem. Berikut adalah simbol-simbol *activity diagram* :

Tabel 2.5 Simbol-Simbol *Activity Diagram.*

| Simbol | Nama | Deskripsi |
|---|---|---|
|  | Status awal | Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal. |
|  | Aktivitas | Aktivitas yang dilakukan sistem, biasanya diawali dengan kata kerja. |
|  | Percabangan/ *decision* | Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu. |
|  | Penggabungan / *join* | Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu. |

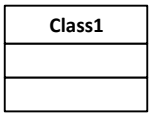
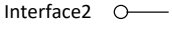
Tabel 2.5 Simbol-Simbol *Activity Diagram (Lanjutan).*

| | | |
|---|---|---|
|  | Status akhir | Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir |
|  | *smiwlane* | Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi |

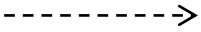
### 2.12.3 *Class Diagram*

Menurut Melda Agarina, Tria Devi Miranti, Sutedi (2019) dalam jurnalnya. *Class diagram* merupakan alat bantu untuk menentukan langkah-langkah kerja yang akan dilakukan oleh pemogram di mulai dari proses pengumpulan data, sampe pembentukan tabel sesuai dengan permasalahan yang ditangani. *Class diagram* ini terlebih dahulu dirancang dalam mendukung rancangan pengolahan data elektronis supaya dapat berjalan dengan baik, dan dengan relasi yang baik akan di peroleh gambaran umum sistem yang akan di persiapkan.

Tabel 2.6 Simbol-Simbol *Class Diagram.*

| **Simbol** | **Deskripsi** |
|---|---|
| Kelas<br>  | Kelas pada struktur sistem. |
| Natarmuka/*interface*<br>  | Sama dengan konsep *interface* dalam pemograman berorientasi objek. |

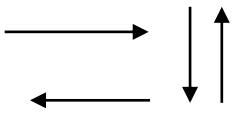
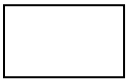
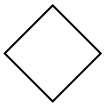
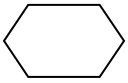
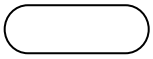
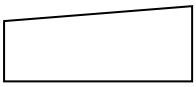
Tabel 2.6 Simbol-Simbol *Class Diagram* (Lanjutan).

| Simbol | Deskripsi |
|---|---|
| Asosiasi  | Relasi antar kelas dalam makna umum, asosiasi biasanya juga disertai dengan *multiplicity*. |
| Asosiasi berarah  | Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan *multiplicity*. |
| Generalisasi  | Relasi antar kelas dengan makna generalisasi-spesialisasi (umum-khusus). |
| Kebergantungan  | Relasi antar kelas dengan makna kebergantungan antar kelas. |
| Agregasi  | Relasi antar kelas dengan makna semua bagian (*whole-part*). |

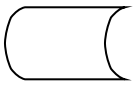
## 2.13 *Flowchart*

Menurut Wibawanto (2017:20), *flowchart* adalah suatu bagan dengan simbol-simbol tertentu yang menggambarkan urutan proses secara mendetail dan hubungan antara suatu proses (instruksi) dengan proses lainnya dalam suatu program. Bagan alir menggunakan serangkaian simbol standar untuk mendeskripsikan melalui gambar prosedur pemrosesan transaksi yang digunakan perusahaan dan arus data yang melalui sistem.

Tabel 2.7 Simbol-Simbol *Flowchart.*

| | | |
|---|---|---|
|  | Arus / *Flow* | Penghubung antara prosedur/proses. |
|  | *Connector* | Simbol keluar/masuk prosedur atau proses dalam lembar/halaman yang sama. |
|  | *Off-line Connector* | Simbol keluar/masuk prosedur atau proses dalam lembar/halaman yang lain. |
|  | *Process* | Simbol yang menunjukkan pengolahan yang dilakukan komputer. |
|  | *Decision* | Simbol untuk kondisi yang menghasilkan kemungkinan jawaban / aksi. |
|  | *Predefined Process* | Simbol untuk mempersiapkan penyimpanan yang akan digunakan sebagai tempat pengolahan didalam *storage.* |
|  | Terminal | Simbol untuk permulaan atau akhir darti suatu program. |
|  | *Manual Input* | Simbol untuk pemasukan data secara manual *on-line keyboard.* |
|  | *Input-Output* | Simbol yang menyatakan proses *input* dan *output* tanpa tergantung dengan jenis peralatannya. |

Tabel 2.7 Simbol-Simbol *Flowchart* (Lanjutan).

| | | |
|---|---|---|
|  | *Document* | Simbol yang menyatakan input berasal dari dokumen dalam bentuk kertas atau *output* di cetak dikertas. |
|  | *Disk and On-line Storage* | Simbol untuk menyatakan input berasal dari disk atau *output* di simpan ke *disk.* |

## 2.14 Pengujian *Black Box*

Menurut M. Sidi Mustaqbal, Roeri Fajri Firdaus, Hendra Rahmadi (2015) dalam Jurnal Ilmiah Teknologi Informasi Terapan, *Black Box Testing* berfokus pada spesifikasi fungsional dari perangkat lunak. *Tester* dapat mendefinisikan kumpulan kondisi input dan melakukan pengetesan pada spesifikasi fungsional program.

## 2.15 Penelitian Terkait

Dalam penyusunan skripsi ini, peneliti terinspirasi dan mereferensi dari penelitian-penelitian sebelumnya yang berkaitan dengan skripsi ini. Daftar penelitian terkait ialah sebagai berikut :

1. Algoritma Pengurutan Data (*Sorting*) Dengan Metode *Insertion Sort* dan *Selection Sort*

   Endang Retnoningsih (2018) menguraikan bahwa, pengurutan merupakan hal yang tidak bisa dipisahkan dari dunia komputer. Adanya kebutuhan terhadap proses pengurutan memunculkan bermacam-macam metode

pengurutan yang bertujuan untuk memperoleh metode pengurutan yang optimal. Data terkadang berada dalam bentuk yang tidak berpola ataupun dengan pola tertentu yang diinginkan. Secara umum ada dua jenis pengurutan data yaitu model urut naik (*ascending*) yang mengurutkan data dari yang mempunyai nilai terkecil sampai terbesar dan model urut turun (*descending*) yang mengurutkan data dari yang mempunyai nilai terbesar sampai terkecil. Perbandingan pengurutan data (*sorting*) dalam sebuah array (L) menggunakan dua algoritma dengan prinsip kerja yang berbeda. Pada penelitian ini membandingkan algoritma metode *selection sort* menggunakan prinsip pertukaran elemen dalam proses *sorting*, dan metode insertion *sort* menggunakan prinsip geser dan sisip elemen dalam proses sorting. Hasil dari penelitian adalah metode *insertion sort* lebih unggul pada jumlah data yang sedikit, sedangkan metode *selection sort* lebih unggul pada jumlah data yang lebih banyak.

2. Pengembangan Sistem Informasi Penjualan *Pet Shop* Berbasis Web (Studi Kasus : Hachiko *Pet And Care*)

   Wulandari, Lestari (2019) menguraikan bahwa, Hachiko *Pet and Care* adalah sebuah usaha kelas menengah yang menjual berbagai macam kebutuhan dan perlengkapan hewan kesayangan seperti makanan, vitamin dan aksesoris. Selain itu. Hachiko *Pet and Care* telah menggunakan jejaring sosial, akan tetapi media pemasaran yang umum digunakan ini belum dilengkapi dengan fasilitas pembelian yang terintegrasi. Sistem informasi berbasis web yang dibangun sebelumnya juga masih belum maksimal dalam hal promosi serta kejelasan status pada pemesanan. Pengembangan website yang didukung SMS *gateway* sebagai media informasi promo atau informasi terbaru dari *Pet Shop* dapat mendukung perkembangan dan peningkatan usaha ini. Tujuan dari sistem informasi berbasis web yang dilengkapi dengan SMS *gateway* ini yaitu mengembangkan sistem informasi penjualan *Pet Shop* berbasis web dan SMS *gateway* di Hachiko *Pet and Care*. Pengembangan sistem pada Hachiko *Petshop* menggunakan PHP sebagai Bahasa pemrograman dan

Adobe *Dreamweaver* yang digunakan sebagai aplikasi perancangan website. Penyimpanan data menggunakan MySQL, sedangkan API sebagai aplikasi *cross-platform* untuk menjembatani atau mengomunikasikan antara database SMS *gateway* dengan SMS *devices*. Sistem informasi berbasis web dan SMS *gateway* yang dibuat dapat membantu kinerja pelayanan juga menghasilkan beberapa keluaran berupa laporan penjualan, penitipan dan pemacakan, serta pesan singkat kepada member baik secara *broadcast* maupun secara tunggal.

3. Aplikasi Lana *Pet Shop* Berbasis *Online*

Hidayatullah Asmy, Ratna Fitriani, Fadilah (2018) menguraikan, dalam penelitian ini data di peroleh dari Lana *Pet Shop* yang menyediakan layanan *grooming* dan penitipan yang berlokasi di jalan karang anyar dekat batas kota banjarbaru dan martapura. Meningkatkan arus transaksi pada toko Lana *Pet Shop* yang semakin padat, namun masih menggunakan pencatatan transaksi secara manual dan dalam hal ini tentu kurang dapat faktor yang utama, pencatatan transaksi secara manual rentan dalam kesalahan ataupun dapat terjadi kehilangan informasi data penitip yang dapat membuat tingkat kepercayaan terhadap layanan kurang menjamin. Dalam pendataan stok barang juga hanya mengandalkan pencatatan pada sebuah buku saja sehinggaa tidak jarang ketika ditinggal pemiliknya, para pegawai mengalami kesulitan dalam melihat harga barang karena harus mencari dulu dalam buku. Oleh karena itu, dibutuhkan suatu aplikasi yang menggantikan pencatatan transaksi secara manual. Aplikasi yang dapat membantu mengurangi tingkat kesalahan dalam pendataan membantu dalam pemantauan jumlah stok barang, membantu memberikan informasi tentang hewan dan promosi menggunakan media penyebaran, dan mempermudah dalam pencarian data saat konsumen menjemputan hewan yang di titipkan.

4. Analisis Perbandingan Algoritma *Bubble Sort* dan *Selection Sort* Pada Sistem Pendukung Keputusan Pemilihan Tempat *Kost* Berbasis Ios (*Iphone Operating System*)

   Dwi Yulian Rizki Saputra, Septi Andryana, Ira Diana Sholihati (2021) menguraikan, sistem pendukung keputusan pemilihan tempat kost berbasis iOS ini dirancang guna mempermudah mahasiswa yang akan melanjutkan pendidikan tinggi, khususnya di Universitas Nasional dan terkendala oleh keterbatasan informasi serta kecepatan mengakses informasi dalam mencari tempat kost idamannya. Adapun di dalam sistem ini terdapat proses analisis perbandingan waktu eksekusi antara algoritma *Bubble Sort* dan *Selection Sort* berdasarkan kriteria yang dipilih oleh pengguna. Kriteria atau kategori yang dapat dipilih oleh pengguna antara lain pemilihan tempat *kost* berdasarkan biaya, jarak tempat kost dengan kampus, luas tempat dan juga jumlah fasilitas yang tersedia. Dalam penggunaannya sistem ini dapat mengatasi permasalahan mahasiswa dalam penentuan tempat kost sesuai kriteria yang disediakan, dengan hasil akhir dari proses pengurutannya dapat digunakan sebagai acuan dalam memilih tempat kost yang diinginkan. Setelah dilakukan implementasi dan analisis dari sisi waktu eksekusi pada data kost sebanyak 15 tempat, algoritma *Selection Sort* dinyatakan sebagai pemilik waktu eksekusi tercepat berdasarkan proses pengurutan terhadap seluruh kriteria yang sudah ditentukan.

5. Perbandingan Efisiensi Algoritma *Sorting* dalam Penggunaan *Bandwidth*

   Desi Anggreani, Aji Prasetya Wibawa, Purnawansyah, Herman (2020) menguraikan, Algoritma yang paling sering digunakan adalah algoritma *sorting.* Telah banyak bermunculan algoritma *sorting* yang dapat digunakan, dalam penelitian ini peneliti mengambil tiga algoritma sorting yaitu : *Insertion Sort, Selection Sort,* dan *Merge Sort.* Adapun penelitian ini akan menganalisis perbandingan waktu eksekusi dan pengunaan memori dengan memperhatikan jumlah masukkan data setiap algoritma yang digunakan. Setelah melakukan implementasi dan analisis dari sisi

waktu eksekusi algoritma *Merge Sort* memiliki waktu eksekusi yang lebih cepat dalam mengurutkan data dengan nilai rata-rata waktu eksekusi sebesar 108.593777 ms pada jumlah data 3000. Sedangkan dalam jumlah data yang sama untuk waktu eksekusi paling lama adalah algoritma Selection Sort dengan besar waktu eksekusi 144.498144 ms, adapun dari sisi penggunaan memori dengan jumlah data 3000 Algoritma *Merge Sort* memiliki penggunaan memori yang paling tinggi dibanding kedua algoritma lainnya yaitu sebesar 21.444 MB sedangkan kedua algoritma lainnya secara berturut-turut memiliki penggunaan memori sebesar 20.837 MB dan 20.325 MB.