



BAB II

TINJAUAN PUSTAKA

2.1 Teori Umum

2.1.1 Aplikasi Mobile

Aplikasi Mobile merupakan sebuah software yang di buat untuk perangkat portable smartphome yang mengharuskan proses mendownload software mobile aplikasi di toko aplikasi agar dapat di gunakan. Sedangkan untuk jenis toko aplikasinya pun bervariasi seperti Apple app store dan Play store. Namun saat ini pasar toko aplikasi yang mampu menguasai pasar aplikasi adalah google playstore atau playstore sehingga bagi kita yang ingin menggunakan aplikasi atau mempublikasikan sebuah aplikasi maka dengan mudah anda melihat dan mendownload di playstore, yang di dalamnya ada banya aplikasi yang di tawarkan.

Meurut Buyens dan Jim (2010) Aplikasi *mobile* berasal dari dua kata, yaitu aplikasi dan *mobile*. Secara istilah, aplikasi adalah program siap pakai yang dibuat untuk melaksanakan suatu fungsi untuk pengguna atau aplikasi yang lain sedangkan *mobile* adalah perpindahan dari suatu tempat ke tempat yang lain. Secara lebih lengkap, aplikasi *mobile* adalah program siap pakai yang melaksanakan fungsi tertentu yang dipasang pada perangkat *mobile*.

Menurut Yonarisa (2012) Aplikasi Mobile adalah sebuah aplikasi yang memungkinkan manusia melakukan mobilitas dengan menggunakan perlengkapan seperti *PDA*, telepon seluler (*handphone*). Dengan menggunakan aplikasi *mobile*, manusia dapat dengan mudah melakukan berbagai macam aktifitas mulai dari hiburan, berjualan, belajar, mengerjakan pekerjaan kantor, *browsing*, *chatting*, *email*, dan lain sebagainya.

Jadi Aplikasi Mobile adalah sebuah aplikasi yang memungkinkan kita melakukan mobilitas dengan menggunakan perlengkapan seperti telepon seluler atau Handphone. Dengan menggunakan aplikasi mobile, kita dapat dengan mudah melakukan berbagai macam aktifitas mulai dari hiburan, berjualan, belajar, mengerjakan pekerjaan kantor, browsing dan lain sebagainya. Pemanfaatan



aplikasi mobile untuk hiburan paling banyak digemari oleh hampir 70% pengguna telepon seluler, karena dengan memanfaatkan adanya fitur game, music player, sampai video player membuat kita menjadi semakin mudah menikmati hiburan kapan saja dan dimanapun.

2.1.2 Rute Terpendek

Persoalan mencari lintasan terpendek di dalam graf merupakan salah satu persoalan optimasi. Graf yang digunakan dalam pencarian lintasan terpendek adalah graf berbobot (weighted graph), yaitu graf yang setiap sisinya diberikan suatu nilai atau bobot. Bobot pada sisi graf dapat menyatakan jarak antar tempat/kota, waktu pengiriman pesan, ongkos pembangunan dan sebagainya. Asumsi yang kita gunakan disini adalah bahwa semua bobot bernilai positif. Kata “terpendek” jangan selalu diartikan secara fisik sebagai panjang minimum, sebab kata “terpendek” berbeda-beda maknanya tergantung pada tipikal persoalan yang akan diselesaikan. Namun secara umum “terpendek” berarti meminimasi bobot pada suatu lintasan di dalam graf.

Menurut Purwananto dan Yudhi (2005) Proses penghitungan rute terpendek adalah proses mencari jarak terpendek atau biaya terkecil suatu rute dari node awal ke node tujuan dalam sebuah jaringan. Pada proses penghitungan rute terpendek terdapat dua macam proses yaitu proses pemberian label dan proses pemeriksaan node.

Menurut Andayani dan Wulan (2014 ; 165) Rute terpendek adalah suatu persoalan untuk mencari lintasan antara dua atau lebih simpul pada graf berbobot yang gabungan bobot sisi graf yang dilalui berjumlah paling minimum. Pada graf berbobot terdapat optimasi yang dapat dinyatakan dalam jarak antar kota, waktu pengiriman pesan, biaya dan sebagainya. Dalam hal ini bobot harus bernilai positif, walau dalam hal lain dapat bernilai negatif. Lintasan terpendek dengan verteks awal s dan verteks tujuan t didefinisikan sebagai lintasan terpendek dari s dan t dengan bobot minimum dan berupa lintasan sederhana (simple path).



Jadi Rute terpendek adalah pencarian rute atau path terpendek antara node yang ada pada graf dan biaya yang dihasilkan adalah minimum.

2.1.3 Algoritma Brute Force

Banyak yang mengatakan bahwa algoritma brute force merupakan jenis algoritma yang sifatnya straight, lurus atau bisa juga disebut sebagai algoritma yang lempeng. Algoritma brute force merupakan bentuk algoritma yang sangat kompleks, karena untuk dapat menyelesaikan masalah dengan teknik straight forward atau lempeng ini, dibutuhkan banyak masukan dan juga pertimbangan secara logis, sehingga dapat diperoleh sebuah keputusan pemecahan masalah yang langsung mengacu atau menuju kepada hasil aygn diinginkan. Algoritma brute force ini biasanya menggunakan pendekatan yang di arahkan pada pernyataan masalah atau problem statement, dan juga definisi konsep yang dilibatkan. Dalam implementasinya, algoritma brute force ini membutuhkan sebuah cara yang jelas namun sederhana.

(Andrew, Yuliant, Izzatul. 2015) Metode *Brute force* merupakan pendekatan yang langsung dalam memecahkan suatu masalah, yang didasarkan pada pernyataan masalah serta mendefinisikan terhadap konsep yang dilibatkan secara langsung. Pemecahan masalah menggunakan Metode *Brute force* sangat sederhana, langsung dan jelas. Karakteristik pada Metode *Brute force* :

1. Jumlah langkah yang di perlukan besar.
2. Dipakai sebagai dasar dalam menemukan suatu solusi yang lebih efisien atau kreatif.
3. Hampir semua masalah dapat diselesaikan dengan metode ini.
4. Digunakan sebagai dasar dalam perbandingan kualitas suatu algoritma.

2.1.3.1 Algoritma *Brute Force* Dengan Teknik *Exhaustive Search* Pada *Traveling Salesman Problem*

Exhaustive search adalah teknik pencarian solusi secara *brute force* untuk masalah yang melibatkan pencarian elemen dengan sifat khusus. Biasanya



di antara objek-objek kombinatorik seperti permutasi, kombinasi, atau himpunan bagian dari sebuah himpunan (Munir, Rinaldi. 2005). Penggunaan *Brute force* pada permasalahan TSP dapat di ilustrasikan sebagai berikut:

Diberikan banyak (n) tempat tujuan pada sebuah kota serta diketahui jarak antara setiap kota satu sama lain. Temukan perjalanan (*tour*) terpendek yang melalui setiap kota lainnya hanya sekali dan kembali lagi ke kota asal keberangkatan. Permasalahan TSP ini tidak lain adalah menemukan sirkuit Hamilton dengan bobot minimum. Langkah-langkah teknik *exhaustive search*:

1. Buat list (Enumerasi) setiap solusi yang mungkin dengan cara yang sistematis.
2. Evaluasi setiap kemungkinan solusi satu per satu, terlebih solusi yang *mirror*, bisa di keluarkan. Kemudian simpan solusi terbaik yang ada sampai saat ini.
3. Setelah pencarian berakhir, umumkan solusi terbaik.
4. Sumber daya yang dibutuhkan dalam pencarian solusi menggunakan *Exhaustive Search* sangat besar, meskipun teknik *exhaustive* secara teoritis menghasilkan solusi.

2.2 Teori Khusus

2.2.1 Konsep *Travelling Salesman Problem* (TSP)

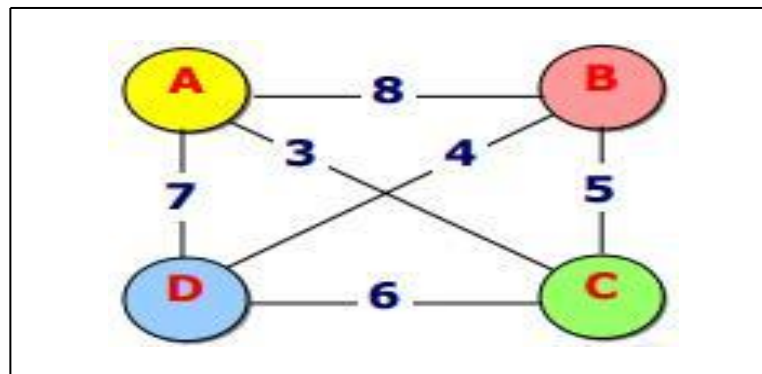
Travelling Salesman Problem (TSP) merupakan masalah kombinasi optimasi dalam operasi penelitian dari teori ilmu komputer. Dengan daftar tempat-tempat yang akan dikunjungi, cara ini sangat tepat untuk menemukan dengan sesingkat mungkin setiap tempat yang akan dikunjungi dengan waktu, dan penggunaan biaya yang tepat dan efisien.

(Tri, 2013) Apa yang dilakukan dalam TSP adalah membentuk sebuah rute perjalanan. Operator yang bisa digunakan untuk masalah TSP adalah pencarian urutan semua lokasi untuk memilih lokasi yang belum pernah terpilih satu demi satu sehingga dihasilkan satu rute kunjungan yang lengkap dari lokasi awal kemudian mengunjungi semua lokasi yang lain tepat satu kali dan akhirnya kembali ke lokasi awal. Sehingga dengan definisi tersebut dapat dikatakan bahwa konsep permasalahan TSP memiliki aturan sebagai berikut:



1. Harus mengunjungi setiap tempat tepat satu kali, tidak boleh kurang ataupun lebih.
2. Semua tempat harus dikunjungi dalam satu kali perjalanan (*tour*).
3. Dimulai dan diakhiri pada kota yang sama.

Sebagai ilustrasi dengan Gambar 2.1, diasumsikan bahwa simpul awal dan simpul akhir adalah 1. Suatu graf TSP dengan 4 simpul tersebut dikonversi menjadi sebuah pohon pencarian yang menghasilkan $(4-1)! = 6$ kemungkinan urutan kunjungan.



Sumber : Tri (2013)

Gambar 2.1 Contoh Graf Rute Perjalanan

Berdasarkan (Suyanto, 2010) TSP dikatakan ada 2 jenis, yaitu:

1. TSP asimetris

TSP jenis asimetris ini, biaya dari kota 1 ke kota 2 tidak sama dengan biaya dari kota 2 ke kota 1. Dengan n kota, besarnya ruang pencarian adalah () jalur yang mungkin.

2. TSP simetris

TSP jenis simetris, biaya dari kota 1 ke kota 2 adalah sama dengan biaya dari kota 2 ke kota 1. Apabila dengan n kota, jumlah jalur yang mungkin adalah jalur yang mungkin.



2.2.2 Use case Diagram

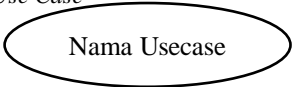


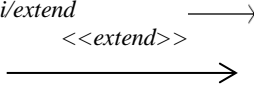
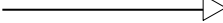
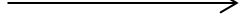
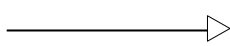
(Asidhiqi dan Hartanto. 2013 :56) *Use case* adalah konstruksi untuk mende-klarasikan bagaimana sistem akan terlihat di mata pengguna potensial. *Use case* terdiri dari sekumpulan skenario yang dilakukan oleh seorang aktor (orang, perangkat keras, urutan waktu atau sistem yang lain). Sedangkan *user case diagram* memfasilitasi komunikasi diantara analis dan pengguna serta diantara analis dan *client*.

(Widodo. 2011:10) Diagram *use case* bersifat *statis*, yang memperlihatkan himpunan *Use Case* dan aktor-aktor (suatu jenis khusus dari kelas) dan menggambarkan apa saja aktifitas yang dilakukan oleh suatu sistem dari sudut pandang pengamatan luar. Diagram ini terutama sangat penting untuk mengorganisasi dan memodelkan perilaku suatu sistem yang dibutuhkan serta diharapkan pengguna yang menjadi persoalan itu apa yang dilakukan bukan bagaimana melakukannya. *Use Case Diagram* menggambarkan fungsionalitas yang diharapkan dari sistem.

(Rosa dan Shalahudin. 2013 : 155) Usecase diagram merupakan pemodelan untuk kelakuan (behavior) sistem infromasi yang akan dibuat. Usecase mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem infromasi yang akan dibuat. Secara kasar, usecase digunakan untuk mengetahui fungsi apa saja yang ada didalam sebuah sistem infromasi dan siapa saja yang berhak menggunakan fungsi-fungsi itu.

Jadi Diagram *Use case* merupakan pemodelan untuk menggambarkan kelakuan (behavior) sistem yang akan dibuat. Diagram *use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem yang akan dibuat. Diagram *use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut.

Tabel 2.1 Simbol *Use Case Diagram*

Simbol	Deskripsi
<p><i>Use Case</i></p> 	Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit dan aktor.
<p>Aktor/<i>Actor</i></p> 	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi.
<p>Asosiasi/<i>association</i></p> 	Komunikasi antar aktor dan <i>Use Case</i> yang berpartisipasi.
<p>Ekstensi/<i>extend</i></p> 	Relasi <i>Use Case</i> tambahan ke sebuah <i>Use Case</i> dimana <i>Use Case</i> yang ditambah dapat berdiri sendiri walau tanpa <i>Use Case</i> tambahan.
<p>Generalisasi/<i>generalization</i></p> 	Hubungan generalisasi dan spesialisasi antara dua buah <i>Use Case</i> yang mana fungsi yang satu lebih umum dari yang lainnya.
<p>Menggunakan include/<i>Use Case</i></p>  	Relasi <i>Use Case</i> tambahan ke sebuah <i>Use Case</i> dimana <i>Use Case</i> yang ditambahkan memerlukan <i>Use Case</i> ini untuk menjalankan fungsinya.

Sumber: Shalahuddin dan Rosa (2013)

2.2.3 Activity Diagram

(Asidhiqi dan Hartanto. 2013 :56) *Activity diagram* adalah teknik untuk mendeskripsikan logika prosedural, proses bisnis dan aliran kerja dalam banyak kasus. *Activity diagram* mempunyai peran seperti halnya *flowchart*, akan tetapi perbedaannya dengan *flowchart* adalah *activity diagram* bisa mendukung perilaku paralel sedangkan *flowchart* tidak bisa.


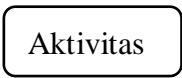
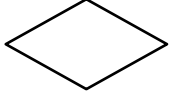


(Rosa dan Shalahudin, 2013 ; 161) Diagram aktivitas atau *activity diagram* menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak. *Activity diagram* digunakan untuk merancang proses bisnis dimana setiap urutan aktivitas yang



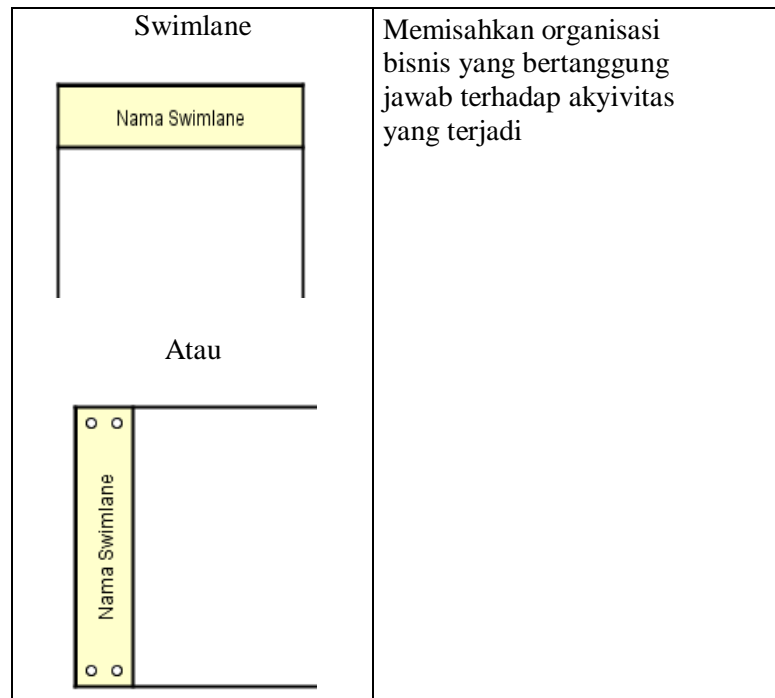
digambarkan merupakan proses bisnis sistem yang didefinisikan serta urutan atau pengelompokan tampilan dari sistem/ *user interface* dimana setiap aktivitas dianggap memiliki sebuah rancangan antarmuka tampilan.

Jadi *Activity Diagram* merupakan alur kerja (*workflow*) atau kegiatan (aktivitas) dari sebuah sistem atau menu yang ada pada perangkat lunak. *Activity Diagram* juga digunakan untuk mendefinisikan urutan atau pengelompokan tampilan dari sistem / *user interface* dimana setiap aktivitas dianggap memiliki sebuah rancangan antar muka tampilan serta rancang menu yang ditampilkan pada perangkat lunak.

Tabel 2.2 Simbol *Activity Diagram*

Simbol	Deskripsi
Status awal 	Status awal aktivitas pada sebuah diagram aktivitas memiliki sebuah status awal.
Aktivitas 	Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja.
Percabangan/ <i>decision</i> 	Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu.
<u>Penggabungan/<i>join</i></u> 	Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu.
Status akhir 	Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir.



Lanjutan tabel 2.2 Simbol Activity Diagram


Sumber: Shalahuddin dan Rosa (2013)

2.2.4 Class Diagram

(Afandi dan Saputra. 2013 :52) Class Diagram adalah suatu diagram yang memperlihatkan atau menampilkan struktur dari sebuah sistem. Class diagram menggambarkan struktur dan deskripsi class, package dan objek beserta hubungan satu sama lain seperti containment, pewarisan, asosiasi, dan lain-lain.

(Asidhiqi dan Hartanto. 2013 : 56) Diagram kelas atau *class diagram* menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut atribut dan metode atau operasi. Atribut merupakan variabel-variabel yang dimiliki oleh suatu kelas. Sedangkan, operasi atau metode adalah fungsi-fungsi yang dimiliki oleh suatu kelas.

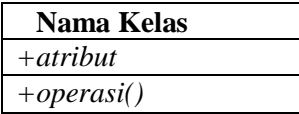


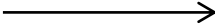
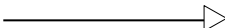
(Rosa dan Shalahudin, 2013 : 141-142) *Class Diagram* atau diagram kelas menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut atribut dan



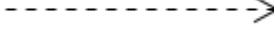
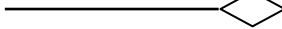
metode atau operasi. Atribut merupakan variabel-variabel yang dimiliki oleh suatu kelas. Sedangkan, operasi atau metode adalah fungsi-fungsi yang dimiliki oleh suatu kelas.

Jadi *Class Diagram* adalah diagram yang menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. kelas memiliki 2 bagian utama yaitu atribut dan metode atau operasi. kelas-kelas yang ada pada struktur sistem harus dapat melakukan fungsi-fungsi sesuai dengan kebutuhan sistem.

Tabel 2.3 Simbol *Class Diagram*

Simbol	Deskripsi
Kelas 	Kelas pada struktur system
Antarmuka/ <i>interface</i>  Nama <i>interface</i>	Sama dengan konsep <i>interface</i> dalam pemrograman berorientasi objek.
Asosiasi/ <i>association</i> 	Relasi antarkelas dengan makna umum, asosiasi biasanya disertai dengan <i>multiplicity</i> .
Asosiasi berarah/ <i>directed association</i> 	Relasi antarkelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan <i>multiplicity</i> .
Generalisasi 	Relasi antarkelas dengan makna generalisasi-spesialisasi (umum khusus).

Lanjutan Tabel 2.3 Simbol *Class Diagram*

Kebergantungan/ <i>dependency</i> 	Relasi antarkelas dengan makna kebergantungan antarkelas.
Agregasi/ <i>aggregation</i> 	Relasi antarkelas dengan makna semua-bagian.

Sumber: Shalahuddin dan Rosa (2013)

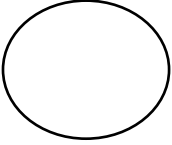
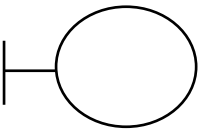
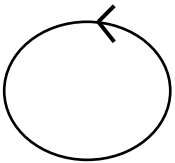
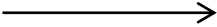
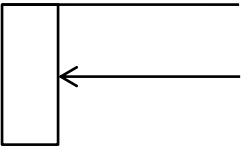
2.2.5 *Sequence Diagram*

(Rosa dan Shalahudin, 2013 : 165) Diagram sekuen atau *sequence diagram* menggambarkan kelakuan objek pada use case dengan mendeskripsikan waktu hidup objek dan *message* yang dikirim dan diterima antar objek. Oleh karena itu untuk menggambarkan diagram sekuen maka harus diketahui objek-objek yang terlibat dalam sebuah *use case* beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek itu.



(Afandi dan Saputra, 2013 : 52) Sequence diagram adalah suatu diagram yang menggambarkan interaksi antar objek dan mengindikasikan komunikasi diantara objek-objek tersebut. Sequence diagram digunakan untuk menunjukkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai respon dari sebuah event untuk menghasilkan output tertentu. Diawali dari apa yang men-trigger aktifitas tersebut, proses dan perubahan apa saja yang terjadi secara internal dan output apa yang akan dihasilkan.

Jadi *Sequence diagram* menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan pesan yang dikirimkan dan diterima antar objek.

Tabel 2.4 Simbol *Sequence Diagram*

Simbol	Deskripsi
	<i>EntityClass</i> , merupakan bagian dari sistem yang berisi kumpulan kelas berupa entitas-entitas yang membentuk gambaran awal sistem dan menjadi landasan untuk menyusun basis data.
	<i>Boundary Class</i> , berisi kumpulan kelas yang menjadi <i>interface</i> atau interaksi antara satu atau lebih aktor dengan sistem, seperti tampilan <i>formentry</i> dan <i>form</i> cetak.
	<i>Control class</i> , suatu objek yang berisi logika aplikasi yang tidak memiliki tanggung jawab kepada entitas, contohnya adalah kalkulasi dan aturan bisnis yang melibatkan berbagai objek.
	<i>Message</i> , simbol mengirim pesan antar <i>class</i> .
	<i>Recursive</i> , menggambarkan pengiriman pesan yang dikirim untuk dirinya sendiri.

Lanjutan Tabel 2.4 Simbol *Sequence Diagram*

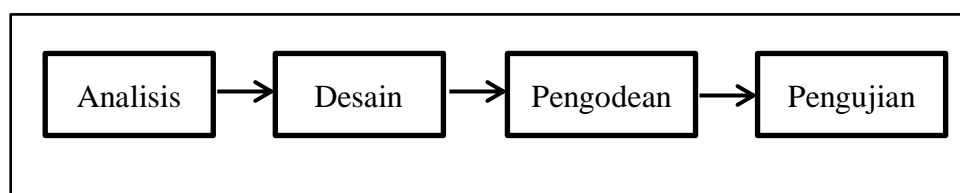
	<p><i>Activation</i>, <i>activation</i> mewakili sebuah eksekusi operasi dari objek, panjang kotak ini berbanding lurus dengan durasi aktivitas sebuah operasi.</p>
	<p><i>Lifeline</i>, garis titik-titik yang terhubung dengan objek, sepanjang <i>lifeline</i> terdapat <i>activation</i>.</p>

Sumber: Shalahuddin dan Rosa (2013)

2.2.6 Metode *Waterfall*

Metode *waterfall* dikerjakan secara bertahap secara satu-persatu mulai dari tahap yang paling atas sampai dengan tahap yang paling bawah. Tahapan yang dilalui meliputi tahap analisis, desain, pengodean, pengujian, dan tahap pendukung (*support*). Metode pengembangan *waterfall* dipilih karena penerapan langkah-langkah dalam *waterfall* sesuai dengan metode pengembangan yang dilakukan penulis.

Langkah-langkah dalam pengembangan yang penulis lakukan berdasarkan *waterfall* digambarkan seperti gambar 3.3 berikut :



Sumber: Shalahuddin dan Rosa (2013)

Gambar 2.2 Waterfall Model



(Rosa dan Shalahudin. 2013 : 29) memecah model waterfall menjadi beberapa tahapan. Berikut adalah penjelasan dari tahap-tahap yang dilakukan di dalam model:

a. Analisis kebutuhan perangkat lunak

Proses pengumpulan kebutuhan dilakukan secara intensif untuk mespesifikasikan kebutuhan perangkat lunak agar dapat dipahami perangkat lunak seperti apa yang dibutuhkan oleh *user*. Spesifikasi kebutuhan perangkat lunak pada tahap ini perlu untuk didokumentasikan.

b. Desain

Desain perangkat lunak adalah proses multi langkah yang fokus pada desain pembuatan program perangkat lunak termasuk struktur data, arsitektur perangkat lunak, representasi antarmuka, dan prosedur pengodean. Tahap ini mentranslasi kebutuhan perangkat lunak dari tahap analisis kebutuhan ke representasi desain agar dapat diimplementasikan menjadi program pada tahap selanjutnya. Desain perangkat lunak yang dihasilkan pada tahap ini juga perlu didokumentasikan.

c. Pembuatan kode program

Desain harus ditranlasikan ke dalam program perangkat lunak. Hasil dari tahap ini adalah program komputer sesuai dengan desain yang dibuat pada tahap desain.

d. Pengujian

Pengujian fokus pada perangkat lunak secara dari segi logik dan fungsional dan memastikan bahwa semua bagian sudah diuji. Hal ini dilakukan untuk meminimalisir kesalahan (*error*) dan memastikan keluaran yang dihasilkan sesuai dengan yang diinginkan.

e. Pendukung (*support*) atau pemeliharaan (*maintenance*)

Tidak menutup kemungkinan sebuah perangkat lunak mengalami perubahan ketika sudah dikiminkan ke *user*. Perubahan busa terjadi karena adanya kesalahan yang muncul dan tidak terdeteksi saat pengujian atau perangkat lunak harus beradaptasi dengan lingkungan baru. Tahap pendukung atau



pemeliharaan dapat mengulangi proses pengembangan mulai dari analisis spesifikasi untuk perubahan perangkat lunak yang sudah ada, tapi tidak untuk membuat perangkat lunak baru.

2.3 Teori Pemrograman

2.3.1 Pengertian Android Studio

Menurut Juansyah (2015 ; 2-3) Android studio adalah IDE (Integrated Development Environment) resmi untuk pengembangan aplikasi Android dan bersifat open source atau gratis. Peluncuran Android Studio ini diumumkan oleh Google pada 16 mei 2013 pada event Google I/O Conference untuk tahun 2013. Sejak saat itu, Android Studio menggantikan Eclipse sebagai IDE resmi untuk mengembangkan aplikasi Android (Sumber: Developer Android Studio) . Android studio sendiri dikembangkan berdasarkan IntelliJ IDEA yang mirip dengan Eclipse disertai dengan ADT plugin (Android Development Tools).

Menurut Rachmawati, Nugraha, dan Awaluddin (2017:49) Android Studio adalah sebuah Integrated Development Environment (IDE) utama Google untuk mengembangkan pada platform Android. Karena Android Studio merupakan IDE dari Google, maka software ini dapat secara langsung terintegrasi dengan Google Maps menggunakan API Key yang dibuat di laman yang disediakan dari Google Maps API untuk mengintegrasikan peta dengan software sehingga peta akan secara otomatis ditampilkan di aplikasi yang dibuat.

Jadi Android Studio Lingkungan Pengembangan Terpadu - Integrated Development Environment (IDE) untuk pengembangan aplikasi Android yang ingin dibuat.

2.3.2 Pengertian PHP (*Hypertext Preprocessor*)

Menurut Winarno dan Zaki (2014 ; 49) PHP adalah sebuah bahasa pemrograman web berbasis server (server-side) yang mampu memarsing kode PHP dari kode web dengan ekstensi .php, sehingga menghasilkan tampilan website yang dinamis di sisi client (browser). Dengan PHP, anda bisa menjadikan



halaman HTML menjadi lebih powerful dan bisa dipakai sebagai aplikasi lengkap, misalnya untuk beragam aplikasi cloud computing.

Menurut Wahana Komputer (2012 ; 76) PHP merupakan bahasa pemrograman berbasis web yang memiliki kemampuan untuk memproses dan mengolah data secara dinamis. PHP dapat dikatakan sebagai sebuah server-side embedded script language, artinya sintak-sintak dan perintah program yang anda tulis akan sepenuhnya dijalankan oleh server tetapi dapat disertakan pada halaman HTML biasa.

Menurut Risnandar, fajar, Nugraha, hafni (2013 ; 57) PHP merupakan bahasa pemrograman yang biasa digunakan untuk membuat halaman HTML. File .php yang dibuat akan diproses didalam server, sedangkan halaman yang akan dikirimkan ke browser pengunjung hanyalah tampilan HTML-nya.

Jadi, PHP (*Hypertext Preprocessor*) merupakan bahasa pemrograman yang dapat gunakan untuk membuat Script yang lebih interaktif. Skrip ini kemudian akan diolah dalam web server yang hasilnya dapat dilihat dalam bentuk HTML. PHP memungkinkan kita membuat situs yang lebih interaktif dan lebih mudah untuk dioperasikan.

2.3.3 Pengertian Database

Menurut Winarno dan Utomo (2010:142) Database atau biasa disebut basis data merupakan kumpulan data yang saling berhubungan. Data tersebut biasanya terdapat dalam table-tabel yang saling berhubungan satu sama lain, dengan menggunakan field/kolom pada tiap tabel yang ada”.

Menurut Risnandar, fajar, Nugraha, hafni (2013: 90-91) Database adalah kumpulan data yang tersimpan dalam tabel-tabel. Tabel-tabel itu disusun berdasarkan baris dan kolom. Tujuan utama pengelolaan data dalam basis data adalah agar kita dapat memperoleh atau meneukan kembali data dengan mudah dan cepat.

Jadi Database adalah sekumpulan data yang sudah disusun sedemikian rupa dengan ketentuan atau aturan tertentu yang saling berelasi sehingga memudahkan



pengguna dalam mengelolanya juga memudahkan memperoleh informasi. Selain itu adapula database sebagai kumpulan file, tabel, atau arsip yang saling terhubung yang disimpan dalam media elektronik.

2.3.4 Pengertian CodeIgniter

Menurut Umar & Anggit (2013: 56) CodeIgniter adalah aplikasi open source yang berupa framework dengan model MVC (Model, View, Controller) untuk membangun website dinamis. Dengan menggunakan PHP CodeIgniter akan memudahkan developer untuk membuat aplikasi web dengan cepat dan mudah dibandingkan dengan membuat dari awal.

Menurut Prabowo (2015: 24) CodeIgniter adalah sebuah framework untuk web yang dibuat dalam format PHP. Format yang dibuat ini selanjutnya dapat digunakan untuk membuat sistem aplikasi web yang kompleks. CodeIgniter dapat mempercepat proses pembuatan web, karena semua class dan modul yang dibutuhkan sudah ada dan programmer hanya tinggal menggunakannya kembali pada aplikasi web yang akan dibuat.

Jadi CodeIgniter adalah sebuah framework php yang bersifat open source dan menggunakan metode MVC (Model, View, Controller). CodeIgniter dibuat dengan tujuan untuk memudahkan developer atau programmer dalam membangun sebuah aplikasi berbasis web tanpa harus membuatnya dari awal.

2.3.5 Pengertian MySql (*My Structured Query Language*)

Menurut Adi Nugroho (2011) MySQL (*My Structured Query Language*) adalah: “ Suatu sistem basis data *relation* atau *Relational Database managemnt System* (RDBMS) yang mampu bekerja secara cepat dan mudah digunakan MySQL juga merupakan program pengakses database yang bersifat jaringan, sehingga sapat digunakan untuk aplikasi *multi user* (banyak pengguna). MySQL didistribusikan gratis dibawah lisensi GPL (*General Public License*). Dimana setiap program bebas menggunakan MySQL namun tidak bisa dijadikan produk turunan yang dijadikan *closed source* atau komersial”.



Menurut Anhar (2010:21) “MySQL (My Structure Query Language) adalah sebuah perangkat lunak sistem manajemen basis data SQL Database Management System atau DBMS dari sekian banyak DBMS seperti Oracle, MS SQL, Postagre SQL dan lainnya”.

Menurut Ramadhani, Anis, Tazkiyatul (2013 ; 480) MySQL adalah sebuah perangkat lunak sistem manajemen basis data SQL (bahasa Inggris: database management system) atau DBMS yang multithread, multi-user, dengan sekitar 6 juta instalasi di seluruh dunia. MySQL AB membuat MySQL tersedia sebagai perangkat lunak gratis dibawah lisensi GNU General Public License (GPL), tetapi mereka juga menjual dibawah lisensi komersial untuk kasus-kasus dimana penggunaannya tidak cocok dengan penggunaan GPL. Relational Database Management System (RDBMS).

Jadi MySql (*My Structured Query Language*) adalah perangkat lunak sebuah program database server yang mampu menerima dan mengirimkan datanya dengan sangat cepat, multi user serta menggunakan perintah standar SQL (*Structured Query Language*).

2.3.6 Pengertian XAMPP

Menurut Sulihati dan Andriyani (2016 ; 21) XAMPP adalah perangkat lunak bebas, yang mendukung banyak sistem operasi, merupakan kompilasi dari beberapa program. Fungsinya adalah sebagai server yang berdiri sendiri (localhost), yang terdiri atas program Apache HTTP Server, MySQL database, dan penerjemah bahasa yang ditulis dengan bahasa pemrograman PHP dan Perl.

Menurut Winarno dan Zaki (2014 ; 1-2) XAMPP adalah software web server yang bisa dipakai untuk mengakomodasi sistem operasi yang Anda pakai (X), Apache (A), MySql (M), PHP (P), dan Perl (P). XAMPP ditujukan untuk pekerjaan pengembangan program lokal saja, dan tidak disarankan untuk tahap produksi, alias dijalankan di internet.



Menurut Wahana(2009:30) XAMPP adalah salah satu paket instalasi apache, PHP, dan MySQL secara instant yang dapat digunakan untuk membantu proses instalasi ketiga produk tersebut”.

Jadi XAMPP adalah sebuah software untuk menghubungkan sebuah data base ke dalam sebuah server kita sendiri (localhost).

2.3.7 Pengertian *Google Maps API*

Menurut Harison dan Syarif (2016 ; 42) *Google Maps Application Programming Interface (API)* merupakan suatu fitur aplikasi yang dikeluarkan oleh *google* untuk memfasilitasi pengguna yang ingin mengintegrasikan *Google Maps* ke dalam *website* masing-masing dengan menampilkan data point milik sendiri. Dengan menggunakan *Google Maps API*, *Google Maps* dapat di-embed pada web site eksternal. Agar aplikasi *Google Maps* dapat muncul di website tertentu, diperlukan adanya *API key*. *API key* merupakan kode unik yang di generasikan oleh *google* untuk suatu website tertentu, agar server *Google Maps* dapat mengenali.

Menurut Mahdia dan Noviyanto (2013 ; 164) *Google Maps API* adalah sebuah layanan (service) yang diberikan oleh *Google* kepada para pengguna untuk memanfaatkan *Google Map* dalam mengembangkan aplikasi. *Google Maps API* menyediakan beberapa fitur untuk memanipulasi peta, dan menambah konten melalui berbagai jenis services yang dimiliki, serta mengizinkan kepada pengguna untuk membangun aplikasi enterprise di dalam websitenya.

Menurut Minarni, dan Febri (2015 ; 33) *Google maps API* adalah fungsi fungsi pemrograman yang disediakan oleh *Google maps* agar *Google maps* bisa di integrasikan kedalam Web atau aplikasi. *Google Maps API* merupakan aplikasi interface yang dapat diakses lewat javascript agar *Google Map* dapat ditampilkan pada sebuah halaman web. Pada *Google Maps JavaScript API* versi 2, untuk dapat mengakses *Google Map* pada sebuah halaman web diperlukan *Google API Key*. *API Key* adalah sederetan kode sebagai izin untuk menampilkan *Google Map* pada sebuah halaman web. Namun untuk versi 3 tidak memerlukan *API Key*,



tetapi pihak *google* menganjurkan menggunakan *API Key* untuk mempermudah mengontrol *Google Maps API*. *API Key* bersifat tunggal, hanya berlaku untuk sebuah URL. Salah satu syarat untuk mendapatkan *API Key* adalah mempunyai akun *google/Gmail* untuk *generate* domain atau URL web pada link <https://code.google.com/apis/console/> . *API Key* bersifat gratis sampai batas 25.000 pengunjung per hari. Jika melebihi 25.000 pengunjung per hari maka diperlukan membeli kuota tambahan.

Jadi *Google Maps API* merupakan fungsi fungsi pemrograman yang disediakan oleh Google maps agar Google maps bisa di integrasikan kedalam Web atau aplikasi yang sedang kita buat.

2.4 Referensi Jurnal

Menurut Eriq, Budi, Tanzil (2017) menjelaskan aplikasi perancangan wisata di Malang Raya dengan algoritma *Greedy* untuk menghemat waktu perjalanan parawisata dengan memilih jarak dan waktu terpendek dalam perjalanan wisata. Bahasa Pemrograman yang digunakan adalah matriks dan algoritma *greedy*, aplikasi ini nantinya memungkinkan wisatawan dalam mentukan perjalanan wisata sehingga proses perjalanan wisata nantinya dapat lebih efektif dan teratur.

Menurut Wilson, Sibaroni, Ummah (2015) menjelaskan analisis ini untuk mengimplementasikan *Brute Force* dengan teknik *Exhaustive Search* pada GPU, dan menganalisis jumlah *threadProcess* pada setiap *thread* dan pengaruh *block* dan *thread* pada GPU. Bahasa Pemrograman yang digunakan adalah *General-Purpose Parallel Computing* dan *Computer Unified Device Architecture (CUDA)*, analisis ini mencari algoritma baru untuk menyelesaikan sebuah permasalahan yang menghasilkan solusi yang lebih optimal dari penelitian sebelumnya.

Menurut Abdullah, Hardi (2017) menjelaskan sistem ini untuk memilih dan menentukan rute-rute terpendek yang mendekati optimal untuk antaran paket dari kota asal ke kota tujuan kemudian kembali ke kota asal. Menggunakan bahasa program *PHP* dan *MySQL*. Dari hasil penelitian ini diharapkan dapat



mempermudah dalam pengantaran paket dan melacak keberadaan paket tersebut dengan akurat.

Menurut Adi (2015) menjelaskan aplikasi yang dibuat untuk membantu petugas marketing support pada divisi *marketing support* PT. Yamaha Mataram Sakti untuk menentukan rute terpendek dalam mengunjungi calon konsumen sehingga dapat mempersingkat jarak yang ditempuh dan menghemat waktu. Sistem ini dibangun menggunakan bahasa pemrograman *PHP* dan *MySQL*. Dari hasil penelitian ini diharapkan petugas marketing support pada divisi *marketing support* dapat mengunjungi konsumen dengan mudah dan lebih efisien dalam segi jarak dan waktu.

Menurut Widia, Sundawa, Azhari (2016) menjelaskan *Search Engine* ini akan menampilkan sebuah hasil pencarian berdasarkan sebuah kata atau beberapa kata yang dicari dan menentukan urutan-urutan teratas berdasarkan kata yang dicari terbanyak. Bahasa pemrograman yang di pakai adalah *MySQL*. Maka Pengimplementasian algoritma Brute Force ini lewat *search engine* dapat memperoleh informasi dan kandungan zat-zat yang terdapat didalam obat tersebut dengan *up-to-date* dan membantu kita dalam menghemat waktu.