BAB 2
LANDASAN TEORI

2.1. Landasan Teori

Landasan terori bertujuan memberikan gambaran dari teori yang terkait dalama pembangunan aplikasi. Landasan teori yang dibahas yaiitu pengertian Metode Simple Additive Weighting, Android, GPS, *Geotagging*, *Java, Web Service, PHP, MySQL*, Metode yang digunakan dan bahasa pemrograman yang digunakan dalam pembangunan aplikasi.

2.1.1. Multi-Atribut Decision Making (MADM)

Pada dasarnya, proses MADM dilakukan melalui 3 tahap, yaitu penyusunan komponen-komponen situasi, analisis, dan sintesis informasi (Rudolphi, 2000). Pada tahap penyusunan komponen, komponen situasi, akan dibentuk tabel taksiran yang berisi identifikasi alternatif dan spesifikasi tujuan, kriteria dan atribut. Salah satu cara untuk menspesifikasikan tujuan situasi $|O_i, i=1,\ldots,t|$ adalah dengan cara mendaftar konsekuensi-konsekuensi yang mungkin dari alternatif yang telah teridentifikasi $|A_i, i=1,\ldots,n|$. Selain itu juga disusun atribut-atribut yang akan digunakan $|a_k, k=1,\ldots,m|$ [1].

Tahap analisis dilakukan melalui 2 langkah. Pertama, mendatangkan taksiran dari besaran yang potensial, kemungkinan, dan ketidakpastian yang berhubungan dengan dampak-dampak yang mungkin pada setiap alternatif. Kedua, meliputi pemilihan dari preferensi pengambil keputusan untuk setiap nilai, dan ketidakpedulian terhadap resiko yang timbul. Pada langkah pertama, beberapa metode menggunakan fungsi distribusi $|p_j(x)|$ yang menyatakan probabilitas kumpulan atribut $|a_k|$ terhadap setiap alternatif $|A_i|$. Konsekuen juga dapat ditentukan secara langsung dari agregasi sederhana yang dilakukan pada informasi terbaik yang tersedia. Demikian pula, ada beberapa cara untuk menentukan preferensi mengambil keputusan pada setiap konsekuen yang dapat dilakukan pada

langkah kedua. Metode yag paling sederhana adalah untuk menurunkan bobot atribut dan kriteria adalah dengan utilitas atau penjumlahan terbobot [1].

Secara umum, model multi-attribute decision making dapat didefinisikan sebagai berikut (Zimermann, 1991): Misalkan A = $\{a_i|\ i=1,\ldots,n\}$ adalah himpunan alternatif-alternatif keputusan dan C = $\{c_i|j=1,...,m\}$ adalah himpunan tujuan yang diharapkan, maka akan ditentukan alternatif x^0 yang memiliki derajat harapan tertinggi terhadap tujuan-tujuan yang relevan c_j [1].

Sebagian besar pendekatan MADM dilakukan melalui 2 langkah, yaitu: pertama, melakukan agregasi terhadap keputusan-keputusan yang tanggal terhadap semua tujuan pada setiap alternatif; kedua, melakukan perangkingan alternatif-alternatif keputtusan tersebut berdasarkan hasil agregasi keputusan [1].

Dengan demikian, bisa dikatakan bahwa, masalah *multi-attribute decision making* (MADM) adalah mengevaluasi m alternatif A_i (i=1,2,…,m) terhadap sekumpulan atribut atau kriteria C_j (j=1,2,…,n), dimana setiap atribut saling tidak bergantung satu dengan yang lainnya. Matriks keputusan setiap alternatif terhadapat setiap atribut X, diberikan sebagai [1]:

$$x = \begin{bmatrix} X_{11} & X_{12} & \cdots & X_{1n} \\ X_{21} & X_{22} & \cdots & X_{2n} \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ X_{m1} & X_{m2} & \cdots & X_{mn} \end{bmatrix}$$

Dimana X_{ij} merupakan rating kinerja alternatif ke-I terhadap atribut ke-j.

Dimana i adalah elemen baris dan j adalah elemen kolom.

Nilai bobot yang menunjukan tingkat kepentingan relatif setiap atribut, diberikan sebagi, w:

$w = \{w_1, w_2, \ldots, w_n\}$

Dimana w adalah Nilai bobot, dan $w_1, w_2, \ldots, w_n$ adalah bobot setiap kriteria, nilai ini dapat di isi secara subjektif maupun objektif secara matematis.

Rating kinerja (X), dan nilai bobot (W) merupakan nilai utama yang merepresentasikan preferensi absolut dari pengambil keputusan. Masalah MADM diakhiri dengan proses perangkingan untuk mendapatkan alternatif terbaik yang diperoleh berdasarkan nilai keseluruhan preferensi yang diberikan (Yeh, 2002) [1].

Ada beberapa metode yang dapat digunakan untuk menyelesaikan masalah MADM, antara lain [1]:

a. Simple Additive Weighting Method (SAW)
b. Weighted Product (WP)
c. ELECTRE
d. Technique for Order Preference by Similarity to Ideal Solution (TOPSIS)
e. Analytic Hierarchy Process(AHP)

Simple Additive Weighting Method (SAW)

Metode SAW sering juga dikenal istilah metode penjumlahan terbobot. Konsep dasar metode SAW adalah mencari penjumlahan terbobot dari rating kinerja pada setiap alternatif pada semua atribut (Fishburn, 1967)(MacCrimmon, 1968). Metode SAW membutuhkan proses normalisasi matriks keputusan (X) ke suatu skala yang dapat diperbandingkan dengan semua rating alternatif yang ada[1].

$$r_{ij}\begin{cases}\dfrac{x_{ij}}{\underset{i}{Max}\ x_{ij}} & jika\ j\ adalah\ atribut\ kruntungan\ (benefit)\\ \dfrac{\underset{i}{Min}\ x_{ij}}{x_{ij}} & jika\ j\ adalah\ atribut\ biaya\ (cost)\end{cases}$$

Dimana

r_{ij} = rating kinerja ternormalisasi dari alternatif A_i pada atribut C_j

A = alternatif

C = kriteria

i = data ke 1,2,…,m

j = data ke 1,2,…,n

m = banyaknya data

n = banyaknya data

Nilai preferensi untuk setiap alternatif (V_i) diberikan sebagai :

$$V_i = \sum_{j=1}^{n} w_j r_{ij}$$

Dimana

V = nilai preferensi untuk setiap alternatif

w = bobot preferensi

r = matriks ternormalisasi r

i = data ke 1,2,…,m

j = data ke 1,2,…,n

m = banyaknya data

n = banyaknya data

Nilai V_i yang lebih besar mengindikasikan bahwa alternatif A_i lebih terpilih

Contoh Kasus:

Suatu perusahaan di Daerah Istimewa Yogyakarta (DIY) ingin membangun sebuah gudang yang akan digunakan sebagai tempat untuk menyimpan sementara hasil produksinya. Ada 3 lokasi yang akan menjadi alternatif, yaitu A_1=Ngemplak, A_2=Kalsan, A_3=Kota Gedhe. Ada 5 kriteria yang dijadikan acuan dalam pengambilan keputusan, yaitu:

a. C_1= Jarak dengan pasar terdekat (km),
b. C_2= Kepadatan penduduk di sekitar lokasi (orang/km^2),
c. C_3= Jarak dari pabrik (km),
d. C_4= Jarak dengan gudang yang sudah ada (km),
e. C_5= Harga tanah untuk lokasi (x1000 Rp/m^2)

Rating kecocokan setiap alternatif pada setiap kriteria, dinilai dengan 1 sampai 5, yaitu:

a. 1= Sangat Buruk,
b. 2= Buruk,
c. 3= Cukup,
d. 4= Baik.

e. 5= Sangat Baik

Tabel 2.1 menunjukan rating kecocokan dari setiap alternatif pada setiap kriteria. Sedangkan tingkat kepentingan setiap kriteria juga dinilai dengan 1 sampai 5, yaitu:

a. 1= Sangat Rendah,
b. 2= Rendah,
c. 3= Cukup,
d. 4= Tinggi,
e. 5= Sangat Tinggi

Tabel 2.1 Rating Kecocokan Dari Setiap Alternatif Pada Setiap Kriteria

Alternatif	Kriteria				
	C_1	C_2	C_3	C_4	C_5
A_1	0.75	2000	18	50	500
A_2	0.50	1500	20	40	450
A_3	0.90	2050	35	35	800

Kriteria C_2 (kepadatan penduduk di sekitasr lokasi) dan C_4 (jarak dengan gudang yang sudah ada) adalah kriteria keuntungan, sedangkan kriteria C_1 (jarak dengan pasar terdekat), C_3 (jarak dari pabrik), dan C_5 (harga tanah untuk lokasi adalah kriteria biaya.

Pengambil keputusan memberikan bobot preferensi sebagai :

w = (5,3,4,4,2)

Pertama-tama dilakukan normalisasi matriks X

$$r_{11} = \frac{\min\{0.75;0.5;0.90\}}{0.75} = \frac{0.5}{0.75} = 0.6667$$

$$r_{12} = \frac{2000}{\max\{2000;1500;2050\}} = \frac{2000}{2050} = 0.9756$$

$$r_{13} = \frac{\min\{18;20;35\}}{18} = \frac{18}{18} = 1$$

$$r_{14} = \frac{50}{\max\{50;40;35\}} = \frac{50}{50} = 1$$

$r_{15} = \frac{\min\{500;450;800\}}{500} = \frac{450}{500} = 0.9$

Dan seterusnya, sehingga diperoleh matriks ternormalisasi r sebagai berikut:

$$r = \begin{bmatrix} 0.6667 & 0.9756 & 1.0000 & 1.0000 & 0.9000 \\ 1.0000 & 0.7317 & 0.9000 & 0.8000 & 1.0000 \\ 0.5556 & 1.0000 & 0.5143 & 0.7000 & 0.5625 \end{bmatrix}$$

Proses perangkingan

V_1 = (5*0.6667)+(3*0.9756)+(4*1)+(4*1)+(2*0.9) = 6.0602;

V_2 = (5*1)+(3*0.7317)+(4*0.9)+(4*0.8)+(2*1) = 15.9951;

V_3 = (5*0.5556)+(3*1)+(4*0.5143)+(4*0.7)+(2*0.5625) = 11.75999.

Nilai terbesar V_2 sehingga alternatif pertama adalah yang terbaik. Dengan kata lain, Ngemplak akan terpilih sebagai lokasi untuk mendirikan geudng baru.

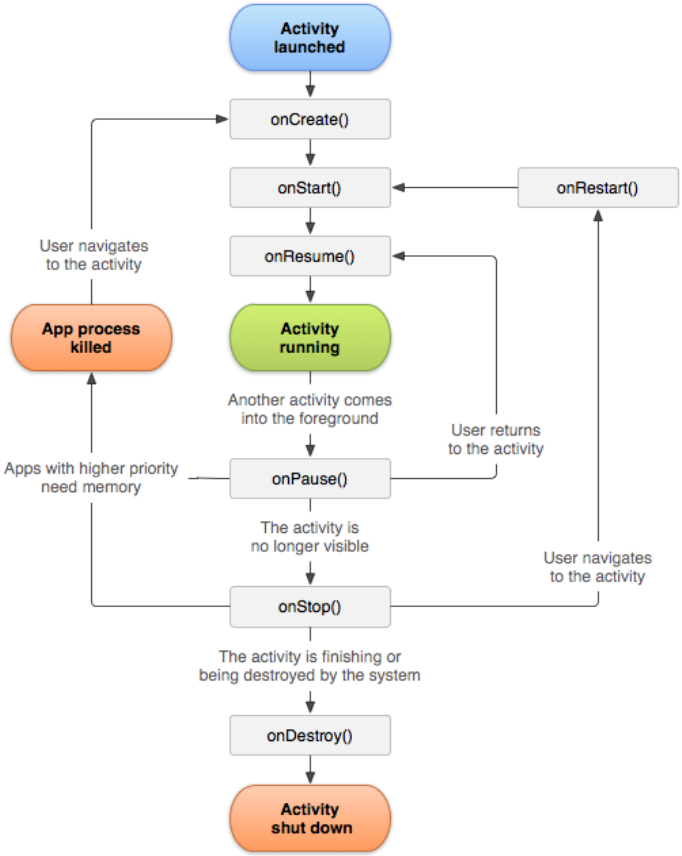
2.1.2. Android

Android adalah sebuah sistem operasi untuk perangkat mobile berbasis linux yang mencakup sistem operasi, *middleware*, dan aplikasi. Android adalah sistem operasi untuk telepon seluler yang berbasis Linux. Android menyediakan *platform* terbuka bagi para pengembang untuk membuat aplikasi mereka sendiri. Pada awalnya dikembangkan oleh *Android Inc*, sebuah perusahaan pendatang baru yang membuat perangkat lunak untuk ponsel yang kemudian dibeli oleh *Google Inc*. Untuk pengembangannya, dibentuklah Open Handset Alliance (OHA), konsorsium dari 34 perusahaan perangkat keras, perangkat lunak, dan telekomunikasi termasuk Google, HTC, Intel, Motorola, Qualcomm, T-Mobile, dan Nvidia [2].

2.1.2.1. Android Life Cycle

Android memiliki paradigma pemrograman lain tidak seperti paradigma pemrograman biasa di mana aplikasi yang dijalankan pada fungsi main(), sistem android menjalankan kode dalam method Activity dengan menerapkan metode callback tertentu yang sesuai dengan tahap tertentu dari siklus hidup. Setiap aplikasi yang berjalan dalam sistem operasi android memiliki siklus hidup yang berbeda dengan aplikasi desktop atau web. Hal ini dikarenakan aplikasi mobile memiliki

tingkat interupsi proses yang lumayan tinggi seperti ketika handling panggilan masuk aplikasi diharuskan menghentikan proses sementara. Penerapan siklus hidup juga berguna untuk memastikan aplikasi tidak menghabiskan sumber daya baterai pengguna [3].



Sumber : https://developer.android.com/guide/components/images/activity_lifecycle.png

Gambar 2.1 Android Life Cycle

Terdapat beberapa state dalam siklus hidup android yang terjadi seperti diilustrasikan pada Gambar 2.1 Siklus Hidup Android, akan tetapi hanya beberapa dari state tersebut yang menjadi statis diantaranya[3]:

1. Resumed

Resumed terjadi ketika aplikasi berjalan setelah state *paused* . State ini akan menjalankan perintah program yang ditulis pada method onResume() [3].

2. Paused

Dalam keadaan ini aktivitas yang terjadi dihentikan secara sementara tetapi masih terlihat oleh pengguna karena terdapat proses yang memiliki prioritas lebih tinggi seperti panggilan telepon. Aplikasi tidak dapat menjalankan perintah apapun ataupun menampilkan apapun dalam state ini [3].

3. Stopped

Dalam keadaan ini, aplikasi benar-benar tidak ditampilkan dan tidak terlihat oleh pengguna tetapi masih meninggalkan service dibackground [3].

State lain seperti Created dan Started bersifat sementara dan sistem dengan cepat menjalankan state berikutnya dengan memanggil metode *life cycle callback* berikutnya. Artinya, setelah sistem OnCreate() dipanggil, dengan cepat sistem akan memanggil method OnStart(), krmudian diikuti oleh onResume().

2.1.3. Flowchart

Flowchart adalah penggambaran secara grafik dari langkah-langkah dan urut-urutan prosedur dari suatu program. Flowchart menolong analis dan programmer untuk memecahkan masalah kedalam segmen-segmen yang lebih kecil dan menolong dalam menganalisis alternatif-alternatif lain dalam pengoperasian. Flowchart biasanya mempermudah penyelesaian suatu masalah khususnya masalah yang perlu dipelajari dan dievaluasi lebih lanjut [8].

Bila seorang analis dan programmer akan membuat flowchart, ada beberapa petunjuk yang harus diperhatikan, seperti [8]:

1. Flowchart digambarkan dari halaman atas ke bawah dan dari kiri ke kanan.
2. Aktivitas yang digambarkan harus didefinisikan secara hati-hati dan definisi ini harus dapat dimengerti oleh pembacanya.
3. Kapan aktivitas dimulai dan berakhir harus ditentukan secara jelas.
4. Setiap langkah dari aktivitas harus diuraikan dengan menggunakan deskripsi kata kerja..
5. Setiap langkah dari aktivitas harus berada pada urutan yang benar.
6. Lingkup dan range dari aktifitas yang sedang digambarkan harus ditelusuri dengan hati-hati. Percabangan-percabangan yang memotong aktivitas yang sedang digambarkan tidak perlu digambarkan pada flowchart yang sama.

Simbol konektor harus digunakan dan percabangannya diletakan pada halaman yang terpisah atau hilangkan seluruhnya bila percabangannya tidak berkaitan dengan sistem.

7. Gunakan simbol-simbol flowchart yang standar.

2.1.4. Global Positioning System (GPS)

Global Positioning System adalah sistem navigasi berbasis satelit terdiri dari jaringan 24 satelit ditempatkan ke orbit oleh Departemen Pertahanan Amerika Serikat yang pertama kali diperkenalkan mulai tahun 1978. Layanan GPS dahulu hanya dipergunakan untuk keperluan militer namun mulai terbuka untuk publik. 24 satelit GPS tersebut berada sekitar 12.000 mil di atas bumi bergerak mengelilingi bumi 12 jam dengan kecepatan 7.000 mil per jam. Satelit GPS berkekuatan energi sinar matahari, memiliki baterai cadangan untuk menjaga agar tetap berjalan pada saat gerhana matahari atau pada saat tidak ada energi matahari dan memiliki roket penguat kecil pada masing-masing satelit agar dapat mengorbit tepat pada tempatnya [4].

GPS adalah singkatan dari Global Positioning System, yang merupakan sistem navigasi dengan menggunakan teknologi satelit yang dapat menerima sinyal dari satelit. Sistem ini menggunakan 24 satelit yang mengirimkan sinyal gelombang mikro ke bumi. Sinyal ini diterima oleh alat penerima (receiver) di permukaan, dimana GPS receiver ini akan mengumpulkan informasi dari satelit GPS, seperti:[4]

a. Waktu. GPS receiver menerima informasi waktu dari jam atom yang mempunyai keakurasian sangat tinggi.
b. Lokasi. GPS memberikan informasi lokasi dalam tiga dimensi:
    1. Latitude
    2. Longitude
    3. Elevasi
c. Kecepatan. Ketika berpindah tempat, GPS dapat menunjukkan informasi kecepatan berpindah tersebut.
d. Arah perjalanan. GPS dapat menunjukkan arah tujuan.

e. Simpan lokasi. Tempat-tempat yang sudah pernah atau ingin dikunjungi bisa disimpan oleh GPS receiver.
f. Komulasi data. GPS receiver dapat menyimpan informasi track, seperti total perjalanan yang sudah pernah dilakukan, kecepatan rata-rata, kecepatan paling tinggi, kecepatan paling rendah, waktu/jam sampai tujuan, dan sebagainya. [4]

2.1.5. JavaScript Object Notation (JSON)

JSON (*JavaScript Object Notation*) adalah format pertukaran data yang ringan, mudah dibaca dan ditulis oleh manusia, serta mudah diterjemahkan dan dibuat (*generate*) oleh komputer. Format ini dibuat berdasarkan bagian dari Bahasa Pemprograman JavaScript, Standar ECMA-262 Edisi ke-3 - Desember 1999. JSON merupakan format teks yang tidak bergantung pada bahasa pemprograman apapun karena menggunakan gaya bahasa yang umum digunakan oleh programmer keluarga C termasuk C, C++, C#, Java, JavaScript, Perl, Python dll. Oleh karena sifat-sifat tersebut, menjadikan JSON ideal sebagai bahasa pertukaran-data [5].
JSON terbuat dari dua struktur [5]:

1. Kumpulan pasangan nama/nilai. Pada beberapa bahasa, hal ini dinyatakan sebagai objek (*object*), rekaman (*record*), struktur (*struct*), kamus (*dictionary*), tabel hash (*hash table*), daftar berkunci (*keyed list*), atau associative array.

2. Daftar nilai terurutkan (*an ordered list of values*). Pada kebanyakan bahasa, hal ini dinyatakan sebagai larik (*array*), vektor (*vector*), daftar (*list*), atau urutan (*sequence*).

Struktur-struktur data ini disebut sebagai struktur data universal. Pada dasarnya, semua bahasa pemprograman moderen mendukung struktur data ini dalam bentuk yang sama maupun berlainan. Hal ini pantas disebut demikian karena format data mudah dipertukarkan dengan bahasa-bahasa pemprograman yang juga berdasarkan pada struktur data ini [5].

2.1.6. Java

Java menurut definisi dari Sun adalah nama untuk sekumpulan teknologi untuk membuat dan menjalankan perangkat lunak pada komputer standalone ataupun pada lingkungan jaringan. Java dikembangkan pada bulan Agustus 1991, dengan nama semula Oak. Pada Januari 1995, karena nama Oak dianggap kurang komersial, maka diganti menjadi Java. Pada Desember 1998, Sun memperkenalkan nama "Java 2" (J2) sebagai generasi kedua dari java platform. Konvensi nama baru ini diterapkan untuk semua edisi Java yaitu Standard Edition (J2SE), Enterprise Edition (J2EE), dan Micro Edition (J2ME) [6].

Ada tiga platform Java yang masing-masing diarahkan untuk tujuan tertentu dan untuk lingkungan komputasi yang berbeda-beda [6]:

1. Standard Edition (J2SE) : J2SE merupakan inti dari bahasa pemrograman Java. J2SE didesain untuk jalan pada komputer desktop dan komputer workstations.
2. Enterprise Edition (J2EE): Dengan built-in mendukung untuk servlets. JSP, dan XML, edisi ini ditujukan untuk aplikasi berbasis server.
3. Micro Edition (J2ME) : Didesain untuk piranti dengan memori terbatas, layar display terbatas dan power pemrosesan yang juga terbatas.

2.1.7. Web services

Web service adalah suatu sistem perangkat lunak yang dirancang untuk mendukung interoperabilitas dan interaksi antar sistem pada suatu jaringan. Web service digunakan sebagai suatu fasilitas yang disediakan oleh suatu web site untuk menyediakan layanan (dalam bentuk informasi) kepada sistem lain, sehingga sistem lain dapat berinteraksi dengan sistem tersebut melalui layanan (service) yang disediakan oleh suatu sistem yang menyediakan web service. Web service menyimpan data informasi dalam format XML, sehingga data ini dapat diakses oleh

sistem lain walaupun berbeda platform, sistem operasi, maupun bahasa compiler [11].

Web service bertujuan untuk meningkatkan kolaborasi antar pemrogram dan perusahaan, yang memungkinkan sebuah fungsi di dalam Web Service dapat dipinjam oleh aplikasi lain tanpa perlu mengetahui detil pemrograman yang terdapat di dalamnya [11].

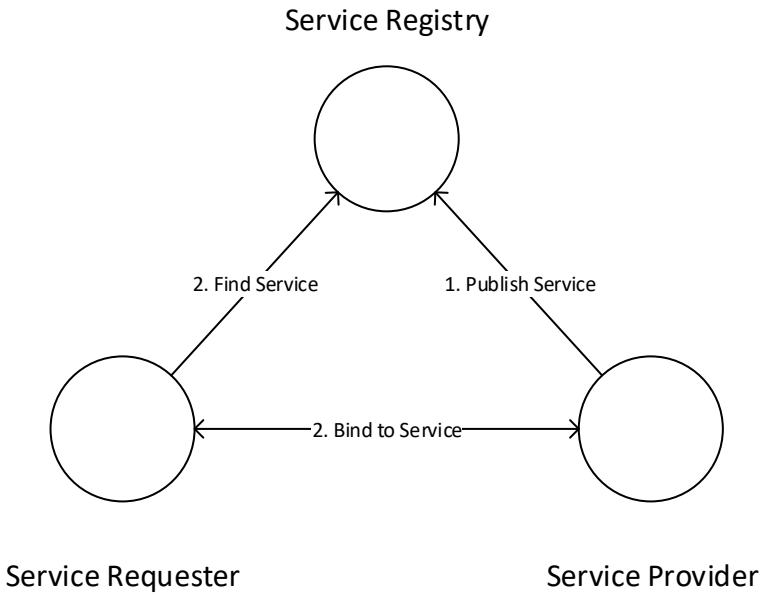
Beberapa alasan mengapa digunakannya web service adalah sebagai berikut:[11]

1. Web service dapat digunakan untuk mentransformasikan satu atau beberapa bisnis logic atau class dan objek yang terpisah dalam satu ruang lingkup yang menjadi satu, sehingga tingkat keamanan dapat ditangani dengan baik [11].
2. Web service memiliki kemudahan dalam proses deployment-nya, karena tidak memerlukan registrasi khusus ke dalam suatu sistem operasi. Web service cukup di-upload ke web server dan siap diakses oleh pihak-pihak yang telah diberikan otorisasi [11].
3. Web service berjalan di port 80 yang merupakan protokol standar HTTP, dengan demikian web service tidak memerlukan konfigurasi khusus di sisi firewall [11].

2.1.7.1. Arsitektur Web Service

Web service memiliki tiga entitas dalam arsitekturnya, yaitu: [11]

1. Service Requester (peminta layanan)
2. Service Provider (penyedia layanan)
3. Service Registry (daftar layanan)



Gambar 2.2 Arsitektur Web Service

a. Service Provider berfungsi untuk menyediakan layanan/service dan mengolah sebuah registry agar layanan-layanan tersebut dapat tersedia [11].
b. Service Registry berfungsi sebagai lokasi central yang mendeskripsikan semua layanan/service yang telah di-register [11].
c. Service Requestor meminta layanan yang mencari dan menemukan layanan yang dibutuhkan serta menggunakan layanan tersebut [11].

2.1.8. Processor Hypertext Protocol (PHP)

PHP merupakan salah satu pemograman server-side di antara beberapa pemograman yang ada. Sejak di luncurkan , PHP mendapat respon yang sangat baik dari kalangan pengembang aplikasi web kemudahannya untuk di pahami, serta sintaksnya yang mirip dengan Bahasa C menjadikan pemograman ini cepat di kenal di kalangan luas [7].

Tanpa mengurangi kemampuan-kemampuan yang sudah ada, PHP 5 hadir dengan fitur lebih kompleks , dan merupakan rilis terbaru yang di keluarkan oleh pengembangnya secara besar besaran pengembang PHP merombak bug yang sering di jumpai daam PHP 4, hal ini akan sangat jelas terasa ketika anda membuat aplikasi berbasis objek [7].

Dengan tujuan untuk lebih meningkatkan kinerja aplikasi, tanpa bermaksud membuang fitur yang ada , PHP 5 menambahkan fitur – fitur baru seperti dukungan terhadap MYSQL. Dukungan ini di wujudkan dalam bentuk fungsi- fungsi lama dan di tulis ulang [7].

2.1.9. HTML

Hyper Text Markup Language (HTML) adalah sebuah bahasa markah yang digunakan untuk membuat sebuah halaman web. Weibsite yang dibuat dengan HTML ini, dapat di lihat oleh semua orang yang terkoneksi dengan internet . Tentunya dengan menggunakan aplikasi penjelajah atau browser [12].

Tiap kali kita mengakses dokumen web, maka sesungguhnya kita mengakses dokumen seseorang yang ditulis dengan menggunakan format HTML. Beberapa orang merasa keberatan jika dikatakan HTML adalah sebuah bahasa pemrograman karena struktur yang dimilikinya dianggap terlalu sederhana, kode-kode dibaca oleh browser baris per baris, dari atas ke bawah. HTML juga tidak memiliki ‘looping’ seperti bahasa pemrograman lain [12].

Pada HTML dipergunakan hypertext link atau hubungan antara teks dan dokumen lain. Dengan demikian pembaca dokumen bisa melompat dari satu dokumen ke dokumen yang lain dengan mudah [12].

2.1.10. MySQL

MySQL adalah salah satu jenis database server yang sangat terkenal dan banyak digunakan untuk membangun aplikasi web yang menggunakan database sebagai sumber dan pengolahan datanya. MySQL dikembangkan oleh perusahaan swedia bernama MySQL AB yang pada saat ini bernama Tcx DataKonsult AB sekitar tahun 1994-1995, namun cikal bakal kodenya sudah ada sejak tahun 1979. Awalnya Tcx merupakan perusahaan pengembang software dan konsultan database, dan saat ini MySQL sudah diambil alih oleh Oracle Corp [9].

Kepopuleran MySQL antara lain karena MySQL menggunakan SQL sebagai bahasa dasar untuk mengakses databasenya sehingga mudah untuk digunakan, kinerja query cepat, dan mencukupi untuk kebutuhan database perusahaan-perusahaan yang berskala kecil sampai menengah, MySQL juga

bersifat open source (tidak berbayar). MySQL merupakan database yang pertama kali didukung oleh bahasa pemrograman script untuk internet (PHP dan Perl). MySQL dan PHP dianggap sebagai pasangan software pembangun aplikasi web yang ideal. MySQL lebih sering digunakan untuk membangun aplikasi berbasis web, umumnya pengembangan aplikasinya menggunakan bahasa pemrograman script PHP [9].

2.1.11. Unified Modelling Language (UML)

Unified Modeling Language (UML) adalah keluarga notasi grafis yang didukung oleh model-model tunggal, yang membantu pendeskripsian dan desain sistem perangkat lunak, khususnya sistem yang dibangun menggunakan pemrograman berorientasi objek (OO). UML merupakan standar yang relatif terbuka yang dikontrol oleh Object Management Group (OMG) [10].

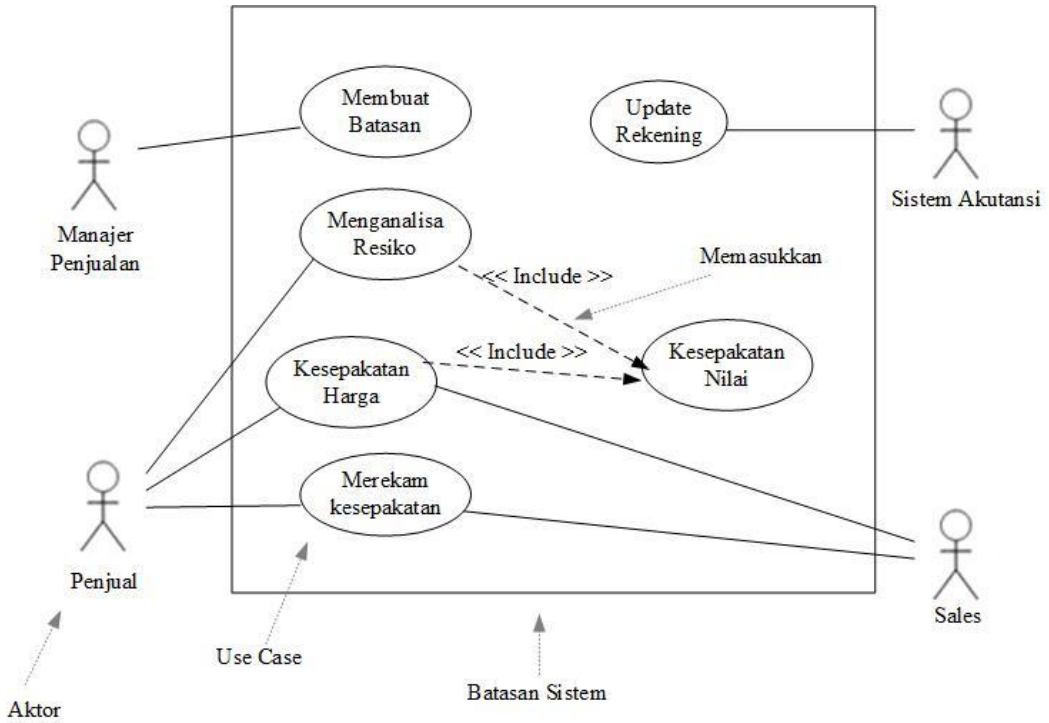
2.1.11.1. Use Case Diagram

Use case adalah teknik untuk merekam persyaratan fungsional sebuah sistem. *Use case* mendeskripsikan interaksi tipikal antara para pengguna sistem dengan sistem itu sendiri, dengan memberi sebuah narasi tentang bagaimana sistem tersebut digunakan. Sebuah *use case* adalah serangkaian skenario yang dikemas menjadi satu oleh tujuan pengguna umum. Dalam bahasa *use case*, para pengguna disebut sebagai aktor. Aktor merupakan sebuah peran yang dimainkan oleh seorang pengguna dalam kaitannya dengan sistem. aktor tidak harus manusia. Jika sebuah sistem melakukan layanan untuk sebuah sistem komputer lain, sistem lain tersebut merupakan aktor [10].

Use case diagram menampilkan aktor, *use case,* dan hubungan antara mereka [10]:

a. Aktor mana yang menggunakan *use case* mana.
b. *Use case* mana yang memasukkan *use case* lain.

Berikut ini adalah contoh dari diagram *use case* :

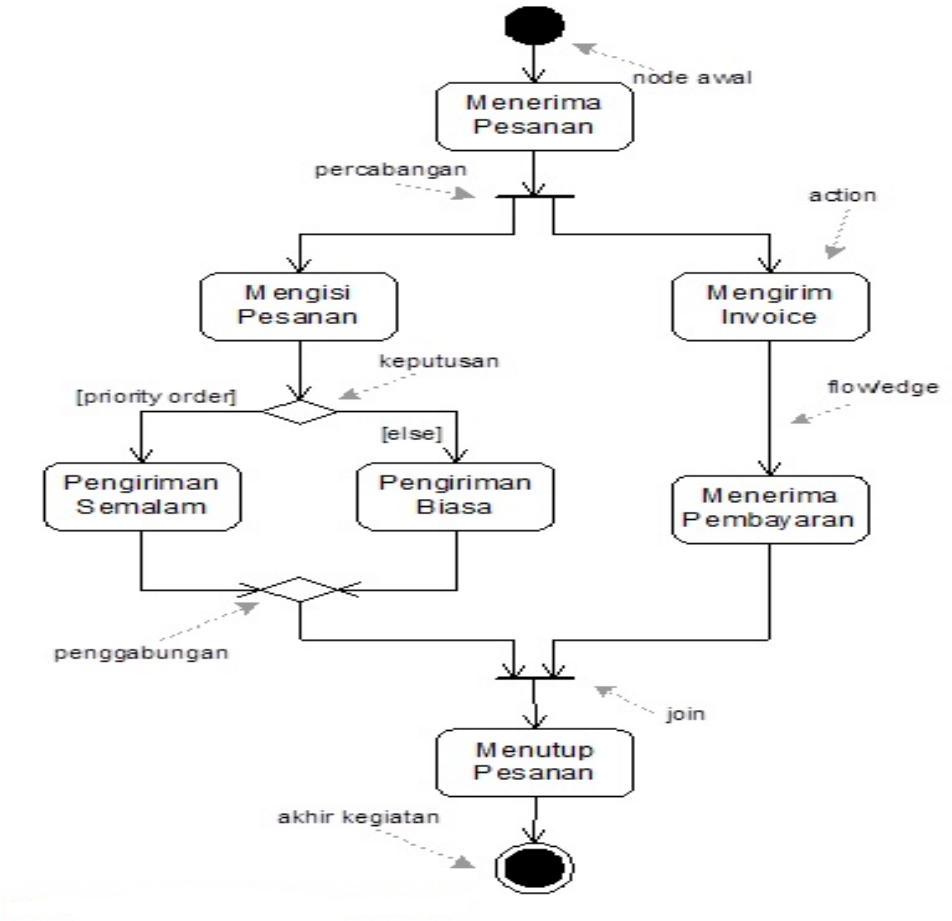


Gambar 2.3 Contoh diagram Use Case

[Cockburnm use case] menjelaskan sebuah skema tingkatan use case. Inti use case berada pada tingkat sea level. Use case sea level khususnya mewakili sebuah interaksi diskrit antara aktor utama dan sistem. Use case yang ada disana hanya karena mereka dimasukkan oleh use case sea level adalah fish level. Lebih tinggi, use case kite level menampilkan bagaimana use case sea level sesuai dengan interaksi bisnis yang lebih luas. Use case kite level biasanya merupakan use case bisnis, sedangkan se case sea dan fish level merupakan use case sistem. Kebanyakan use case Anda berada pada tingkatan sea level [10]

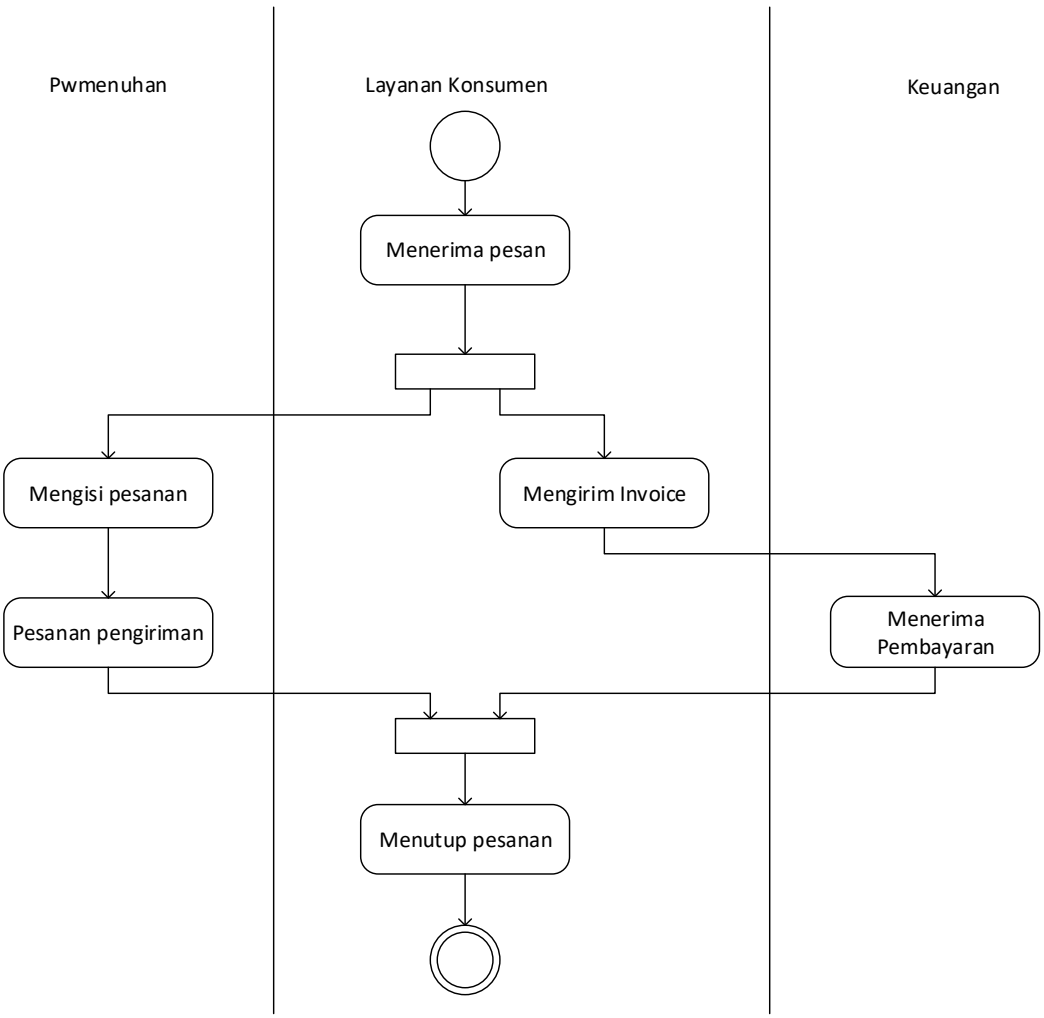
2.1.11.2. Activity Diagram

Diagram *activity* adalah teknik untuk menggambarkan logika prosedural, proses bisnis, dan jalur kerja. [10].



Gambar 2.4 Contoh Diagram Aktivitas

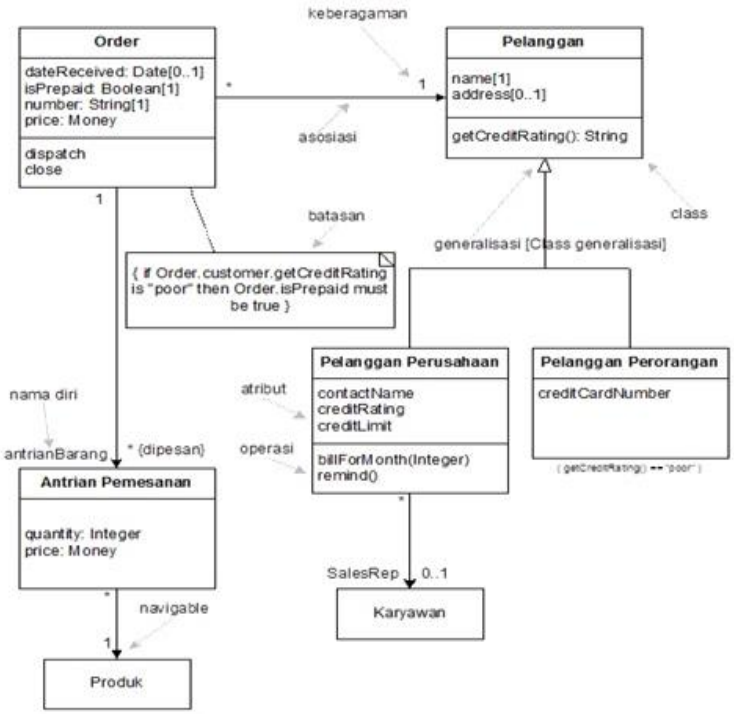
Activity diagram memberi tahu anda tentang apa yang terjadi, tetapi diagram ini tidak memberi tahu anda tentang siapa yang melakukan apa. Jika anda ingin menunjukkan siapa melakukan apa, anda dapat memisahkan activity diagram ke dalam partisi-pasrtisi yang menampilkan action mana yang dilakukan oleh sebuah class atau organisasi. Pembuatan partisi pada Gambar 2.5 merupakan pembuatan partisi sederhana satu dimensi. Model ini sering disebut sebagai swim lanes.



Gambar 2.5 Partisi Pada Activity Diagram

2.1.11.3. Diagram Kelas

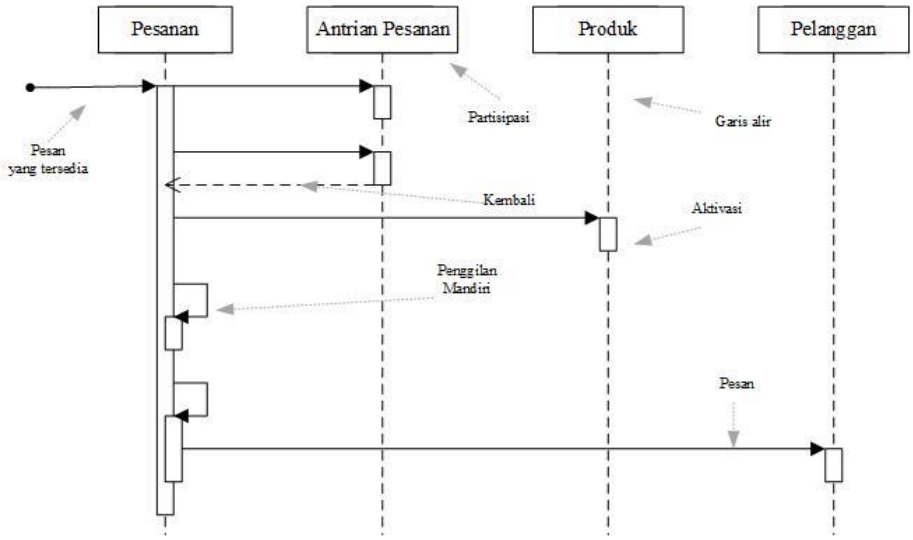
Diagram *class* mendeskripsikan jenis-jenis objek dalam sistem dan berbagai macam hubungan statis yang terdapat diantara mereka. Diagram *class* juga menunjukan properti dan operasi sebuah *class* dan batasan-batasan yang terdapat pada sebuah hubungan-hubungan objek. UML menggunakan istilah fitur sebagai istilah umum yang meliputi properti dan operasi sebuah *class.* Didalam class dibagi menjadi tiga bagian. Nama *class*, atributnya, dan operasinya. [10].



Gambar 2.6 Contoh Diagram Class

2.1.11.4. Sequence Diagram

Diagram *sequence* secara khusus menjabarkan *behavior* sebuah skenario tunggal. Diagram tersebut menunjukan sejumlah objek contoh dan pesan-pesan yang melewati objek-objek di dalam *use case.* Diagram *sequence* menunjukkan interaksi dengan menampilkan setiap partisipan dengan garis alir secara vertikal dan pengurutan pesan dari atas ke bawah [10].



Gambar 2.7 Contoh Diagram Sequence