BAB 2
TINJAUAN PUSTAKA

2.1 Landasan Teori

Landasan teori menjelaskan beberapa teori yang berkaitan dengan permasalahan yang akan dibahas sebagai pemahaman sistem yang akan dibangun

2.1.1 Sembilan Bahan Pokok (Sembako)

Sembilan Bahan Poko (Sembako) adalah adalah sembilan jenis kebutuhan pokok masyarakat menurut keputusan Menteri Industri dan Perdagangan no. 115/mpp/kep/2/1998 tanggal 27 Februari 1998 [1]. Sembako sudah menjadi kebutuhan utama dalam kehidupan sehari-hari. Berikut adalah daftar nama anggota bahan pokok sembako, yaitu : beras dan sagu, jagung, sayur-sayuran dan buah-buahan, daging (sapi dan ayam), susu, gula pasir, garam yang mengandung yodium/iodium, minyak goreng dan margarin, minyak tanah atau gas elpiji.

2.1.2 *Business Process Modelling Notation* (BPMN)

BPMN adalah standar untuk memodelkan proses bisnis dan proses-proses *web services*. Diusulkan oleh BPMI (*Business Process Management Initiative*) pada tahun 2004. BPMN dirancang bukan hanya mudah digunakan dan dipahami, tetapi juga memiliki kemampuan untuk memodelkan proses bisnis yang kompleks dan secara spesifik dirancang dengan mempertimbangkan *web services*. BPMN menyediakan notasi yang dapat dengan mudah dipahami oleh semua pengguna bisnis, termasuk juga analisis bisnis yang menciptakan draf awal dari proses sampai pengembang teknis yang bertanggung jawab untuk mengimplementasikan teknologi yang digunakan utuk menjalankan proses-proses tersebut [4].

Berikut aturan dalam memodelkan suatu proses bisnis :

1. Memodelkan kejadian-kejadian yang memulai proses, proses yang dilakukan dan hasil akhir dari aliran proses.
2. Keputusan bisnis atau percabangan aliran dimodelkan dengan *gateways*. Sebuah *gateway* mirip dengan *symbol* keputusan dalam *flowchart*.

3. Sebuah proses dalam aliran dapat mengandung sub-proses, yang secara grafis dapat ditujukkan dengan BPD (*Business Process Diagram*) lain yang tersambung melalui *task* yaitu level proses paling rending.
4. Sebuah tanda '+' pada *symbol* proses menunjukkan bahwa proses ini didekomposisi, jika tidak ada tanda '+' maka proses ini disebut sebuah *task*.

Sebuah BPD terdiri dari sekumpulan elemen grafis dan kategori dasar dari elemen BPD adalah :

a. *Flow Objects*
b. *Connecting Objects*
c. *Swimlanes*
d. *Artifacts*

Berikut contoh BMPN dapat dilihat pada gambar 2.1.



Gambar 2.1 Contoh BPMN

2.1.3 Aplikasi

Aplikasi merupakan program yang berisikan perintah-perintah untuk melaksanakan pengolahan data, aplikasi secara umum adalah suatu proses dari cara manual yang ditransformasikan ke komputer dengan membuat sistem atau program agar data diolah lebih berdaya guna secara optimal. Aplikasi (*application*) juga adalah *software* yang dibuat oleh suatu perusahaan komputer untuk mengerjakan tugas-tugas tertentu, misalnya Microsoft Word dan Microsoft Excel [5].

Dari pengertian di atas dapat disimpulkan bahwa aplikasi merupakan *software* yang ditransformasikan ke komputer yang berisikan perintah-perintah yang berfungsi untuk melakukan berbagai bentuk pekerjaan atau tugas-tugas tertentu seperti penerapan, penggunaan dan penambahan data. Selain itu aplikasi dapat mengintegrasikan berbagai kemampuan komputer.

2.1.4 Android

Android adalah sistem operasi berbasis Linux yang dimodifikasi untuk perangkat bergerak (*mobile devices*) yang terdiri dari sistem operasi, *middleware*, dan aplikasi-aplikasi utama. Awalnya, Android dikembangkan oleh oleh Android Inc. Perusahaan ini kemudian dibeli oleh google pada tahun 2005. Sistem operasi Android kemudian diluncurkan bersamaan dengan dibentuknya organisasi *Open Handset Alliance* tahun 2007. Selain Google, nama-nama besar juga ikut serta dalam *Open Handset Alliance*, antara lain Motorola, Samsung, LG, Sony Ericsson, T-Mobile, Vodafone, Toshiba dan Intel [6].



Gambar 2.2 Logo Android

2.1.4.1 Android SDK (Software Development Kit)

Android SDK adalah tools API (*Application Programming Interface*) yang diperlukan untuk mulai mengembangkan aplikasi pada *platform* Android menggunakan bahasa pemrograman Java. Android merupakan subset perangkat lunak untuk ponsel yang meliputi sistem operasi, *middleware* dan aplikasi kunci yang di-*release* oleh Google. Saat ini disediakan Android SDK (*Software Development Kit*) sebagai alat bantu dan API untuk mulai mengembangkan aplikasi pada *platform* Android menggunakan bahasa pemrograman Java. Sebagai *platform* aplikasi-netral, Android memberi kesempatan untuk membuat aplikasi yang dibutuhkan [6].

2.1.4.2 ADT (Android Development Tools)

Android Develpoment Tools adalah *plugin* yang di desain untuk IDE Eclipse yang memberikan kemudahan dalam mengembangkan aplikasi android dengan menggunakan IDE Eclipse. Dengan menggunakan ADT untuk Eclipse akan memudahkan dalam membuat aplikasi *project Android*, membuat GUI aplikasi, dan menambahkan komponen-komponen yang lainnya. Mengembangkan aplikasi Android dengan menggunakan ADT di Eclipse sangat dianjurkan dan sangat mudah untuk memulai mengembangkan aplikasi Android. Berikut adalah versi ADT untuk Eclipse yang sudah dirilis [6]:

- ADT 12.0.0 (July 2011)
- ADT 11.0.0 (June 2011)
- ADT 10.0.1 (March 2011)
- ADT 10.0.0 (February 2011)
- ADT 9.0.0 (January 2011)
- ADT 8.0.1 (December 2010)
- ADT 8.0.0 (December 2010)
- ADT 0.9.9 (September 2010)
- ADT 0.9.8 (September 2010)
- ADT 0.9.7 (May 2010)
- ADT 0.9.6 (March 2010)
- ADT 0.9.5 (December 2009)
- ADT 0.9.4 (October 2009)

Semakin tinggi platform Android yang digunakan, dianjurkan menggunakan ADT yang lebih terbaru, karena biasanya munculnya platform baru diikuti oleh munculnya versi ADT yang terbaru.

2.1.4.3 Arsitektur Android

Secara garis besar Arsitektur Android dapat di jelaskan dan di gambarkan sebagai berikut [7] :

1. *Application and Widgets*

Application dan Widgets adalah layer dimana user berhubungan dengan aplikasi saja, dimana biasanya user men-download aplikasi, melakukan instalasi dan menjalankan aplikasi.

2. *Applications Frameworks*

Android adalah "*Open Development Platform*" yaitu Android menawarkan kepada pengembang atau member kemampuan untuk membangun aplikasi yang inovatif.

3. *Libraries*

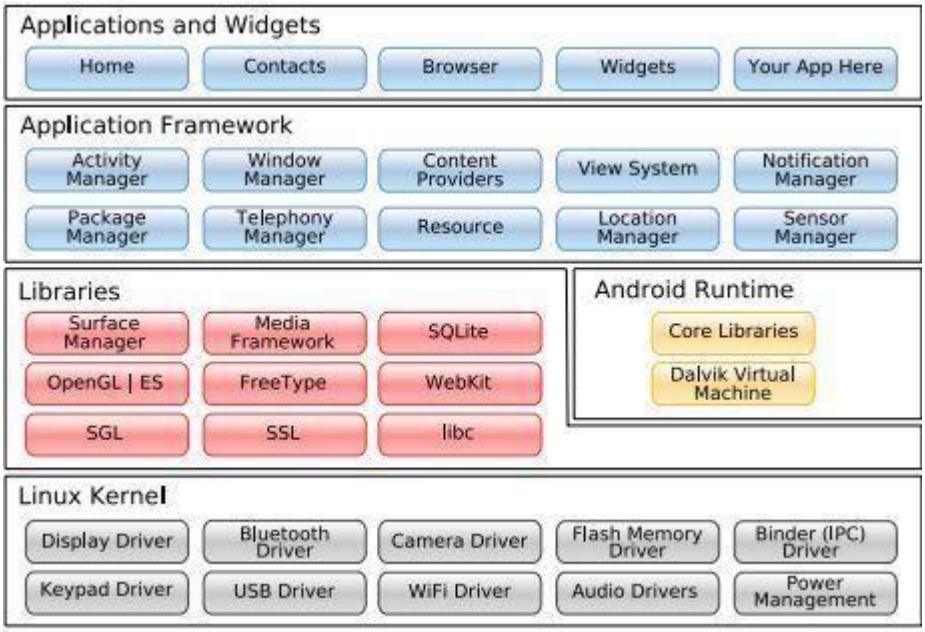
Libraries adalah *layer* dimana fitur-fitur Android berada, biasanya para pembuat aplikasi mengakses *libraries* untuk menjalankan aplikasinya.

4. *Android Run Time*

Layer yang membuat aplikasi Android dapat djalankan dimana dalam prosesnya menggunakan Implementasi Linux.

5. *Linux Kernel*

Linux Kernel adalah *layer* dimana inti dari *operating system* dari Android itu berada. Berisi file-file sistem yang mengatur sistem *processing*, *memory*, *resource*, *drivers*, dan sistem-sistem opersi Android lainnya.



Gambar 2.3 Arsitektur Android

2.1.4.4 Fundamental Aplikasi

Aplikasi Android ditulis dalam bahasa pemrograman Java. Kode Java dikompilasi bersama dengan data file *resource* yang dibutuhkan oleh aplikasi, dimana prosesnya di-*package* oleh *tools* yang dinamakan "*apt tools*" ke dalam paket Android sehingga menghasilkan file dengan ekstensi apk. Ada empat jenis komponen pada aplikasi Android yaitu [7]:

1) *Activities*
2) *Service*
3) *Broadcast Receiver*
4) *Content Provider*

2.1.4.5 Versi Android

Android telah mengalami sejumlah pembaruan sejak pertama kali dirilis . Rata-rata, versi terbaru dari Android dirilis setiap 6 bulan. Tabel 2.1 menunjukkan beberapa jenis Android dan nama kodenya. Penamaan kode menggunakan nama makanan dan huruf depannya urut sesuai abjad [7].

Tabel 2.1 Daftar Android

Versi	Tanggal Rilis	Kode
1.1	9 Februari 2009	
1.5	30 April 2009	Cupcake
1.6	15 September 2009	Donut
2.0/2.1	26 Oktober 2009	Eclair
2.2	20 Mei 2010	Frozen Yoghurt (Froyo)
2.3	6 Desember 2010	Gingerbread
3.0	22 Februari 2011	Honeycomb
4.0	19 Oktober 2011	Ice Cream Sandwich
4.1	27 Juni 2012	Jelly Bean
4.2	29 Oktober 2012	Jelly Bean
4.3	24 Juli 2013	Jelly Bean
4.4	3 September 2013	Kitkat

2.1.5 Android Studio

Android Studio adalah IDE resmi untuk pengembangan aplikasi Android, berdasarkan IntelliJ IDEA. Di atas IntelliJ yang kuat *code editor* dan pengembang alat, Android Studio menawarkan lebih banyak fitur yang meningkatkan produktivitas Anda ketika membangun aplikasi Android, seperti [8] :

1. Sebuah *fleksibel* berbasis *Gradle* membangun sistem
2. Membangun varian dan beberapa generasi file APK
3. Kode template untuk membantu Anda membangun fitur aplikasi umum
4. Sebuah *layout editor* kaya dengan dukungan untuk *drag* dan *drop tema editing*
5. alat *Lint* untuk menangkap kinerja, kegunaan, kompatibilitas versi, dan masalah lainnya
6. Kode menyusut dengan ProGuard dan sumber daya menyusut dengan Gradle
7. Built-in dukungan untuk Google *Cloud Platform*, sehingga mudah untuk mengintegrasikan Google *Cloud Messaging* dan App *Engine*.

2.1.5.1 Struktur Proyek

Setiap proyek di Android Studio berisi satu atau lebih modul dengan file kode sumber dan file sumber daya. Berbagai jenis modul meliputi :

1) *Android app modules*
2) *Test modules*
3) *Library modules*
4) *App Engine modules*

Secara *default*, Android Studio menampilkan file proyek Anda dalam tampilan proyek Android. Pandangan ini diselenggarakan oleh modul untuk menyediakan akses cepat ke file kunci sumber proyek Anda. Semua file membangun terlihat di tingkat atas di bawah *Script Gradle* dan setiap modul aplikasi mengandung tiga unsur berikut :

1. Manifests : file Manifest.
2. Java : Sumber file kode.
3. Res : file Sumber Daya.

Struktur proyek Android pada disk berbeda dari representasi pipih ini. Untuk melihat struktur file yang sebenarnya dari proyek, pilih Proyek dari Proyek *drop-down*. Anda juga dapat menyesuaikan tampilan dari file proyek untuk fokus pada aspek-aspek tertentu dari pengembangan aplikasi Anda. Misalnya, memilih Masalah tampilan proyek Anda menampilkan link ke file sumber yang berisi setiap diakui coding dan sintaksis kesalahan, seperti hilang elemen XML tag penutup dalam *file layout*.

2.1.5.2 Versi Android Studio

Versi Android Studio sebagai berikut [8]:

1) Android Studio v0.1.x (Mei 2013)
2) Android Studio v0.2.x (Juli 2013)
3) Android Studio v0.3.2 (Oktober 2013)
4) Android Studio v0.4.2 (Januari 2014)
5) Android Studio v0.4.6 (Maret 2014)
6) Android Studio v0.5.2 (Mei 2014)
7) Android Studio v0.8.0 (Juni 2014)
8) Android Studio v0.8.6 (Agustus 2014)
9) Android Studio v0.8.14 (Oktober 2014)
10) Android Studio v1.0 (Desember 2014)
11) Android Studio v1.0.1 (Desember 2014)
12) Android Studio v1.1.0 (Februari 2015)
13) Android Studio v1.2.0 (April 2015)
14) Android Studio v1.2.1 (Mei 2015)
15) Android Studio v1.2.2 (Juni 2015)
16) Android Studio v1.3.0 (Juli 2015)
17) Android Studio v1.3.1 (Agustus 2015)
18) Android Studio v1.3.2 (Agustus 2015)
19) Android Studio v1.4.0 (September 2015)
20) Android Studio v1.4.1 (Oktober 2015)
21) Android Studio v1.5.0 (November 2015)
22) Android Studio v1.5.1 (Desember 2015)

23) Android Studio v2.0.0 (April 2016)
24) Android Studio v2.1.0 (April 2016)
25) Android Studio v2.1.1 (Mei 2016)

2.1.6 Java

Java adalah bahasa pemrograman yang dapat dijalankan di berbagai komputer termasuk telepon genggam. Bahasa ini awalnya dibuat oleh James Gosling saat masih bergabung di *Sun Microsystems* saat ini merupakan bagian dari *Oracle* dan dirilis tahun 1995. Bahasa ini banyak mengadopsi sintaksis yang terdapat pada C dan C++ namun dengan sintaksis model objek yang lebih sederhana serta dukungan rutin-rutin aras bawah yang minimal. Aplikasi-aplikasi berbasis java umumnya dikompilasi ke dalam *p-code* (*bytecode*) dan dapat dijalankan pada berbagai Mesin *Virtual Java* (JVM). Java merupakan bahasa pemrograman yang bersifat umum/non-spesifik (*general purpose*), dan secara khusus didisain untuk memanfaatkan dependensi implementasi seminimal mungkin. Karena fungsionalitasnya yang memungkinkan aplikasi java mampu berjalan di beberapa platform sistem operasi yang berbeda, java dikenal pula dengan slogannya, "Tulis sekali, jalankan di mana pun". Saat ini java merupakan bahasa pemrograman yang paling populer digunakan, dan secara luas dimanfaatkan dalam pengembangan berbagai jenis perangkat lunak aplikasi ataupun aplikasi berbasis web [9].



Gambar 2.4 Logo Java

2.1.6.1 Sejarah Perkembangan Java

Bahasa pemrograman Java terlahir dari *The Green Project*, yang berjalan selama 18 bulan, dari awal tahun 1991 hingga musim panas 1992. Proyek tersebut belum menggunakan versi yang dinamakan Oak. Proyek ini dimotori oleh Patrick Naughton, Mike Sheridan, James Gosling dan Bill Joy, beserta sembilan pemrogram lainnya dari *Sun Microsystems*. Salah satu hasil proyek ini adalah maskot Duke yang dibuat oleh Joe Palrang.

Pertemuan proyek berlangsung di sebuah gedung perkantoran Sand Hill Road di Menlo Park. Sekitar musim panas 1992 proyek ini ditutup dengan menghasilkan sebuah program Java Oak pertama, yang ditujukan sebagai pengendali sebuah peralatan dengan teknologi layar sentuh (*touch screen*), seperti pada PDA sekarang ini. Teknologi baru ini dinamai "*7" (Star Seven).

Setelah era Star Seven selesai, sebuah anak perusahaan Tv kabel tertarik ditambah beberapa orang dari proyek *The Green Project*. Mereka memusatkan kegiatannya pada sebuah ruangan kantor di 100 Hamilton Avenue, Palo Alto.

Perusahaan baru ini bertambah maju, jumlah karyawan meningkat dalam waktu singkat dari 13 menjadi 70 orang. Pada rentang waktu ini juga ditetapkan pemakaian Internet sebagai *medium* yang menjembatani kerja dan ide di antara mereka. Pada awal tahun 1990-an, Internet masih merupakan rintisan, yang dipakai hanya di kalangan akademisi dan militer.

Mereka menjadikan perambah (*browser*) *Mosaic* sebagai landasan awal untuk membuat perambah Java pertama yang dinamai *Web Runner*, terinsipirasi dari film 1980-an, Blade Runner. Pada perkembangan rilis pertama, *Web Runner* berganti nama menjadi Hot Java.

Pada sekitar bulan Maret 1995, untuk pertama kali kode sumber Java versi 1.0a2 dibuka. Kesuksesan mereka diikuti dengan untuk pemberitaan pertama kali pada surat kabar San Jose Mercury News pada tanggal 23 Mei 1995.

Sayang terjadi perpecahan di antara mereka suatu hari pada pukul 04.00 di sebuah ruangan hotel Sheraton Palace. Tiga dari pimpinan utama proyek, Eric Schmidt dan George Paolini dari Sun Microsystems bersama Marc Andreessen, membentuk Netscape.

Nama Oak, diambil dari pohon oak yang tumbuh di depan jendela ruangan kerja "Bapak Java", James Gosling. Nama Oak ini tidak dipakai untuk versi release Java karena sebuah perangkat lunak lain sudah terdaftar dengan merek dagang tersebut, sehingga diambil nama penggantinya menjadi "Java". Nama ini diambil dari kopi murni yang digiling langsung dari biji (kopi tubruk) kesukaan Gosling. Konon kopi ini berasal dari Pulau Jawa. Jadi nama bahasa pemrograman Java tidak lain berasal dari kata Jawa (bahasa Inggris untuk Jawa adalah Java).

2.1.6.2 Versi Java

Versi awal Java ditahun 1996 sudah merupakan versi *release* sehingga dinamakan Java Versi 1.0. Java versi ini menyertakan banyak paket standar awal yang terus dikembangkan pada versi selanjutnya :

1. Java.lang : Peruntukan kelas elemen-elemen dasar.
2. Java.io : Peruntukan kelas input dan output, termasuk penggunaan berkas.
3. Java.util : Peruntukan kelas pelengkap seperti kelas struktur data dan kelas penaggalan.
4. Java.net : Peruntukan kelas TCP/IP, yang memungkinkan berkomunikasi dengan komputer lain menggunakan jaringan TCP/IP.
5. Java.awt : Kelas dasar untuk aplikasi antarmuka dengan penggunaan (GUI).
6. Java.applet : Kelas dasar aplikasi antar muka untuk diterapkan pada penjelajah web.

2.1.6.3 Kelebihan Java

Kelebihan dari java adalah sebagai berikut :

1) **Multiplatform.** Kelebihan utama dari Java ialah dapat dijalankan di beberapa *platform*/sistem operasi komputer, sesuai dengan prinsip tulis sekali, jalankan di mana saja. Dengan kelebihan ini pemrogram cukup menulis sebuah program Java dan dikompilasi (diubah, dari bahasa yang dimengerti manusia menjadi bahasa mesin/*bytecode*) sekali lalu hasilnya dapat dijalankan di atas beberapa platform tanpa perubahan.
2) **OOP (*Object Oriented Programming* - Pemrogram Berorientasi Objek)**
3) **Perpustakaan Kelas Yang Lengkap**, Java terkenal dengan kelengkapan library/perpustakaan (kumpulan program program yang disertakan dalam

pemrograman java) yang sangat memudahkan dalam penggunaan oleh para pemrogram untuk membangun aplikasinya. Kelengkapan perpustakaan ini ditambah dengan keberadaan komunitas Java yang besar yang terus menerus membuat perpustakaan-perpustakaan baru untuk melingkupi seluruh kebutuhan pembangunan aplikasi.

4) **Bergaya C++**, memiliki sintaks seperti bahasa pemrograman C++ sehingga menarik banyak pemrogram C++ untuk pindah ke Java. Saat ini pengguna Java sangat banyak, sebagian besar adalah pemrogram C++ yang pindah ke Java. Universitas-universitas di Amerika Serikat juga mulai berpindah dengan mengajarkan Java kepada murid-murid yang baru karena lebih mudah dipahami oleh murid dan dapat berguna juga bagi mereka yang bukan mengambil jurusan komputer.
5) **Pengumpulan Sampah Otomatis**, memiliki fasilitas pengaturan penggunaan memori sehingga para pemrogram tidak perlu melakukan pengaturan memori secara langsung (seperti halnya dalam bahasa C++ yang dipakai secara luas).

2.1.6.4 Kekurangan Java

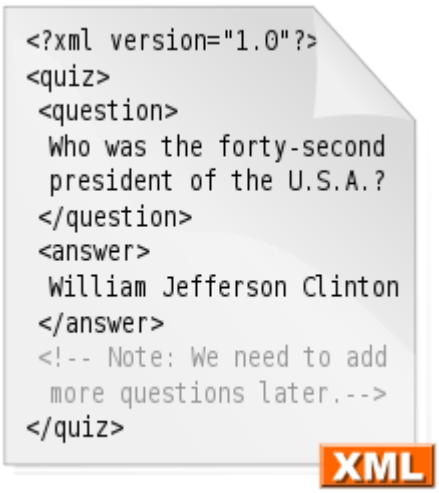
Kekurangan dari java adalah sebagai berikut :

1) **Tulis sekali, jalankan di mana saja** - Masih ada beberapa hal yang tidak kompatibel antara platform satu dengan platform lain. Untuk J2SE, misalnya SWT-AWT bridge yang sampai sekarang tidak berfungsi pada Mac OS X.
2) **Mudah didekompilasi**. Dekompilasi adalah proses membalikkan dari kode jadi menjadi kode sumber. Ini dimungkinkan karena kode jadi Java merupakan bytecode yang menyimpan banyak atribut bahasa tingkat tinggi, seperti nama-nama kelas, metode, dan tipe data. Hal yang sama juga terjadi pada Microsoft .NET Platform. Dengan demikian, algoritma yang digunakan program akan lebih sulit disembunyikan dan mudah dibajak/di-reverse-engineer.
3) **Penggunaan memori yang banyak**. Penggunaan memori untuk program berbasis Java jauh lebih besar daripada bahasa tingkat tinggi generasi sebelumnya seperti C/C++ dan Pascal (lebih spesifik lagi, Delphi dan Object Pascal). Biasanya ini bukan merupakan masalah bagi pihak yang menggunakan teknologi terbaru (karena trend memori terpasang makin murah), tetapi menjadi

masalah bagi mereka yang masih harus berkutat dengan mesin komputer berumur lebih dari 4 tahun.

2.1.7 XML (*Extensible Markup Language*)

XML (*Extensible Markup Language*) adalah bahasa markup untuk keperluan umum yang disarankan oleh W3C untuk membuat dokumen markup keperluan pertukaran data antar sistem yang beraneka ragam. XML merupakan kelanjutan dari HTML (*HyperText Markup Language*) yang merupakan bahasa standar untuk melacak Internet [10].



Gambar 2.5 Contoh Dokumen XML

2.1.7.1 Pengenalan XML

XML didesain untuk mempu menyimpan data secara ringkas dan mudah diatur. Kata kunci utama XML adalah data (jamak dari datum) yang jika diolah bisa memberikan informasi. XML menyediakan suatu cara terstandarisasi namun bisa dimodifikasi untuk menggambarkan isi dari dokumen. Dengan sendirinya, XML dapat digunakan untuk menggambarkan sembarang *view database*, tetapi dengan suatu cara yang standar.

2.1.7.2 Tipe XML

XML memiliki tiga tipe file, yaitu :

1. XML, merupakan standar format dari struktur berkas (file).
2. XSL, merupakan standar untuk memodifikasi data yang diimpor atau diekspor.
3. XSD, merupakan standar yang mendefinisikan struktur database dalam XML.

2.1.8 PHP (*Hypertext Preprocessor*)

PHP adalah sebuah bahasa pemograman yang berjalan dalam sebuah *web server* (*server side*). PHP adalah bahasa skrip yang sangat populer digunakan untuk mengembangkan aplikasi-aplikasi web, meskipun sebenarnya PHP juga dapat digunakan untuk mengembangkan aplikasi desktop, baik *console* maupun yang berbasis GUI [11].

2.1.8.1 Konsep Dasar PHP

Kode PHP diawali dengan tanda lebih kecil (<) dan diakhiri dengan tanda lebih besar (>). Ada empat cara untuk menuliskan skrip PHP, yaitu :

1. <? Hallo ?>
2. <? php Hallo ?>
3. <% Hallo %>
4. <SCRIPT LANGUAGE="php">Hallo</script>

Pemisah antara instruksi adalah titik koma (;) dan untuk membuat atau menambahkan komentar/standar penulisan adalah : /* komentar */, // komentar, # komentar. Cara penulisan dibedakan menjadi Embedded dan Non Embedded script.

2.1.8.2 Variable PHP

Variabel PHP digunakan untuk menyimpan data yang nilainya dapat berubah-ubah. Dalam bahasa PHP, variabel dimulai dengan tanda "$". Aturan penulisan variabel antara lain sebagai berikut :

a. Hanya ada 3 karakter yang dapat digunakan untuk nama variable, yaitu huruf, angka dan garis bawah.
b. Karakter pertama setelah tanda "$" harus huruf atau garis bawah.
c. Jika nama variable lebih dari satu kata, tidak boleh ada tanda spasi diantara keduanya.

2.1.8.3 Kelebihan PHP

PHP (*Hypertext Preprocessor*) adalah skrip yang berjalan dalam *server side* yang ditambahkan dalam HTML. Skrip ini akan membuat suatu aplikasi dapat diintegrasikan ke dalam HTML sehingga suatu halaman HTML tidak lagi bersifat statis, namun menjadi bersifat dinamis. Sifat *server side* ini membuat pekerjaan skrip tersebut dikerjakan di *server* sedangkan yang dikirimkan ke *browser* adalah

hasil proses dari skrip tersebut yang sudah berbentuk HTML. Keunggulan dari sifatnya yang *server side* tersebut adalah :

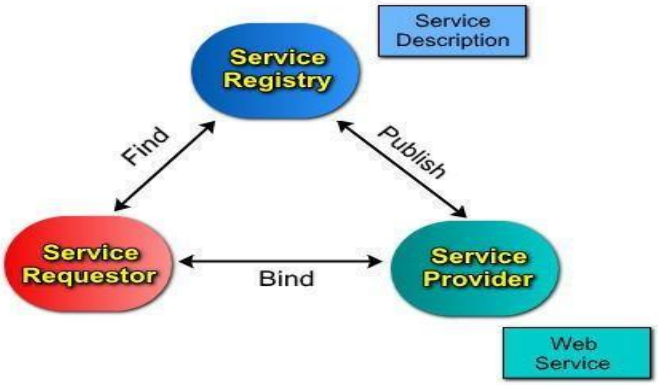
a. Tidak diperlukan adanya kompatibilitas *browser* atau harus menggunakan *browser* tertentu, karena *server*-lah yangakan mengerjakan skrip tersebut. Hasil yang dikirimkan kembali ke *browser* biasanya dalam bentuk teks ataupun gambar sehingga dapat dikenali oleh *browser* apapun.
b. Dapat memanfaatkan sumber-sumber aplikasi yang dimiliki oleh *server*, contohnya : hubungan kedalam database.
c. Skrip asli tidak terlihat sehingga keamanam lebih terjamin.

2.1.9 Web Service

Web service adalah salah satu bentuk sistem perangkat lunak yang didesain untuk mendukung interaksi mesin-ke-mesin melalui jaringan. *Web service* memiliki *interface* yang dideskripsikan dalam format yang dapat dibaca oleh mesin. Sistem-sistem lainnya berinteraksi dengan web service menggunakan pesan SOAP yang umumnya dikirim melalui HTTP dalam bentuk XML [12].

Definisi diatas diberikan oleh *World Wide Web Consortium*(W3C) yang merupakan badan yang menciptakan dan mengembangkan standar *web service*. Tetapi secara umum, *web service* tidak terbatas hanya pada standar SOAP saja. Salah satu pustaka yang mengulas lengkap tentang *web service* menyebutkan definisi yang lebih umum: *web service* adalah aplikasi yang diakses melalui internet menggunakan protokol standar internet dan menggunakan XML sebagai format pesannya.

2.1.9.1 Arsitektur Web Service



Gambar 2.6 Arsitektur Web Service

Pada gambar diatas, ada tiga komponen yang membuat *web service* berjalan. Ketiga komponen itu adalah [12]:

1. *Service provider*, merupakan pemilik *web service* yang berfungsi menyediakan kumpulan operasi dari *web service*.
2. *Service requestor*, merupakan aplikasi yang bertindak sebagai klien dari *web service* yang mencari dan memulai interaksi terhadap layanan yang disediakan.
3. *Service registry*, merupakan tempat dimana *service provider* mempublikasikan layanannya. Pada arsitektur *web service*, *service registry* bersifat optional. Teknologi *web service* memungkinkan kita dapat menghubungkan berbagai jenis software yang memiliki platform dan sistem operasi yang berbeda.

2.1.10 JSON (JavaScript Object Notation)

JSON (*JavaScript Object Notation*) adalah format pertukaran data yang ringan, mudah dibaca dan ditulis oleh manusia, serta mudah diterjemahkan dan dibuat (*generate*) oleh komputer. 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. Kelebihan-kelebihan dari JSON adalah [13]:

1. Format Penulisan : Untuk merepresentasikan sebuah struktur data yang rumit dan berbentuk hirarkis penulisan JSON relatif lebih terstruktur dan mudah.

2. Ukuran karakter yang dibutuhkan JSON lebih kecil dibandingkan XML untuk data yang sama.
3. Proses parsing merupakan proses pengenalan token atau bagian-bagian kecil dalam rangkaian dokumen XML/JSON.

JSON terbuat dari dua struktur :

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*).

2.1.11 API (*Application Programming Interface*)

API adalah sekumpulan perintah, fungsi, dan protokol yang dapat digunakan saat membangun perangkat lunak untuk sistem operasi tertentu. API memungkinkan programmer untuk menggunakan fungsi standar untuk berinteraksi dengan system operasi. API atau *Application Programming Interface* juga merupakan suatu dokumentasi yang terdiri dari antar muka, fungsi, kelas, struktur untuk membangun sebuah perangkat lunak.

Dengan adanya API, maka memudahkan seorang *programmer* untuk membongkar suatu *software* untuk kemudian dapat dikembangkan atau diintegrasikan dengan perangkat lunak yang lain. API dapat dikatakan sebagai penghubung suatu aplikasi dengan aplikasi lainnya. Suatu rutin standar yang memungkinkan *developer* menggunakan *system function*. Proses ini dikelola melalui *operating system*. Keunggulan dari API ini adalah memungkinkan suatu aplikasi dengan aplikasi lainnya untuk saling berinteraksi [14].

Keuntungan dengan menggunakan API adalah sebagai berikut:

1. Portabilitas.

Developer yang menggunakan API dapat menjalankan programnya dalam sistem operasi mana saja asalkan sudah terinstal API tersebut.

2. Lebih Mudah Dimengerti

API menggunakan bahasa yang lebih terstruktur dan mudah dimengerti daripada bahasa *system call*. Hal ini sangat penting dalam hal editing dan pengembangan.

Cara menggunakan API :

1) Dilakukan dengan mengimpor *package*/kelas.
2) Ada beberapa kelas bernama sama dipackage yang berbeda, yaitu:

a. Import salah satu dan gunakan nama lengkap untuk yang lain.
b. Gunakan nama lengkap semua kelas

Kebanyakan Sistem Operasi seperti Windows, menyediakan fasilitas API sehingga *programmer* dapat melakukan aktivitas programming dengan lebih konsisten. Meskipun API didesain untuk programer, namun API juga baik untuk user karena setidaknya dapat menjamin bahwa program tersebut memiliki interface yang sama, sehingga lebih mudah untuk dipelajari.

2.1.12 GPS (*Global Position Service*)

GPS (*Global Position Service*) adalah sistem navigasi berbasis satelit yang dikembangkan oleh U.S. Department of Defense (DoD). pada awal tahun 1970 awalnya GPS dikembangkan sebagai sistem militer untuk memenuhi kebutuhan militer. Namun, yang terakhir dibuat tersedia untuk warga sipil, dan sekarang sistem penggunaan ganda yang dapat diakses oleh pengguna militer dan sipil. GPS memberikan posisi terus menerus dan informasi waktu setiap tempat di dunia di bawah setiap kondisi cuaca. Karena, melayani jumlah pengguna yang tidak terbatas serta digunakan untuk alasan keamanan. GPS menggunakan *one-way-ranging* (*passive*) sistem. Artinya, pengguna hanya dapat menerima sinyal satelit [15].

Satelit-satelit GPS harus selalu berada pada posisi orbit yang tepat untuk menjaga akurasi data yang dikirim ke GPS *receiver*, sehingga harus selalu dipelihara agar posisinya tepat. Posisi satelit-satelit tersebut selalu dipantau oleh stasiun pengendali. Stasiun-stasiun pengendali di bumi ada di Hawaii, Ascension Islan, Diego Garcia, Kwajalein dan Colorado Spring. Untuk dapat mengetahui posisi seseorang maka diperlukan alat yang diberi nama GPS *receiver* yang berfungsi untuk menerima sinyal yang dikirim dari satelit GPS. GPS *receiver*

mengambil informasi tersebut dan melakukan perhitungan triangulation untuk menentukan lokasi pengguna dengan tepat. GPS *receiver* membandingkan waktu sinyal dikirim dengan waktu sinyal tersebut diterima untuk mengetahui jarak satelit. Dengan mengetahui jarak tersebut, GPS *receiver* dapat melakukan perhitungan dan menentukan posisi pengguna dan menampilkan dalam peta elektronik. Setelah menentukan posisi pengguna, selanjutnya GPS dapat menghitung informasi lain, seperti kecepatan, arah yang dituju, jalur, tujuan perjalanan, jarak tujuan, matahari terbit dan matahari terbenam dan masih banyak lagi.

Satelit GPS sangat presisi dalam mengirim informasi karena satelit tersebut menggunakan jam atom. Jam atom yang ada pada satelit merupakan partikel atom yang diisolasi, sehingga dapat menghasilkan jam yang akurat dibandingkan dengan jam biasa. Keistimewaan GPS adalah mampu bekerja dalam berbagai kondisi cuaca, siang atau malam. Keakuratan sebuah perangkat GPS bisa mencapai 15 meter, bahkan model terbaru yang dilengkapi teknologi *Wide Area Augmentation System* (WAAS) keakuratannya sampai 3 meter. Karena GPS bekerja mengandalkan satelit, maka penggunaannya disarankan di tempat terbuka. Penggunaan di dalam ruangan, atau di tempat yang menghalangi arah satelit (di angkasa), maka GPS tidak akan bekerja secara akurat dan maksimal. Perhitungan waktu yang akurat sangat menentukan akurasi perhitungan untuk menentukan informasi lokasi. Semakin banyak sinyal satelit yang dapat diterima maka akan semakin presisi data posisi yang dihasilkan. Selain itu, ketinggian juga mempengaruhi proses kerja GPS, karena semakin tinggi maka semakin bersih atmosfer, sehingga gangguan semakin sedikit.

2.1.13 LBS (*Location Based Service*)

LBS merupakan konsep baru yang menunjukkan aplikasi yang mengintegrasikan lokasi geografis (yaitu, spasial koordinat) dengan gagasan umum layanan. Konsep LBS dapat didefinisikan sebagai layanan yang mengintegrasikan lokasi perangkat mobile atau posisi dengan informasi lain sehingga dapat memberikan nilai tambah untuk pengguna. Lokasi layanan telah ada sejak tahun 1970-an dengan kedudukan global di seluruh dunia yang dikenal sebagai GPS(*Global Position Service*) yang dikembangkan oleh Departemen Pertahanan

Amerika Serikat. Tapi dalam dekade berikutnya ketika pemerintah luar AS memutuskan untuk membuat data posisi sistem bebas tersedia untuk industri lainnya di seluruh dunia, yang industri ini telah diambil sampai kesempatan untuk mengakses data posisi melalui GPS dan sekarang menggunakannya untuk meningkatkan produk dan layanan mereka [3].

2.1.14 Database

Database atau basis data adalah kumpulan data yang disimpan secara sistematis di dalam komputer dan dapat diolah atau dimanipulasi menggunakan perangkat lunak (program aplikasi) untuk menghasilkan informasi. Pendefinisian basis data meliputi spesifikasi berupa tipe data, struktur, dan juga batasan-batasan data yang akan disimpan. Basis data merupakan aspek yang sangat penting dalam sistem informasi dimana basis data merupakan gudang penyimpanan data yang akan diolah lebih lanjut. Basis data menjadi penting karena dapat menghidari duplikasi data, hubungan antar data yang tidak jelas, organisasi data, dan juga update yang rumit.

Proses memasukkan dan mengambil data ke dan dari media penyimpanan data memerlukan perangkat lunak yang disebut dengan sistem manajemen basis data (*database management system* | DBMS). DBMS merupakan sistem perangkat lunak yang memungkinkan user untuk memelihara, mengontrol, dan mengakses data secara praktis dan efisien. Dengan kata lain semua akses ke basis data akan ditangani oleh DBMS. Ada beberapa fungsi yang harus ditangani DBMS yaitu mengolah pendefinisian data, dapat menangani permintaan pemakai untuk mengakses data, memeriksa sekuriti dan integriti data yang didefinisikan oleh DBA (*Database Administrator*), menangani kegagalan dalam pengaksesan data yang disebabkan oleh kerusakan sistem maupun disk, dan menangani unjuk kerja semua fungsi secara efisien.

Tujuan utama dari DBMS adalah untuk memberikan tinjauan abstrak data kepada user (pengguna). Jadi sistem menyembunyikan informasi tentang bagaimana data disimpan, dipelihara, dan tetap dapat diambil (akses) secara efisien. Pertimbangan efisien disini adalah bagaimana merancang struktur data yang

kompleks tetapi masih tetap bisa digunakan oleh pengguna awam tanpa mengetahui kompleksitas strukturnya. Database dibagi menjadi dua jenis yaitu [16]:

1. Basis Data Flat File

Basis data flat-file ideal untuk data berukuran kecil dan dapat dirubah dengan mudah. Pada dasarnya, mereka tersusun dari sekumpulan string dalam satu atau lebih file yang dapat diurai untuk mendapatkan informasi yang disimpan. Basis data flat-file baik digunakan untuk menyimpan daftar atau data yang sederhana dan dalam jumlah kecil. Basis data flat-file akan menjadi sangat rumit apabila digunakan untuk menyimpan data dengan struktur kompleks walaupun dimungkinkan pula untuk menyimpan data semacam itu. Salah satu masalah menggunakan basis data jenis ini adalah rentan pada korupsi data karena tidak adanya penguncian yang melekat ketika data digunakan atau dimodifikasi.

2. Basis Data Relasional

Basis data ini mempunyai struktur yang lebih logis terkait cara penyimpanan. Kata "relasional" berasal dari kenyataan bahwa tabel-tabel yang berada di basis data dapat dihubungkan satu dengan lainnya. Basis data relasional menggunakan sekumpulan tabel dua dimensi yang masing-masing tabel tersusun atas baris (*tupel*) dan kolom (atribut). Untuk membuat hubungan antara dua atau lebih tabel, digunakan key (atribut kunci) yaitu *primary key* di salah satu tabel dan *foreign key* di tabel yang lain. Saat ini, basis data relasional menjadi pilihan karena keunggulannya. Beberapa kelemahan yang mungkin dirasakan untuk basis data jenis ini adalah implementasi yang lebih sulit untuk data dalam jumlah besar dengan tingkat kompleksitasnya yang tinggi dan proses pencarian informasi yang lebih lambat karena perlu menghubungkan tabel-tabel terlebih dahulu apabila datanya tersebar di beberapa tabel.

2.1.15 MySQL

MySQL merupakan *software database* yang paling populer di kalangan para pengembang aplikasi database. MySQL juga adalah suatu perangkat lunak yang menganut atau mengimplementasikan model basis data relasional maka MySQL disebut sebagai *Relational Database Management System* (RDBMS). MySQL merupakan *software* RDBMS (atau *server database*) yang dapat mengelola

database dengan cepat, dapat menampung data dalam jumlah sangat besar, dapat diakses oleh banyak user (*multi-user*), dan dapat melakukan suatu proses secara sinkron atau berbarengan (*multi-threaded*) [17].

Saat ini, MySQL banyak digunakan di berbagai kalangan untuk melakukan penyimpanan dan pengolahan data, mulai dari kalangan akademis sampai ke industri, baik industri kecil, menengah, maupun besar. Lisensi MySQL terbagi menjadi dua. Anda dapat menggunakan MySQL sebagai produk open source dibawah GNU *General Public License* (gratis) atau dapat membeli lisensi dari versi komersialnya. MySQL versi komersial tentu memiliki nilai lebih atau kemampuan-kemampuan yang tidak disertakan pada versi gratis. Pada kenyataannya, untuk keperluan industri menengah kebawah, versi gratis masih dapat digunakan dengan baik. Beberapa contoh aplikasi yang menggunakan MySQL adalah Joomla, Wordpress, MyBB, phpBB, dan masih banyak lagi.

2.1.15.1 Kelebihan MySQL

Beberapa kelebihan yang menjadikan para pengembang aplikasi database memilih untuk menggunakan MySQL dibanding dengan RDBMS yang lain. Banyaknya pengguna MySQL yang kian meningkat secara drastis ini mengakibatkan nama MySQL menjadi melambung tinggi dan menjadikannya se-populer seperti saat sekarang. Berikut ini beberapa keuntungan dari MySQL [17]:

1. Fleksibel

MySQL dapat digunakan untuk mengembangkan aplikasi desktop maupun aplikasi web dengan menggunakan teknologi yang bervariasi. Ini berarti bahwa MySQL memiliki fleksibilitas terhadap teknologi yang akan digunakan sebagai pengembang aplikasi, apakah itu PHP, JSP, Java, Delphi, C++, maupun yang lainnya dengan cara menyediakan plug-in dan driver yang spesifik untuk masing-masing teknologi tersebut. Dalam database MySQL, juga memiliki dukungan terhadap stored procedure, fungsi, trigger, view, SQL standar ANSI, dll, yang akan mempermudah dan mempercepat proses pengembangan aplikasi.

2. Performa Tinggi

MySQL memiliki mesin *query* dengan performa tinggi, dengan demikian proses transaksional dapat dilakukan dengan sangat cepat. Hal ini terbukti dengan digunakannya MySQL sebagai database dari beberapa aplikasi web memiliki traffic (lalu lintas) sangat tinggi.

3. Lintas Platform

MySQL dapat digunakan pada platform atau lingkungan (dalam hal ini sistem operasi) yang beragam, bisa Microsoft Windows, Linux, atau UNIX. Ini menyebabkan proses migrasi data (bila dibutuhkan) antar sistem operasi dapat dilakukan secara lebih mudah. Misal, jika kita ingin mengganti sistem operasi pada mesin server.

4. Gratis

MySQL dapat digunakan secara gratis. Meskipun demikian, ada juga software MySQL yang bersifat komersil. Biasanya yang sudah ditambahi dengan kemampuan spesifik dan mendapat pelayanan dari technical support.

5. Proyeksi Data yang Handal

Perlindungan terhadap keamanan data merupakan hal nomor satu yang dilakukan oleh para profesional di bidang database. MySQL menyediakan mekanisme yang powerful untuk menangani hal tersebut, yaitu dengan menyediakan fasilitas manajemen user, enkripsi data, dan lain sebagainya.

6. Komunitas Luas

Karena penggunanya banyak maka MySQL memiliki komunitas yang luas. Hal ini berguna jika kita menemui suatu permasalahan dalam proses pengolahan data menggunakan MySQL. Dengan mengikuti salah satu atau beberapa komunitas tertentu, kita dapat menanyakan atau mendiskusikan permasalahan tersebut melalui forum. Harapannya adalah solusi akan permasalahan tersebut akan cepat diperoleh.

2.1.15.2 Bahasa *MySQL*

Seperti halnya SQL engine yang lain, Perintah dalam SQL diklasifikasikan menjadi lima bagian besar, yaitu [17]:

a. *Data Defenition Language* (DDL), Kumpulan perintah yang berfungsi untuk mendefinisikan *database*, struktur tabel, indeks, dan objek-objek lain dalam

database. Perintah-perintah yang digolongkan dalam DDL adalah *create*, *alter* dan *drop*.

b. *Data Manipulation Language* (DML), Kumpulan perintah yang digunakan untuk melakukan manipulasi data di dalam tabel. Perintah-perintah yang digolongkan dalam DML adalah *update*, *insert*, dan *delete*.
c. *Data Control Language* (DCL), Kumpulan perintah yang digunakan untuk mengontrol (memberi/mencabut) hak akses ke/dari seorang user. Dua perintah utama dalam DCL adalah grant dan *revoke*. *Grant* digunakan untuk memberikan hak akses kepada user tertentu, sedangkan *revoke* mencabut hak akses dari user tertentu dalam database.
d. *Transactional Control Language* (TCL), Kumpulan perintah yang digunakan untuk mengontrol (menyimpan/membatalkan) transaksi data. Perintah-perintah yang digolongkan dalam TCL adalah *commit* dan *rollback*. *Commit* digunakan untuk menyimpan perubahan-perubahan yang dilakukan terhadap database secara permanen, sedangkan *rollback* digunakan untuk membatalkan transaksi atau perubahan-perubahan yang telah dilakukan ke dalam database.
e. *Data Query Language* (DQL), Kumpulan perintah yang digunakan untuk melakukan seleksi data.

2.1.16 HTTPS (Hypertext Transfer Protocol)

HTTPS (*Hypertext Transfer Protocol Secure*) adalah protokol komunikasi internet yang melindungi integritas dan kerahasiaan data pengguna antara komputer pengguna dan situs. Data yang dikirim menggunakan HTTPS diamankan melalui protokol *Transport Layer Security* (TLS), yang memberikan tiga lapis perlindungan kunci [18] :

1. Enkripsi: mengenkripsi data pertukaran untuk menjaga keamanannya dari penyadap. Artinya, saat pengguna menjelajahi situs web, tidak ada yang dapat "menguping" percakapan, melacak aktivitas di berbagai laman, atau mencuri informasi mereka.
2. Integritas data: data tidak dapat diubah atau dirusak selama transfer, dengan sengaja atau tidak, tanpa terdeteksi.

3. Autentikasi: membuktikan bahwa pengguna Anda berkomunikasi dengan situs web yang diinginkan. Hal tersebut melindungi dari serangan *Man In The Middle* (MITM) dan membangun kepercayaan pengguna, yang dapat memberikan keuntungan lain untuk bisnis Anda.

2.1.16.1 Fungsi dan Tujuan HTTPS

Fungsi khusus HTTPS adalah untuk menjaga kemanan data dari hacker yang berniat membajak sebuah dokumen secara ilegal. Sedangkan tujuan HTTPS adalah untuk mengantisipasi terjadinya eror program sebuah data dan kerusakan server akibat kegagalan koneksi yang disebabkan perusahaan yang di buat oleh pakar hacker yang tidak bertanggung jawab [18].

2.1.16.2 Kelebihan dan Kelemahan HTTPS

a. Kelebihan HTTPS :

Beberapa kelebihan HTTPS yang membuatnya masih digunakan hingga saat ini :

1. Jalur yang digunakan untuk transfer data sangat aman karena terenkripsi.
2. Sertifikat yang didapatkan dari badan pemberi sertifikat dijamin keamanannya oleh badan tersebut.
3. Dapat menangkal serangan *eavesdroppers*, dan *man in the middle attacks*.
4. Banyak situs-situs besar yang menggunakan metode ini, sehingga komunitas pengguna HTTPS cukup banyak.

b. Kelemahan HTTPS :

Beberapa kelemahan yang dimiliki dari menggunakan metode HTTPS, adalah sebagai berikut :

1. Tidak semua *web browser* mempunyai kemampuan untuk menginstall sertifikat dengan baik. Oleh karena itu sertifikat harus mempunyai *browser recognition rate*.
2. Kerja komputer semakin tinggi karena melibatkan enkripsi, maka *client* dan *server* membutuhkan waktu, *resource*, dan *bandwidth* lebih untuk melakukan kegiatannya.
3. Kebutuhan *bandwidth* bertambah karena data yang dikirim lebih besar daripada data yang tidak dienkripsi.

4. Proses mendapatkan sertifikat yang cukup rumit, sehingga membuat para pembuat situs individu sulit untuk membuat sertifikat yang sesuai dengan kerja dari situsnya.

2.1.17 OOAD (Object Oriented Analysis and Design)

Konsep OOAD mencakup analisis dan desain sebuah sistem dengan pendekatan objek, yaitu analisis berorientasi objek (OOA) dan desain berorientasi objek (OOD). OOA adalah metode analisis yang memeriksa requirement (syarat/keperluan) yang harus dipenuhi sebuah sistem dari sudut pandang kelas-kelas dan objek-objek yang ditemui dalam ruang lingkup sistem. Sedangkan OOD adalah metode untuk mengarahkan arsitektur *software* yang didasarkan pada manipulasi objek-objek sistem atau subsistem [19].

2.1.18 OOP (Object Oriented Programming)

Secara spesifik, pengertian berorientasi objek berarti bahwa mengorganisasi perangkat lunak sebagai kumpulan dari objek tertentu yang memiliki struktur data dan perilakunya. Hal ini yang membedakan dengan pemograman konvensional dimana struktur data dan perilaku hanya berhubungan secara terpisah. Terdapat beberapa cara untuk menentukan karakteristik dalam pendekatan berorientasi objek, tetapi secara umum mencakup empat hal, yaitu identifikasi, klasifikasi, *polymorphism* (polimorfisme) dan *inheritance* (pewarisan) [20].

2.1.18.1 Konsep Dasar Berorientasi Object

Pengembangan sistem berorientasi objek berbeda dari pengembangan konvensional yang memandang perangkat lunak sebagai fungsi dan data yang terisolasi. Program perangkat lunak adalah sekumpulan mekanisme yang melakukan aksi-aksi tertentu pada data tertentu. Pada pendekatan konvensional, kebanyaan berfokus pada fungsi. Namun juga terdapat pendekatan yang fokus pada data terutama pada kubu basis data dan pemodelan informasi. Sementara itu, pandangan berorientasi objek berpusat pada objek yang mengkombinasikan data dan fungsionalitas. Berikut ini adalah beberapa kriteria-kriteria penting pendekatan objek yang harus dipahami tentang metodologi berorientasi objek [20]:

a. *Seamlessness*

Konsep berorientasi objek dapat diterapkan di seluruh siklus hidup pengembang sistem (*System Developmnent life cycle*) dari analisis, perancangan, sampai implementasi.

b. Kelas

Pendekatan berorientasi objek berdasar pada pentingnya kelas. Kelas adalah elemen perangkat lunak yang mendeskripsikan tipe data abstrak (ADT – *Abstract data type*) dimana implementasinya dapat secara parsial atau total. Implementasi parsial bila di kelas terdapat operas/metode abstrak.

c. Assertion

Fitur tipe data abstrak harus secara formal menspesifikasikan properti kelas Assertion berperan pada prakondisi (*precondition*), poskondisi (*postcondition*), invarian kelas. *Assertion* mendeskripsikan efek fitur-fitur di objek independen terhadap implementasi fitur-fitur itu.

d. Kelas Merupakan Cara Pemodulan

Pendekatan berorientasi objek merupakan teknik pembentukan arsitektur perangkat lunak. Dampak utama penerapannya adalah pada struktur modul dari sistem perangkat lunak. Peran utamanya di sistem berorientasi objek dimainkan oleh kelas. Pada pendekatan murni berorientasi objek maka kelas merupakan satu-satunya pemodulan.

e. Kelas merupakan *type*

Kelas cukup ampuh untuk menghilangkan kebutuhan mekanisme typing yang lain. Setiap type harus didasarkan pada kelas.

f. *Exception handling*

Sistem perangkat lunak seharusnya mampu menangani kondisi abnormal secara terstruktur dengan baik.

g. *Genericity*

Agar penerapan typing menjadi praktis adalah kemampuan untuk mendefinisikan *type-parameterized* classes atau disebut generik.

h. Pewarisan

Pendekatan berorientasi objek harus memungkinkan mendefinisikan kelas yang merupakan turunan kelas yang lain. Pewarisan dapat berupa pewarisan tunggal (*single inheritance*) dan pewarisan jamak (*multiple inheritance*).

i. *Redefinition*

Kita mungkin perlu mengubah implementasi atau properti lain dari fitur-fitur yang diwariskan. Redefinisi dapat mempengaruhi fitur, penanda (tipe argumen dan hasil) dan spesifikasi.

j. *Polymorphism*

Dimungkinkan untuk entitas ditautkan ke objek-objek beragam tipe. *Polymorphism* dikendalikan oleh pewarisan.

k. *Dynamic binding*

Dynamic binding mempunyai pengaruh utama pada aplikasi berorientasi objek. Mekanisme ini memungkinkan pengembang menulis panggilan secara sederhana untuk beberapa panggilan yang dapat dimungkinkan, bergantung pada situasi saat sebenarnnya panggilan itu dieksekusi.

l. *Run-time type interogation*

Perangkat lunak memerlukan mekanisme untuk mengakses objek dengan cara yang aman tanpa melanggar konstrain *static typing*. Mekanisme ini harus dirancang agar tidak meniadakan manfaat-manfaat dari *dynamic binding* dan *polymorphism*.

2.1.19 UML (Unified Modelling Language)

Unified Modelling Language (UML) adalah bahasa grafis untuk mendokumentasi, menspesifikasikan, dan membangun sistem perangkat lunak. UML berorientasi objek, menerapkan banyak level abstraksi, tidak bergantung proses pengembangan, tidak bergantung bahasa dan teknologi, pemaduan beberapa notasi di beragam metodologi, usaha bersama dari banyak pihak, didukung oleh kakas-kakas yang diintegrasikan lewat XML (XMI). Standar UML dikelola oleh OMG (*Object Management Group*) [20].

UML adalah bahasa pemodelan untuk menspesifikasikan, memvisualisasikan, membangun dan mendokumentasikan artifak-artifak dari sistem. Notasi UML terutama diturunkan dari 3 notasi yang telah ada sebelumnya yaitu : *Grady Booch*

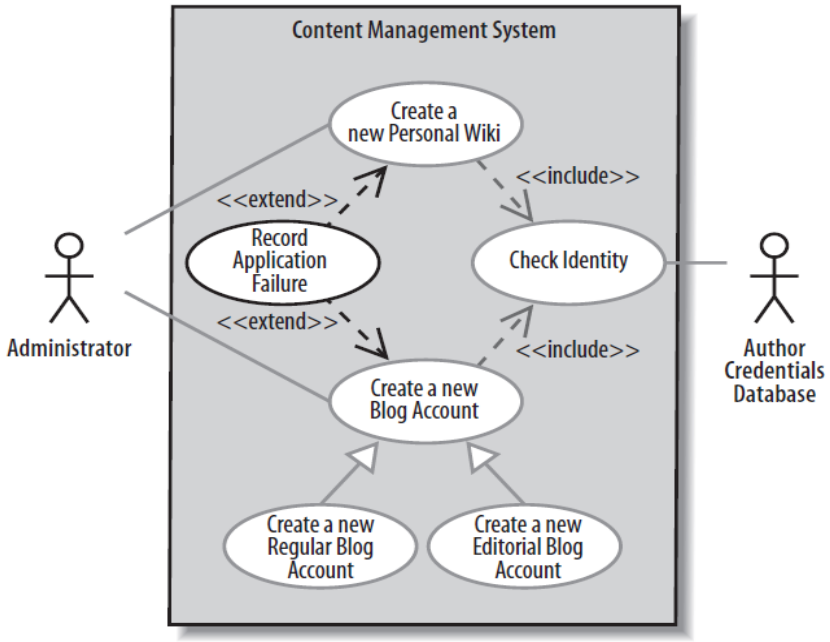
OOD (*Object Oriented Design*), James Rumbaugh OMT (*Object Modeling Technique*) dan Ivar Jacobson OOSE (*Object Oriented Software Engineering*).

2.1.19.1 *Use Case* Diagram

Use case Diagram merupakan salah satu diagram untuk memodelkan aspek perilakau sistem. Masing-masing diagram *use case* menunjukkan sekumpulan *use case*, aktor, dan hubungannya. Diagram *use case* adalah penting untuk memvisualisasikan, menspesifikasikan, dan mendokumentasikan kebutuhan perilaku sistem. Diagram-diagram *use case* merupakan pusat pemodelan perilaku sistem, subsistem, dan kelas. Diagram *use case* digunakan untuk mendeskripsikan apa yang seharusnya dilakukan oleh sistem. Ada empat hal utama pada *use case* yaitu pendefinisian apa yang disebut aktor dan use case [20] :

1. Sistem yaitu sesuatu yang hendak kita bangun.
2. Relasi adalah relasi antara aktor dengan *use case*.
3. Aktor merupakan orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang.
4. *Use case* merupakan fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor.

   Berikut Contoh *Use Case* Diagram dapat dilihat pada Gambar 2.7.



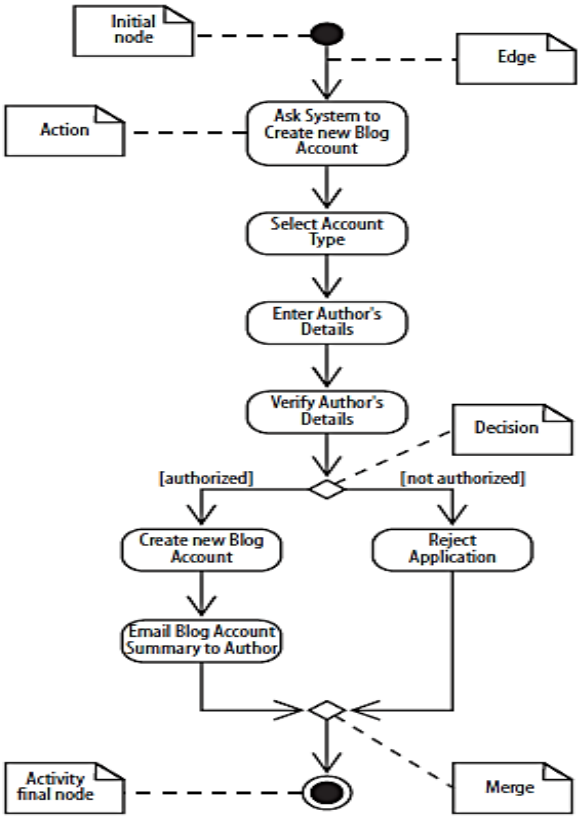
Gambar 2.7 Contoh *Use Case* Diagram

2.1.19.2 *Activity* Diagram

Diagram aktivitas adalah diagram *flowchart* yang diperluas yang menunjukan aliran kendali satu aktivitas ke aktivitas lain di sistem. Diagram aktivitas ini digunakan untuk memodelkan aspek dinamis sistem. Diagram aktivitas mendeskripsikan aksi-aksi dan hasilnya. Diagram aktivitas juga banyak digunakan untuk mendefinisikan hal-hal berikut [20]:

1. Rancangan proses bisnis dimana setiap urutan aktivitas yang digambarkan merupakan proses bisnis sistem yang didefinisikan.
2. Urutan atau pengelompokkan tampilan dari sistem/*user interface* dimana setiap aktivitas dianggap memiliki sebuah rancangan antarmuka tampilan.
3. Rancangan pengujian dimana setiap aktivitas dianggap memerlukan sebuah pengujian yang perlu didefinisikan kasus ujinya.
4. Rancangan menu yang ditampilkan pada perangkat lunak.

Berikut Contoh *Activity* Diagram dapat dilihat pada Gambar 2.8.

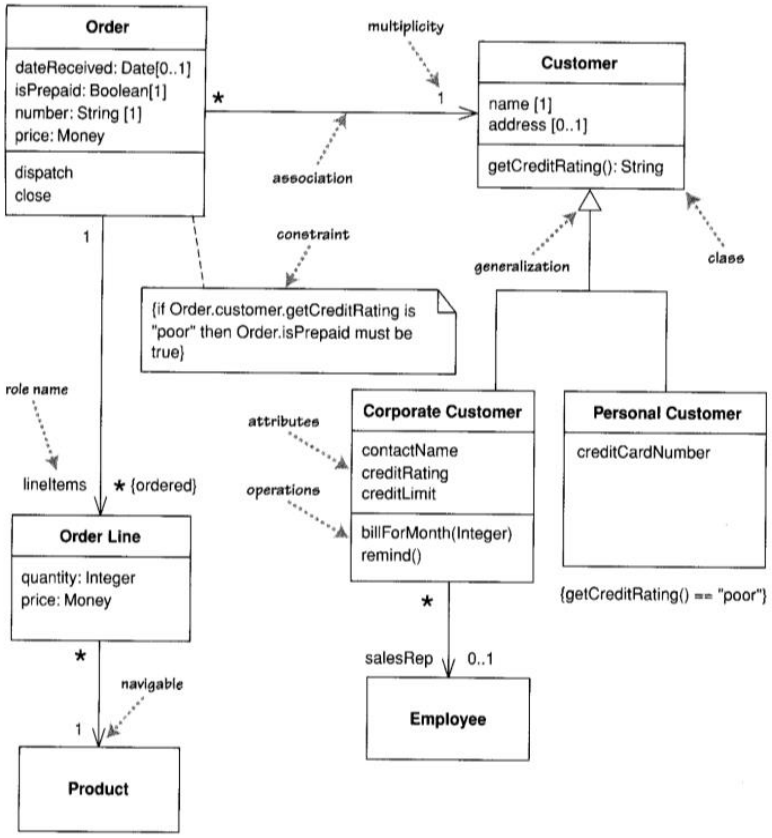


Gambar 2.8 Contoh *Activity* Diagram

2.1.19.3 *Class* Diagram

Class diagram merupakan diagram yang selalu ada di pemodelan system berorientasi objek. *Class* diagram menunjukkan hubungan antar class dalam system yang sedang dibangun dan bagaimana mereka saling berkolaborasi untuk mencapai satu tujuan. Kelas pada kelas diagram terdiri dari 3 bagian utama yaitu nama kelas, isi *property* dari kelas beserta metode yang ada pada kelas tersebut. Kelas juga memiliki jenis-jenis hubungan seperti asosiatif, dependensi, agregasi, komposisi, spesifikasi dan generalisasi. Hubungan ini digunakan untuk menggambarkan bagaimana hubungan dan interaksi yang terjadi antar kelas. Masing-masing komponen penyusun kelas memiliki hak akses seperti *public*, *private* dan *protected* [20].

Berikut contoh *Class* Diagram dapat dilihat pada Gambar 2.9.

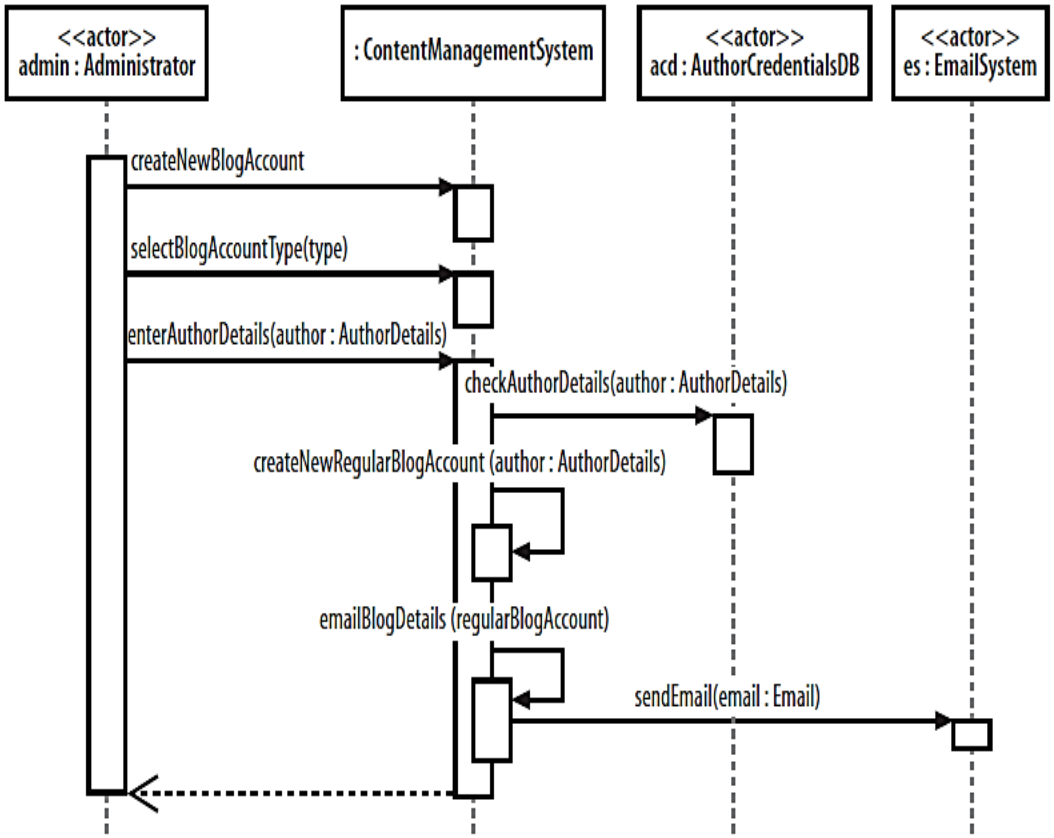


Gambar 2.9 Contoh Class Diagram

2.1.19.4 *Sequence* Diagram

Sequence Diagram menunjukan interaksi yang terjadi antar objek. Diagram ini merupakan pandangan dinamis terhadap sistem. Diagram ini menekankan pada sisi basis keberurutan waktu dari pesan-pesan yang terjadi. Diagram sekuen menunjukan objek sebagai garis vertikal dan tiap kejadian sebagai panah horisontal dari objek pengirim ke objek penerima. Waktu berlalu dari atas ke bawah dengan lama waktu tidak relevan. Diagram ini hanya menunjukan barisan kejadian, bukan perwaktuan nyata. Kecuali untuk sistem waktu nyata yang mengharuskan konstrain barisan terjadi [20].

Berikut Contoh *sequence* diagram dapat dilihat pada gambar 2.10.



Gambar 2.10 Contoh Sequence Diagram