BAB 2
LANDASAN TEORI

BAB 1

2.1 Tinjauan Perusahaan

Pada tinjauan perusahaan ini akan dibahas mengenai Profil perusahaan, visi misi, motto, serta penghargaan dan pencapaian yang diperoleh perusahaan.

2.1.1 Profil Perusahaan

JG Motor adalah dealer resmi Yamaha dengan badan hukum PT Jayamandiri Gemasejati. Bergerak dibidang penjualan retail kendaraan otomotif khususnya motor dengan merk Yamaha. Didirikan pada tanggal 29 Oktober 1994 berdasarkan Akta Pendirian No. 256 tanggal 29/10/1994 secara operasional menjalankan bisnis retail otomotif Yamaha pada bulan September 2001 dengan menggunakan nama JG Motor Group.

Nama JG itu sendiri berasal dari penyingkatan nama Jayamandiri Gemasejati. Cabang pertama yang dibuka adalah JG Cibeureum yang berlokasi di Jl. Raya Cibeureum No. 39B Cimahi-Bandung dengan status dealer 1S, dimana hanya melayani penjualan unit saja tanpa bengkel dan *spareparts*. Jumlah karyawan pada saat itu hanya 8 orang saja. Hingga bulan Juni tahun 2011 JG Motor Group telah memiliki 27 cabang dengan status 3S (Sales, *Service* dan *Spareparts*), 2 cabang dengan status 2S (*Service* dan *Spareparts*) dan 1 cabang dengan status 1S (Sales) yang tersebar di wilayah DKI dan Jawa Barat dengan total jumlah karyawan tercatat sebanyak 1.083 karyawan.

2.1.2 Visi, Misi, dan Motto Perusahaan

1. Visi: Menjadi perusahaan terkemuka dalam jaringan retail otomotif.

2. Misi: Memiliki jaringan retail yang kuat dan menguntungkan, serta operasional yang handal.
3. Motto: Motto JG Motor yang terkenal adalah "JG Jagonya Yamaha, Yamaha Jagonya Motor" Diciptakan pada tahun 2006 merupakan cerminan langsung dari visi perusahaan untuk membawa JG Motor Group menjadi yang terkemuka dalam bisnis retail otomotif khususnya motor merk Yamaha. Sampai saat ini JG Motor terus berusaha mengembangkan pelayanan untuk penjualan dan servis sepeda motor Yamaha. Baik dalam hal penjualan, maupun dalam hal layanan purna jual. Guna memberikan layanan yang terbaik bagi konsumennya dari waktu ke waktu.

Dari waktu ke waktu, senantiasa ada tantangan yang harus dilalui. Terutama pada saat memasuki tahun 2009, ada banyak tantangan yang terbentang didepan. Namun, apapun tantangan yang akan dihadapi tetap menjadi komitmen untuk bisa memberikan layanan yang terbaik. Dan tentunya, semuanya itu tidak terlepas dari suatu upaya pembinaan yang dilakukan secara kontinu dan konsisten, hingga didapat suatu hal yang lebih baik dari waktu ke waktu.

Dengan demikian, moto menjadi suatu titik awal cerminan segala komitmen JG Motor untuk senantiasa mengembangkan diri dan memberikan yang terbaik. Sehingga, JG boleh menjadi idiom dan kata baru yang dikenal umum yang senantiasa menjadi pengingat bahwa *"JG… is Just Great!"*.

2.1.3 Cabang JG Motor

JG Motor memiliki jaringan bengkel yang tersebar luas di wilayah Jawa Barat dan JABODETABEK. Untuk cabang area Bandung diantaranya sebagai berikut:

1. JG MOTOR ASIA-AFRIKA
   Jl. Asia Afrika No.172, Bandung.
2. JG MOTOR KOPO
   Jl. Kopo No. 601, Bandung.
3. JG MOTOR BANDUNG

Jl. BKR No. 5, Lingkar Selatan-Bandung.

4. JG MOTOR CIBEUREUM
   Jl. Raya Cibeureum No. 70, Cimahi.
5. JG MOTOR UJUNG BERUNG
   Jl. Raya Ujungberung No. 228, Bandung.
6. JG MOTOR CIWASTRA
   Jl. Ciwastra No. 151, Ds. Margasari, Kec. Margacinta, Bandung.

2.1.4 Penghargaan dan Pencapaian

Dalam perjalanannya perusahaan JG Motor mendapatkan penghargaan yang sejalan dengan pencapaian prestasinya. Penghargaan dan pencapaian tersebut dintaranya sebagai berikut:

1. *The Best National* 3S Dealer 2008 (JG Motor Cibeureum).
2. Peraih *Champion* BSEA (Bandung *Service Excellence Award*) 2009 Kategori Otomotif.
3. Pemenang Yamaha *Frontliner Championship* 2009.
4. Lola Husniati – JG Cirebon
5. Heri Mulyanto – JG Siliwangi
6. Peraih President Award
7. JG Asia Afrika – Best Dealer 2007
8. JG Cianjur – Best Dealer 2009
9. Peraih *Champion* BSEA (Bandung *Service Excellence Award*) 2010 Kategori Otomotif.

2.1.5 Klasifikasi Kendaraan

Kendaraan yang dapat diservis dan tersedia sparepart di seluruh cabang YSS JG Motor adalah semua jenis kendaraan Yamaha dengan tipe Moped, *Matic*, Premium, dan *Sport*[CITATION JGM18 \l 1057]. Diantaranya sebagai berikut:

1. Tipe Moped

1. Jupiter Z1
2. Jupiter MX King 150

3. Vega Force

2. Tipe Matic

1. Fino Grande 123
2. Fino Premium 125
3. Mio M3 123
4. Mio M3 125 AKS SSS
5. Mio S
6. Mio Soul GT AKS
7. Mio Z

3. Tipe Premium

1. Aerox 155
2. Aerox 155 R
3. Lexi
4. Lexi S
5. NMAX 155
6. NMAX 155 ABS
7. XMAX

4. Tipe Sport

1. Vixion
2. R15
3. R25
4. R25 ABS

2.1.6 Rekomendasi Pergantian Sparepart

Perawatan sepeda motor terutama yang berbasis *Fuel Injection* (FI), haruslah dilakukan secara rutin. Sepeda motor jenis ini dirancang sedemikian rupa untuk memberikan kemudahan transportasi. Pemilik haruslah peka serta teliti, tidak hanya menggunakannya saja. Sebab, ada masa tertentu yang harus diketahui untuk segera mengganti part (suku cadang). Tujuannya agar Vixion, Mio J, Jupiter Z1 atau motor Yamaha lainnya, dapat dikendarai dengan baik, aman sehingga

memberikan rasa nyaman ketika berkendara untuk menempuh setiap tempat tujuan.

Rekomendasi penggantian suku cadang di YSS Yamaha JG Motor didasari oleh 2 faktor yang menentukan bahwa sparepart harus diganti. Faktor tersebut diantaranya:

1. Berdasarkan pengecekan servis yang telah dilakukan
2. Berdasarkan jumlah jarak tempuh KM kendaraan yang memungkinkan terjadinya keausan.

Berikut ini tabel perawatan dan penggantian suku cadang berdasarkan jarak tempuh, dilengkapi informasi gejala-gejala yang dapat ditimbulkannya:

Tabel.2.1 Rekomendasi Penggantian Sparepart

Komponen	**Kilometer Disarankan untuk Penggantian**	**Akibat yang Timbul jika Tidak Dilakukan Penggantian**
Busi	6.000 km	Performa dan tenaga mesin turun· Gejala panas berlebihan· Bahan bakar boros
Filter Oli	9.000 km	Pelumasan mesin tidak optimal, komponen mesin cepat aus· Kualitas oli turun, mesin mudah panas
Weight (*Roller*)	24.000 km	Bahan bakar boros. Performa dan tenaga mesin turun. Timbul suara kasar (*noise*)
Kanvas Kopling	24.000 km	Bahan bakar boros. Saat akselerasi tenaga kurang. Timbul panas berlebihan
Filter Udara	9.000 km	Bahan bakar boros. Performa dan tenaga mesin turun. Debu masuk ke ruang bakar, timbul suara kasar, komponen mesin aus/rusak
Kanvas Rem	12.000 km	Piringan cakaram atau teromol aus. Timbul suara berderit, kemampuan pengereman berkurang
V-Belt	25.000 km	Bahan bakar boros.

		Performa dan tenaga mesin turun. Timbul suara kasar (*noise*). V-Belt bias putus, komponen Pulley rusak
Gear dan Rantai	15.000 km	Bahan bakar boros. Performa dan tenaga mesin turun. Timbul suara kasar (*noise*). Rantai bias putus, rawan kecelakaan

Sumber Tabel : "https://jgmotor.co.id/2012/11/26/tabel-perawatan-dan-penggantian-suku-cadang-berdasarkan-jarak-tempuh/"[CITATION JGM18 \l 1057]

2.2 Landasan Teori

Landasan Teori merupakan definisi, konsep yang telah disusun secara sistematis dan dasar yang kuat dalam sebuah penelitian. Landasan teori yang digunakan dalam penyusunan Pembangunan Aplikasi YSS Yamaha JG Motor Area Bandung.

2.2.1 Pengertian Service

Kata service dalam kamus bahasa inggris yaitu memperbaiki, penulis membuat kesimpulan bahwa memperbaiki disini adalah membuat sepeda motor yang tadinya rusak atau ada salah satu bagian motor yang tidak berfungsi menjadi baik atau berfungsi kembali seluruh bagian-bagian yang rusak tadi.

2.2.2 Pengertian Sparepart

Sparepart diterjemahkan kedalam bahasa Indonesia yaitu sukucadang, penulis juga mempunyai kesimpulan bahwa sukucadang disini adalah komponen-komponen yang ada pada sepeda motor.

2.2.3 Pengertian Aplikasi

Menurut Buyens[CITATION Jim01 \l 1057] aplikasi adalah satu unit perangkat lunak yang dibuat untuk melayani kebutuhan akan beberapa aktivitas.

Jika ingin mengembangkan program aplikasi sendiri, maka untuk menulis program aplikasi tersebut, dibutuhkan suatu bahasa pemrograman, yaitu language software, yang dapat berbentuk *assembler*, *compiler* ataupun *interpreter*. Jadi *language software* merupakan bahasanya dan program yang ditulis merupakan program aplikasinya. *Language software* berfungsi agar dapat menulis program dengan bahasa yang lebih mudah, dan akan menterjemahkannya kedalam bahasa mesin supaya bisa dimengerti oleh komputer.

Apabila hendak mengembangkan suatu program aplikasi untuk memecahkan permasalahan yang besar dan rumit, maka supaya program aplikasi tersebut dapat berhasil dengan baik, dibutuhkan prosedur dan perencanaan yang baik dalam mengembangkannya. Sekarang, banyak sekali program-program aplikasi yang tersedia dalam bentuk paket-paket program. Yaitu program-program aplikasi yang sudah ditulis oleh orang lain atau perusahaan-perusahaan perangkat lunak.

Beberapa perusahaan perangkat lunak telah memproduksi paket-paket perangkat lunak yang mempunyai reputasi internasional. Program-program paket tersebut dapat diandalkan, dapat memenhi kebutuhan pemakai, dirancang dengan baik, relatif bebas dari kesalahan-kesalahan, *user friendly* (mudah digunakan), mempunyai dokumentasi manual yang memadai, mampu dikembangkan untuk kebutuhan mendatang, dan didukung perkembangannya. Akan tetapi, bila permasalahannya bersifat khusus dan unik, sehingga tidak ada paket-paket program yang sesuai untuk digunakan, maka dengan terpaksa harus mengembangkan program aplikasi itu sendiri.

2.2.4 Mobile

Mobile diartikan sebagai perpindahan yang mudah dari satu tempat ke tempat yang lain, misalnya telepon *mobile* berarti bahwa terminal telepon yang dapat berpindah dengan mudah dari satu tempat ke tempat lain tanpa terjadi pemutusan atau terputusnya komunikasi.

2.2.5 Aplikasi Mobile

Menurut Buyens [CITATION Jim01 \l 1057] aplikasi *mobile* berasal dari kata *application* dan *mobile*. *Application* yang artinya penerapan, lamaran, penggunaan. Secara istilah aplikasi adalah program siap pakai yang direka untuk melaksanakan suatu fungsi bagi pengguna atau aplikasi yang lain dan dapat digunakan oleh sasaran yang dituju sedangkan *mobile* dapat di artikan sebagai perpindahan dari suatu tempat ke tempat yang lain. Maka aplikasi mobile dapat di artikan sebuah program aplikasi yang dapat dijalankan atau digunakan walaupun pengguna berpindah-pindah dari satu tempat ke tempat yang lain serta mempunyai ukuran yang kecil. Aplikasi mobile ini dapat di akses melalui perangkat nirkabel, pager, PDA, telepon seluler, *smartphone*, dan perangkat sejenisnya.

2.2.6 Android

Android adalah sebuah sistem operasi untuk perangkat mobile berbasis linux yang mencakup sistem operasi, *middleware* dan aplikasi. Android menyediakan *platform* yang terbuka bagi para pengembang untuk menciptakan aplikasi mereka. Awalnya, Google Inc. membeli Android Inc. yang merupakan pendatang baru yang membuat peranti lunak untuk ponsel/*smartphone*. Kemudian untuk mengembangkan Android, dibentuklah Open Handset Aliance yang merupakan konsorsium dari 34 perusahaan peranti keras, peranti lunak, dan telekomunikasi, termasuk Google, HTC, Intel, Motorola, Qualcomm, T-Mobile, dan Nvidia. Pada saat perilisan perdana Android, 5 November 2007, Android bersama Open Handset Alliance menyatakan mendukung pengembangan *open source* pada perangkat *mobile*.[CITATION Saf11 \l 1057]

2.2.6.1 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.

2.2.6.2 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:

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

2.2.6.3 Versi Android

Berikut merupakan perkembangan dari android dari versi 1.1 hingga sekarang 8.0 android oreo:

1. Android Versi 1.1
2. Android Versi 1.5 (*Cupcake*)
3. Android Versi 1.6 (*Donut*)
4. Android Versi 2.0/2.1 (*Éclair*)
5. Android Versi 2.2 (*Froyo*)
6. Android Versi 2.3 (*Gingerbread*)
7. Android Versi 3.0 (*Honeycomb*)
8. Android Versi 4.0 (*Ice Cream Sandwich*)
9. Android Versi 4.1 (*Jelly Bean*)
10. Android Versi 4.4 (*KitKat*)
11. Android Versi 5.0 (*Lollipop*)
12. Android Versi 6.0 (*Marshmallow*)
13. Android Versi 7.0 (*Nougat*)

14. Android Versi 7.1 (*Nougat*)
15. Android Versi 8.0 (*Oreo*)

2.2.7 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[CITATION Jav18 \l 1057].

2.2.7.1 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 kelas penanggalan.
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 pengguna (GUI)
6. java.applet : Kelas dasar aplikasi antar muka untuk diterapkan pada penjelajah web.

2.2.7.2 Kelebihan Java

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*)
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).

2.2.7.3 Kekurangan Java

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.

2.2.8 Web Services

Web service adalah sistem perangkat lunak yang menyediakan interaksi program ke program lain melalui jaringan *website*. Format standar dalam pendistribusian data dari satu perangkat ke perangkat lain menggunakan XML dan JSON. *Web service* menggunakan internet sebagai jalur pendistribusian. Pemanggilan *webservice* dengan cara memanggil di setiap fungsinya yang ada di *webservice*. Dalam satu *webservice* memiliki banyak fungsi. Jalur disetiap fungsinya disebut *Application Programing Interface* (API). *Webservice* pada umumnya menggunakan *Restfull* API yang berarti seluruh fungsi hanya dapat diakses menggunakan API yang telah dibuat. Dengan menggunakan *webservice*

yang terhubung dengan database yang ada dengan membuat API yang mana akan menghasilkan suatu JSON untuk diimplementasikan di aplikasi android. Sehingga pertukaran data dapat dilakukan dan dapat diketahui informasi data secara real-time.

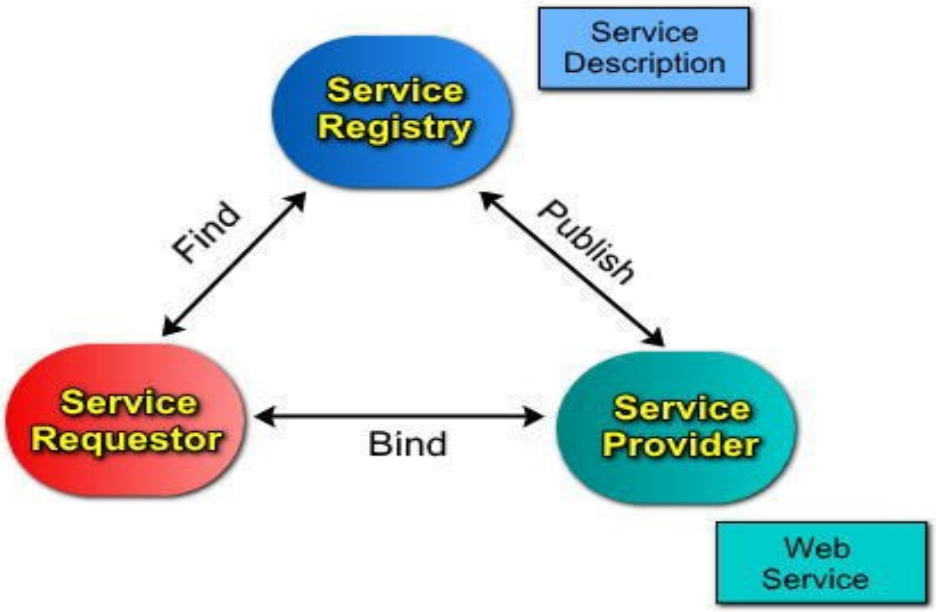
Beberapa karakteristik dari web service adalah:

1. *Message-based*
2. *Standards-based*
3. *Programming language independent*
4. *Platform-neutral*

Beberapa *key standard* di dalam web service adalah: JSON, XML, SOAP, WSDL and UDDI.[CITATION Ave07 \l 1057]

2.2.8.1 Arsitektur Web service

Berikut ini gambar arsitektur *web service* yang merupakan komponen *web service* dimana membuat *web servis* bisa berjalan.



Sumber gambar: Web Services Architecture, [Online]. Available: http://www.w3.org/TR/2004/NOTE-ws-arch-20040211/#whatis [CITATION Web18 \l 1057]

Gambar 2.1 Arsitektur Web Service

Pada gambar diatas, ada tiga komponen yang membuat web service berjalan. Ketiga komponen itu adalah[CITATION Web18 \l 1057]:

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.

2.2.9 API (Application Programming Interface)

Application programming interface (API) merupakan suatu dokumentasi yang terdiri dari interface, fungsi, kelas, struktur dan sebagainya untuk membangun sebuah perangkat lunak. Dengan adanya API ini, maka memudahkan programmer untuk "membongkar" suatu software, kemudian dapat dikembangkan atau diintegrasikan dengan perangkat lunak yang lain. API dapat dikatakan sebagai penghubung suatu aplikasi dengan aplikasi lainnya yang memungkinkan programmer menggunakan sistem *function*. Proses ini dikelola melalui sistem operasi. Keunggulan dari API ini adalah memungkinkan suatu aplikasi dengan aplikasi lainnya dapat saling berhubungan dan berinteraksi. [CITATION Blo \l 1057]

2.2.10 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

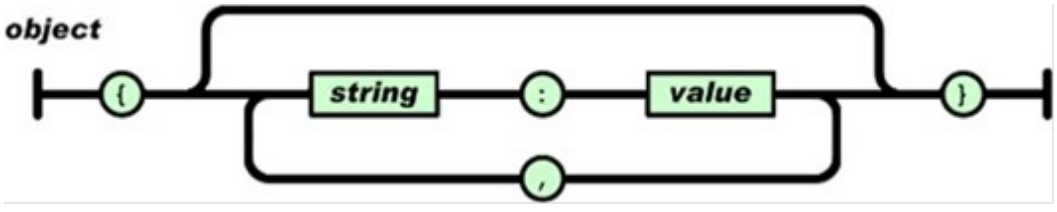
1999. JSON merupakan format teks yang tidak bergantung pada bahasa pemrograman apapun karena menggunakan gaya bahasa yang umum digunakan oleh programmer keluarga C termasuk C, C++, C#, Java, JavaScript, Perl, Python. Oleh karena sifat-sifat tersebut, menjadikan JSON ideal sebagai bahasa pertukaran-data. 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*).

Struktur-struktur data ini disebut sebagai struktur data universal. Pada dasarnya, semua bahasa pemprograman modern 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. JSON menggunakan bentuk sebagai berikut:

1. Objek

   Objek adalah sepasang nama / nilai yang tidak terurutkan. Objek dimulai dengan { (kurung kurawal buka) dan diakhiri dengan } (kurung kurawal tutup). Setiap nama diikuti dengan : (titik dua) dan setiap pasangan nama/nilai dipisahkan oleh , (koma). Objek biasanya digunakan untuk menyimpan data tunggal dalam bentuk JSON. Dapat dilihat pada Gambar 2 .2 Objek JSON.

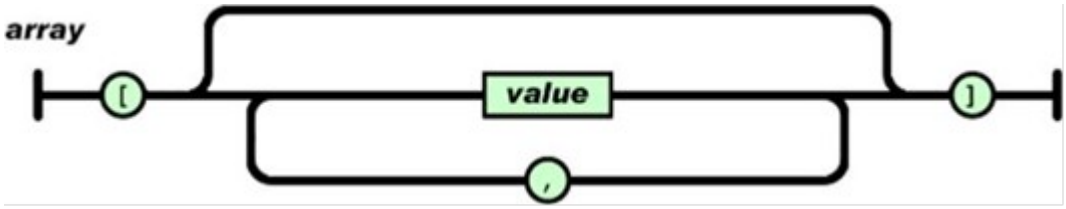


Sumber gambar: Adam Mucharil Bachtiar, Boby Indra Pratama "[CITATION Bac \l 1057]*"*

Gambar 2.2 Objek JSON

2. Larik

   Larik adalah kumpulan nilai yang terurutkan. Larik dimulai dengan [(kurung kotak buka) dan diakhiri dengan] (kurung kotak tutup). Setiap nilai dipisahkan oleh , (koma). Larik dalam JSON dapat digunakan sebagai value dari JSON object hal ini dapat berguna jika JSON menyimpan data bertingkat. Dapat dililihat pada gambar Gambar 2 .3 Array JSON.



Sumber gambar: Adam Mucharil Bachtiar, Boby Indra Pratama "[CITATION Bac \l 1057]*"*

Gambar 2.3 Array JSON

Bentuk data JSON objek dan larik dapat saling dikombinasikan untuk mendukung struktur data yang lebih kompleks. JSON mendukung beberapa tipe data untuk menjadi value seperti Angka, String, Bollean dan nilai NULL.

Berikut adalah deskripsi Arsitektur Perangkat Lunak Pada Platform Android:

1. Perangkat *mobile* pengguna melakukan *request* data ke server melalui API.
2. Server menerima request data dan menentukan jenis *request* yang diminta.
3. Jika server menerima permintaan lokasi maka permintaan data akan diteruskan ke server google place.
4. Jika server menerima permintaan data gambar maka permintaan data akan diarahkan ke server.
5. Jika server menerima permintaan data text maka server akan langsung mengambil data yang ada di database.
6. Setelah server menerima data yang diminta data tersebut akan dikembalikan dalam bentuk JSON untuk diproses perangkat mobile pengguna.

2.2.11 Firebase

Firebase adalah penyedia layanan cloud dengan backend sebagai servis yang berdiri sejak April 2012 yang berbasis di San Fransisco, California. *Firebase* menyediakan *realtime* database dan *backend* sebagai layanan yang memungkinkan pengembangan API untuk disinkronisasikan ke client. Layanan *firebase* diantaranya:

1. *Firebase Cloud Messaging*
2. *Firebase Authentication*
3. *Firebase Real Time Database*.

Firebase memiliki banyak *library* yang memungkinkan untuk mengitegrasikan layanan ini dengan Android, iOS, Javascript, Java, Objective-C dan Node.JS.

2.2.11.1 Firebase Cloud Messaging (FCM)

Firebase Cloud Messaging (FCM) adalah salah satu fitur messaging yang disediakan Firebase. Firebase Cloud Messaging (FCM) adalah penerus dari Google Cloud Messaging yang digunakan untuk memberikan pesan secara *crossplatform* dan tanpa biaya. Sehingga dengan Firebase Cloud Messaging (FCM) dapat dengan mudah memberikan pesan berupa notifikasi kepada user ketika ada update sistem, sampai mengirimkan update informasi.

Aplikasi yang ter-instal pada smartphone pengguna akan secara otomatis meregistrasi *smartphone* android ke dalam FCM. Selanjutnya FCM akan mengirim token ke smartphone pengguna sebagai penanda, token yang dikirim bersifat *unique* sehingga berbeda-beda pada tiap *smartphone*. *Smartphone* yang telah mendapat token, akan mengirimkannya ke dalam server, lalu server akan menyimpan token tersebut. Jika terjadi perubahan nilai data dari database, maka FCM akan mendeteksi perubahan nilai data tersebut. FCM akan mengetahui dari mana data dikirim dan akan dikirim melalui token yang telah terdaftar, kemudian FCM akan mengirim notifikasi ke *smartphone* pengguna, *smartphone* yang menerima notifikasi akan memanggil *broadcast receiver* kemudian *service* notifikasi akan muncul pada layar smartphone pengguna.

Contoh *syntak* untuk Firebase Cloud Messaging adalah sebagai berikut:

```
{
 "message":{
  "token":"bk3RNwTe3H0:CI2k_HHwgIpoDKCIZvvDMExUdFQ3P1...",
  "notification":{
   "title":"Portugal vs. Denmark",
   "body":"great match!"
  }
 }
}
```

2.2.11.2 Firebase Authentication

Firebase authentication adalah layanan yang diberikan oleh Firebase untuk fungsi *user membership*. Fitur-fitur yang diberikan adalah register/login dengan beberapa metode :

3. Alamat email dan *password*
4. Akun Google
5. Akun Facebook.

User melakukan login dengan salah satu metode yang didukung Firebase (email-password, Google, Facebook).

Untuk menggunakan layanan autentikasi, Harus diaktifkan dikonsol Firebase. Buka halaman Metode Masuk di bagian Autentikasi Firebase untuk mengaktifkan masuk Email/Sandi dan penyedia identitas lain yang diinginkan untuk aplikasi.

Buat metode *createAccount* baru yang mengambil alamat email dan kata sandi, validasi dan kemudian buat pengguna baru dengan metode *createUserWithEmailAndPassword*.

Contoh syntak dari firebase authentication adalah sebagai berikut:

1. Periksa status autentikasi

Deklarasikan instance `FirebaseAuth`

```
private FirebaseAuth mAuth;
```

Di metodh `onCreate()`, inisialisasi instance `FirebaseAuth`

```
// Initialize Firebase Auth
mAuth = FirebaseAuth.getInstance();
```

Saat menginisialisasi *activity*, periksa untuk melihat pengguna saat ini masuk atau tidak.

```
@Override
public void onStart() {
    super.onStart();
    // Check if user is signed in (non-null) and update UI
accordingly.
    FirebaseUser currentUser = mAuth.getCurrentUser();
    updateUI(currentUser);
}
```

2. Pendaftaran akun baru oleh user

Buat method `createAccount` yang mana akan digunakan alamat email dan kata sandi, memvalidasinya dan kemudian membuat pengguna baru dengan method `createUserWithEmailAndPassword`.

```
mAuth.createUserWithEmailAndPassword(email, password)
        .addOnCompleteListener(this, new
OnCompleteListener<AuthResult>() {
            @Override
            public void onComplete(@NonNull Task<AuthResult>
task) {
                if (task.isSuccessful()) {
                    // Sign in success, update UI with the
signed-in user's information
                    Log.d(TAG, "createUserWithEmail:success");
                    FirebaseUser user =
mAuth.getCurrentUser();
                    updateUI(user);
                } else {
                    // If sign in fails, display a message to
the user.
                    Log.w(TAG, "createUserWithEmail:failure",
```

```
task.getException());
                    Toast.makeText(EmailPasswordActivity.this,
"Authentication failed.",
                            Toast.LENGTH_SHORT).show();
                    updateUI(null);
                }

                // ...
            }
        });
```

Tambahkan formulir untuk mendaftarkan pengguna baru dengan email dan kata sandi dan panggil method baru ini ketika dikirimkan.

3. *Sign in users* yang telah melakukan pendaftaran

Buat method `signIn` baru yang mengambil alamat email dan kata sandi, validasi, dan kemudian memverifikasi pengguna dengan method `signInWithEmailAndPassword`.

```
mAuth.signInWithEmailAndPassword(email, password)
        .addOnCompleteListener(this, new
OnCompleteListener<AuthResult>() {
            @Override
            public void onComplete(@NonNull Task<AuthResult>
task) {
                if (task.isSuccessful()) {
                    // Sign in success, update UI with the
signed-in user's information
                    Log.d(TAG, "signInWithEmail:success");
                    FirebaseUser user =
mAuth.getCurrentUser();
                    updateUI(user);
                } else {
                    // If sign in fails, display a message to
the user.
                    Log.w(TAG, "signInWithEmail:failure",
task.getException());
                    Toast.makeText(EmailPasswordActivity.this,
```

```
"Authentication failed.",
                                Toast.LENGTH_SHORT).show();
                    updateUI(null);
                }

                // ...
            }
        });
```

Tambahkan formulir untuk sign pengguna dengan email dan kata sandinya dan panggil metode baru ini saat dikirim.

2.2.12 GPS (Global Positioning System)

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:

1. Waktu.

GPS *receiver* menerima informasi waktu dari jam atom yang mempunyai keakurasian sangat tinggi.

2. Lokasi.

GPS memberikan informasi lokasi dalam tiga dimensi:

a. Latitude

b. Longitude

c. Elevasi

3. Kecepatan.

Ketika berpindah tempat, GPS dapat menunjukkan informasi kecepatan berpindah tersebut.

4. Arah perjalanan.

GPS dapat menunjukkan arah tujuan.

5. Simpan lokasi.

Tempat-tempat yang sudah pernah atau ingin dikunjungi bisa disimpan oleh GPS *receiver*.

6. 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.[CITATION Whi12 \l 1057]

2.2.13 Latitude dan Longitude

Latitude dan longitude adalah sebuah koordinat yang ada di bumi dan digunakan untuk menentukan kedudukan kita di atas bumi.[CITATION Win10 \l 1057]

2.2.13.1 Latitude

Latitude adalah garis yang melintang dari kutub utara dan kutub selatan. Titik 0 adalah sudut ekuator, tanda + menunjukan arah ke atas menuju kutub utara, sedangkan tanda minus di koordinat Latitude menuju ke kutub selatan. Titik yang dipakai dari 0 ke 90 derajat arah kutub utara, dan 0 ke -90 derajat ke kutub selatan. [CITATION Win10 \l 1057]

2.2.13.2 Longitude

Longitude adalah garis lintang. Angka dari sudut bundar bumi horisontal. Titik diawali dari 0 ke 180 derajat, dan - ke arah sebaliknya. Titik 0 dimulai dari garis negara Inggris. Mengarah ke Indonesia akan menjadi angka positif. Kebalikannya koordinat Longitude minus adalah arah kebalikan. [CITATION Win10 \l 1057]

2.2.14 LBS

LBS (*Location Base Service*) adalah sebuah layanan berbasis lokasi, yaitu sebuah layanan berbasis internet yang mampu menampilkan posisi secara geografis

dari perangkat bergerak atau ponsel, atau memberi informasi lokasi dari alamat yang diinginkan. LBS merupakan istilah umum yang digunakan untuk menggambarkan teknologi yang digunakan untuk menemukan lokasi perangkat yang digunakan.

Dua unsur utama dari LBS adalah:

1. **Location Manager (API Maps)**

   Menyediakan *tools* atau *source* untuk LBS, *Aplication Programming Interface* (API) Maps menyediakan fasilitas untuk menampilkan, memanipulasi maps atau peta beserta fitur-fitur lainnya seperti tampilan satelit, *street* 'jalan', maupun gabungannya. Paket ini berada pada *com.google.android.maps*. [CITATION Saf11 \l 1057]

2. **Location Providers**

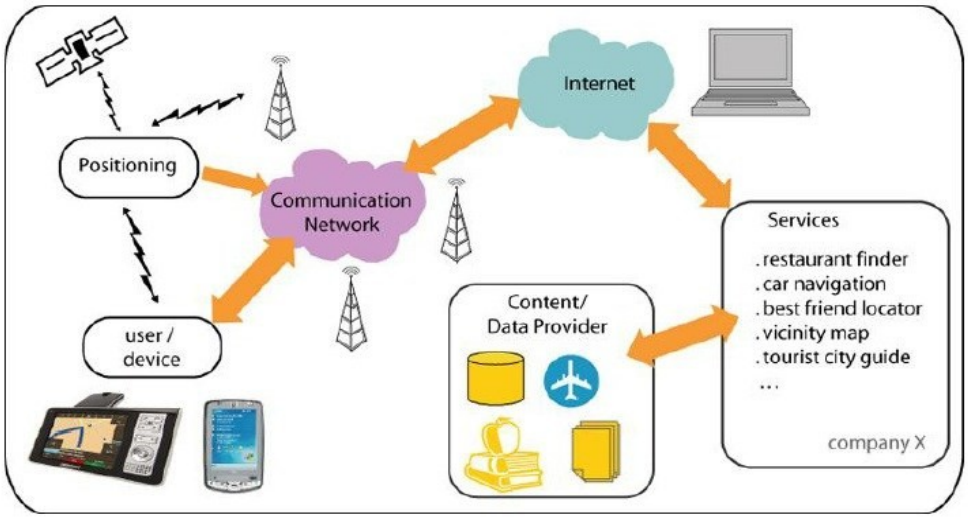
   Menyediakan teknologi pencarian lokasi yang digunakan oleh *device* 'perangkat'. API *Location* berhubungan dengan data GPS (*Global Positioning System*) dan data lokasi *real-time*. API *Location* berada pada paket Android yaitu dalam paket android *location*. Dengan *Location Manager*, dapat ditentukan lokasi saat ini, *track* 'gerakan/perpindahan', serta kedekatan dengan lokasi tertentu dengan mendeteksi perpindahan. [CITATION Saf11 \l 1057]

Data lokasi pengguna biasanya di dapatkan melalui jaringan telepon seluler ataupun menggunakan GPS. LBS memiliki komponen-komponen yang menunjang dalam prosesnya yaitu:

1. Perangkat *Mobile*, pengguna membutuhkan perangkat mobile untuk menggunakan layanan LBS ini. Sepertihalnya *smartphone*, tablet dan lain-lain.
2. Jaringan Komunikasi, jaringan kominkasi digunakan untuk menghubungkan perangkat mobile dengan perangkat lainya.
3. Komponen pengambil posisi latitude dan longitude (satelit), satelit merupakan alat yang menentukan posisi pengguna. Seperti jarak, lokasi dan lain-lain.
4. Data dan provider *content*, data yang di dapatkan akan di proses di server dan dikirim kembali ke pengguna berupa data yang telah akurat.

5. Web Map Server (WMS), merupakan server dimana tempat pengumpulan dan pemrosesan data.

Berikut ini adalah cara kerja dari LBS :



Sumber Gambar : http://eprints.umm.ac.id/34206/1/jiptummpp-gdl-anggaefend-45496-1pendahul-x.pdf

Gambar 2.4 Cara Kerja Location Based Service

1. Pertama Smartphone membuka aplikasi LBS yang sudah ter *install*/jika menggunakan aplikasi yang berbasis browser, maka buka browser dan ketik alamat tujuan situsnya.
2. Aplikasi LBS akan melakukan sambungan dengan jaringan provider yang dipakai oleh *User*.
3. Jaringan mengirimkan request ke satelit untuk menentukan longitude (garis bujur) dan latitude (garis lintang) dari si pengguna aplikasi tersebut.
4. Provider menghubungkan aplikasi (di smartphone) dengan server LBS dan meminta data yang diinginkan *User*.
5. User mendapatkan data dan ditampilkan di Smartphone.

Pada dasarnya, LBS terbagi menjadi 2 yaitu:

1. Pull Service

Pengguna secara aktif mengirimkan informasi yang dibutuhkan. Sama seperti mengakses sebuah halaman web di browser, saat memasukan halaman web yang di tuju diperolah informasi dari halaman web yang tampil di browser. Pull

service terbagi menjadi dua bagian yaitu berdasarkan fungsional seperti memesan taksi dengan menekan tombol pada *device* atau layanan *service* mencari lokasi restoran yang terdekat.

2. Push Service

Memberikan informasi kepada pengguna yang mana tidak secara langsung diminta oleh pengguna. Aplikasi Push Service cocok digunakan untuk memata - matai seseorang melalui *SmartPhone* yang digunakannnya, misalnya orang tua yang ingin tau anaknya sedang berada dimana, maka dengan adanya aplikasi Push Service ini orang tua bisa mendapatkan Data lokasi anaknya yang dikirim ke *smartphone* orang tua yang berasal dari *smartphone* anaknya secara Push Service. sehingga si anak tidak mengetahui kalau orang tua nya sedang memata-matai pergerakan dari si Anak dan menghindari hal-hal yang tidak dinginkan seperti penculikan.

Penggunaan teknologi LBS pada pembangunan aplikasi ini bertujuan untuk mendapatkan lokasi keberadaan bengkel yang paling dekat dengan pengguna.

2.2.15 Google Maps

GoogleMaps adalah peta online atau membuka peta secara online, dapat dilakukan secara mudah melalui layanan gratis dari Google. Bahkan layanan ini menyediakan API (*Application Programming Interface*) yang memungkinkan *developer* lain untuk memanfaatkan aplikasi ini di aplikasi buatannya. Tampilan GoogleMaps pun dapat dipilih, berdasarkan foto asli atau peta gambar rute saja. GoogleMaps adalah suatu peta dunia yang dapat digunakan untuk melihat suatu daerah. Dengan kata lain, GoogleMaps merupakan suatu peta yang dapat dilihat dengan menggunakan suatu browser. GoogleMaps API adalah suatu library yang berbentuk JavaScript. Cara membuat GoogleMaps untuk ditampilkan pada suatu web atau blog sangat mudah hanya dengan membutuhkan pengetahuan mengenai HTML serta JavaScript, serta koneksi Internet yang sangat stabil. Dengan menggunakan GoogleMaps API, dapat menghemat waktu dan biaya untuk membangun aplikasi peta digital yang handal, sehingga dapat fokus hanya pada data-data yang akan ditampilkan. Dengan kata lain, hanya membuat suatu data

sedangkan peta yang akan ditampilkan adalah milik Google sehingga tidak dipusingkan dengan mambuat peta suatu lokasi, bahkan dunia.

2.2.16 Basis Data

Sistem basis data adalah sistem terkomputerisasi yang tujuan utamanya adalah memelihara data yang sudah diolah atau informasi dan membuat informasi tersedia saat dibutuhkan. Pada intinya basis data adalah media untuk menyimpan data agar dapat diakses dengan mudah dan cepat. [CITATION Suk14 \l 1057] Sistem informasi tidak dapat dipisahkan dengan kebutuhan akan basis data apa pun bentuknya, entah berupa file teks ataupun *Database Management System* (DMBS). Kebutuhan basis data dalam sistem informasi meliputi:

1. Masukkan, menyimpan, dan mengambil data
2. Membuat laporan berdasarkan data yang telah disimpan

2.2.17 DBMS

DBMS (*Database Management System*) atau dalam bahasa Indonesia sering disebut sebagai Sistem Manajemen Basis Data adalah suatu sistem aplikasi yang digunakan untuk menyimpan, mengelola, dan menampilkan data [CITATION Suk14 \l 1057]. Suatu sistem aplikasi disebut DBMS jika memenuhi persyaratan minimal sebagai berikut:

1. Menyediakan fasilitas untuk mengelola akses data
2. Mampu menangani integritas data
3. Mampu menangani akses data yang dilakukan secara bersamaan 4. Mampu menangani backup data.

2.2.18 Structure Query Language (SQL)

Structure Query Language atau SQL merupakan suatu bahasa yang digunakan untuk mengakses database. SQL juga sering disebut query. Secara umum SQL terdiri dari dua bahasa, yaitu Data Definition Language (DDL) dan Data Manipulation Language (DML) [CITATION Ach10 \l 1057].

DDL digunakan untuk mendefinisikan, mengubah, serta menghapus database dan objek – objek yang diperlukan database. Secara umum, DDL yang digunakan adalah *Create*, *Alter*, *Use*, *Drop* yang biasanya digunakan oleh administrator dalam pembuatan suatu aplikasi database. DML digunakan untuk memanipulasi data yang ada dalam suatu tabel. Perintah yang dapat dilakukan adalah *Select, Insert, Update, Delete* [CITATION Ach10 \l 1057].

Fungsi Agregat, yaitu fungsi khusus yang melibatkan sekelompok data, secara umum fungsi agregat adalah SUM, COUNT, AVG, MAX, dan MIN. Fungsi ini digunakan pada bagian *select*, syarat untuk fungsi ini adalah diletakkan pada bagian having, bukan where. Sub query adalah suatu query yang sudah menjadi kompleks terutama melibatkan lebih dari satu tabel atau fungsi agregat [CITATION Ach10 \l 1057].

2.2.19 Pemrograman Berorientasi Objek

Metodologi berorientasi objek adalah suatu strategi pembangunan perangkat lunak yang mengorganisasikan perangkat lunak sebagai kumpulan objek yang berisi data dan operasi yang diberlakukan terhadapnya. Metodologi berorientasi objek merupakan suatu cara bagaimana sistem perangkat lunak dibangun melalui pendekatan objek secara sistematis. Metode berorientas objek didasarkan pada penerapan prinsip-prinsip pengelolaan kompleksitas. Metode berorientasi objek meliputi rangkaian aktivitas analisis berorientasi objek, perancangan berorientasi objek, pemrograman berorientasi objek, dan pengujian berorientasi objek. [CITATION Suk14 \l 1057]

Pada saat ini, metode berorientasi objek banyak dipilih karena metodologi lama banyak menimbulkan masalah seperti adanya kesulitan pada saat mentransformasi hasil dari satu tahap pengembangan ke tahap berikutnya, misalnya pada metode pendekatan terstruktur, jenis aplikasi yang dikembangkan saat ini berbeda dengan masa lalu. Aplikasi yang dikembangkan pada saat ini beragam dengan platform yang berbeda-beda, sehingga menimbulkan tuntutan kebutuhan metodologi pengembangan yang dapat mengakomodasi kesemua jenis

aplikasi tersebut. Keuntungan menggunakan metodologi berorientasi objek adalah sebagai berikut:

6. Meningkatkan produktivitas

   Karena kelas dan objek yang ditemukan dalam suatu masalah masih dapat dipakai ulang untuk masalah lainnya yang melibatkan objek tersebut (*reusable*).

7. Kecepatan pengembangan

   Karena sistem yang dibangun dengan baik dan benar pada saat analisis dan perancangan akan menyebabkan berkurangnya kesalahan pada saat pengodean.

8. Kemudahan pemeliharaan

   Karena dengan model objek, pola-pola yang cenderung tetap dan stabil dapat dipisahkan dan pola-pola yang mungkin sering berubah-ubah.

9. Adanya konsistensi

   Karena sifat pewarisan dan penggunaan notasi yang sama pada saat analisis, perancangan maupun pengkodean.

10. Meningkatkan kualitas perangkat lunak

    Karena pendekatan pengembangan lebih dekat dengan dunai nyata dan adanya konsistensi pada saat pengembangannya, perangkat lunak yang dihasilkan akan mampu memenuhi kebutuhan pemakai serta mempunyai sedikit kesalahan.

2.2.20 Jaringan Internet

Internet adalah suatu jaringan computer global yang terbentuk dari jaringan-jaringan komputer lokal dan regional yang memungkinkan komunikasi data antar komputer yang terhubung ke jaringan tersebut. Internet awalnya merupakan suatu rencana dari Departemen Pertahanan Amerika Serikat (US Department of Defense) pada sekitar tahun 1960. Dimulai dari suatu proyek yang dinamakan ARPANET atau *Advanced Research Project Agency Network*. Beberapa universitas di Amerika Serikat diantaranya UCLA, Stanford, UC Santa Barbara dan University of Utah, diminta bantuan dalam mengerjakan proyek ini

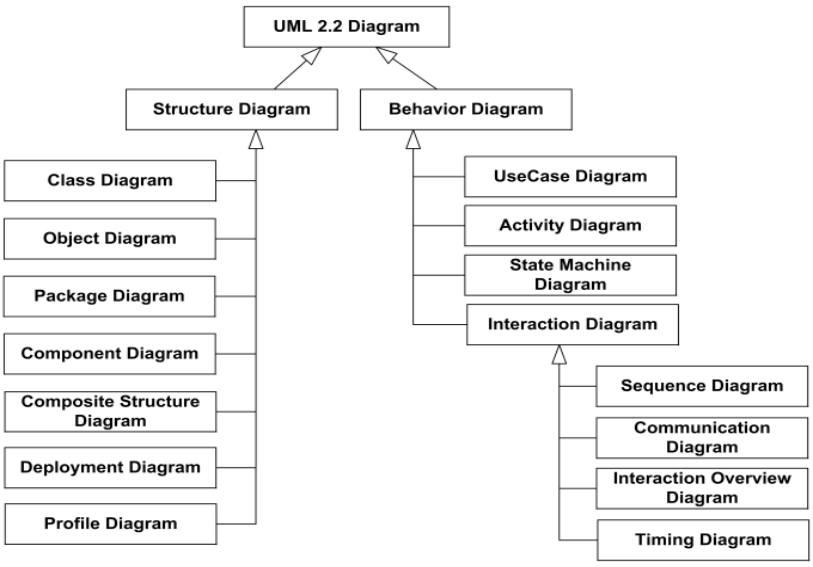
dan awalnya telah berhasil menghubungkan empat komputer di lokasi universitas berbeda tersebut.

2.2.21 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*) [CITATION Har04 \l 1057]. 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*).

2.2.22 UML Diagram

Pada UML terdiri dari macam-macam diagram yang dikelompokan dalam 2 kategori. Pembagian kategori dan macam-macam diagram tersebut dapat dilihat pada Gambar 2 .5 UML Diagram.



Sumber gambar: K. Hamilton dan R. Miles, "Learning UML 2.0" [CITATION Ham06 \l 1057]

Gambar 2.5 UML Diagram

Berikut ini penjelasan singkat dari pembagian kategori tersebut :

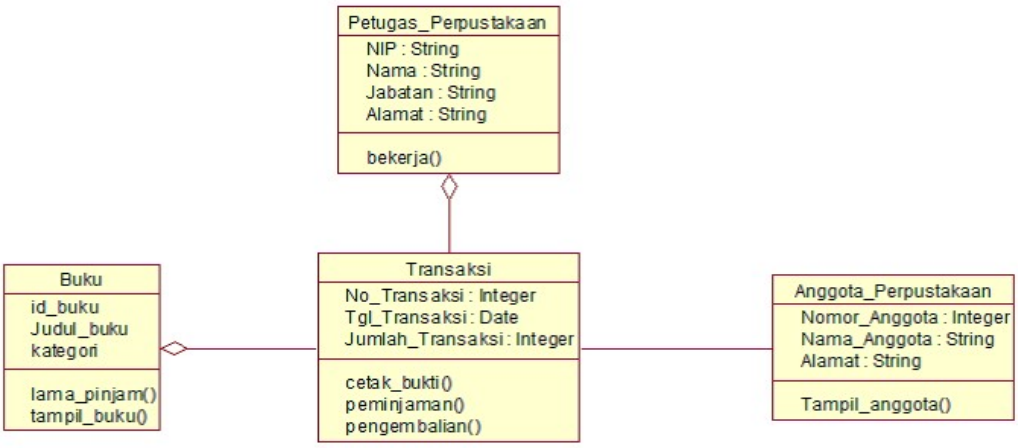
1. *Structure diagrams* yaitu diagram untuk memvisualisasikan, mendokumentasi, menspesifikasikan, dan membangun aspek statis dari sistem yang dimodelkan.
2. *Behavior diagrams* yaitu diagram untuk memvisualisasikan, mendokumentasi, menspesifikasikan, dan membangun aspek dinamis dari sistem yang dimodelkan.

2.2.23 Class Diagram

Diagram Kelas menunjukkan sekumpulan kelas, *interface* dan kolaborasi dan keterhubungan. Diagram kelas ditujukan untuk pandangan statik terhadap sistem. Kelas memiliki apa yang disebut nama, atribut, metode atau operasi [CITATION Har04 \l 1057].

1. Nama kelas harus unik. Nama akan menjadi identifier di program, seharusnya sedini mungkin dipilih nama yang memenuhi aturan (semua) bahasa pemrograman.
2. Atribut adalah properti bernama di kelas yang mendeskripsikan range nilai yang dipunyai instan kelas. Kelas dapat mempunyai sejumlah atribut atau tidak sama sekali.
3. Operasi atau metode adalah implementasi layanan yang dapat diminta pada sembarang objek kelas untuk mempengaruhi perilaku sistem. Kelas dapat mempunyai sejumlah operasi atau tidak sama sekali.

Berikut Contoh *Class Diagram* dapat dilihat pada *Gambar 2 .6 Contoh Class Diagram*.



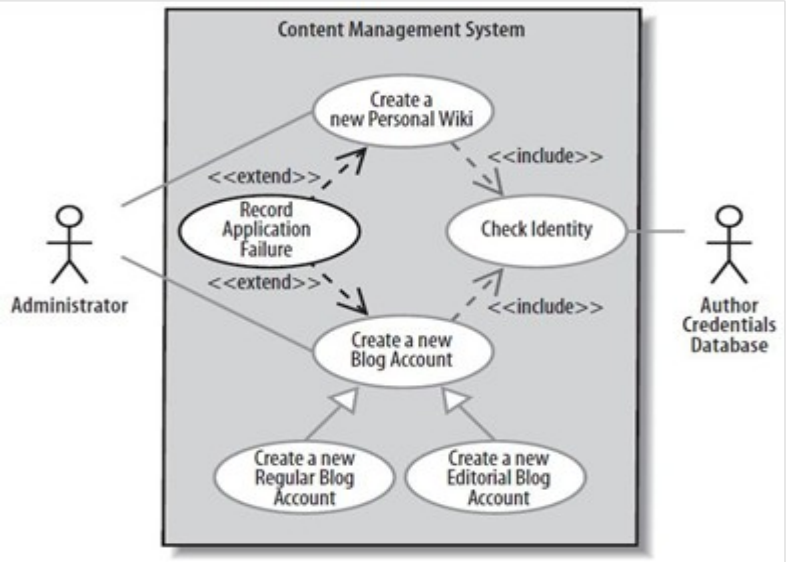
Gambar 2.6 Contoh *Class Diagram*

2.2.24 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* [CITATION Har04 \l 1057].

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 Contoh Use Case Diagram.



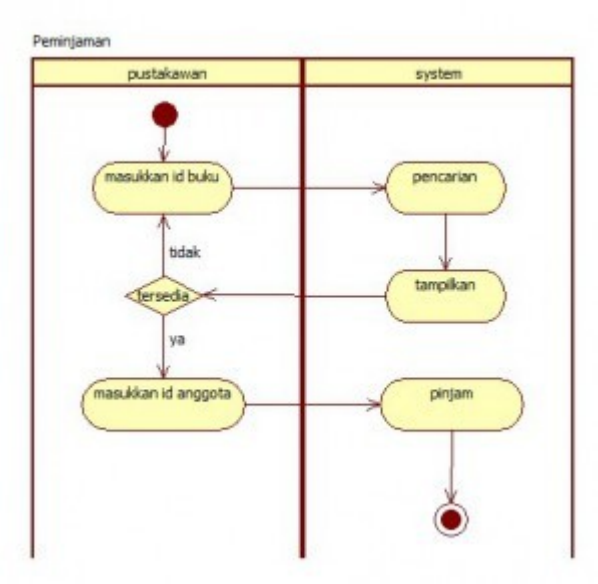
Gambar 2.7 Contoh *Use Case Diagram*

2.2.25 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[CITATION Har04 \l 1057]:

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.

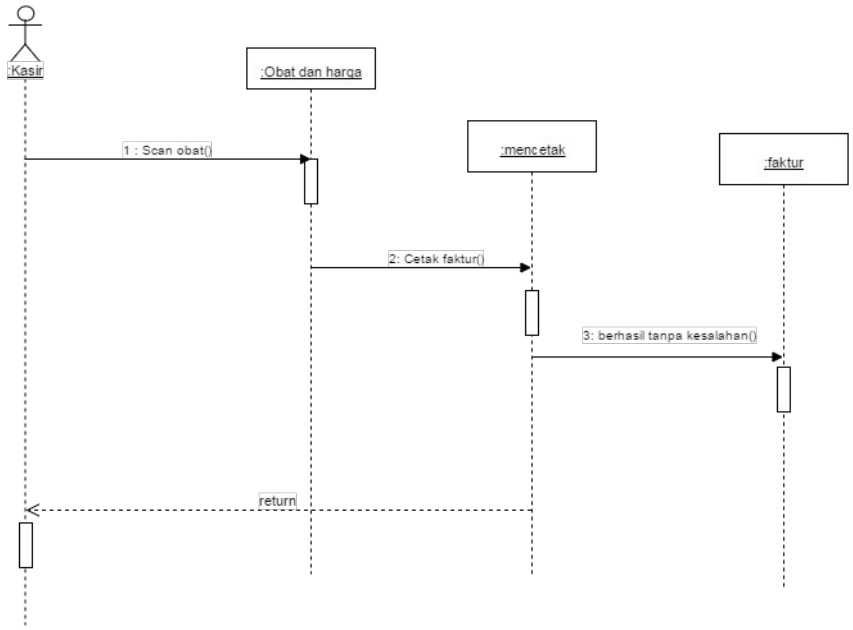
Berikut contoh *Activity Diagram* dapat dilihat pada *Gambar 2 .8 Contoh Activity Diagram.*



Gambar 2.8 Contoh *Activity Diagram*

2.2.26 Sequence Diagram

Diagram sekuen 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 [CITATION Har04 \l 1057]. Berikut Contoh *sequence diagram* dapat dilihat pada Gambar 2 .9 Contoh Sequence Diagram.



Gambar 2.9 Contoh Sequence Diagram

2.2.27 Metode Pengujian Sistem

Metode pengujian sistem terdiri dari Pengujian *black box* dan *beta* yang dilakukan untuk mengetahui efektifitas dari perangkat lunak (*software*) yang digunakan.

2.2.27.1 Pengujian Black Box

Pengujian yang dilakukan hanya mengamati hasil eksekusi melalui data uji dan memeriksa fungsional dari perangkat lunak. Pengujian ini dianalogikan seperti melihat suatu kotak hitam yang hanya bisa dilihat penampilan luarnya saja, tanpa tahu ada apa dibalik bungkus hitamnya. *Black box* testing melakukan evaluasi untuk menemukan kesalahan dalam kategori berikut[20]:

1. Fungsi tidak benar atau hilang.
2. Kesalahan *interface* atau antarmuka.
3. Kesalahan dalam struktur data atau akses *database eksternal*.
4. Kesalahan kinerja atau perilaku dan kesalahan inisialisasi dan terminasi.

2.2.27.2 Pengujian *Beta*

Pengujian *beta* merupakan pengujian yang dilakukan secara objektif dimana diuji secara langsung ke lapangan, dengan menggunakan kuesioner mengenai tanggapan masyarakat/customer terhadap aplikasi yang telah dibangun. Adapun metode penilaian pengujian yang digunakan yaitu metode kuantitatif berdasarkan data sampel dari masyarakat/customer.

2.2.27.3 Kuesioner

Menurut Depdikbud kuesioner atau angket adalah suatu alat pengumpul data yang berupa serangkaian pertanyaan yang diajukan pada responden untuk mendapat jawaban. Angket merupakan teknik pengumpulan data yang dilakukan dengan mengadakan komunikasi dengan sumber data. Menurut KBBI (Kamus Besar Bahasa Indonesia) kuesioner merupakan alat riset atau survei yang terdiri atas serangkaian pertanyaan tertulis, bertujuan mendapatkan tanggapan dari kelompok orang terpilih melalui wawancara pribadi atau melalui pos daftar pertanyaan.

Kuesioner merupakan teknik pengumpulan data yang dilakukan dengan cara memberi seperangkat pertanyaan atau pernyataan tertulis kepada responden untuk dijawabnya. Selain itu, kuesioner juga cocok digunakan bila jumlah responden cukup besar dan tersebar di wilayah yang luas. kuesioner dapat berupa pertanyaan/pernyataan tertutup atau terbuka, dapat diberikan kepada responden secara langsung atau dikirim melalui pos, atau internet. Bila penelitian dilakukan pada lingkup yang tidak terlalu luas, sehingga kuesioner dapat diantarkan langsung dalam waktu yang tidak terlalu lama, maka pengiriman kuesioner kepada responden tidak perlu melalui pos [22].