BAB 2
TINJAUAN PUSTAKA

2.1. Profil Perusahaan

PT. Dirgantara Indonesia(Persero) adalah salah satu perusahaan di Asia yang bergerak dalam dunia penerbangan. PT. Dirgantara Indonesia(Persero) mempunyai kompetensi utama dalam bidang pengembangan dan perancangan pesawat, pembuatan struktur pesawat, perakitan, serta perawatan untuk pesawat sipil dan militer ukuran kecil sampai menengah. Sejak awal berdirinya di Bandung pada tahun 1976, PT. Dirgantara Indonesia(Persero) telah berhasil menjadi industri penerbangan. Dalam dunia penerbangan, PT. Dirgantara Indonesia(Persero) telah memproduksi berbagai macam pesawat terbang. Salah satunya pesawat CN235 yang digunakan sebagai alat transportasi untuk sipil dan militer, patroli maritim dan penjaga pantai.

Dengan persetujuan kerjasama antara PT. Dirgantara Indonesia(Persero) dan Airbus Defence and Space, PT. Dirgantara Indonesia(Persero) memproduksi pesawat NC212i (pengembangan pesawat C212-400), juga sebagai pusat pengiriman dan perakitan akhir serta penyedia komponen pesawat CN295.

Disamping pesawat dengan jenis *fixed wing*, PT. Dirgantara Indonesia(Persero) juga memproduksi beberapa tipe pesawat jenis *rotary wing* atau helikopter seperti NAS-332, AS 725-Cougar, AS365N3-Dauphin dibawah lisensi Airbus Helikopter dan BELL-412 EP dibawah lisensi Bell Textron. Selain itu, PT. Dirgantara Indonesia(Persero) juga memproduksi komponen, *tools* dan bagian dari pesawat – pesawat Airbus tipe A320/321/330/340/350/380, MK2 dan EC725 milik Airbus Helicopters, serta pesawat CN235 dan C295 milik Airbus Defence and Space.

2.1.1. Sejarah Perusahaan

Aktivitas kedirgantaraan di Indonesia dimulai tahun 1946 dengan dibentuknya Biro Rencana dan Konstruksi Pesawat di lingkungan Tentara Republik Indonesia Angkatan Udara di Madiun, yang kemudian dipusatkan di andir,

Bandung. Tahun 1953, kegiatan tersebut mendapat wadah baru dengan nama Seksi Percobaan yang pada tahun 1957 berubah menjadi Sub Depot Penyelidikan, Percobaan dan Pembuatan Pesawat Terbang. Tahun 1960, Sub Depot ini ditingkatkan menjadi Lembaga Persiapan Industri Penerbangan (LAPIP) yang kemudian berubah menjadi Komando Pelaksanaan Industri Pesawat Terbang (KOPELAPIP) yang pada tahun 1966 digabung dengan PN Industri Pesawat Terbang Berdikari menjadi Lembaga Industri Penerbangan Nurtanio (LIPNUR).

Pada tahun 1975, PT. Pertamina membentuk Divisi Advanced Technology dan Teknologi Penerbangan (ATTP) yang bertujuan menyiapkan infrastruktur bagi industri kedirgantaraan di Indonesia. Berdasarkan Akta Notaris No. 15, tanggal 24 April 1976, didirikan PT. Industri Pesawat Terbang Nurtanio, dipimpin oleh Prof. Dr. Ing. B.J.Habibie. Perusahaan ini merupakan penggabungan antara LIPNUR dan ATTP. Kemudian pada bulan April 1986, melalui Keputusan Presiden (KEPRES) N0. 15/1986 dan Rapat Umum Pemegang Saham Perusahaan, nama perusahaan diganti menjadi PT. Industri Pesawat Terbang Nusantara (IPTN) dan tanggal 24 Agustus 2000, nama perusahaan secara resmi diubah oleh Presiden Republik Indonesia saat itu menjadi PT. Dirgantara Indonesia (PT. DI). Pada tahun 1998, berdasarkan Peraturan Pemerintah Nomor 35 Tahun 1998 tentang Penyertaan Modal Negara Republik Indonesia Untuk Pendirian Perusahaan Perseroan (Persero) Di Bidang Industri, saham negara pada PT. IPTN (Persero) dialihkan menjadi penyertaan pada PT. Bahana Pakarya Industri Strategis (Persero) (PT. BPIS), dengan demikian status PT. IPTN berubah menjadi anak perusahaan PT. BPIS.

Kemudian pada tahun 2002, berdasarkan Peraturan Pemerintah Nomor 52 Tahun 2002 tentang Penyertaan Modal Negara Republik Indonesia Ke Dalam Modal Saham PT. Dirgantara Indonesia, PT. Industri Kereta Api, PT. Industri Telekomunikasi Indonesia Dan PT. LEN Industri Dan Pembubaran Perusahaan Perseroan (Persero) PT. Bahana Pakarya Industri Strategis, PT. Dirgantara Indonesia berubah menjadi badan hukum persero.

2.1.2. Visi dan Misi Perusahaan

Visi PT. Dirgantara Indonesia(Persero) adalah menjadi perusahaan kelas dunia dalam industri berbasis pada penguasaan teknologi tinggi dan mampu bersaing dalam pasar global dengan mengandalkan keunggulan biaya.

Misi PT. Dirgantara Indonesia(Persero) adalah sebagai pusat keunggulan di bidang industri dirgantara terutama dalam rekayasa, rancang bangun, manufaktur, produksi dan pemeliharaan untuk kepentingan komersial dan militer dan juga aplikasi di luar industri dirgantara. Menjalankan usaha dengan selalu berorientasi pada aspek bisnis dan komersial dan dapat menghasilkan produk jasa yang memiliki keunggulan biaya.

2.1.3. Logo Perusahaan

Adapun logo perusahaan PT. Dirgantara Indonesia (Persero) ditunjukan pada Gambar 2.1 Logo PT Dirgantara Indonesia.

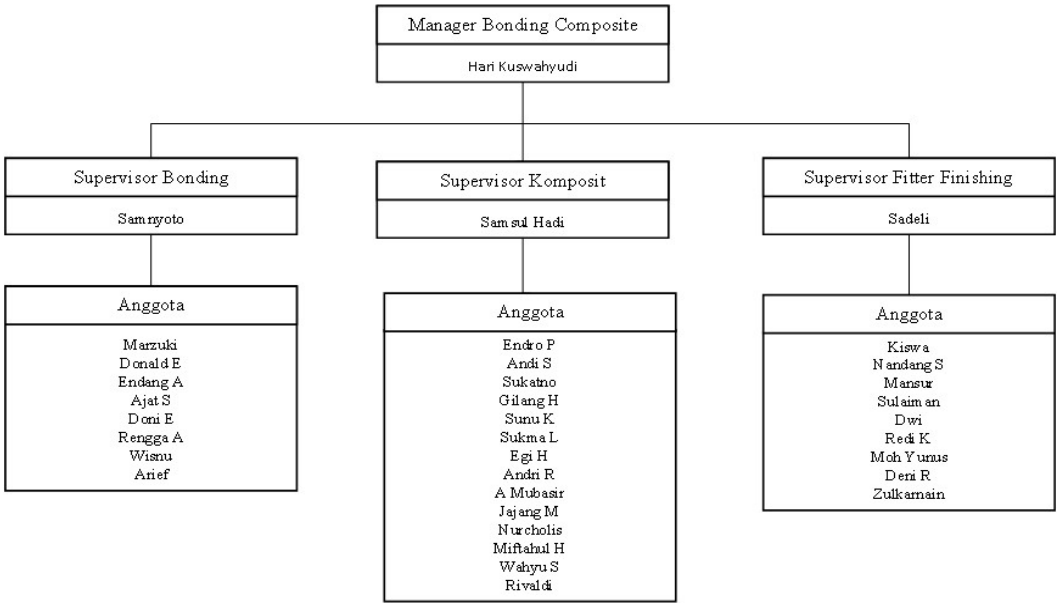
Gambar 2.1 Logo PT Dirgantara Indonesia

Sumber Gambar : http://www.indonesian-aerospace.com/kiosk-web/gallery.html

2.1.4. Departemen Bonding Komposit

Departemen Bonding Komposit adalah salah satu departemen produksi di PT Dirgantara Indonesia yang termasuk ke dalam *special proses*. Adapun kegiatannya adalah membuat semua jenis part pesawat yang berasal dari material komposit dan part-part lainnya yang membutuhkan proses bonding.

Adapun struktur organisasi bonding komposit dapat dilihata pada Gambar 2.2 Struktur Organisasi.



Gambar 2.2 Struktur Organisasi

Selain karyawan yang bekerja dibawah departemen bonding komposit, kegiatan produksinya juga dibantu oleh beberapa fungsi lain seperti *Quality Assurance (QA),* Petugas Gudang dan *Maintenance.* QA melaksanakan proses inspeksi pada setiap kegiatan produksi, Petugas Gudang mendukung ketersedian material untuk produksi dan *Maintenance* membantu menangani fasilitas yang dipakai seperi pemeliharaan dan perbaikan.

2.2. Aplikasi

Pengertian aplikasi menurut para ahli adalah sebagai berikut Pengertian aplikasi menurut para ahli adalah sebagai berikut:

1. Menurut Jogiyanto adalah penggunaan dalam suatu komputer, instruksi (instruction) atau pernyataan (statement) yang disusun sedemikian rupa sehingga komputer dapat memproses input menjadi output.
2. Menurut Kamus Kamus Besar Bahasa Indonesia adalah penerapan dari rancang sistem untuk mengolah data yang menggunakan aturan atau ketentuan bahasa pemrograman tertentu. Aplikasi adalah suatu program komputer yang dibuat untuk mengerjakan dan melaksanakan tugas khusus dari pengguna.

3. Menurut Rachmad Hakim S, adalah perangkat lunak yang digunakan untuk tujuan tertentu, seperti mengolah dokumen, mengatur Windows &, permainan (game), dan sebagainya.
4. Menurut Harip Santoso, adalah suatu kelompok file (form, class, rePort) yang bertujuan untuk melakukan aktivitas tertentu yang saling terkait, misalnya aplikasi payroll, aplikasi fixed asset [6].

2.3. Monitoring

Monitoring adalah proses pengumpulan dan analisis informasi berdasarkan indikator yang ditetapkan secara sistematis dan kontinu tentang kegiatan/program sehingga dapat dilakukan tindakan koreksi untuk penyempurnaan program/kegiatan itu selanjutnya. *Monitoring* adalah pemantauan yang dapat dijelaskan sebagai kesadaran (*awareness*) tentang apa yang ingin diketahui, pemantauan berkadar tingkat tinggi dilakukan akan dapat membuat pengukuran melalui waktu yang menunjukan pergerakan kearah tujuan atau menjauh dari itu.

Monitoring akan memberikan informasi tentang status dan kecenderungan bahwa pengukuran dan evaluasi yang diselesaikan berulang dari waktu ke waktu, pemantauan umumnya dilakukan untuk tujuan tertentu, untuk memeriksa terhadap proses berikut objek. Proses *Monitoring* adalah proses rutin pengumpulan data dan pengukuran kemajuan atas objektif program. Memantau perubahan yang fokus pada proses dan keluaran [7].

2.4. Material

Material adalah sesuatu yang disusun atau dibuat oleh bahan [8]. Selain itu, material dapat diartikan sebagai bahan baku yang diolah perusahaan industri dapat diperoleh dari pembelian lokal, impor atau pengolahan yang dilakukan sendiri.

Material memiliki beberapa sifat [8]. Diantaranya ebagai berikut :

1. Sifat listrik (daya hantar atau *conductivity*).
2. Sifat kimia (segregasi, ketahanan korosi).
3. Sifat fisik (massa jenis, struktur).
4. Sifat teknologi (mampu mesin, mampu keras).

5. Sifat magnetik (permeabilitas, histeresis).
6. Sifat thermal (panas jenis pemuaian, konduktifitas).
7. Sifat mekanik (kekuatan, kekerasan, nilai impak).

2.5. Komposit

Komposit adalah suatu jenis bahan baru hasil rekayasa yang terdiri dari dua atau lebih bahan dimana sifat masing-masing bahan berbeda satu sama lainnya baik itu sifat kimia maupun fisikanya dan tetap terpisah dalam hasil akhir bahan tersebut (bahan komposit). Dengan adanya perbedaan dari material penyusunnya maka komposit antar material harus berikatan dengan kuat, sehingga perlu adanya penambahan wetting agent [2].

Tujuan dari dibentuknya komposit antara lain sebagai berikut :

1. memperbaiki sifat mekanik dan sifat spesifik tertentu
2. mempermudah design yang sulit pada manufaktur
3. keleluasaan dalam design yang dapat menghemat biaya
4. menjadikan bahan lebih ringan

Komposit pada umumnya terdiri dari 2 fasa:

1. Matriks

Matriks adalah fasa dalam komposit yang mempunyai bagian atau fraksi volume terbesar (dominan).

Matriks mempunyai fungsi sebagai berikut :

a. Mentransfer tegangan ke serat. Membentuk ikatan koheren, permukaan matrik/serat.
b. Melindungi serat.
c. Memisahkan serat.
d. Melepas ikatan.
e. Tetap stabil setelah proses manufaktur.

Ilustrasi contoh matriks dapat dilihat pada Gambar 2.3 Ilustrasi Contoh Matriks.

Gambar 2.3 Ilustrasi Contoh Matriks

2. *Reinforcement atau Filler atau Fiber*

Salah satu bagian utama dari komposit adalah *reinforcement* (penguat) yang berfungsi sebagai penanggung beban utama pada komposit dan dapat dilihat pada Gambar 2.4 Contoh Fiber.

Gambar 2.4 Contoh Fiber

Adanya dua penyusun komposit atau lebih menimbulkan beberapa daerah dan istilah penyebutannya; Matrik (penyusun dengan fraksi volume terbesar), Penguat (Penahan beban utama), *Interphase* (pelekat antar dua penyusun), *interface* (permukaan phase yang berbatasan dengan phase lain).

2.5.1. Pre Impregnation (Prepreg)

Prepreg adalah material komposit yang komponen penyusunya berupa serat dan resin dalam kondisi setengah curing, atau tercampur dengan perbandingan tertentu. Prepreg harus tersimpan di dalam ruangan yang bersih dan suhunya selalu terjaga, karena bila disimpan dalam ruangan yang suhunya tidak terjaga (sesuai *requirement),* maka prepreg tersebut akan mengering dan tidak dapat digunakan sebagaimana mestinya. Prepreg komposit dapat dilihat pada Gambar 2.5 Prepeg Komposit

Gambar 2.5 Prepeg Komposit

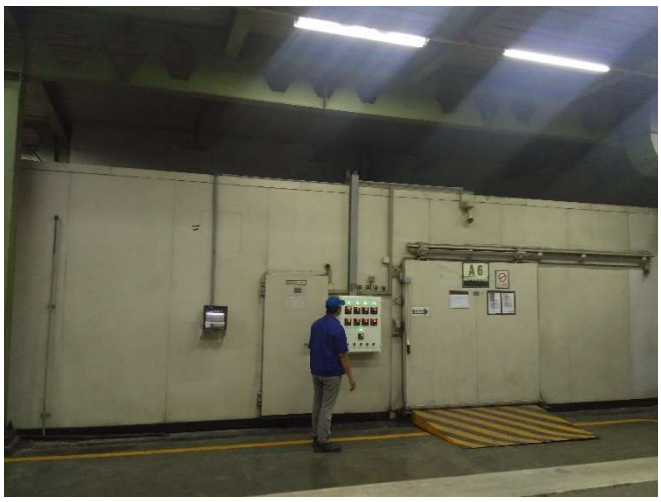
Ruangan yang dipakai untuk menyimpan prepreg biasanya berupa *cold storage* dengan suhu yang diatur minimal -18 ^{0} C, dan harus selalu terkontrol. Karena jika ada kenaikan suhu maka akan mempengaruhi kualitas dari material tersebut. Penyimpanan prepreg komposit dapat dilihat pada Gambar 2.6 Penyimpanan Prepreg komposit.

Gambar 2.6 Penyimpanan Prepreg komposit

2.6. *Cold storage*

Cold storage merupakan suatu alat mesin pendingin yang menampung benda-benda yang akan mengalami proses pendinginan. Unit *cold storage* biasa digunakan dalam kehidupan sehari-hari untuk mendinginkan atau mengawetkan

makanan. Adapun penggunaan *cold storage* di industri diantaranya untuk mendinginkan bahan baku atau bahan jadi dari suatu produk. Salah satu tujuan dari *cold storage* adalah untuk memperpanjang umur penyimpanan dengan cara pendinginan [9]. *Cold storage* dapat dilihat pada Gambar 2.7 *Cold storage* Bonding Komposit



Gambar 2.7 *Cold storage* Bonding Komposit

Cold storage memiliki perbedaan dengan alat refrigerasi lainnya. letak perbedaannya yaitu *cold storage* memiliki dimensi yang lebih besar atau dapat dikatakan sebagai big freezer dengan mengingat kapasitas penyimpanan,kebutuhan jalur bongkar muat dan kebutuhan distribusialiran udara di sekitar produk.

Jenis-jenis *cold storage* diantaranya sebagai berikut :

1. *Chilled room* adalah ruang pendingin temperatur rendah antara 1^0C s/d 7^0 C, digunakan untuk menyimpan bahan makanan fresh food, seperti sayur sayuran, buah buahan, dan menyimpan bahan lainnya, sehingga bahan makanan tersebut bisa tahan sampai 1 bulan. Bisa difungsikan untuk thawing room bagi industri makanan dengan merubah temperatur setting ke 10^0 C.
2. *Freezer room* merupakan ruangan pendingin yang temperatur kerjanya antara -15^0 C s/d -20^0 C, untuk gudang penyimpanan ikan, daging, ayam, sosis, susu, keju, kentang dan untuk semua jenis bahan makanan, dan bahan-bahan lainnya yang membutuhkan temperatur beku. juga bisa diatur sampai -40^0 C untuk produk khusus.

3. *Blast Chiller* digunakan untuk pendinginan cepat setelah proses memasakselesai dengan target temperatur 1^0 C sampai dengan 4^0 C.
4. *Blast freezer* digunakan untuk pendinginan beku secara cepat untuk makanan olahan maupun untuk daging, ikan maupun udang. Target temperatur -20^0 C sampai dengan -25^0 C.

Komponen *cold storage* adalah ruang penyimapanan dan system refrigerasi. *Cold storage* harus memiliki dimensi ruang penyimpanan yang lebih besar karena kebutuhan penyimpanan produk yang lebih besar pula. Untuk ukuran *cold storage* dihitung dan ditentukan berdasarkan kapasitas penyimpanan, kebutuhan jalur bongkar muat dan kebutuhan distribusi aliran udara di sekitar produk. System refrigerasi pada *cold storage* memiliki kesamaan dengan system refrigerasi pada umumnya terdiri dari kondensor, kompresor, evaporator dan katup ekspansi. *Cold storage* dapat dilihat pada Gambar 2.8 Bagian Dalam *Cold storage* Bonding Komposit

Gambar 2.8 Bagian Dalam *Cold storage* Bonding Komposit

2.7. *Autoclave*

Autoclave adalah alat yang dipakai untuk memanaskan material yang juga membutuhkan *tekanan*. Untuk operasionalnya sendiri membutuhkan media lain yaitu *boiler* dan *compressor.* Boiler digunakan sebagai pemasok panas yang

nantinya akan dipakai oleh *autoclave*. Boiler harus dinyalakan terlebih dahulu dan mencapai panas yang selisihnya minimal diatas 30^0 C dari suhu yang dibutuhkan oleh *autoclave*. Sedangkan compressor adalah media yang dipakai sebagai pemasok udara yang nantinya dipakai dalam proses *vacuum* dan *Pressure* saat pemanasan.

Berikut ini adalah gambar *autoclave* seperti yang terlihat pada Gambar 2.9 *Autoclave* Bonding Komposit

Gambar 2.9 *Autoclave* Bonding Komposit

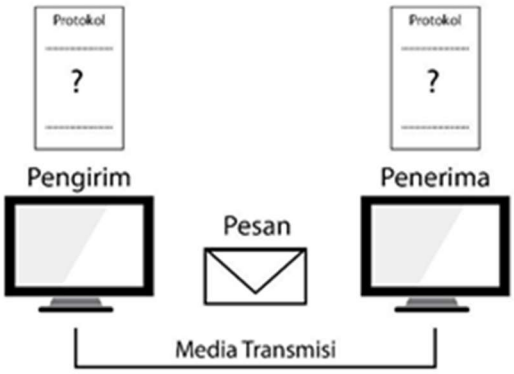
2.8. Komunikasi Data

Komunikasi data adalah hubungan antara dua atau lebih peralatan "data *processing*" (contoh : Komputer) melalui media transmisi untuk melakukan pertukaran informasi. Unsur pokok dalam komunikasi data, adalah :

a. Sumber atau pengirim
b. Media Transmisi
c. Tujuan atau Penerima
d. Informasi atau data yang dikirim

Pendekatan yang lebih realistis dapat mengambil contoh yakni saat seseorang berkomunikasi dengan orang lain, maka akan terjadi saling tukar menukar informasi satu sama lainnya. Tukar menukar informasi ini dapat terjadi secara lokal atau secara remote. Antar dua individu yang berbeda, komunikasi kasi secara lokal terjadi antara *face to face*, sedangkat secara remote dapat terjadi jika terdapat perbedaan jarak, waktu dan tempat.

Komunikasi data adalah proses pertukaran data antara dua perangkat (*device*) komunikasi melalui media transmisi seperti kabel dan wireless.



Gambar 2.10 Komponen – komponen Dasar Komunikasi Data

Pada Gambar 2.10 Komponen – komponen Dasar Komunikasi Data, terlihat terdapat beberapa komponen dasar pembentuk komunikasi data. Komponen – komponen dasar tersebut meliputi :

1. Pesan (*Message*). Merupakan informasi (data) yang akan ditransmisikan. Pesan tersebut dapat berupa teks, angka, gambar, suara, atau video serta kombinasi dari keseluruhan pesan tersebut.
2. *Sender*. Merupakan perangkat yang berfungsi untuk mengirimkan informasi. Sender dapat berupa komputer, workstation, handset pesawat telepon ataupun kamera video.
3. *Receiver.* Merupakan perangkat yang berfungsi untuk menerima informasi atau pesan dari *sender*. Dapat berupa komputer, workstation, handset pesawat telepon atau TV.

4. Media transmisi. Merupakan media transmisi tempat data atau informasi dilewatkan dari *sender* ke *receiver*. Dapat berupa kabel, fiber optik, laser, gelombang radio.
5. Protokol. Protokol merupakan seperangkat aturan yang digunakan untuk saling berkomunikasi. Dapat dianalogikan seperti bahasa yang digunakan oleh manusia.

2.9. Wifi

Wifi atau Wi-Fi, kependekan dari *Wireless Fidelity,* adalah sekumpulan standar yang digunakan untuk Jaringan Lokal Nirkabel (Wireless Local Area Network - WLAN). Didasari pada spesifikasi IEEE 802.11, yang kemudian berkembang dengan beberapa spesifikasi, antara lain 802.11a, 802.11b, 802.11g, dan 802.11n. Dengan Wifi, kita dapat membangun atau terhubung dengan jaringan area lokal (LAN). Selanjutnya, jika suatu LAN terhubung dengan segmen LAN lain, kita akan juga bisa terhubung dengan perangkat pada segmen LAN tersebut. Demikian juga, jika suatu LAN terhubung dengan jaringan internet, perangkat atau komputer dalam LAN akan bisa mengakses internet.

2.9.1. Frekuensi Wifi

Di Indonesia, standar Wifi yang paling banyak dipakai adalah 802.11b, dan 802.11g yang bekerja pada frekuensi 2.400 MHz sampai dengan 2.483,50 MHz dan terbagi dalam 11 channel (masing – masing 5 MHz), yaitu :

a. Channel 1 – 2.412 MHz
b. Channel 2 – 2.417 MHz
c. Channel 3 – 2.422 MHz
d. Channel 4 – 2.427 MHz
e. Channel 5 – 2.432 MHz
f. Channel 6 – 2.437 MHz
g. Channel 7 – 2.442 MHz
h. Channel 8 – 2.447 MHz
i. Channel 9 – 2.452 MHz
j. Channel 10 – 2.457 MHz

k. Channel 11 – 2.462 MHz

Setiap pemancar Wifi akan menggunakan salah satu channel (frekuensi) dan Wifi lain yang akan mengoneksinya harus menggunakan channel yang sama. Pembagian channel ini juga memperkecil kemungkinan interferensi pada beberapa Wifi yang berlainan jaringan (jika menggunakan channel yang berbeda).

2.9.2. Mode Koneksi Wifi

Koneksi dengan Wifi bisa dilakukan dalam mode Ad – Hoc atau mode Infrastructure.

Mode Ad – Hoc adalah koneksi antara dua komputer, di mana saru komputer berfungsi sebagai server dan komputer lainnya menjadi client. Koneksi semacan ini sering disebut sebagai koneksi peer – to – peer.

Mode Infrastructure adalah koneksi antara dua komputer atau lebih, dengan Access Point (AP) sebagai pengatur lalu lintasnya. Access Point adalah suatu perangkat yang dapat memancarkan sinyal Wifi dalam jangkauan tertentu (sering disebut sebagai hotspot). Melalui sinyal Wifi tersebut, beberapa client bisa terkoneksi ke jaringan dan AP – lah yang akan mengatur lalu lintasnya.

2.10. Mikrokontroler

Mikrokontroler adalah sebuah sistem komputer yang dibangun pada sebuah keping (*chip*) tunggal. Jadi, hanya dengan sebuah keping IC saja dapat dibuat sebuah sistem komputer yang dapat dipergunakan untuk mengontrol alat.

Saat ini sebagian besar peralatan elektronika dikontrol dengan mikrokontroler, misalnya mesin fax, mesin foto – copy, mesin cuci otomatis, sampai handphone. Peralatan tersebut tidak akan dapat dibuat dengan ukuran yang cukup kecil jika tidak menggunakan kontrol dengan menggunakan mikrokontroler.

2.10.1. Komponen Penyusun Mikrokontroler

Mikrokontroler disusun oleh beberapa komponen, yaitu CPU (*Central Processing Unit*), ROM (*Read Only Memory*), RAM(*Random Access Memory*), dan I/O(*Input/Output*). Keempat komponen ini secara bersama – sama membentuk sistem komputer dasar. Beberapa mikrokontroler memiliki tambahan komponen lain, misalnya ADC (*Analog to Digital Converter*), Timer/Counter.

2.10.2. CPU *(Central Processing Unit)*

CPU terdiri atas dua bagian, yaitu unit pengendali (*control unit*) serta unit aritmatika dan logika (ALU). Fungsi utama unit pengendali adalah mengambil, mengkodekan, dan melaksanakan urutan instruksi sebuah program yang tersimpan dalam memori.

Unit pengendali menghasilkan dan mengatur sinyal pengendali yang diperlukan untuk menyerempakkan operasi, aliran, dan instruksi program. Unit aritmatika dan logika berfungsi untuk melakukan proses perhitungan yang diperlukan selama program dijalankan serta mempertimbangkan suatu kondisi dan mengambil keputusan yang diperlukan untuk instruksi – instruksi berikutnya.

2.10.3. Bus Alamat

Bus alamat berfungsi sebagai sejumlah lintasan saluran pengalamatan antara alat dengan komputer. Pengalamatan ini harus ditentukan terlebih dahulu untuk menghindari terjadinya kesalahan pengiriman sebuah instruksi dan terjadinya bentrok antara dua buah alat yang bekerja secara bersamaan.

2.10.4. Bus Data

Bus data merupakan sejumlah lintasan saluran keluar – masuknya data dalam suatu mikrokontroler. Pada umumnya saluran data yang masuk sama dengan saluran data yang keluar.

2.10.5. Bus Kontrol

Bus kontrol atau bus pengendali ini berfungsi untuk menyerempakkan operasi mikrokontroler dengan operasi rangkaian luar.

2.10.6. Memori

Di dalam sebuah mikrokontroler terdapat suatu memori yang berfungsi untuk menyimpan data atau program. Ada beberapa jenis memori, di antaranya adalah RAM dan ROM. Ada beberapa tingkatan memori, di antaranya adalah register internal, memori utama dan memori massal. Register internal adalah memori di dalam ALU. Waktu akses register sangat cepat, umumnya kurang dari 100ns.

Memori utama adalah memori yang ada pada suatu sistem. Waktu aksesnya lebih lambat dibandingkan register internal, yaitu antara 200 sampai 1.000ns. Memori masal dipakai untuk penyipanan berkapasitas tinggi, biasanya berbentuk disket, pita magnetik, atau kaset.

2.10.7. Wemos

Wemos merupakan salah satu modul board yang dapat berfungsi dengan arduino khususnya untuk project yang mengusung konsep IOT. Wemos dapat *running stand- alone* tanpa perlu dihubungkan dengan mikrokontroler, berbeda dengan modul wifi lain yang masih membutuhkan mikrokontroler sebagai pengrontrol atau otak dari rangkaian tersebut, wemos dapat *running stand-alone* karena didalammnya sudah terdapat CPU yang dapat memprogram melalui serial port atau via OTA serta transfer program secara *wireless* [10].

Wemos memiliki 2 buah *chipset* yang digunakan sebagai otak kerja antara lain :

1. ESP8266 merupakan sebuah *chip* yang memiliki fitur Wifi dan mendukung stack TCP/IP. Modul kecil ini memungkinkan sebuah mikrokontroler terhubung kedalam jaringan Wifi dan membuat koneksi TCP/IP hanya dengan menggunakan *command* yang sederhana. Dengan *clock* 80 MHz *chip* ini dibekali dengan 4MB eksternal RAM serta mendukung format IEEE 802.11 b/g/n sehingga tidak menyebabkan gangguan bagi yang lain.
2. CH340 adalah *chipset* yang mengubah USB serial menjadi serial *interface*, contohnya adalah aplikasi *converter to* IrDA atau aplikasi USB *converter to Printer*. Dalam mode serial *interface*, CH340 mengirimkan sinyal penghubung yang umum digunakan pada modem. CH340 digunakan untuk mengubah perangkat serial *interface* umum untuk berhubungan dengan bus USB secara langsung.

2.11. Bahasa Pemrograman (*programming language*)

Merupakan notasi untuk memberikan instruksi secara tepat pada komputer. Berbeda dengan bahasa manusia yang merupakan bahasa alamial (*natural language*), sintaks atau struktur bahasa pemrograman (komputer) ditentukan secara

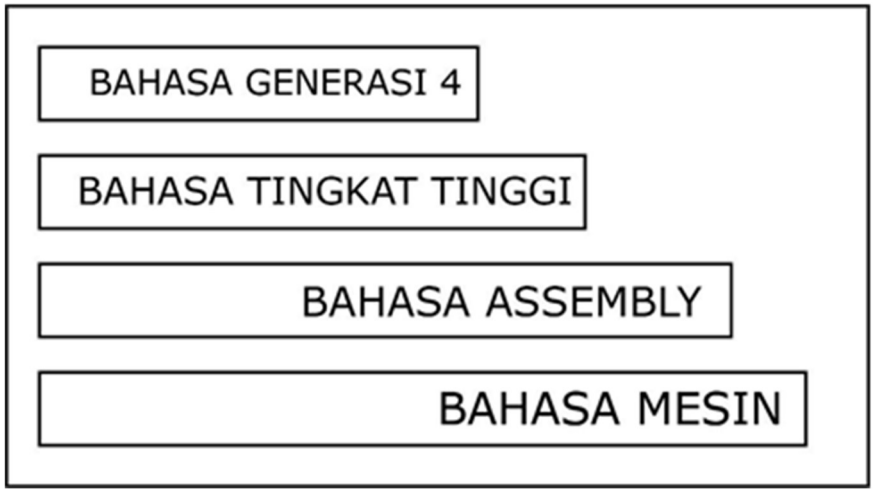
kaku, sehingga bahasa pemrograman juga disebut sebagai bahasa formal (*formal language*).

Jadi, dalam bahasa pemrograman yang digunakan sebagai alat komunikasi untuk memberikan perintah kepada komputer terdapat aturan – aturan main tertentu yang telah ditetapkan secara ketat.

Generasi bahasa pemrograman :

a. Generasi I: *Machine Language*
b. Generasi II : *Assembly Language* : Assembler.
c. Generasi III : *high – level programming language* : C, PASCAL, Java, Python, HTML.
d. Generasi IV : 4 GL (*fourth – generation language*) : SQL

Tingkatan Bahasa pemograman dapat dilihat pada Gambar 2.11 Tingkatan Bahasa Pemrograman brikut ini.



Sumber Gambar : E-Book Dasar – Dasar Pemrograman Pascal [11]

Gambar 2.11 Tingkatan Bahasa Pemrograman

2.11.1. Bahasa Tingkat Rendah (*low – level language*)

Merupakan bahasa *assembly* atau bahasa mesin. Bahasa ini dikatakan tergolong dalam bahasa tingkat rendah dikarenakan lebih dekat ke mesin (*hardware*), bahasa yang tergolong dalam bahasa tingkat rendah adalah Bahasa Mesin (*Machine Language*) dan Bahasa Rakitan (*Assembly Language*).

2.11.2. Bahasa Mesin

Bahasa mesin merupakan representasi tertulis *machine code* (kode mesin), yaitu kode operasi suatu mesin tertentu. Bahasa ini bersifat khusus untuk mesin tertentu dan "dimengerti" langsung oleh mesin, sehingga pelaksanaan proses sangat cepat. Bahasa mesin kelompok komputer tertentu berlainan dengan bahasa mesin kelompok komputer yang lain.

Abstraksi bahasa ini adalah kumpulan kombinasi kode biner "0" dan "1" yang sangat tidak alamiah bagi kebanyakan orang, kecuali untuk para ahli pembuat mesin komputer. Karena tidak alamiah bagi kebanyakan orang, bahasa mesin juga disebut bahasa tingkat rendah

2.11.3. Bahasa *Assembly*

Bahasa rakitan (*assembly language*) merupakan notasi untuk menyajikan bahasa mesin yang lebih mudah dibaca dan dipahami oleh manusia. Bahasa ini sudah menggunakan simbol alphabet yang bermakna (*mnemonic*). Contoh "MOV AX 1111", dipindahkan ke register AX nilai 1111.

Proses data oleh komputer berdasarkan perintah bahasa rakitan adalah cepat. Meski demikian masih merepotkan bahkan bagi kebanyakan pemrogram, karena masih harus mengingat – ingat tempat penyimpanan data. Bahasa rakitan juga bersifat khusus untuk mesin tertentu seperti Assembler.

2.11.4. Bahasa Tingkat Tinggi (*high – level language*)

Adalah bahasa pemrograman yang dekat dengan bahasa manusia, kelebihan utama dari bahasa ini adalah mudah untuk dibaca, tulis, maupun diperbaharui, sebelum bisa dijalankan program harus terlebih dahulu di – compile atau diterjemahkan. Contoh bahasa yang tergolong dalam bahasa tingkat tinggi adalah Ada, Algol, Basic, Cobol, C, C++, Fortran, Pascal, Java.

Pada generasi bahasa pemrograman terakhir sekarang ini, kedua cara interpretasi dan kompilasi digabungkan dalam satu lingkungan pengembangan terpadu (IDE = *Integrated Development Environment*).

Cara interpretasi memudahkan dalam pembuatan program secara interaktif dan cara kompilasi menjadikan eksekusi program lebih cepat. Pertama program

dikembangkan interaktif, kemudian setelah tidak ada kesalahan keseluruhan program dikompilasi. Contoh bahasa program seperti ini adalah Visual Basic yang berbasis Basic dan Delphi yang berbasis Pascal.

2.11.5. *Fourth – Generation Language* (4GL)

Lebih dekat ke bahasa manusia dibandingkan dengan *high – level programming languages*. Biasanya dipakai untuk mengakses *database*. Contoh pemrograman tingkat ini adalah *Structured Query Language* (SQL).

2.12. 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 system perangkat lunak dibangun melalui pendekatan objek secara sistematis. Metode berorientasi objek didasarkan pada penerapan prinsip-prinsip pengelolaan kompleksitas. Metode berorientasi objek meliputi rangkaian aktivitas analisis berorientasi objek, perancangan objek, pemrograman berorientasi objek, dan pengujian berorientasi objek [12].

Keuntungan menggunakan metodologi berorientasi objek adalah sebagai berikut:

1. Kecepatan pengembangan

   Karena system yang dibangun dengan baik dan benar pada saat analisis dan perancangan akan menyebabkan berkurangnya kesalahan pada saat pengkodean
2. Kemudahan pemeliharaan.

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

   Karena sifat pewarisan dan penggunaan notasi yang sama pada saat analisis, perancangan maupun pengkodean.
4. Meningkatkan kualitas perangkat lunak

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

5. Meningkatkan produktivitas

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

Pendekatan berorientasi objek merupakan suatu teknik atau cara pendekatan dalam melihat permasalahan dan system (system perangkat lunak, system informasi, atau system lainnya). Pendekatan berorientasi objek akan memandang system yang akan dikembangkan sebagai suatu kumpulan objek yang berkorespondensi dengan objek-objek dunia nyata.

Ada banyak cara untuk mengabstraksikan dan memodelkan objek-objek tersebut, mulai dan abstraksi objek, kelas, hubungan antarkelas sampai abstraksi system. Saat mengabstraksikan dan memodelkan objek, data dan proses-proses yang dipunyai oleh objek akan dienkapsulasi (dibungkus) menjadi satu kesatuan.

Dalam rekayasa perangkat lunak, konsep pendekatan berorientasi objek dapat diterapkan pada tahap analisis, perancangan, pemrograman, dan pengujian perangkat lunak. Ada berbagai teknik yang dapat digunakan pada masing-masing tahap tersebut, dengan aturan dan alat bantu pemodelan tertentu.

Sistem berorientasi objek merupakan sebuah system yang dibangun dengan berdasarkan metode berorientasi objek adalah sebuah system yang komponennya dibungkus (dienkapsulasi) menjadi kelompok data dan fungsi. Setiap komponen dalam system tersebut dapat mewarisi atribut dan sifat dan komponen lainnya. Dan dapat berinteraksi satu sama lain.

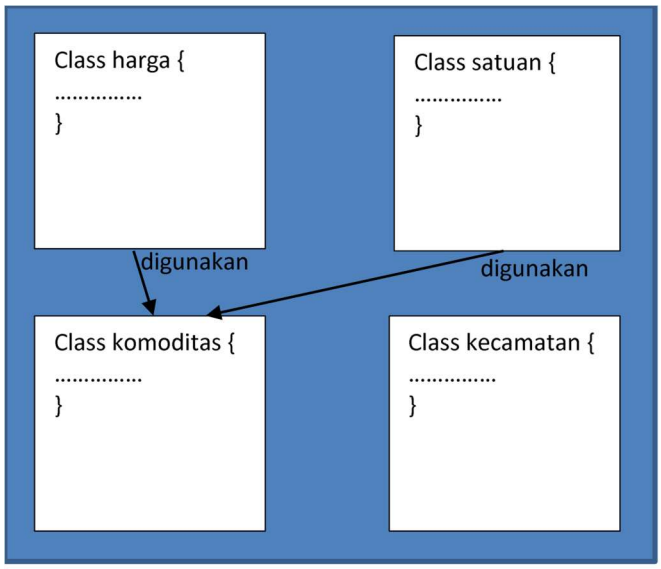
Berikut ini adalah beberapa konsep dasar yang harus dipahami tentang metodologi berorientasi objek:

1. Kelas (*class*)

Kelas adalah kumpulan objek-objek dengan karakteristik yang sama. Kelas merupakan definisi static dan himpunan objek yang sama yang mungkin lahir atau diciptakan dan kelas tersebut. Sebuah kelas akan mempunyai sifat (atribut). Kelakuan (operasi/metode), hubungan (*relationship*) dan arti. Suatu kelas dapat

diturunkan dan kelas yang lain, dimana atribut dan kelas semula dapat diwariskan ke kelas yang baru.

Secara teknis, kelas adalah sebuah struktur tertentu dalam pembuatan perangkat lunak. Kelas merupakan bentuk struktur pada kode program yang menggunakan metodologi berorientasi objek. Ilustrasi dari sebuah kelas dapat dilihat pada Gambar 2.12 Ilustrasi Kelas.



Gambar 2.12 Ilustrasi Kelas

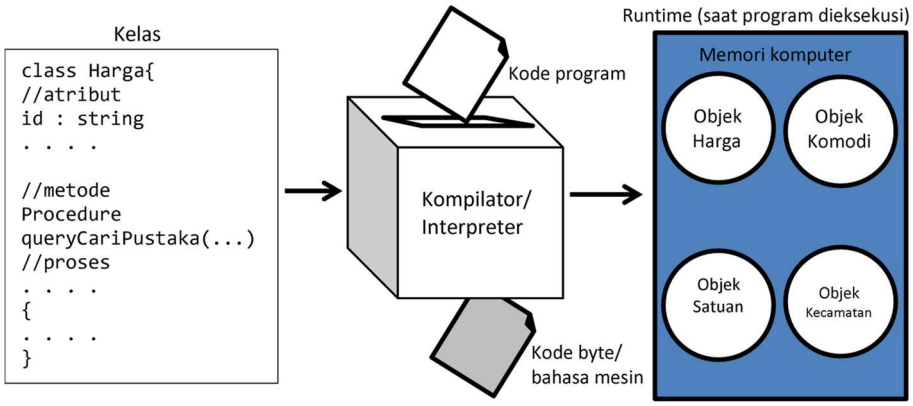
Kelas secara fisik adalah berkas atau file yang berisi kode program, dimana kode program merupakan semua hal yang terkait dengan nama kelas.

2. Objek (*object*)

Objek adalah abstraksi dan sesuatu yang mewakili dunia nyata seperti benda, manusia, satuan organisasi, tempat, kejadian, struktur, status, atau hal - hal lain yang bersifat abstrak. Objek merupakan suatu entitas yang mampu menyimpan informasi (status) dan mempunyai operasi (kelakuan) yang dapat diterapkan atau dapat berpengaruh pada status objeknya. Objek mempunyai siklus hidup yaitu diciptakan, dimanipulasi, dan dihancurkan.

Secara teknis, sebuah kelas saat program dieksekusi maka akan di buat sebuah objek. Objek dilihat dari segi teknis adalah elemen pada saat runtime yang akan diciptakan, dimanipulasi, dan dihancurkan saat eksekusi sehingga sebuah objek hanya ada saat sebuah objek hanya ada saat sebuah program dieksekusi, jika

masih dalam bentuk kode, disebut sebagai kelas jadi pada saat runtime (saat sebuah program dieksekusi), yang kita punya adalah objek, di dalam teks program yang kita lihat hanyalah kelas. Ilustrasi kelas dan objek dapat dilihat pada Gambar 2.13 Ilustrasi Kelas dan Objek



Gambar 2.13 Ilustrasi Kelas dan Objek

3. Metode (*method*)

Operasi atau metode atau method pada sebuah kelas hamper sama dengan fungsi atau prosedur pada metodologi struktural. Sebuah kelas boleh memiliki lebih dari satu metode atau operasi. Metode atau operasi yang berfungsi untuk memanipulasi objek itu sendiri. Operasi atau metode merupakan fungsi atau transformasi yang dapat dilakukan terhadap objek atau dilakukan oleh objek.

Metode atau operasi dapat berasal dari event, aktivitas atau aksi keadaan, fungsi, atau kelakuan dunia nyata. Atau kelakuan dunia nyata. Contoh metode atau operasi misalnya read, write, move copy, dan sebagainya. Kelas sebaiknya memiliki metode get dan set untuk setiap atribut agar konsep enkapsulasi tetap terjaga. Metode get digunakan untuk memberikan akses kelas ini dalam mengakses atribut, dan set adalah metode yang digunakan untuk mengisi atribut, agar kelas ini tidak mengakses atribut secara langsung.

a. Atribut (*attribute*)

Atribut dari sebuah kelas adalah variabel global yang dimiliki sebuah kelas. Atribut dapat berupa nilai atau elemen-elemen data yang dimiliki oleh objek dalam kelas objek. Atribut dipunyai secara individual oleh sebuah objek,

misalnya berat, jenis, nama, dan sebagainya. Atribut sebaiknya bersifat privet untuk menjaga konsep enkapsulasi.

b. Abstraksi (*abstraction*)

Prinsip untuk merepresentasikan dunia nyata yang kompleks menjadi satu bentuk model yang sederhana dengan mengabaikan aspek-aspek lain yang tidak sesuai dengan permasalahan.

c. Enkapsulasi (*encapsulation*)

Pembungkusan atribut data dan layanan (operasi-operasi) yang dipunyai objek untuk menyembunyikan implementasi dan objek sehingga objek lain tidak mengetahui cara kerjanya.

d. Pewarisan (*inheritance*)

Mekanisme yang memungkinkan satu objek mewarisi sebagian atau seluruh definisi dan objek lain sebagai bagian dan dirinya.

e. Antarmuka (*interface*)

Antarmuka atau interface sangat mirip dengan kelas, tapi tanpa atribut kelas dan memiliki metode yang dideklarasikan tanpa isi. Deklarasi metode pada sebuah interface dapat diimplementasikan oleh kelas lain. Sebuah kelas dapat mengimplementasikan lebih dari satu antarmuka dimana kelas ini akan mendeklarasikan metode pada antarmuka yang dibutuhkan oleh kelas itu sekaligus mendefinisikan isinya pada kode program kelas itu. Metode pada antarmuka yang diimplementasikan pada suatu kelas harus sama persis dengan yang ada pada antarmuka. Antarmuka atau *interface* biasanya digunakan agar kelas yang lain tidak mengakses langsung ke suatu kelas, mengakses antarmuka.

f. Reusability

Pemanfaatan kembali objek yang sudah didefinisikan untuk suatu permisalan pada permasalahan lainnya yang melibatkan objek tersebut. Misalkan dalam sebuah aplikasi peminjaman buku diperlukan kelas anggota, maka ketika membuat aplikasi penyewaan VCD, kelas anggota ini bisa digunakan kembali dengan sedikit perubahan untuk aplikasi penyewaan VCD tanpa harus membuat dari awal kembali.

g. Generalisasi dan Spesialisasi

Menunjukkan hubungan antara kelas dan objek uang umum dengan kelas dan objek yang khusus. Misalnya kelas yang lebih umum (generalisasi) adalah kendaraan darat dan kelas khususnya (spesialisasi) adalah mobil, motor, dan kereta.

h. Komunikasi Antar Objek

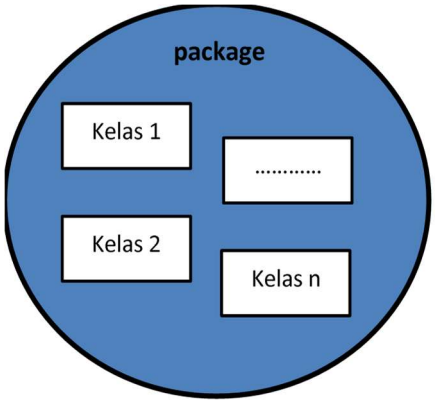
Komunikasi antar-objek dilakukan lewat pesan (message) yang dikirim dan satu objek ke objek lainnya.

i. Polimorfisme (*polymorphism*)

Kemauan suatu objek untuk digunakan di banyak tujuan yang berbeda dengan nama yang sama sehingga menghemat baris program.

j. Package

Package adalah sebuah container atau kemasan yang dapat di gunakan untuk mengelompokkan kelas-kelas sehingga memungkinkan beberapa kelas yang bernama sama disimpan dalam package yang berbeda. Ilustrasi dari sebuah package dapat dilihat pada Gambar 2.14 Package



Sumber gambar: Buku Rekayasa Perangkat Lunak oleh Rosa A.S dan M. Shalahuddin (2013) [12]

Gambar 2.14 Package

2.13. *Database* (Basis Data)

Database/basis data terdiri dari 2 susku kata, yaitu data dan base/basis. Data dapat diartikan sebagai representasi fakta dunia nyata yang mewakili suatu objek,

misalnya manusia, hewan, barang, peristiwa, konsep, dan lain sebagainya yang direkam dalam bentuk huruf, teks, simbol, angka, suara, gambar, dan lainnya. Sedangkan basis/base dapat diartikan sebagai tempat berkumpul, sarang atau gudang untuk menyimpan sesuatu. Dengan demikian, basis data/*database* dapat diartikan sebagai tempat berkumpul/menyimpan data – data suatu benda atau kejadian yang saling berhubungan.

Database juga dapat diartikan sebagai kumpulan file, arsip yang saling berhubungan yang disimpan di dalam media elektronis. Data yang disimpan tersebut tidak dibiarkan begitu saja, namun dikelola dan diorganisasi yang dikenal dengan *Database* Management System (DBMS). Dengan demikian, data yang tersimpan di dalam *database* dapat disusun dengan rapi dan terstruktur sehingga memudahkan dalam mendapatkan informasi yang bersangkutan dengan *database*.

2.13.1. Manfaat *Database*

Database sangat bermanfaat untuk mengatasi berbagai masalah yang sering terjadi dalam penyusunan data. Masalah dalam penyusunan data dapat berupa redudansi dan inkonsistensi data atau kesulitan pengaksesan data. Lebih jelas akan dipaparkan manfaat *database* sebagai berikut :

1. Redudansi dan Inkonsistensi Data

Redudansi adalah penyimpanan data yang sama dalam dua tempat (tabel) yang berbeda sehingga dapat menyebabkan inkonsistensi data. Inkonsistensi data, yaitu data yang sama dan disimpan pada dua tabel dapat memiliki nilai yang berbeda. Sebagai contoh, dalam *database* perpustakaan, di mana data nama anggota disimpan di tabel anggota dan tabel transaksi peminjaman (redudansi). Apabila nama anggota pada tabel anggota diubah, maka nama pada tabel transaksi tidak berubah (inkonsistensi data) padahal data tersebut menjelaskan objek yang sama.

2. Kesulitan Pengaksesan data

Kesulitan pengaksesan data pada *database* dengan program yang dapat muncul dapat diatasi dengan DBMS. DBMS mampu mengambil data secara langsung dengan bahasa yang sudah familiar dan mudah digunakan, misalnya SQL. Sebagai contoh, apabila pada suatu waktu akan dibutuhkan laporan tentang

adventori buku yang terbit tahun 2009, sedangkan program untuk itu belum dibuat, maka dengan mudah DBMS menyelesaikannya.

3. Isolasi Data untuk Standarisasi

Isolasi data untuk standarisasi berkaitan dengan format penyimpanan data tersebut. Sebagai contoh, data disimpan pada file dengan bahasa tertentu, misalnya Java, C/C++,Visual Basic, Pascal, dan lainnya, tentunya hal ini menyulitkan programmer untuk mengakses data tersebut. Untuk itu, data dalam suatu *database* harus disimpan dalam format yang sama. Sebagai contoh, dalam format JavaDB/Derby, MySQL, MsQL.

4. Multiple User atau Banyak Pemakai

Database dirancang agar data diakses oleh banyak pemakai(*user*) dan banyak program yang sama dalam waktu bersamaan maupun berbeda. Hal ini mungkin terjadi dengan *database*, dikarenakan dalam rangka mempercepat daya guna sistem dan mendapatkan responsi waktu yang cepat, beberapa sistem mengizinkan banyak pemakai untuk meng-*update* data secara simultan. Salah satu alasan mengapa basis data dibangun adalah karena nantinya data tersebut akan digunakan oleh banyak orang dalam waktu yang berbeda, diakses oleh program yang sama, tetapi berbeda orang dan waktu. Semua ini mungkin akan terjadi karena data yang diolah dan tidak tergantung dan menyatu dalam program, tetapi terlepas dalam satu kelompok data.

5. Masalah Keamanan

Walaupun *database* dibuat untuk diakses oleh banyak pemakai, *database* tetap menjaga keamanan data. Keamanan data dilakukan dengan memberikan hak akses tertentu bagi para user tersebut dan tidak semua user diizinkan untuk mengakses *database*. Sebagai misal data transaksi peminjaman buku tidak boleh diakses oleh anggota. Keamanan dapat diatur melalui program yang dirancang oleh programmer atau melalui fasilitas keamanan sistem operasi.

6. Masalah Kesatuan Data

Database memiliki tabel – tabel yang saling berkaitan/berhubungan. Hubungan antartabel secara teknis dikaitkan/dihubungkan dengan field kunci, sehingga dikenal dengan kunci primer (*primary key*) dan kunci tamu (*foreign key*) dalam suatu tabel. Sebagai contoh tabel buku dan tabel transaksi peminjaman buku, secara teknis buku mempunyai field kunci, sebagai misal field *id* dan tabel peminjaman juga mempuyai

field *id* di mana mempunyai struktur yang sama dengan field *id* pada tabel buku. Maka field *id* pada tabel buku disebut kunci primer dan field *id* pada tabel peminjaman disebut kunci tamu. Kedua field yang disebut kunci inilah yang menghubungkan antar tabel, sehingga dalam *database* mempunyai kesatuan data yang saling berkaitan.

7. Masalah Kebebasan Data

Kebebasan data yang dimaksud adalah kemudahan pembacaan data dalam *database* dapat diakses dengan mudah dalam menampilkan informasi yang dibutuhkan. Sebagai contoh, ketidakbebasan data adalah aplikasi program *database* yang telah dibuat secara khusus, di mana apabila suatu struktur *database* diubah strukturnya, maka progammer harus mengubah program aplikasi tersebut untuk mengakses data pada struktur *database* yang baru. Hal ini berbeda dengan *database* DBMS yang bersifat bebas, di mana data dengan mudah ditampilkan, dibaca, diubah, dan sebagainya dengan perintah – perintah tertentu.

8. Kemandirian Program Data

Program aplikasi berbasis *database* terpisah dengna data yang digunakannya, di mana data disimpan dalam DBMS dengan format tertentu. Dengan demikian, program aplikasi dan *database* berdiri sendiri – sendiri, namun antara program aplikasi dan *database* mempunyai hubungan yang erat. Hubungan tersebut adalah program aplikasi dapat mengakses data yang disimpan dalam DBMS. Hal ini dapat meningkatkan pengembangan aplikasi dan produktivitas perawatan.

9. Meningkatkan Kualitas Data

Untuk meningkatkan kualitas data dapat dilakukan dengan aturan validasi data melalui program aplikasi. Validasi data adalah penyaringan data – data yang hanya boleh disimpan dalam *database*, misal field *id* hanya boleh menyimpan data dengan format A11 (satu karakter diikuti 2 angka), atau hanya menerima input data berupa angka.

10. Akses Data Lebih Baik atau Responsif

Dengan menggunakan bahasa Structured Query *Language* (SQL), pengaksesan data jauh lebih baik dan cepat. Hal ini dikarenakan DBMS

menggunakan bahasa SQL yang mempunyai bahasa yang simpel dan mudah digunakan serta telah menjadi standar bahasa pembuatan *database*.

2.13.2 *Database* Management System (DBMS)

Database Management System atau disingkat DBMS adalah perangkat lunak (*software*) yang berfungsi untuk mengelola *database*. Mulai dari membuat *database* itu sendiri, sampai dengan proses yang berlaku dalam *database* tersebut, baik berupa entry, edit, hapus, query terhadap data, membuat laporan dan lain sebagainya secara efektif dan efisien. Salah satu jenis DBMS yang sangat terkenal saat ini adalah Relational DBMS (RDBMS). RDBMS merepresentasikan data dalam bentuk tabel – tabel yang saling berhubungan. Sebuah tabel disusun dalam bentuk baris (*record*) dan kolom (*field*).

Ada tiga kelompok perintah yang digunakan dalam mengelola dan mengorganisasikan data dalam RDBMS, yaitu :

1. Data Definition *Language*

Merupakan perintah yang digunakan oleh seorang *Database* Administrator untuk mendefinisikan struktur *database*, baik membuat tabel baru, menentukan struktur penyimpanan tabel, model relasi antar tabel dan validasi data.

2. Data Manipulation *Language* (DML)

Perintah – perintah yang digunakan untuk memanipulasi dan mengambil data pada suatu *database*. Manipulasi yang dapat dilakukan terhadap data adalah :

a. Penambahan data
b. Penyisipan data
c. Penghapusan data
d. Pengubahan data

DML merupakan bahasa yang memudahkan pengguna dalam mengakses *database*. Ada dua jenis DML, yaitu :

Prosedural, mengharuskan pengguna menentukan spesifikasi data apa yang dibutuhkan dan bagaimana cara mendapatkannya. Contoh paket bahasanya adalah dBase III, FoxBase, FoxPro.

Non Prosedural, pengguna hanya menentukan data apa yang dibutuhkan tanpa harus tahu cara mendapatkannya. Contoh paket bahasanya diberi nama Structural Query *Language* (SQL).

3. Data Control *Language*

Bagian ini berkenaan dengan cara mengendalikan data, seperti siapa saja yang bisa melihat isi data, bagaimana data bisa digunakan oleh banyak user. Lebih mengarah ke segi sekuritas data.

2.14. MySQL

MySQL pertama kali dirintis oleh seorang programmer *database* bernama Michael Widenius. MySQL *database* server adalah RDBMS (*Relational Database Management System*) yang dapat menangani data yang bervolume besar. Meskipun begitu, tidak menuntut *resource* yang besar. MySQL adalah *database* yang paling populer diantara *database – database* yang lain.

MySQL adalah program *database* yang mampu mengirim dan menerima data dengan sangat cepat dan multi user. MySQL memiliki dua bentuk lisensi yaitu free software dan shareware. Untuk penelitian ini akan menggunakan MySQL yang free software karena bebas menggunakan *database* ini untuk keperluan pribadi atau usaha tanpa harus membeli atau membayar lisensi, yang berada di bawah lisensi GNU/GPL (*general public license*), yang dapat didapatkan pada *website* resminya di http://www.mysql.com . MySQL sudah cukup lama dikembangkan beberapa fase penting dalam pengembangan MySQL adalah sebagai berikut :

a. MySQL dirilis pertama kali secara internal pada 23 Mei 1995
b. Versi Windows dirilis pada 8 Januari 1998 untuk Windows 95 dan Windows NT.
c. Versi 3.23 : beta dari Juni 2000, dan dirilis pada Januari 2001.
d. Versi 4.0 : beta dari Agustus 2002, dan dirilis pada Maret 2003 (unions).
e. Versi 4.1 : beta dari bulan Juni 2004, dirilis pada bulan Oktober 2004 (R-trees dan B-trees, subqueries, prepared statements).
f. Versi 5.0 : beta dari bulan Maret 2005, dirilis pada Oktober 2005.
g. Sun Microsystems Membeli MySQL AB pada tanggal 26 Februari 2008.

h. Versi 5.1 : dirilis 27 November 2008 (event scheduler, partitioning, plug – in API, row – based replication, server log table).

2.14.1. Kelebihan dan Keuntungan Menggunakan MySQL

Ketika menggunakan sebuah software DBMS perlu diperhitungkan apa kebutuhan yang paling penting. Apakah performa, support, fitur – fitur SQL, kondisi keamanan dalam license atau harga. Dengan pertimbangan – pertimbangan tersebut, *database* MySQL memiliki beberapa kelebihan dan keuntungan dibandingkan dengan *database* lain. Beberapa diantaranya adalah :

a. MySQL merupakan Software DBMS yang bersifat OpenSource sehingga bebas digunakan oleh perseorangan atau instansi tanpa harus membeli atau membayar pada pembuatnya.
b. MySQL dapat diakses melalui protokol ODBS (*Open Database Connectivity*) buatan Microsoft. Ini menjadikan MySQL dapat diakses oleh banyak software.
c. Semua klien dapat mengakses server dalam satu waktu, tanpa harus menunggu yang lain untuk mengakses *database*.
d. *Database* MySQL mengerti bahasa SQL (Structured Query *Language*).
e. *Database* MySQL dapat diakses darimana saja melalui internet dengan hak akses tertentu.
f. MySQL dapat berjalan di berbagai Sistem Operasi seperti Linux, Windows dan Solaris.

2.14.2. SQL

SQL adalah singkatan dari *Structured Query Language.* SQL sering dibaca dengan “Sequel” adalah bahasa terstruktur yang digunakan untuk query, meng – update dan memanipulasi *database*. Bahasa ini digunakan untuk memuat, mengurutkan dan menyaring suatu data, sehingga dihasilkan data yang spesifik dari sebuah *database*.

2.15. 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

Pemrograman 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 dll. Oleh karena sifat-sifat tersebut, menjadikan JSON ideal sebagai bahasa pertukaran-data.

Penulis menggunakan metode JSON dalam pengiriman data yang dilakukan, karena JSON memiliki beberapa kelebihan - kelebihan dibandingkan XML, kelebihan – kelebihan tersebut adalah:

1. Format Penulisan Untuk merepresentasikan sebuah struktur data yang rumit dan berbentuk hirarkis penulisan JSON relatif lebih terstruktur dan mudah.
2. Ukuran-ukuran karakter yang dibutuhkan JSON lebih kecil dibandingkan XML untuk data yang sama. Hal ini tentu berpengaruh pula pada kecepatan pertukaran data, walaupun tidak signifikan untuk data yang kecil, namun cukup berarti jika koneksi yang digunakan relatif lambat untuk mengakses aplikasi web kaya fitur yang memanfaatkan pertukaran data. Di sini JSON lebih unggul dibandingkan XML, kecuali jika data dikompresi terlebih dahulu sebelum dikirimkan, perbedaan JSON dan XML yang telah dikompresi tidaklah signifikan.
3. Browser Parsing Proses parsing merupakan proses pengenalan token atau bagian-bagian kecil dalam rangkaian dokumen XML/JSON. Contohnya, terdapat data text dalam format JSON. Data tersebut harus di-parsing terlebih dahulu sebelum dapat diakses dan dimanipulasi. Browser parsing berarti proses parsing yang terjadi pada sisi client/browser. Melakukan browser parsing pada JSON lebih sederhana dibandingkan pada XML, JSON menggunakan function JavaScript eval() untuk melakukan parsing. Sementara dokumen XML di-parsing oleh XMLHttpRequest. Rata-rata survei menobatkan JSON sebagai pemenang jika diadu kecepatan parsingnya.

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 pemrograman moderen mendukung struktur data ini dalam bentuk yang sama maupun berlainan. Hal ini pantas disebut demikian karena format data mudah dipertukarkan dengan bahasa-bahasa pemrograman yang juga berdasarkan pada struktur data ini [13].

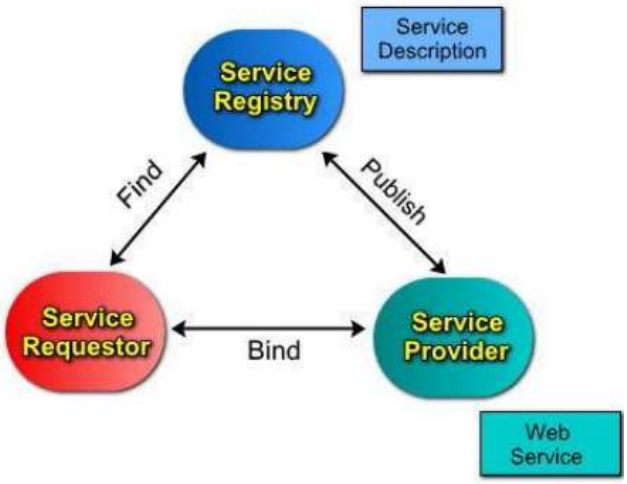
2.16. Web Service

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.

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 [14].

2.16.1. Arsitektur *Web Service*

Secara umum, arsitektur web service dapat dilihat pada Gambar 2.15 Arsitektur Web Service.



Sumber gambar : Jurnal Komputer (2007) [14]

Gambar 2.15 Arsitektur Web Service

Pada Gambar 2.15, ada tiga komponen yang membuat web service berjalan. Ketiga komponen itu adalah:

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.

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 [14].

2.17. Android

Menurut Nazarudin Safaat H, "Android adalah sebuah sistem operasi untuk perangkat *mobile* berbasis linux yang mencakup sistem operasi, *middleware,* dan aplikasi". Android adalah sistem operasi untuk telepon seluler yang berbasis linux. Android menyediakan *platform* terbuka bagi para pengembang untuk membuat aplikasi mereka sendiri. Pada awalnya dikembangkan oleh Android Inc, sebuah perusahaan pendatang baru yang membuat perangkat lunak untuk ponsel yang kemudian dibeli oleh Google Inc. untuk pengembangannya, dibentuklah *Open*

Handset Alliance (OHA), konsorsium dari 34 perusahaan perangkat keras, perangkat lunak, dan telekomunikasi termasuk Google, HTC, Intel, Motorola Qualcomm, T-*Mobile*, dan Nvidia [15].

Android dipuji sebagai "*platform mobile* pertama yang lengkap, terbuka dan bebas" [16].

1. Lengkap (*Complete Platform*)
   Para desainer dapat melakukan pendekatan yang komprehensif ketika mereka sedang mengembangkan platform Android. Android merupakan sistem operasi yang aman dan banyak menyediakan tools dalam membangun software dan memungkinkan untuk peluang pengembangan aplikasi.
2. Terbuka (*Open Source Platform*)
   Platform Android disediakan melalui lisensi *open source*. Pengembang dapat dengan bebas untuk mengembangkan aplikasi. Android sendiri menggunakan Linux Kernel 2.6.
3. Free (*Free Platform*)
   Android adalah *platform* aplikasi yang bebas untuk *develop*. Tidak ada lisensi atau biaya royalti untuk dikembangkan pada *platform* Android. Tidak ada biaya keanggotaan diperlukan. Tidak diperlukan biaya pengujian. Tidak ada kontrak yang diperlukan. Aplikasi untuk android dapat didistribusikan dan diperdagangkan dalam bentuk apapun [15].

2.18. Android Studio

Android Studio adalah sebuah IDE untuk Android Development yang diperkenalkan google pada acara Google I/O 2013. Android Studio merupakan pengembangan dari Eclipse IDE, dan dibuat berdasarkan IDE Java populer, yaitu IntelliJ IDEA. Android Studio merupakan IDE resmi untuk pengembangan aplikasi Android.

Sebagai pengembangan dari Eclipse, Android Studio mempunyai banyak fitur-fitur baru dibandingkan dengan Eclipse IDE. Berbeda dengan Eclipse yang menggunakan Ant, Android Studio menggunakan Gradle sebagai build environment. Fitur-fitur lainnya adalah sebagai berikut :

1. Menggunakan Gradle-based build system yang fleksibel.
2. Bisa mem-build multiple APK .
3. Template support untuk Google Services dan berbagai macam tipe perangkat.
4. Layout editor yang lebih bagus.
5. Built-in support untuk Google Cloud Platform, sehingga mudah untuk integrasi dengan Google Cloud Messaging dan App Engine.
6. Import library langsung dari Maven repository

Jika dibandingkan dengan Android Studio memang dari sisi build lebih baik dibandingkan Eclipse, karena Android Studio menggunakan Gradle. Pada Android Studio, kita tidak perlu lagi dipusingkan dengan *dependencies package* berbeda dengan Eclipse. Satu hal tambahan lagi yang membuat Android Studio unggul adalah dukungan layout xml editor secara visual yang jauh lebih baik daripada Eclipse. Walaupun begitu, Android Studio saat ini masih dalam tahap beta dan belum mempunyai dukungan untuk NDK/Native Development Kit.

2.19. UML (*Unified modelling language*)

Unified Modeling Language (UML) adalah suatu alat untuk memvisualisasikan dan mendokumentasikan hasil analisa dan desain yang berisi sintak dalam memodelkan sistem secara visual. Juga merupakan satu kumpulan konvensi pemodelan yang digunakan untuk menentukan atau menggambarkan sebuah sistem *software* yang terkait dengan objek [17].

Sejarah UML sendiri terbagi dalam dua fase sebelum dan sesudah munculnya UML. Dalam fase sebelum, UML sebenarnya sudah mulai diperkenalkan sejak tahun 1990, namun notasi yang dikembangkan oleh para ahli analisis dan desain berbeda-beda, sehingga dapat dikatakan belum memiliki standarisasi.

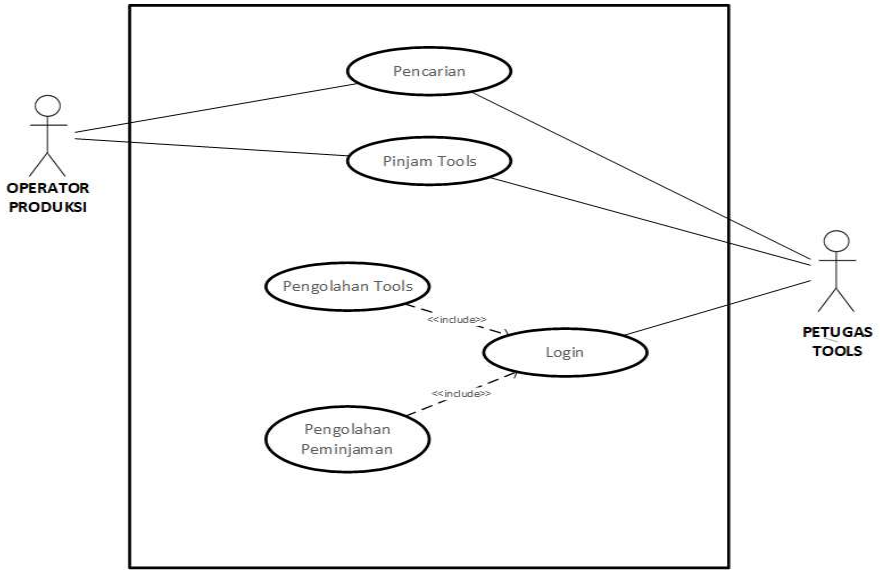
Saat ini sebagian besar para perancang sistem informasi dalam menggambarkan informasi dengan memanfaatkan UML diagram dengan tujuan utama untuk membantu tim proyek berkomunikasi, mengeksplorasi potensi desain, dan memvalidasi rancangan arsitektur perangkat lunak atau pembuat program. Secara filosofi UML, diilhami oleh konsep yang telah ada yaitu konsep pemodelan

Object Oriented karena konsep ini menganalogikan sistem seperti kehidupan nyata yang didominasi oleh objek dan digambarkan atau dinotasikan dalam simbol-simbol yang cukup spesifik [17].

UML memiliki beberapa diagram antara lain : *use case diagram, class diagram, statechart diagram, activity diagram, sequence diagram, collaboration diagram, component diagram, deployment diagram*.

2.19.1. *Use case diagram*

Use case diagram merupakan sebuah gambaran fungsionalitas sebuah sistem. Sebuah *use case* merepresentasikan interaksi antara aktor dengan sistem. *Use case* sangat menentukan karakteristik sistem yang sedang dibuat. Seorang/sebuah aktor adalah sebuah entitas manusia atau mesin yang berinteraksi dengan sistem untuk melakukan pekerjaan-pekerjaan tertentu [17]. Contoh *Use case diagram* dapat dilihat pada Gambar 2.16 Contoh *Use case diagram*.

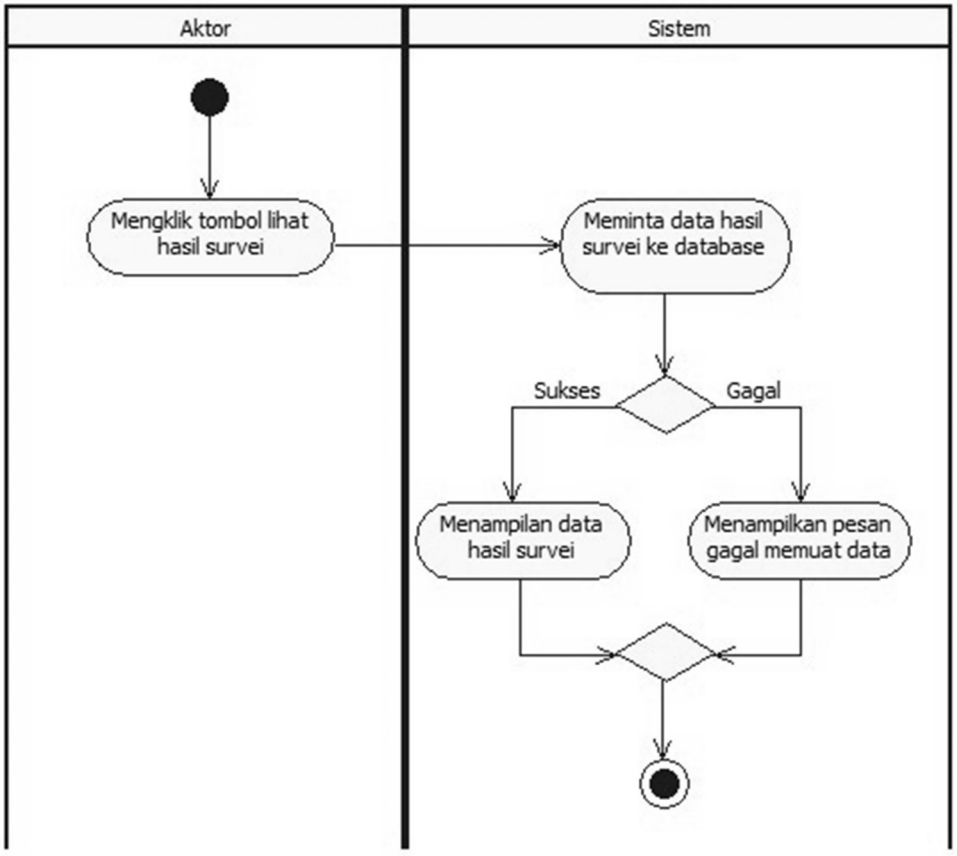


Gambar 2.16 Contoh *Use case diagram*

2.19.2. *Activity diagram*

Activity diagram merupakan state diagram khusus, dimana sebagian besar state adalah *action* dan sebagian besar transisi di-*trigger* oleh selesainya state sebelum (*internal processing*). Oleh karena itu *activity diagram* tidak menggambarkan *behavior internal* sebuah sistem (dan interaksi antar subsistem) secara eksak, tetapi lebih menggambarkan proses-proses dan jalur-jalur aktivitas

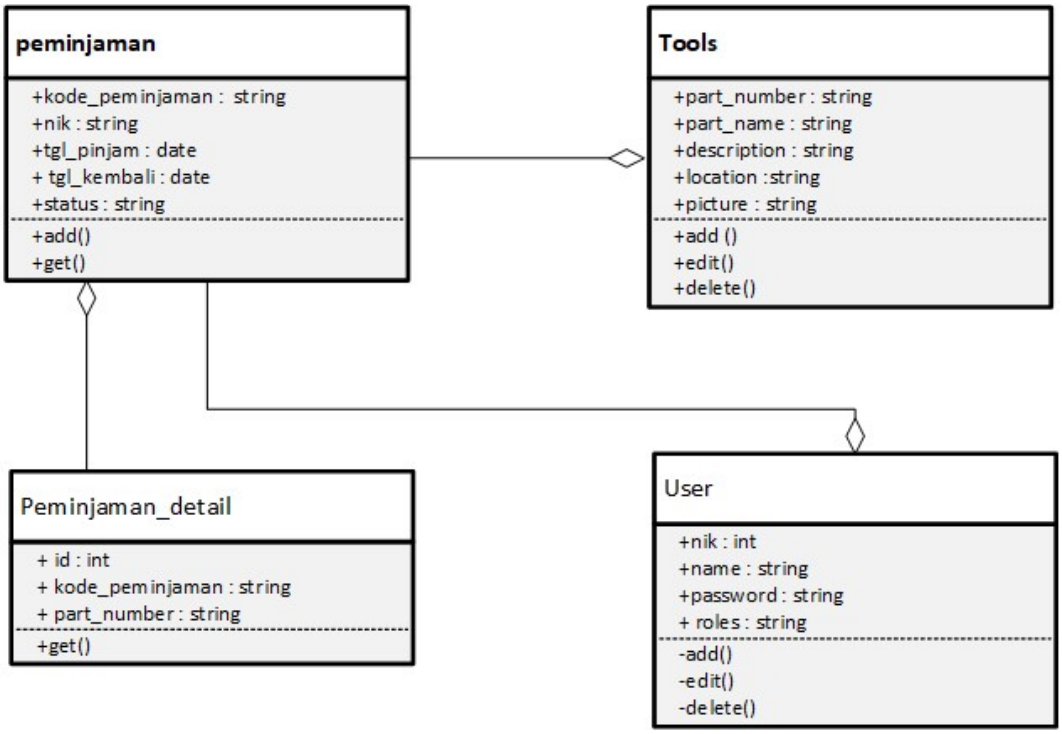
dari level atas secara umum. Sebuah aktivitas menggambarkan proses yang berjalan, sementara *use case* menggambarkan bagaimana aktor menggunakan sistem untuk melakukan aktivitas [17]. Conroh *Activity diagram* dapat dilihat pada Gambar 2.17 Contoh *Activity diagram*.



Gambar 2.17 Contoh *Activity diagram*

2.19.3. *Class diagram*

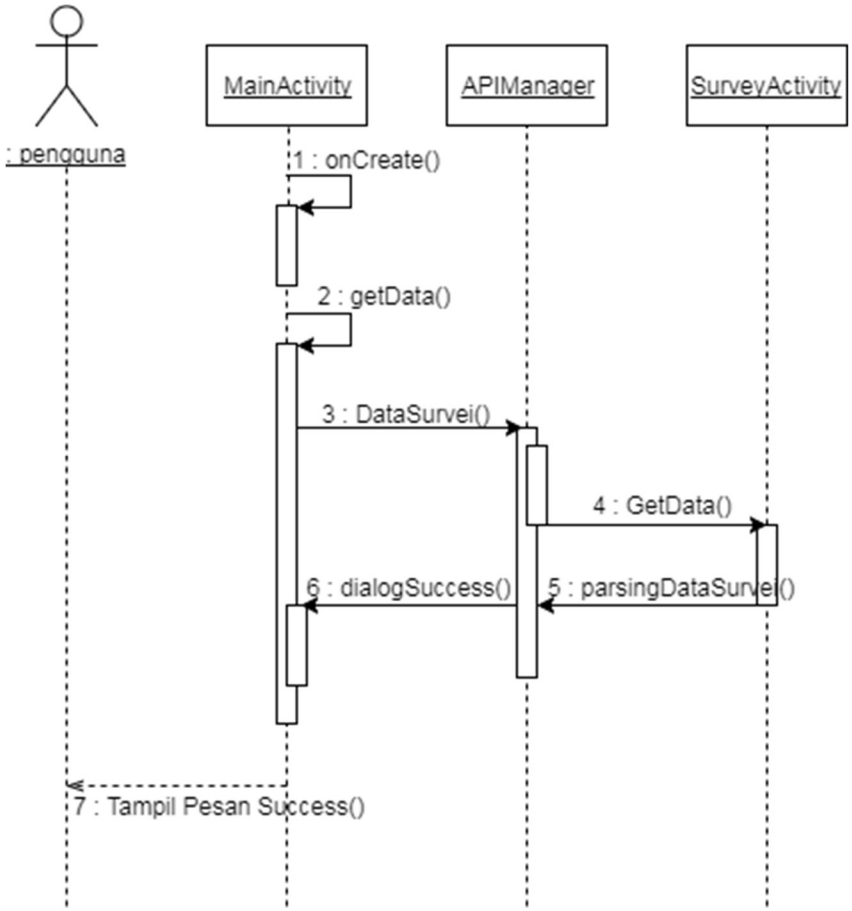
Class merupakan inti dari pengembangan dan desain berorientasi objek. *Class* menggambarkan keadaan (atribut/properti) suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (metoda/fungsi). *Class diagram* menggambarkan struktur dan deskripsi *class, package* dan objek yang berhubungan satu sama lain seperti *containment*, asosiasi, dan lain-lain. Berikut adalah simbol-simbol yang ada pada *class diagram* [17]. Contoh *Class diagram* dapat dilihat pada Gambar 2.18 Contoh *Class diagram*.



Gambar 2.18 Contoh *Class diagram*

2.19.4. *Sequence diagram*

Sequence diagram menggambarkan interaksi antar objek di dalam dan di sekitar sistem (termasuk pengguna, *display*, dan sebagainya) berupa *message* yang digambarkan terhadap waktu. *Sequence diagram* terdiri antar dimensi vertikal (waktu) dan dimensi horizontal (objek-objek yang terkait). *Sequence diagram* biasa digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai respon dari sebuah *event* untuk menghasilkan *output* tertentu. Diawali dari apa yang men-*trigger* aktivitas tersebut, proses dan perubahan apa saja yang terjadi secara internal dan *output* apa yang dihasilkan [16]. Conto *Sequence diagram* dapat dilihat pada Gambar 2.19 Contoh *Sequence diagram*.



Gambar 2.19 Contoh *Sequence diagram*

2.20. Apache

Apache merupakan web server yang dapat di jalankan di berbagai sistem operasi yang berbeda seperti UNIX, BSD, Linux, Windows dan Novell Netware yang bertugas untuk melayani fasilitas web dengan menggunakan *protocol* HTTP.

Apache sendiri bersifat *open source*, yang artinya dapat digunakan secara bebas oleh semua pengguna. Apache dikenal memiliki fitur-fitur canggih seperti penanganan pesan kesalahan yang dapat di konfigurasi serta otentikasi berbasis data dan memiliki tampilan GUI yang memungkinkan penangan server menjadi mudah.

Web Server Apache akan bekerja saat pengguna mengetikkan *request* melalui protokol http:// untuk membuat suatu halaman. Apache akan menjawab permintaan yang diketikan pengguna dan kemudian menampilkan halaman yang diminta. Dalam hal ini Apache punya peranan penting untuk menjalankan sebuah *request* halaman bahasa pemrograman.

2.21. XAMPP

Web server merupakan sebuah bentuk server yang khusus digunakan untuk menyimpan halaman *website* atau *homepage*. Apache merupakan turunan dari *web server* yang dikeluarkan oleh NSCA yaitu NSCA HTTPd sekitar tahun 1995. Pada dasarnya, Apache adalah "APatCHy" (*patch*) dan pengganti dari NCSA HTTPd. Apache web server merupakan tulang punggung permintaan dari *client*yang menggunakan *browser*, seperti Netscape Navigator, Internet Explorer, Mozilla, lynx dan lain-lain. *Web Server* dalam berkomunikasi dengan kliennya menggunakan protokol HTTP (*Hyper Text Transfer Protocol*). Apache berada di bawah GNU, *General Public License* yang bersifat *free* sehingga Apache dapat di *download* gratis pada alamat http://www.apache.org. Adapun pertimbangan dalam memilih Apache adalah :

1. Apache termasuk dalam kategori *free software* (software gratis).
2. Instalasi apache sangat mudah.
3. Mampu beroperasi pada banyak *platform* sistem operasi, seperti Linux, Windows dan lain-lain.

Apache Web Server merupakan *web server* yang bersifat *open source* dan mempunyai *performance* yang sangat bagus, fleksibel dan mendukung berbagai macam *platform* sistem operasi seperti Windows NT/9x, UNIX, Netware 5x, OS/2 dan berbagai macam sistem operasi lainnya. Apache sangat cepat sekali mengeluarkan *update* terbarunya, sehingga mengurangi munculnya *bugs* dan kelemahan program.

2.22. *Hyper text markup language* (HTML)

HTML adalah bahasa standar penulisan dokumen web. Semua informasi yang akan diletakkan diweb menggunakan format penulisan HTML. *File* HTML adalah *file* teks yang dilengkapi simbol-simbol untuk keperluan *display*. Simbol-simbol tadi disebut *tag*. HTML kependekan dari *Hyper text markup language*. Dokumen HTML adalah *file* teks murni yang dapat dibuat dengan *editor* teks sembarang. Dokumen ini dikenal sebagai *webpage*. Dokumen HTML merupakan dokumen

yang disajikan dalam *browser web surfer*. Dokumen ini umumnya berisi informasi ataupun *interface* aplikasi didalam internet.

Dokumen HTML disusun oleh elemen-elemen. Elemen merupakan istilah bagi komponen-komponen dasar pembentuk dokumen pembentuk HTML. Beberapa contoh HTML adalah: *head*, *body*, *table*, *paragraph*, dan *list*. Untuk menandai berbagai elemen dalam suatu dokumen HTML, dapat menggunakan *tag*. *Tag* HTML terdiri atas sebuah kurung sudut kiri (<. Tanda lebih kecil), sebuah nama *tag*, dan sebuah kurung sudut kanan (>, tanda lebih besar). *Tag*umumnya berpasangan (misalnya <H1> dengan </H1>, *tag* yang berpasangan selalu diawali dengan karakter garis miring (/). *Tag-tag* yang pertama menunjukan *tag* awal yang berarti awal elemen, dan yang kedua menunjukan tag akhir, berarti akhir elemen.

Elemen yang dibutuhkan untuk membuat suatu dokumen HTML dinyatakan dengan *tag*<html>,<head>, dan <body> berikut *tag-tag* pasanganya. Setiap dokumen terdiri atas *tag head* dan *body*. Elemen *head* berisi informasi tentang dokumen tersebut, dan elemen *body* berisi tentang teks yang sebenarnya yang tersusun dari link, grafik, paragraf, dan elemen lainya.

2.23. *Cascading style sheet* (CSS)

Cascading style sheet (CSS) adalah suatu bahasa *stylesheet* yang digunakan untuk mengatur tampilan suatu dokumen yang ditulis dalam bahasa *markup*. Penggunaan yang paling umum dari CSS adalah untuk memformat halaman web yang ditulis dengan HTML dan XHTML. Walaupun demikian, bahasanya sendiri dapat dipergunakan untuk semua jenis dokumen XML termasuk SVG dan XUL. Spesifikasi CSS diatur oleh *World Wide Web Consortium* (W3C).

2.24. *Hypertext preprocessor* (PHP)

Menurut *Firdaus* (2007:18) PHP (*Hypertext preprocessor*) merupakan bahasa yang hanya dapat berjalan pada server yang hasilnya dapat ditampilkan pada klien. Dalam mengeksekusi kode PHP pada sisi server (disebut *server side*) berbeda dengan mesin maya Java yang mengeksekusi program pada sisi klien (*client side*). Proses eksekusi kode PHP yang disisipkan pada halaman HTML.

PHP merupakan bahasa standar yang digunakan dalam dunia *website*. PHP adalah bahasa pemograman yang berbentuk *script* yang diletakan didalam server web. Sekitar tahun 1994, Rasmus Lerdorf telah meletakan bersama *Perl script* untuk membuat siapa yang telah melihat resumenya terkesan. Kemudian sedikit demi sedikit *user* mulai menyukai *script* ini.

2.25. *Codeigniter*

Codeigniter merupakan aplikasi *open source* yang berupa *framework* (kerangka kerja) PHP dengan model MVC (*Model*, *View*, *Controller*) untuk membangun *website* dinamis dengan menggunakan PHP. *Codeigniter* memudahkan pengembang web untuk membuat aplikasi web dengan cepat mudah dibandingkan dengan membuatnya dari awal. *Codeigniter* dirilis pertama kali pada 28 Februari 2006.

Adapun beberapa keuntungan dari menggunakan *framework Codeigniter* adalah [18]:

1. Gratis
2. Ditulis menggunakan PHP4
3. Berukuran kecil
4. Menggunakan konsep MVC (*Model, View,* dan *Controller)*
5. *URL* (*Uniform Resource Locator)* yang sederhana
6. Memiliki paket *library* yang lengkap
7. Tidak memerlukan *template engine*
8. Dokumentasi lengkap dan jelas

2.26. *Ionic Framework*

Ionic adalah sebuah *framework* dapat digunakanuntuk membangun sebuah aplikasi *mobile* dengan menggunakan teknologi weh seperti HTML5, CSS dan JavaScript. Tetapi bukan hanya itu yang dapat dilakukan oleh *Ionic*. *Ionic* menyediakan komponen – komponen yang dapat digunakan oleh pengembang untuk mengakses fitur – fitur aplikasi *mobile* layaknya aplikasi *mobile* sebenarnya (*native*). Beberapa fitur – fitur seperti *gesture, popups, modals* semuanya

disediakan *ionic* untuk digunakan oleh pengembang untuk melengkapi kebutuhan aplikasi yang diperlukan.

Framework ionic dibangun menggunakan AngularJS, yang merupakan sebuah *framework* JavaScript yang telah banyak digunakan. Sebelumnya, membangun arsitektur aplikasi – aplikasi hybrid sangat sulit untuk dilakukan, tetapi dengan menggunakan Angulaer, kita dapat membuat aplikasi – aplikasi *mobile* menggunakan sebuah teknik yang dinamakan SPA (*Single Page Application*). Angular juga menjadikan aplikasi – aplikasi yang dibangun lebih mudah dikelola untuk pengembangan dan bekerja secara *team work* karena *ionic* memungkinkan pengembang untuk dapat dengan mudah menambah, mengkostumisasi fitur – fitur atau library – library aplikasi.

2.26.1. Fitur *Ionic*

Ionic menyediakan beberapa fitur yang dapat membantu pengembang untuk membuat aplikasi hybrid dengan fungsi dan tampilan yang menarik dalam waktu singkat. Fitur – fitur dari *ionic* ada dalam 3 kategori, seperti berikut :

1. Fitur CSS

Ionic dihadirkan dengan CSS library yang menyediakan pengembang untuk membuat *User Interface* yang menarik. CSS *Style* pada *ionic* di *generate* menggunakan SASS, sebuah *preprocessor* CSS yang digunakan untuk manipulasi yang lebig *advance.*

2. Fitur JavaScript

Fitur JavaScript merupakan hal yang paling penting untuk membangun aplikasi menggunakan *framework Ionic*. Fitur-fitur ini memungkinkan pengembang untuk mengkostumisasi aplikasi atau bahkan menyediakan fungsi-fungsi pembantu yang dapat digunakan untuk memberikan kemudahan dalam pembuatan aplikasi.

3. *Ionic* CLI

Ionic CLI adalah alat yang sangat penting dalam *framework Ionic* yang memungkinkan pengembang atau *programmer* untuk menggunakan perintah-perintah *Ionic* dengan menggunakan *command line / terminal.* Dengan alat ini juga kita dapat mengakses fitur-fitur *Ionic* yang membuat pengembang aplikasi kita menjadi lebih singkat. Bisa dibilang bahwa bagian ini merupakan bagian paling

penting dari *ionic* dan juga merupakan fitur yang akan paling banyak digunakan dalam pengembangan aplikasi.