BAB 2
TINJAUAN PUSTAKA

2.1 Profil Instansi

Profil instansi membahas bagaimana instansi ini berdiri dan berkembang hingga sampai sekarang ini. Ditinjau dari sejarah, visi dan misi, tujuan dan sasaran, struktur organisasi, deskripsi tugasnya di instansi dan lain sebagainya yang menjadi hal yang wajib ada.

2.1.1 Sejarah Instansi

Dinas Kelautan dan Perikanan Provinsi Jawa Barat berdasarkan PERDA Provinsi Jawa Barat Nomor 15 Tahun 2001 merupakan bagian dari Pemerintah Daerah Provinsi Jawa Barat, melaksanakan tugas pokok sebagian pemerintah daerah di bidang perikanan dalam merumuskan kebijaksanaan operasional dan eksploitasi kelautan serta melaksanakan kewenangan desentralisasi provinsi dan kewenangan yang dilimpahkan kepada Gubernur. Sedangkan fungsi yang dijalankan Dinas Perikanan Provinsi Jawa Barat dalam melaksanakan tugas pokok tersebut adalah melaksanakan :

A. Perumusan kebijaksanaan operasional pengelolaan perikanan dan eksplorasi kelautan.
B. Pelaksanaan pelayanan umum pengelolaan perikanan dan eksplorasi kelautan.
C. Fasilitasi pelaksanaan pengelolaan perikanan dan eksplorasi kelautan meliputi kegiatan program, produksi, dan konservasi, prasarana, usaha, eksplorasi keluatan serta unit pelaksanaan teknis dinas serta penyelengaraan urusan ketatausahaan Dinas.

2.1.2 Visi dan Misi Instansi

- Visi Instansi

Dinas perikanan telah menetapkan visi : " Prima Dalam Pelayanan Menuju Masyarakat Perikanan Jawa Barat Yang Tangguh,Dinamis Dan Mandiri ". Visi dinas perikanan tersebut mengandung beberapa pemahaman sebagai berikut:

A. Prima, berarti sempurna atau sangat baik.
B. Tangguh, berarti kuat, sukar dikalahkan, tidak lemah, tabah dan tahan menderita, serta kukuh.
C. Dinamis, berarti selalu ada peningkatan.
D. Mandiri, berarti dapat memiliki wewenang untuk menentukan ketentuan dan keputusan.

- Misi Instansi

Agar visi Dinas Kelautan dan Perikanan dapat terlaksana dan berhasil dengan baik maka ditetapkan misi Dinas sebagai berikut:

A. Meningkatkan kualitas dan produktivitas sumberdaya manusia perikanan dan kelautan yang berdaya saing.
B. Mendorong peningkatan pendapatan masyarakat melalui pemanfaatan sumberdaya perikanan yang bernilai ekonomis dengan penerapan teknologi berwawasan lingkungan.
C. Meningkatkan produk perikanan dan kelautan yang berkualitas untuk pemenuhan kebutuhan gizi masyarakat dan bahan baku industri secara berkelanjutan.
D. Meningkatkan pelestarian sumberdaya perikanan dan kelautan.

2.1.3 Tujuan dan Sasaran

- Tujuan

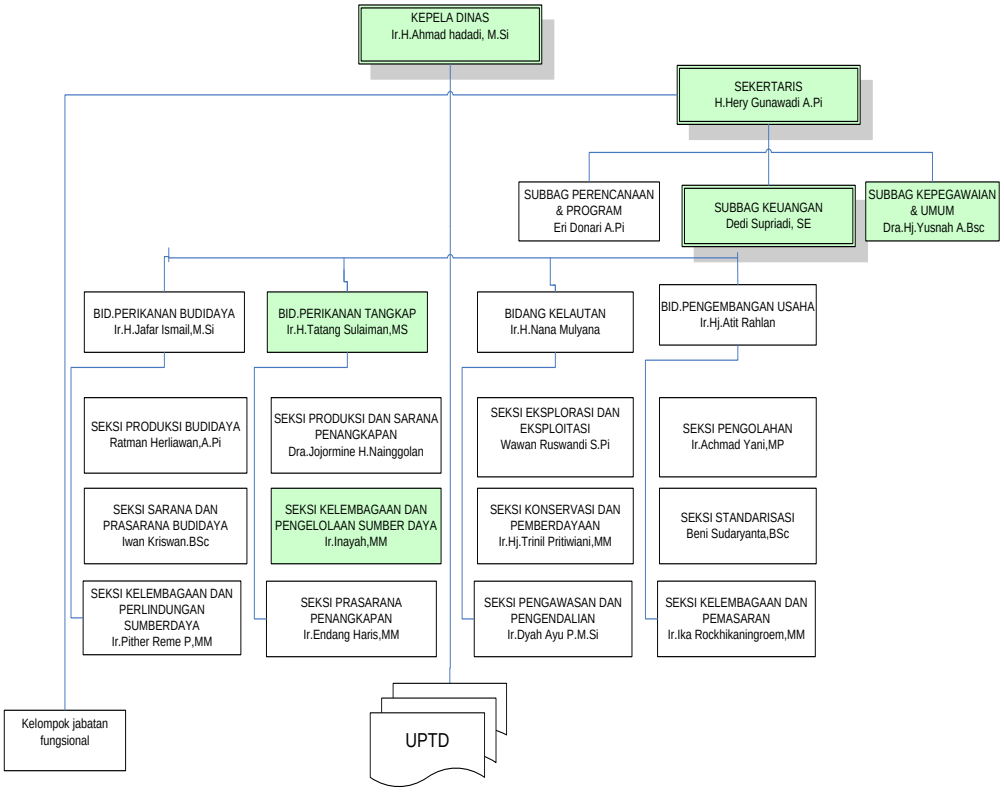
Tujuan utama dari pembangunan perikanan dan kelautan adalah memanfaatkan sumberdaya ikan dan kelautan untuk memenuhi kebutuhan gizi manusia dan meningkatkan kesejahteraan hidup masyarakat, khususnya yang terkait dengan peningkatan pendapatan dan peningkatan devisa negara.

Untuk mewujudkan Visi-Misi di atas, Dinas Kelautan dan Perikanan Provinsi Jawa Barat memposisikan diri sebagai berikut:

A. Inovator dalam pengembangan teknologi perikanan dan kelautan.
B. Motivator dalam pemberdayaan ekonomi masyarakat.
C. Fasilitator dalam pengembangan sarana prasarana.
D. Mediator dalam penyediaan permodalan dan pengembangan usaha.
E. Dinamisator tumbuh kembangnya peran sertanya masyarakat dalam pembangunan perikanan yang berkelanjutan dan berwawasan lingkungan, sekaligus mengantisipasi perubahan perkembangan situasi dan kondisi regional, nasional, lingkungan strategis serta kecenderungan global dalam perubahan yang begitu cepat.

2.1.4 Struktur Organisasi Instansi

Adapun struktur organisasi dari Dinas Kelautan dan Perikanan Provinsi Jawa Barat adalah sebagai berikut:



Gambar 2.1 Struktur Organisasi Instansi

2.1.5 Deskripsi Jabatan

A. Kepala Dinas

1. Kepala Dinas mempunyai tugas pokok memimpin, mengkoordinasikan dan mengendalikan pelaksanaan kegiatan tugas pokok dinas.
2. Untuk menyelenggarakan tugas pokok sebagaimana yang dimaksud pada ayat (1) pasal ini, Kepala Dinas mempunyai fungsi:
   a. Pengaturan dan koordinasi pelaksanaan kebijakan operasinal bidang perikanan dan eksplorasi kelautan.
   b. Fasilitasi dan pengendalian pelaksanaan pengelolaan perikanan.
   c. Penyelenggaraan koordinasi dan kerjasama dalam rangka pelaksanaan tugas pokok Dinas.

B. Wakil Kepala Dinas

1. Wakil Kepala Dinas mempunyai tugas pokok mengkoordinasikan perencanaan dan program, pelaksanaan, evaluasi dan pelaporan serta tugas lain yang dilimpahkan oleh Kepala Dinas.
2. Untuk menyelenggarakan tugas pokok sebagaimana yang dimaksud dalam ayat (1) pasal ini, Wakil Kepala Dinas mempunyai fungsi:
   a. Pengkoordinasian dan pengendalian kegiatan internal Dinas.
   b. Pembinaan administrasi kegiatan Dinas dan UPTD.
   c. Pelaksanaan tugas operasional lain sesuai dengan pelimpahan Kepala Dinas.
3. Rincian Tugas Wakil Kepala Dinas:
   a. Mengkoordinasi kegiatan internal Dinas.
   b. Melaksanakan pembinaan administrasi kegiatan Dinas dan UPTD yang meliputi pembinaan administrasi kepegawaian, keuangan, perlengkapan, umum dan kesisteman.
   c. Mengkoordinasikan perumusan perencanaan strategis (RENSTRA) dan Laporan Akuntabilitas Kinerja Instansi Pemerintah (LAKIP) Dinas.
   d. Melaksanakan tugas opersional lain sesuai dengan pelimpahan Kepala Dinas.

C. Sekertariat

1. Sekertariat mempunyai tugas pokok menyelenggarakan koordinasi perencanaan dan program Dinas, pengkajian perencanaan dan program, pengelolaan keuangan, kepegawaian dan umum.
2. Rincian Tugas Sekertariat:
   a. Menyelenggarakan pengkajian dan koordinasi perencanaan dan program Dinas.
   b. Menyelenggarakan pengkajian perencanaan dan program sekertariat.
   c. Menyelenggarakan pengelolaan administrasi.
   d. Menyelenggarakan pengkajian anggaran belanja.

e. Menyelenggarakan penata ushaan, kelembagaan dana ketata laksanaan.
f. Menyelenggarakan pengelolaan urusan rumah tangga dan perlengkapan.

D. Bidang Perikanan Tangkap

1. Bidang perikanan tangkap mempunyai tugas pokok menyelenggarakan pengkajian bahan kebijakan teknis dan fasilitasi perikanan tangkap.
2. Rincian Tugas Bidang Perikanan Tangkap:
   a. Menyelenggarakan pengkajian program kerja bidang perikanan tangkap.
   b. Menyelenggarakan pengkajian bahan kebijakan teknis perikanan tangkap.
   c. Menyelenggarakan pengkajian bahan fasilitas produksi dan sarana penangkapan.
   d. Menyalengarakan bahan koordinasi bidang perikanan tangkap.
   e. Menyelenggarakan telaah staf sebagai pertimbangan pengambilan kebijakan.
   f. Menyelenggarakan koordinasi dengan Badan Koordinasi Pemerintah dan Pembangunan Wilayah dalam pelaksanaan tugas di Kabupaten atau Kota.
   g. Menyelenggarakan pelaporan dan evaluasi kegiatan Bidang Perikanan Tangkap.
   h. Menyelenggarakan koordinasi dengan unit kerja terkait.
3. Bidang Perikanan Tangkap Membawakan:

a. Seksi produksi dan sarana penangkapan mempunyai tugas pokok melaksanakan penyusunan bahan kebijakan teknis dan fasilitasi produksi dan sarana penangkapan.

   Rincian Tugas Seksi Produksi dan Sarana Penangkapan:

   1. Melaksanakan penyusunan program kerja seksi produksi dan saran penangkapan.

2. Melaksanakan penyusunan bahan kebijakan produksi dan sarana penangkapan.
3. Melaksanakan penyusunan bahan kebijakan pemberdayaan nelayan kecil di wilayah laut dan perairan umum.
4. Melaksanakan penyusunan bahan kebijakan penempatan dan pemanfaatan alat bantu penangkapan ikan di wilayah laut.

b. Seksi Kelembagaan dan Pengelolaan Sumberdaya Perikanan mempunyai tugas pokok pelaksanaan penyusunan bahan kebijakan teknis dan fasilitas kelembagaan dan pengelolaan sumberdaya perikanan tangkap.

   Rincian Tugas Seksi Kelembagaan dan Pengelolaan Sumberdaya Perikanan:

   1. Melaksanakan penyusunan program kerja seksi kelembagaan dan pengelolaan sumberdaya perikanan.
   2. Melaksanakan penyusunan bahan kebijakan kelembagaan dan sumberdaya meliputi inventarisasi, identifikasi dan analisis data serta penyusunan sistem informasi potensi sumberdaya manusia , kelembagaan dan pengelolaan sumberdaya perikanan tangkap.
   3. Melaksanakan penyusunan bahan fasilitas dan kerjasama pengembangan sumberdaya manusia serta kelembagaan nelayan.
   4. Melaksanakan penyusunan bahan kebijakan konservasi dan rehabilitasi sumberdaya perikanan tangkap secara berkelanjutan.
   5. Melaksanakan penyusunan bahan inventarisasi dan kerusakan fisik dan nilai kerugian sebagai akibat pencemaran dan kerusakan sumberdaya perikanan.

c. Seksi Prasarana Penangkapan mempunyai tugas pokok melaksanakan penyusunan bahan kebijakan teknis dan fasilitasi prasarana penangkapan.

   Rincian Tugas Seksi Prasarana Penangkapan:

   1. Melaksanakan penyusunan program kerja Seksi Prasrana Penangkapan.

2. Melaksanakan penyusunan bahan kebijakan pembangunan, pemanfaatan dan pemeliharaan prasarana penangkapan.
3. Melaksanakan penyusunan bahan kebijakan pengembangan jasa-jasa pelabuhan perikanan/pangkalan pendaratan ikan.
4. Melaksanakan pelaksanaan dan evaluasi kegiatan seksi prasarana pengkapan.
5. Melaksanakan penyusunan bahan telaahan staf sebagai bahan pertimbangan kebijakan.

2.1.6 Aspek Kegiatan Instansi

Aspek kegiatan yang dilakukan di Dinas Perikanan Dan Kelautan Provinsi Jawa Barat khususnya di Bidang Perikanan Tangkap adalah:

A. Menyelenggarakan pengkajian program kerja bidang perikanan tangkap.
B. Menyelenggarakan pengkajian bahan kebijakan teknis perikanan tangkap.
C. Menyelenggarakan pengkajian bahan fasilitasi produksi dan sarana penangkapan.
D. Menyelenggarakan pengkajian bahan fasilitasi kelembagaan dan pengelolaan sumberdaya perikanan tangkap.
E. Menyelenggarakan pegkajian bahan fasilitasi prasarana penangkapan.

2.2 Landasan Teori

Landasan teori merupakan penjelasan berbagai konsep dasar dan teori-teori yang berkaitan dalam pembangunan aplikasi M-Nelayan berbasis android.

Dimulai dengan teori nelayan, teori web dan *mobile technology*, teori informatika umum sampai dengan *tools* pembangun aplikasi, diantaranya basis data, jaringan, internet,web, HTTP, web *service*, web server, sistem operasi Android, UML, *eclipse*, notepad plus-plus, MySQL, WAMP, dan mozilla firefox. Hal ini digunakan untuk mengetahui gambaran tempat

penelitian Tugas Akhir serta menjelaskan teori-teori yang berhubungan dengan masalah yang dibahas.

2.2.1 Nelayan

Nelayan adalah golongan dalam masyarakat yang menggantungkan hidupnya dari menangkap ikan. Ikan hasil tangkapan tidak lagi sekedar untuk memenuhi kebutuhan hidupnya sendiri untuk keluarga akan tetapi sebagian besar untuk tujuan dipertukarkan dengan kebutuhan hidup lainnya secara langsung atau dijual secara tunai di pasar. Mereka yang hanya mencari ikan untuk sekedar lauk-pauk tidak digolongkan sebagai nelayan. Disepanjang garis pantai indonesia yang sangat panjang, hidup jutaan rakyat yang menggantungkan hidupnya dari menangkap ikan dan karena kemampuannya berlayar dengan ilmu navigasi turun temurunnya, mereka juga ambil bagian dalam perniagaan secara terbatas menghubungkan daerah-daerah yang terisolasi oleh lautan. Mereka secara tradisional telah menjadi penghubung pertama antara Jawa dengan Sumatera, Kalimantan, Sulawesi, Kepulauan Maluku, Papua, Nusa Tenggara atau sebaliknya. Kaum nelayan di pesisir menghadapi penindasan dan penghisapan feodal bahkan kapitalis dari para pemilik kapital besar seperti pemilik kapal besar dan alat tangkap besar serta modern. Nelayan ini juga mengahadapi maslah peribaan yang sangat berat terutama pada saat angin kencang dan gelombang tinggi di mana intensitas melaut sangat terbatas.(2)

Daratan indonesia juga memilki rawa-rawa yang luas yang didalamnya juga mengandung ikan berbagai jenis dalam jumlah besar. Bahkan karena keterbatasan tanah, mereka juga membangun rumah di atas rawa-rawa pasang surut tersebut. Sekalipun ada aktivitas menanam padi dan palawija di tanah yang lebih tinggi atau pada saat air surut, menangkap ikan adalah pendapatan utama mereka untuk menghidupi keluarganya. Ini juga digolongkan sebagai nelayan.

Di dalam masyarakat setengah jajahan dan setengah feodal yang selalu menciptakan kelas-kelas dalam masyarakat(3), nelayan-nelayan ini juga telah terbagi dalam lapisan-lapisan, utamanya sebagai berikut:

1. Nelayan kaya adalah nelayan yang memiliki dan menggunakan kapal motor, jala ikan yang besar, dan perlengkapan pemancingan ikan yang lengkap. Mereka mempekerjakan nelayan miskin dan pendapatannya lebih dari cukup bagi keluarganya.
2. Nelayan sedang yaitu mereka yang memiliki kapal sedang, jala ikan berukuran sedang, dan memiliki perlengkapan lainnya yang lebih sederhana jika dibandingkan dengan nelayan kaya. Penghasilan dari hasil tangkapan mereka hanya cukup bagi keluarganya. Dan dalam keadaan krisis ekonomi maupun karena cuaca buruk mereka dapat jatuh menjadi nelayan kecil.
3. Nelayan kecil yaitu mereka memiliki sampan kecil tradisional yang hanya mampu mengarungi lautan yang dekat dengan pantai di mana ikan sudah jauh berkurang. Penghasilannya sangat kecil sehingga untuk menutupi kekurangan pendapatannya, mereka bekerja dan berutang pada nelayan kaya.

2.2.2 Web dan *Mobile Technology*

Web *mobile* bertujuan untuk mengakses layanan data secara *wireless* dengan menggunakan perangkat *mobile* seperti handphone, pda dan perangkat portable yang tersambung ke sebuah jaringan telekomunikasi selular. Web *mobile* yang diakses melalui perangkat *mobile* seperti sebuah handphone yang memiliki sebuah layar dengan ukuran yang terbatas ataupun beberapa keterbatasan pada sebuah perangkat *mobile*. Pada bulan Mei 2005 *mobile* di keluarkan oleh W3C dengan nama *Mobile* Web *Initiative* (MWI) dengan tujuan membuat web dapat diakses dari sebuah perangkat *mobile* secara sederhana seperti mengakses web dari sebuah komputer dekstop. Dalam membuat sebuah *mobile* web membutuhkan implementasi untuk perbaikan dari segi *interoperability, usability* dan

accessibility pada sebuah *mobile* web. *Mobile* Web umumnya berukuran ringan disetiap halamannya yang ditulis dengan *Extensible Hypertext Markup Language* (XHTML) atau *Wireless Markup Language* (WML) untuk mengirimkan konten ke perangkat *mobile*. Selain itu beberapa teknik seperti dengan menggunakan *Adobe Flash Lite* atau Sun's J2ME yang memungkinkan untuk membuat perangkat *mobile* yang lebih bervariasi. *Mobile* web turut berkembang seiring dengan perkembangan teknologi web. Contohnya saat teknologi Web 2.0 muncul dimana teknologi web 2.0 ke sebuah *mobile* device. Dapat digambarkan seperti membuka sebuah web 2.0 (contohnya *Friendster, Flicker,m blog*) kedalam sebuah perangkat *mobile.* Tetapi secara sederhananya perangkat *mobile* berbeda dengan komputer desktop, sehingga banyak faktor yang menjadi batasan yang patut di perhatikan ketika membangun sebuah *mobile* web, salah satunya adalah ketersediaannya data secara *real time* atau update berkala ketika terjadi perubahan data pada web. Maka untuk membangun sebuah *mobile* 2.0 yang mempertimbangkan proses update data secara berkala dapat dibangun dengan menggunakan teknik *AJAX* ataupun *Flash Lite.*(4)

2.2.3 Pengertian Data

Data adalah representasi dari suatu fakta, yang dimodelkan dalam bentuk gambar, kata dan angka. Manfaat data adalah sebagai satuan representasi yang dapat diingat, direkam, dan dapat diolah menjadi informasi. Karakteristiknya, data bukanlah fakta, namun representasi dari fakta. Kata sederhananya, data adalah catatan tentang fakta, atau data merupakan rekaman catatan tentang fakta. Data yang baik, adalah yang sesuai dengan faktanya. Data dapat berupa angka, huruf, gambar atau bentuk rekaman lainnya, yang dapat diakses oleh manusia, yang dapat dimasukan, disimpan dan diolah dalam komputer, atau dilewatkan pada kanal digital. Komputer merepresentasikan data dalam bentuk kode binary. Data yang bermanfaat adalah data yang lengkap mendukung pembentukan

informasi, serta akurat dan mutakhir, karena selalu diverifikasi dan diperbaharui sesuai perkembangan fakta.

2.2.4 Informasi

Informasi adalah data yang diolah menjadi bentuk yang berguna dan menjadi berarti bagi penerimanya. Kegunaan informasi adalah untuk mengurangi ketidakpastian di dalam proses pengambilan keputusan tentang suatu keadaan. Suatu informasi dikatakan bernilai bila manfaatnya lebih efektif dibandingkan dengan biaya untuk mendapatkan informasi tersebut.

Informasi merupakan sesuatu yang harus dimilki oleh suatu perusahaan atau organisasi untuk bisa tetap eksis dan bertahan, karena kurangnya informasi yang dimiliki akan mengakibatkan perusahaan atau organisasi akan hancur dan berakhir.

Menurut (MC Leod, 2002:40-41) Kualitas informasi sangat dipengaruhi atau ditentukan oleh beberapa hal yaitu:

a. Relevan *(Relevancy)*
b. Akurat *(Accurancy)*
c. Tepat Waktu *(Time liness)*
d. Ekonomis *(Economy)*
e. Efisien *(Efficiency)*
f. Ketersediaan *(Availability)*
g. Dapat dipercaya *(Reliability)*
h. Konsisten

2.2.5 Database

Database dapat diartikan sebagai kumpulan data yang terdiri atas satu atau lebih tabel yang terintegrasi satu sama lain, di mana setiap pemakai (user) diberi wewenang (otorisasi) untuk dapat mengakses (mengubah, menghapus, menganalisis, menambah, memperbaiki) data dalam tabel-tabel tersebut.

2.2.5.1 *Fungsi Database*

Adapun fungsi database diantaranya adalah sebagai berikut:

1. salah satu komponen penting dalam sistem informasi, karena merupakan dasar dalam menentukan informasi.
2. Menentukan kualitas informasi. Informasi dapat dikatakan bernilai apabila manfaatnya lebih efektif dibandingkan dengan biaya mendapatkannya.
3. Mengurangi duplikasi data *(data redudancy).*
4. Hubungan data dapat ditingkatkan *(data relatability).*
5. Mengurangi pemborosan tempat simpanan luar.

2.2.5.2 *Kriteria Database*

Dari penjelasan sebelumnya dapat disimpulkan bahwa basis data mempunyai beberapa kriteria penting, yaitu:

1. bersifat data *oriented* dan bukan *program oriented*
2. dapat digunakan oleh beberapa program aplikasi tanpa perlu mengubah basis datanya.
3. Dapat dikembangkan dengan mudah, baik volume maupun strukturnya.
4. Dapat memenuhi kebutuhan sistem-sistem baru secara mudah, dapat digunakan dengan cara-cara yang berbeda.

2.2.6 Internet

Internet atau *Interconnected Networking* merupakan dua komputer atau lebih yang saling berhubungan membentuk jaringan komputer hingga meliputi jutaan komputer di dunia, yang saling berinteraksi dan bertukar informasi.

Internet mempunyai sejarah yang sangat kompleks dan mencakup banyak aspek seperti teknologi, organisasi dan komunitas. Pengaruh internet tidak hanya terhadap bidang teknik komunikasi komputer saja tetapi, terhadap masalah sosial misalnya dengan menggunakan alat-alat

bantu *online* untuk mencapai bisnis elektronik (*e-commerce*), kepemilikan informasi dan interaksi dengan masyarakat.

Ada beberapa fasilitas dari *internet* yang sering kita temui bahkan digunakan antara lain :

1. *E-mail*
2. *World Wide Web (WWW)*
3. *Newsgroup*
4. *Telnet*
5. *Chat*
6. *File Transfer Protocol (FTP)*

2.2.6.1 *Network*

Jaringan atau *network* saat ini menjadi istilah yang sangat penting dalam dunia pengolahan data dan informasi. Pengertian sederhana dari jaringan komputer adalah sekumpulan perangkat yang dapat digunakan untuk menyimpan dan memanipulasi data elektronis dan pesan-pesan, saling terkait satu dengan yang lainnya di mana dengan cara tersebut pengguna dapat menyimpan, menggali dan saling berbagi terhadap informasi yang tersedia.

Istilah *"Network"* mengacu pada perangkat keras dan lunak yang digunakan untuk menghubungkan antar komputer satu dengan komputer lainnya, sehingga memungkinkannya untuk berkomunikasi satu sama lainnya. Sistem jaringan tidak lagi hanya akan melayani sebuah mesin besar saja. Sebaliknya jaringan-jaringan akan merupakan sarana bantu yang memungkinkan sebuah organisasi besar untuk melakukan penyesuaian yang sesuai, antara kebutuhan informasi dengan besarnya aplikasi serta investasi perangkat keras dan lunaknya.

2.2.7 Web Server

Web server adalah sebuah bentuk server yang khusus digunakan untuk menyimpan halaman *website* atau *homepage*. Komputer dapat dikatakan

web server jika komputer tersebut memiliki suatu program server yang disebut *Personal Web Server* (PWS).

Macam-macam *web server* antara lain:

1. Apache (*Open Source*)
2. Xitami
3. IIs
4. PWS (*Personal Web Server*)

Website (Situs Web) merupakan alamat (URL) yang berfungsi sebagai tempat penyimpanan data dan informasi dengan berdasarkan topik situs tertentu atau web dapat dikategorikan menjadi 2 yaitu:

a. Web statis, yaitu:

Web yang berisi atau menampilkan informasi-informasi yang sifatnya statis (tetap).

b. Web Dinamis, yaitu:

Web yang menampilkan informasi serta dapat berinteraksi dengan user yang bersifat dinamis.

2.2.8 HTTP

HTTP (*Hypertext Transfer Protocol*) merupakan protokol yang digunakan untuk mendistribusikan sistem informasi yang berbasis *hypertext.* Protokol ini merupakan protokol standar yang digunakan untuk mengakses HTML. HTTP diprakarsai oleh *World Wide Web* sistem informasi yang menyeluruh sejak tahun 1990. Apabila pada penjelajahan web dan pada alamat tertulis http://www.google.com; ini merupakan salah satu penggunaan protokol HTTP dalam web.

2.2.9 Web atau Situs

Secara terminologi, website adalah kumpulan dari halaman-halaman situs, yang biasanya terangkum dalam sebuah *domain* atau *subdomain*, yang tempatnya berada di dalam *World Wide Web* (WWW) di internet. Sebuah halaman *web* adalah dokumen yang ditulis dalam format HTML (*HyperTextMarkupLanguage*), yang hampir selalu bisa diakses melalui

HTTP, yaitu protokol yang menyampaikan informasi dari server website untuk ditampilkan kepada para pemakai melalui *webbrowser*. Semua publikasi dari website-website tersebut dapat membentuk sebuah jaringan informasi yang sangat besar.

2.2.10 Enkripsi MD5

Dalam kriptografi, MD5 (*Message-Digest algorithm* 5) ialah fungsi *hash* kriptografik yang digunakan secara luas dengan *hash* value 128-bit. Pada standard internet (RFC 1321), MD5 telah dimanfaatkan secara bermacam-macam pada aplikasi keamanan, dan MD5 juga umum digunakan untuk melakukan pengujian integritas sebuah file. MD5 didesain oleh Ronald Rivers pada tahun 1991 untuk menggantikan *hash function* sebelumnya, MD4. Pada tahun 1996, sebuah kecacatan ditemukan dalam desainnya, walau bukan kelemahan fatal, pengguna kriptografi mulai menganjurkan menggunakan algoritma lain seperti SHA-1. Pada tahun 2004, kecacatan-kecacatan yang lebih serius ditemukan menyebabkan penggunaan algoritma tersebut dalam tujuan untuk keamanan jadi makin dipertanyakan.

2.2.11 Web Service

Web *service* adalah sekumpulan *application logic* beserta objek-objek dan metode-metode yang dimilikinya yang terletak di suatu server yang terhubung ke internet sehingga dapat diakses menggunakan *protocol* HTTP dan SOAP (*Simple Object Access Protocol*). Dalam penggunaannya, web service dapat digunakan dari hanya untuk memeriksa data user yang login ke sebuah website ataupun untuk digunakan pada transaksi perbankan *online* yang rumit. Tujuan dari teknologi ini adalah untuk memudahkan beberapa aplikasi atau komponennya untuk saling berhubungan dengan aplikasi lain dalam sebuah organisasi maupun diluar organisasi menggunakan standar yang tidak terikat *platform*(*platform-neutral*) dan tidak terikat akan bahasa pemrograman yang digunakan (*language-neutral*). Hal tersebut dapat terjadi karena penggunaan XML

standar yang didukung oleh banyak perusahaan besar di dunia, yang digunakan untuk bertukar data. Selain daripada itu, penggunaan SOAP menjadikan metode-metode dari objek-objek yang ada dalam sebuah web service dapat di akses dari aplikasi lain seperti halnya aplikasi tersebut mengakses metode lokal.

2.2.12 JSON

JSON (*Java Script 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 dll. Oleh karena sifat-sifat tersebut, menjadikan JSON ideal sebagai bahasa pertukaran-data. (5)

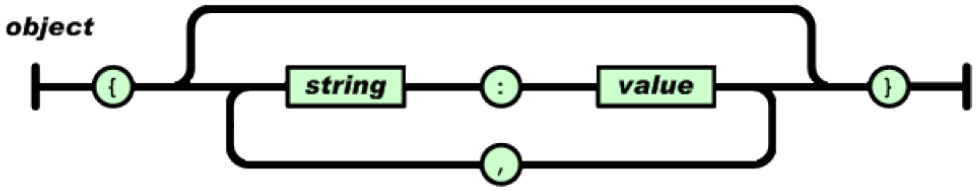
JSON terbuat dari dua struktur:

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 moderen mendukung struktur data ini dalam bentuk yang sama maupun berlainan. Hal ini pantas disebut demikian karena format data mudah dipertukarkan dengan bahasa-bahasa pemprograman yang juga berdasarkan pada struktur data ini. JSON menggunakan bentuk sebagai berikut:

1. Objek

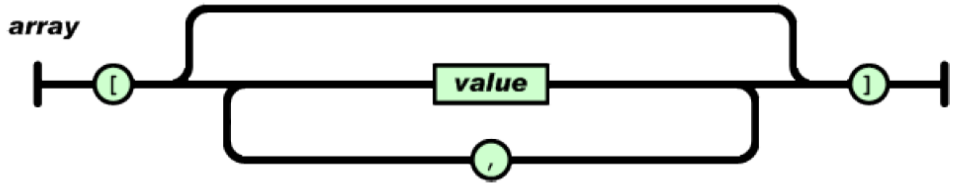
Objek adalah sepasang nama/nilai yang tidak terurutkan. Objek dimulai dengan { (kurung kurawal buka) dan diakhiri dengan } (kurung kurawal tutup). Setiap nama diikuti dengan : (titik dua) dan setiap pasangan nama/nilai dipisahkan oleh , (koma).



Gambar 2.2 Objek JSON

2. Larik

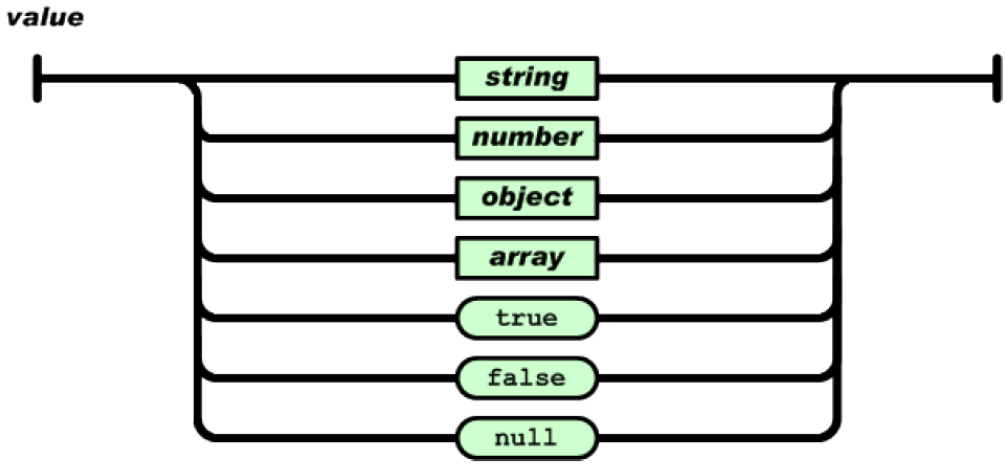
Larik adalah kumpulan nilai yang terurutkan. Larik dimulai dengan [(kurung kotak buka) dan diakhiri dengan] (kurung kotak tutup). Setiap nilai dipisahkan oleh , (koma).



Gambar 2.3 Larik JSON

3. Nilai

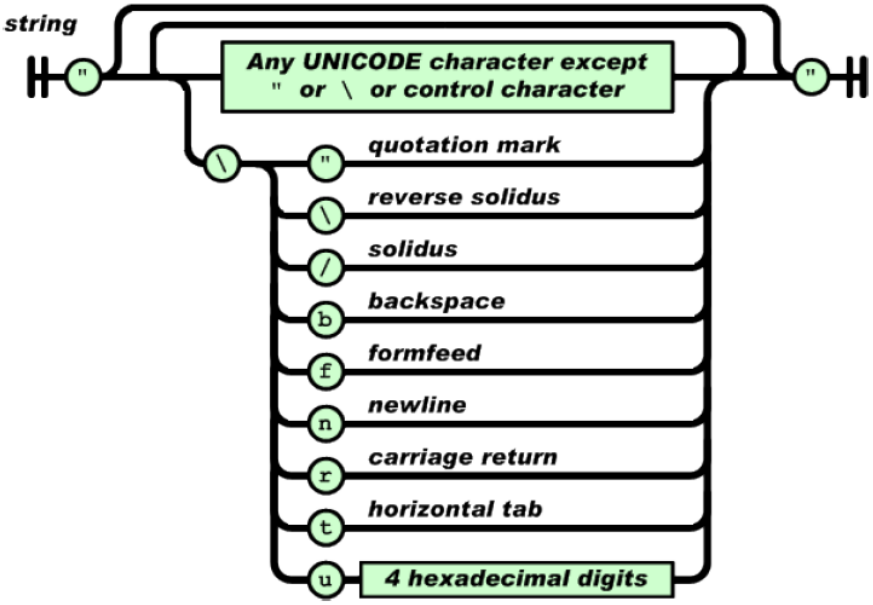
Nilai (*value*) dapat berupa sebuah string dalam tanda kutip ganda, atau angka, atau *true* atau *false* atau *null*, atau sebuah objek atau sebuah larik. Struktur-struktur tersebut dapat disusun bertingkat.



Gambar 2.4 Nilai JSON

4. String

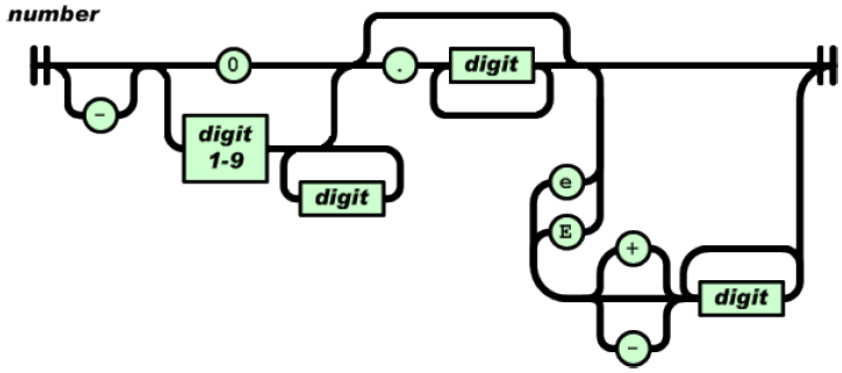
String adalah kumpulan dari nol atau lebih karakter *Unicode*, yang dibungkus dengan tanda kutip ganda. Di dalam string dapat digunakan backslash escapes "\" untuk membentuk karakter khusus. Sebuah karakter mewakili karakter tunggal pada string. String sangat mirip dengan string C atau Java.



Gambar 2.5 String JSON

5. Angka

Angka adalah sangat mirip dengan angka di C atau Java, kecuali format oktal dan heksadesimal tidak digunakan.



Gambar 2.6 Angka JSON

2.2.13 Android

Android adalah sistem operasi untuk telepon selular (*mobile)* seperti pada *smartphone* dan komputer tablet. Sistem operasi Android berbasiskan Linux yang bersifat terbuka (*opensource)* bagi pengembang untuk menciptakan aplikasi maupun mengembangkan Android dengan *tools* yang disediakan yaitu Android SDK (*Software Development Kit)* dan API (*Application Programming Interface)* dengan menggunakan bahasa pemrograman Java. Dikembangkan Oleh *Google* Inc, HTC, Intel, Motorola, *Qualcom*, *T-Mobile*, dan Nvidia yang tergabung dalam *Open Handset Alliance* (OHA). Dengan tujuan mendukung sebuah standar terbuka pada perangkat seluler. Kode-kode yang dirilis oleh Google untuk Android berada dibawah *license Apache*, sebuah *license* perangkat lunak dan standar terbuka bagi perangkat seluler.

2.2.13.1 *Sejarah*

Pada Juli 2000, Google bekerjasama dengan Android Inc., perusahaan yang berada di Palo Alto, California Amerika Serikat. Para pendiri Android Inc. bekerja pada Google, di antaranya Andy Rubin, Rich Miner, Nick Sears, dan Chris White. Saat itu banyak yang menganggap fungsi Android Inc. hanyalah sebagai perangkat lunak pada telepon seluler. Sejak saat itu muncul rumor bahwa Google hendak memasuki pasar telepon

seluler. Di perusahaan Google, tim yang dipimpin Rubin bertugas mengembangkan program perangkat seluler yang didukung oleh kernel Linux. Hal ini menunjukkan indikasi bahwa Google sedang bersiap menghadapi persaingan dalam pasar telepon seluler. Saat ini Android 4.0.3 atau Ice Cream Sandwich adalah versi terbaru dari sistem operasi Android. Sekitar September 2007 sebuah studi melaporkan bahwa Google mengajukan hak paten aplikasi telepon seluler (akhirnya Google mengenalkan Nexus One, salah satu jenis telepon pintar GSM yang menggunakan Android pada sistem operasinya. Telepon seluler ini diproduksi oleh HTC Corporation dan tersedia di pasaran pada 5 Januari 2010). Pada 9 Desember 2008, diumumkan anggota baru yang bergabung dalam program kerja Android ARM Holdings, Atheros Communications, diproduksi oleh Asustek Computer Inc, Garmin Ltd, Softbank, Sony Ericsson, Toshiba Corp, dan Vodafone Group Plc. Seiring pembentukan Open Handset Alliance, OHA mengumumkan produk perdana mereka, Android, perangkat bergerak (*mobile*) yang merupakan modifikasi kernel Linux 2.6. Sejak Android dirilis telah dilakukan berbagai pembaruan berupa perbaikan bug dan penambahan fitur baru. Telepon pertama yang memakai sistem operasi Android adalah HTC Dream, yang dirilis pada 22 Oktober 2008. Pada penghujung tahun 2009 diperkirakan di dunia ini paling sedikit terdapat 18 jenis telepon seluler yang menggunakan Android**.**(6)

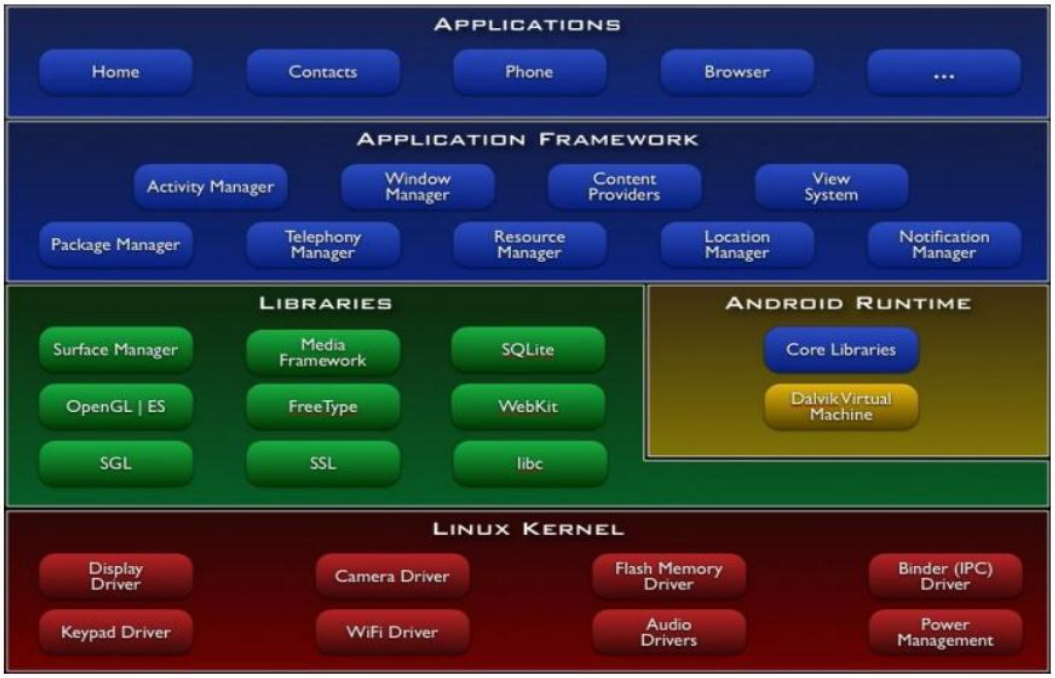
2.2.13.2 *Fitur*

Fitur yang tersedia di Android adalah:

1. Kerangka aplikasi yang memungkinkan penggunaan dan penghapusan komponen yang tersedia.
2. Dalvik mesin virtual merupakan mesin virtual dioptimalkan untuk perangkat *mobile*.
3. Grafik di 2D dan grafis 3D berdasarkan pustaka OpenGL.
4. SQLite untuk penyimpanan data.

5. Mendukung media seperti audio, video, dan berbagai format gambar (MPEG4, H.264, MP3, AAC, AMR, JPG, PNG, GIF).
6. Mendukung GSM, Bluetooth, EDGE, 3G, dan WiFi (tergantung *hardware).*
7. Kamera, *Global Positioning System* (GPS), kompas, dan *accelerometer* (tergantung hardware).
8. Lingkungan *Development* yang lengkap seperti perangkat *emulator, tools,*dan *plugin* untuk Eclpise IDE.
9. *Multi-touch* kemampuan Android sebagai perangkat lunak moderen yang mendukung intraksi sentuh dengan layar dengan menggunakan dua jari.
10. *Google Play* merupakan sarana tempat penjualan aplikasi resmi untuk Android dari Google. *Google Play* berisikan katalog aplikasi yang dapat di-*download* untuk di-*install* pada handphone melalui internet.

2.2.13.3 *Arsitektur Android*



Gambar 2.7 Arsitektuk Android

1. Aplikasi

Android memiliki satu set aplikasi inti yang terdiri dari email *client*, program SMS, kalender, peta, *browser*, kontak telepon, dan sebagainya. Semua aplikasi inti dibangun dengan menggunakan bahasa pemrograman Java.

2. Framework Aplikasi

Pengembang memiliki akses penuh menuju API framework yang sama dengan yang digunakan oleh aplikasi inti. Arsitektur aplikasi dirancang agar komponen dapat digunakan kembali (*reuse*) dengan mudah. Setiap aplikasi dapat memanfaatkan kemampuan ini (sesuai dengan batasan keamanan yang didefinisikan oleh *framework*). Mekanisme yang sama memungkinkan komponen untuk diganti oleh pengguna.

Semua aplikasi merupakan rangkaian set layanan dan sistem, termasuk:

a. *Views* yang kaya dan extensible yang dapat digunakan untuk membangun aplikasi, termasuk *list*, *grids*, kotak teks, tombol, dan bahkan sebuah embedded web.

b. *Content Provider* yang memungkinkan aplikasi untuk mengakses data (seperti dari daftar kontak telepon) atau membagi data tersebut.

c. *Resource Manager*, yang menyediakan akses ke kode nonsumber lokal seperti string, gambar, dan tata letak file.

d. *Notifications* Manager yang memungkinkan semua aplikasi menampilkan alert yang bisa dikostumisasi di dalam status bar.

e. *Activity Manager* yang mengelola siklus hidup aplikasi dan menyediakan navigasi umum *backstack*.

3. Libraries

Android menyertakan libraries C/C++ yang digunakan oleh berbagai komponen dari sistem Android. Kemampuan ini disediakan kepada *Developer* aplikasi melalui *Framework* Aplikasi Android. Beberapa inti libraries tercantum dibawah ini:

1. *System* C *Library* – Variasi dari implementasi BSD-berasal pelaksana sistem standar C library (*libc*), sesuai untuk perangkat *embedded* berbasis Linux.
2. Media *Libraries* – PacketVideo berdasarkan OpenCORE; *library* medukung pemutaran rekaman dan populer banyak format audio dan video, serta file gambar, termasuk MPEG4, H.264, MP3, AAC, AMR, JPG, dan PNG.
3. *Surface* Manager – mengelola akses ke subsistem layar, lapisan komposit 2D dan grafis 3D dari beberapa aplikasi.
4. LibWebCore – mesin web modern yang *powerfull* yang baik browser Android *embedded* web.
5. 3D *Libraries* – sebuah pelaksana berdasarkan openGL ES 1,0 API; perpustakaan baik menggunakan perangkat keras akselerasi 3D (apabila tersedia) atau yang disertakan, sangat optimal 3D *software rasterizer*.
6. FreeType – bitmap dan vektor *font rendering*.
7. SQLite – mesin database yang kuat dan ringan, dan penghubung tersedia untuk semua aplikasi.

4. Android Runtime

Android terdiri dari satu set core *libraries* yang menyediakan sebagian besar fungsi yang sama dengan yang terdapat pada *core libraries* bahasa pemrograman Java. Setiap aplikasi menjalankan prosesnya sendiri dalam Android, dengan masing-masing instan dari mesin virtual Dalvik (Dalvik VM). Dalvik dirancang agar perangkat dapat menjalankan *multiple* VMs secara efisien. Mesin Virtual Dalvik mengeksekusi file dalam Dalvik executable (.dex), sebuah format yang dioptimalkan untuk memori yang kecil. Dalvik VM berbasis, berjalan, dan dikompilasi oleh compiler bahasa Java yang telah ditransformasikan ke dalam .dex format oleh tool “dx” yang telah

disertakan. Dalvik VM bergantung pada kernel Linux untuk berfungsi seperti *threading* dan manajemen memori tingkat rendahnya.

5. Kernel Linux

   Pada layanan sistem inti Android bergantung pada Linux versi 2.6 seperti layanan keamanan, manjemen memori, manajemen proses, *stack* jaringan, dan model *driver*. Kernel bertindak sebagai lapisan abstrak antara perangkat keras dengan seluruh *stack software*.

2.2.14 Tools Implementasi

Tools implementasi merupakan peralatan-peralatan yang dibutuhkan dalam membangun perangkat lunak. Adapun *tools* implementasi yang digunakan untuk membuat aplikasi *mobile* Keuskupan adalah JDK sebagai perangkat pembangun Java, Eclipse IDE sebagai lingkungan pengembangan perangkat lunak, dan SQL sebagai sistem manajemen basis data.

2.2.14.1 *JDK*

JDK (Java Development Kit) adlah sebuah perangkat peralatan yang digunakan untuk membangun perangkat lunak dengan menggunakan bahasa pemrograman Java. JDK berjalan diatas sebuah *virtual machine* yang dinamakan JVM (*Java Virtual Machine*). Dokumentasi JDK berisi spesifikasi API, deskripsi fitur, panduan pengembang, referensi halaman untuk perkakas JDK dan utilitas, demo, dan link ke informasi terkait. Dokumentasi ini juga tersedia dalam bundel yand dapat diunduh dan diinstal pada komputer.

2.2.14.2 *Eclipse IDE*

Eclipse adalah sebuat komunitas proyek open source yang difokuskan pada membangun sebuah platform pengembangan yang dapat diperluas, runtime dan kerangka aplikasi untuk membangun, menyebarkan dan mengelola perangkat lunak di seluruh daur hidup perangkat lunak.

Komunitas Eclipse memiliki lebih dari 60 proyek *opensource*. Proyek-proyek ini secara konsep terbagi menjadi 7 kategori :

1. *Enterprise Development*
2. *Embedded and Device Development*
3. *Rich Client Platform*
4. *Rich Internet Applications*
5. *Application Frameworks*
6. *Application Lifecycle Management (ALM)*
7. *Service Oriented Architecture (SOA)*

Komunitas Eclipse juga didukung oleh ekosistem besar dan dinamis dari penyedia solusi teknologi informasi utama, *start-up* inovatif, universitas dan lembaga penelitian, dan individu yang memperluas, mendukung, dan melengkapi *platform* Eclipse.

2.2.14.3 *Notepad plus-plus*

Notepad plus-plus adalah editor kode program yang bisa didapatkan secara gratis dan tersedia untuk Windows, Mac, Linux, dan Solaris. Platform aplikasi yang memungkinkan pengembang untuk secara cepat membuat web, perusahaan, desktop, dan aplikasi *mobile* menggunakan platform Java, serta PHP, JavaScript, Ajax, Groovy dan Grails, dan C / C + +. Notepad plus – plus menyediakan kinerja meningkat secara signifikan dan pengalaman coding yang berbeda, dengan kemampuan baru analisis kode statis di editor dan lebih cerdas dalam pemindaian.

2.2.15 UML

Unified *Modeling Language* (UML) adalah bahasa spesifikasi standar untuk mendokumentasikan, menspesifikasikan, dan membangun sistem perangkat lunak. UML adalah himpunan struktur dan teknik untuk pemodelan desain program berorientasi objek (OOP) serta aplikasinya. UML adalah metodologi untuk mengembangkan sistem OOP dan sekelompok perangkat tool untuk mendukung pengembangan sistem tersebut. UML mulai diperkenalkan oleh *Object Management Group*,

sebuah organisasi yang telah mengembangkan model, teknologi, dan standar OOP sejak tahun 1980-an. Sekarang UML sudah mulai banyak digunakan oleh para praktisi OOP. UML merupakan dasar bagi perangkat (tool) desain berorientasi objek dari IBM.UML adalah suatu bahasa yang digunakan untuk menentukan, memvisualisasikan, membangun, dan mendokumentasikan suatu sistem informasi. UML dikembangkan sebagai suatu alat untuk analisis dan desain berorientasi objek oleh Grady Booch, Jim Rumbaugh, dan Ivar Jacobson. Namun demikian UML dapat digunakan untuk memahami dan mendokumentasikan setiap sistem informasi. Penggunaan UML dalam industri terus meningkat. Ini merupakan standar terbuka yang menjadikannya sebagai bahasa pemodelan yang umum dalam industri peranti lunak dan pengembangan sistem. (7)

2.2.15.1 *Sejarah UML*

Sampai era tahun 1990 puluhan metodologi pemodelan berorientasi objek telah bermunculan di dunia. Diantaranya adalah: metodologi booch, metodologi coad, metodologi OOSE, metodologi OMT, metodologi *shlaer-mellor*, metodologi *wirfs-brock*, dan sebagainya. Masa itu terkenal dengan masa perang metodologi (*method war*) dalam pendesainan berorientasi objek. Masing-masing metodologi membawa notasi sendiri-sendiri, yang mengakibatkan timbul masalah baru apabila kita bekerjasama dengan kelompok/perusahaan lain yang menggunakan metodologi yang berlainan.

Dimulai pada bulan Oktober 1994 Booch, Rumbaugh dan Jacobson, yang merupakan tiga tokoh yang boleh dikata metodologinya banyak digunakan mempelopori usaha untuk penyatuan metodologi pendesainan berorientasi objek. Pada tahun 1995 direlease draft pertama dari UML (versi 0.8). Sejak tahun 1996 pengembangan tersebut dikoordinasikan oleh *Object Management Group*.

2.2.15.2 *Diagram UML*

UML menyediakan 10 macam diagram untuk memodelkan aplikasi berorientasi objek, yaitu:

1. *Use Case* Diagram untuk memodelkan proses bisnis.
2. *Conceptual* Diagram untuk memodelkan konsep-konsep yang ada di dalam aplikasi.
3. *Sequence* Diagram untuk memodelkan pengiriman pesan (*message*) antar objects
4. *Collaboration* Diagram untuk memodelkan interaksi antar objek.
5. *State* Diagram untuk memodelkan perilaku objects di dalam sistem.
6. *Activity* Diagram untuk memodelkan perilaku *Use Cases* dan *objects* di dalam *system*.
7. *Class* Diagram untuk memodelkan struktur kelas.
8. *Object* Diagram untuk memodelkan struktur *object*.
9. *Component* Diagram untuk memodelkan komponen *object*.
10. *Deployment* Diagram untuk memodelkan distribusi aplikasi.

Berikut akan dijelaskan empat macam diagram yang paling sering digunakan dalam pembangunan aplikasi berorientasi *object*, yaitu *use case* diagram, *sequence* diagram, *activity* diagram, dan *class* diagram.

1. *Use Case* Diagram

Use case diagram digunakan untuk memodelkan bisnis proses berdasarkan perspektif pengguna sistem. *Use case* diagram terdiri atas diagram untuk *use case* dan *actor*. *Actor* merepresentasikan orang yang akan mengoperasikan atau orang yang berinteraksi dengan sistem aplikasi. *Use case* merepresentasikan operasi-operasi yang dilakukan oleh aktor. *Use case* digambarkan berbentuk elips dengan nama operasi dituliskan di dalamnnya. *Actor* yang melakukan operasi dihubungkan dengan garis lurus ke *use case*.

2. *Sequence* Diagram

Sequence diagram menjelaskan secara detil urutan proses yang dilakukan dalam sistem untuk mencapai tujuan dari use case: interaksi

yang terjadi antar class, operasi apa saja yang terlibat, urutan antar operasi, dan informasi yang diperlukan oleh masing-masing operasi.

3. *Activity* Diagram

Activity Diagram menggambarkan workflow (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis. Yang perlu diperhatikan disini adalah bahwa diagram aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan aktor, jadi aktivitas yang dapat dilakukan oleh sistem.

Diagram aktivitas juga banyak digunakan untuk mendefinisikan hal-hal berikut ini :

1. Rancangan proses bisnis dimana setiap urutan aktivitas yang digambarkan merupakan proses bisnis sistem yang didefinisikan.
2. Urutan atau pengelompokan 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. *Class* Diagram

Class diagram merupakan diagram yang selalu ada di permodelan sistem berorientasi objek. Class diagram menunjukkan hubungan antar class dalam sistem yang sedang dibangun dan bagaimana mereka saling berkolaborasi untuk mencapai suatu tujuan.

2.2.16 MySQL

MySQL adalah sebuah perangkat lunak sistem manajemen basis data SQL (bahasa Inggris: *database management* sistem) atau DBMS yang *multithread*, *multi-user*, dengan sekitar 6 juta instalasi di seluruh dunia.. MySQL adalah *server database* SQL (*Structured Query Language)* yang paling banyak diminati karena kecepatan kerja dan kemudahan dalam penggunaannya. Kelebihan MySQL sebagai basis data, antara lain : 1.

Mendukung standar yang telah ada, yaitu standar ODBC level 0-2 2. Mampu membuat tabel dengan ukuran besar

Pada MySQL sebuah database mengandung satu atau sejumlah tabel. Tabel terdiri dari sejumlah baris dan setiap baris mengandung satu atau beberapa kolom. Untuk mengelolah data MySQL ada beberapa cara yaitu melalui *prompt* DOS *(tool command line)* dan dapat juga menggunakan program *utility* seperti:

1. PHP MyAdmin
2. MySQLGUI
3. MySQL Manager Java Based
4. MySQL Administrator for windows

2.2.17 XAMPP

Menurut Bunafit Nugroho (2008 : 2) XAMPP adalah suatu bundel *web server* yang populer digunakan untuk coba-coba di *Windows* karena kemudahan instalasinya. Bundel program *open source*tersebut berisi antara lain server web, Apache, Interpreter PHP, dan basis data MySQL. Setelah menginstall XAMPP, kita bisa memulai pemrograman PHP di komputer sendiri maupun mencoba menginstall aplikasi-aplikasi web.

2.2.18 Mozilla FireFox

Mozilla Firefox (aslinya bernama Phoenix dan kemudian untuk sesaat dikenal sebagai Mozilla Firebird) adalah penjelajah *web* antar-*flatform* gratis yang dikembangkan oleh yayasan mozilla. Yayasan mozilla bertujuan untuk mengembangkan sebuah *browser web* yang kecil, cepat, dan simpel melalui firefox. Salah satu fitur populer firefox adalah pemblokir *pop-up* yang sudah terpasang di dalamnya dan sebuah mekanisme pengembangan untuk menambah fungsionalitas tambahan.

2.2.19 Penelitian Kuantitatif

Penelitian kuantitatif merupakan jenis penelitian ilmiah yang terkait dengan pengumpulan dan analisis data dalam bentuk numerik[3]. Kumpulan data dalam skala besar yang menggambarkan fakta/fenomena yang ada, menjadi objek penekanan pada penelitian ini untuk mengembangkan teori maupun menguji hipotesis yang telah dibuat. Salah satu pendekatan yang digunakan dalam melakukan penelitian kuantitatif adalah survei. Survei merupakan metode dalam melakukan pengumpulan data dengan menggunakan beberapa pertanyaan yang telah dibuat secara terstruktur dalam kuesioner. Pertanyaan-pertanyaan tersebut ditanyakan kepada sampel yang merepresentasikan populasi yang telah ditentukan. Setelah data terkumpul, analisis data selanjutnya dilakukan untuk mencapai tujuan penelitian.

2.2.20 Skala Likert

Skala likert merupakan skala yang menyatakan tingkat persetujuan individu terhadap suatu pernyataan. Skala ini sering digunakan dalam berbagai penelitian yang menggunakan pendekatan survei, dimana kuesioner dijadikan sebagai alat untuk memperoleh data. Ada beberapa tingkat Skala Likert yang sering digunakan dalam penelitian, diantaranya adalah 5-level, 7-level dan 9-level[18].

Berikut adalah tingkat persetujuan apabila digunakan skala Likert 5-Level :

Tabel 2.1 Skala Likert

No.	Keterangan	Skor Positif	Skor Negatif
1	Sangat Setuju	5	1
2	Setuju	4	2
3	Ragu-ragu	3	3
4	Tidak Setuju	2	4
5	Sangat Tidak Setuju	1	5

Apabila data yang diperoleh melalui jawaban kuesioner telah terkumpul, maka angka dalam bentuk skala Liker yang dipilih responden (pada setiap pertanyaan kuesioner) selanjutnya dikalkulasi. Perhitungan ini menggunakan teknik statistik sehingga dapat dianalisis lebih lanjut.

2.2.21 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.[9] 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.

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 ke semua jenis aplikasi tersebut.

Keuntungan menggunakan metodologi berorientasi objek adalah sebagai berikut:

1. Meningkatkan produktivitas

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

2. Kecepatan pengembangan

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

3. Kemudahan pemeliharaan

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

4. Adanya konsistensi

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

5. 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.

2.2.22 Konsep Dasar Berorientasi Objek

Pendekatan berorientasi objek merupakan suatu teknik atau cara pendekatan dalam melihat permasalahan dan sistem (sistem perangkat lunak, sistem informasi, atau sistem lainnya.[9] Pendekatan berorientasi objek akan memandang sistem 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 antar kelas sampai abstraksi sistem. Saat mengabstraksikan dan memodelkan objek, data dan proses-proses yang dipunyai oleh objek akan dienkapsulasi (dibungkus) menjadi suatu 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 sistem yang dibangun dengan berdasarkan metode berorientasi objek adalah sebuah sistem yang komponennya dibungkus (dienkapsulasi) menjadi kelompok data dan fungsi. Setiap komponen dalam sistem 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:

a. Kelas (*Class*)

Kelas adalah sekumpulan objek-objek dengan karakteristik yang sama. Kelas merupakan definisi statis dan himpunan objek yang sama yang mungkin lahir atau diciptakan dan kelas tersebut. Sebuah kelas akan mempunyai sifat (atribut), kelakuan (metode/operasi), 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 dalam pembuatan perangkat lunak. Kelas merupakan bentuk struktur pada kode program yang menggunakan metodologi berorientasi objek.

b. Objek (*object*)

Objek adalah abstraksi dari sesuatu yang mewakili dunia nyata benda, manusia, satu 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 dibuat sebuah objek. Objek dilihat darisegi teknis adalah elemen pada saat *runtime* yang akan diciptakan, dimanipulasi, dan dihancurkan saat eksekusi sehinga 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.

c. Metode (*method*)

Operasi atau metode pada sebuah kelas hampir 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*, aktifitas atau aksi keadaan, fungsi, atau kelakuan dunia nyata. Contoh metode atau operasi misalnya *Read, Write, Move, Copy,* dan sebagainya.

d. 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.

e. 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.

f. Enkapsulasi (*encapsulation*)

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

g. Pewarisan (*inheritance*)

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

h. Antarmuka (*interface*)

Antarmuka 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.

i. *Reusability*

Pemanfaatan kembali objek yang sudah didefinisikan untuk suatu permasalahan pada permasalahan lainnya yang melibatkan objek tersebut.

j. Generalisasi dan Spesialisasi

Menunjukkan hubungan antara kelas dan objek yang 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.

k. Komunikasi Antarobjek

Komunikasi antarobjek dilakukan lewat pesan (*message*) yang dikirim dari satu objek ke objek lainnya.

l. Polimorfisme (*polymorphism*)

Kemampuan suatu objek digunakan di banyak tujuan yang berbeda dengan nama yang sehingga menghemat baris program.

m. *Package*

Package adalah sebuah kontainer atau kemasan yang dapat digunakan untuk mengelompokkan kelas-kelas sehingga memungkinkan beberapa kelas yang bernama sama disimpan dalam *package* yang berbeda.