BAB II
LANDASAN TEORI

2.1 Tinjauan Perusahaan

PD. Bank Perkreditan Rakyat (BPR) Kota Bandung adalah badan usaha milik daerah yang diharapkan dapat meningkatkan pembangunan ekonomi Kota Bandung, salah satunya dapat memperluas kesempatan kerja dan berusaha melalui penyaluran kredit dengan bunga kompetitif dan agunan yang tidak memberatkan. Usaha dalam pemberian kredit saat ini terus berkembang pesat terumata di PD. Bank Perkreditan Rakyat Kota Bandung sehingga memicu persaingan antar bank untuk mengembangkan usaha kreditnya dengan baik.

2.1.1 Sejarah Singkat

Perusahaan Daerah Bank Perkreditan Rakyat Kota Bandung adalah badan usaha milik Pemerintah Daerah Kota Bandung. PD BPR Kota Bandung didirikan dengan maksud dan tujuan membantu dan mendorong pertumbuhan perekonomian dan pembangunan daerah di segala bidang serta sebagai salah satu sumber Pendapatan Asli Daerah dalam rangka meningkatkan taraf hidup rakyat. PD. BPR Kota Bandung juga merupakan salah satu alat kelengkapan otonomi daerah di bidang keuangan/perbankan dan menjalankan usahanya sebagai Bank Perkreditan Rakyat sesuai dengan ketentuan perundang-undangan yang berlaku.

PD. Bank Perkreditan Rakyat Kota Bandung didirikan pada tanggal 1 Agustus 1968 dengan Instruksi Walikotamadya Bandung, Nomor: 476/67/DPP dengan nama Bank Pasar Kotamadya Daerah Tingkat II Bandung yang statusnya merupakan Seksi dari Dinas Pendapatan Daerah.Tanggal 22 Oktober 1996 PD. Bank Pasar Kotamadya Daerah Tingkat II Bandung berubah nama menjadi PD. BANK PERKREDITAN RAKYAT KOTAMADYA DAERAH TINGKAT II BANDUNG, sesuai Surat Keputusan Walikotamadya Nomor: 539/SK 642-Bdg/1996.

Saat ini PD. BPR Kota Bandung berdasarkan pada perda No.21 Tahun 2002 yang telah diubah dan ditambah dengan Perda No.05 Tahun 2005. Pendirian

perusahaan daerah bank perkreditan rakyat Kota Bandung telah ditetapkan dengan Peraturan Daerah Kota Bandung no 21 tahun 2002 tentang Perusahaan Daerah Bank Perkreditan Rakyat Kota Bandung sebagaimana telah diubah beberapa kali dengan Peraturan Daerah Kota Bandung Nomor 05 Tahun 2005. Sedangkan mengenai organisasi dan tata kerja dari PD BPR Kota Bandung juga telah diatur melalui Peraturan Walikota Bandung No. 226 Tahun 2007 tentang Organisasi dan Tata Kerja PD. BPR Kota Bandung. Selanjutnya dalam rangka meningkatkan pelayanan kepada masyarakat dan sebagai salah satu sumber pendapatan asli daerah serta untuk lebih mendorong kinerja Perusahaan Daerah Bank Perkreditan Rakyat maka di keluarkan peraturan daerah kembali mengenai PD BPR Kota Bandung No. 04 Tahun 2009 tentang PD. BPR Kota Bandung.

2.1.2 Visi dan Misi PD.BPR Kota Bandung

Sebagai perusahaan daerah di bidang perbankan PD.BPR Kota Bandung memiliki visi dan misi untuk meningkatkan kinerja dan dapat mencapai tujuan yang diinginkan.Adapun visi dan misi pada PD.BPR Kota Bandung, yaitu :

2.1.2.1 Visi PD.BPR Kota Bandung

Menjadi Bank Perkreditan yang unggul, handal, profesional dan terpercaya.

2.1.2.2 Misi PD.BPR Kota Bandung

Tumbuh bersama untuk meningkatkan kesejahteraan dan perekonomian masyarakat serta memberikan kontribusi kepada stakeholder melalui jasa perbankan yang di kelola secara profesional.

2.1.3 Logo PD.BPR Kota Bandung



Gambar 2.1 Logo PD. BPR Kota Bandung

Keterangan :

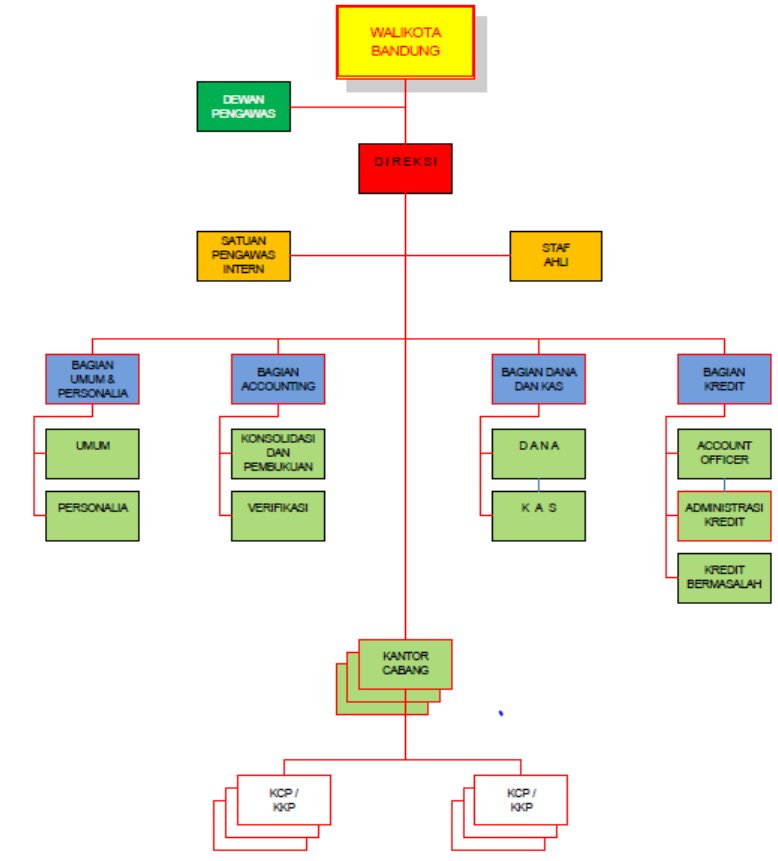
1. Ditetapkan dengan Peraturan Daerah Kota besar Bandung tahun 1953, tertanggal 8 Juni 1953, yang diijinkan dengan Keputusan Presiden tertanggal 28 april 1953 No. 104 dan diundangkan dalam Berita Propinsi Jawa Barat tertanggal 28 Agustus 1954 No. 4 lampiran No. 6 Lambang tersebut bertokoh PERISAI yang berbentuk JANTUNG. Perisai tersebut terbagi dalam dua bagian oleh sebuah BALOK- LINTANG mendatar bertajuk empat buah, yang berwarna HITAM dengan pelisir berwarna PUTIH (PERAK) pada pinggir sebelah atasnya:
   a. Bagian atas latar KUNING (EMAS) dengan lukisan sebuah GUNUNG berwaarna HIJAU yang bertumpu pada blok-lintang dan bagian bawah latar PUTIH (PERAK) dengan lukisan empat bidaang jalur mendatar berombak yang berwarna BIRU.
   b. Di bawah perisai itu terlukis sehelai PITA berwarna KUNING (EMAS) yang melambai pada kedua ujungnya, Pada pita itu tertulis dengan huruf-huruf besar latin berwarna HITAM amsal dalam bahasa KAWI, yang berbunyi GEMAH RIPAH WIBAWA MUKTI.
   c. Sebagai tokoh lambang itu diambil bentuk perisai atau tameng, yang dikenal kebudayaan dan peradaban sebagai senjata dalam perjuangan untuk mencapai sesuatu tujuan dengan melindungi diri. Perkakas perjuangan yang demikian itu dijadikan lambang yang mempunyai arti menahan segala mara bahaya dan kesukaran.
2. Warna

a. KUNING (EMAS), melambangkan kesejahteraan, keluhungan.
b. HITAM (SABEL), melambangkan kokoh, tegak, kuat.
c. HIJAU (SINOPEL), melambangkan kemakmuran sejuk.
d. PUTIH (PERAK), melambangkan kesucian.
e. BIRU (AZUUR), melambangkan kesetiaan.

2.1.4 Stuktur Organisasi

Struktur organisasi PD. BPR Kota Bandung berkembang secara dinamis karena didorong faktor internal dan eksternal. Struktur organisasi di PD.BPR Kota Bandung masih bersifat sentralisasi. Jadi semua keputusan, kebijakan, wewenang menjadi tanggung jawab Kepala Kantor. Struktur organisasi di PD.BPR Kota Bandung ditetapkan dalam suatu keputusan tersendiri dengan mengikuti pola struktur fungsional dan staff.

Bila kita lihat struktur organisasi yang dimiliki oleh PD. BPR Kota Bandung merupakan gabungan dari jenis organisasi dalam bentuk lini dan staf. Dimana wewenang dari pucuk pimpinan mengalir langsung kepada kepala bagian yang memimpin satuan-satuan organisasi menurut jenjang organisasi. Adapun struktur organisasi PD. BPR Kota Bandung dapat dilihat pada gambar 2.2



Sumber : PD. BPR Kota Bandung

Gambar 2.2 Struktur Organisasi

2.1.5 Deskripsi Tugas

Deskripsi pekerjaan *(Job Description)* dari masing-masing departemen yang ada di PD. BPR Kota Bandung adalah sebagai berikut :

Tabel 2.1 Deskripsi Tugas

No	Jabatan	Deskripsi Tugas
1	Dewan Pengawas	a) Mengkoordinir penyusunan tata cara pengawasan dan pengelolaan kebijakan umum untuk pelaksanaan manajemen PD.BPR Kota Bandung. b) Mengkoordinir pelaksanaan pengendalian dalam bentuk petunjuk dan pengarahan kepada Direksi dalam pelaksanaan tugas. c) Mengkoordinir pelaksanaan dan pengawasan atas pengurusan PD. BPR Kota Bandung dengan melakukan audit pengawas baik secara periodik sesuai dengan jadwal yang telah ditentukan; atau sewaktu-waktu apabila dipandang perlu. d) Melakukan pembinaan dan pengembangan PD. BPR Kota Bandung seperti menghadiri rapat manajemen pembahasan kinerja dan kesehatan bank atau rapat direksi yang terkait untuk memberikan masukan dan saran, baik diminta maupun tidak. e) Mengikuti rapat pengawas dengan direksi yang diadakan paling sedikit 2 kali dalam 1 tahun atas undangan Dewan Pengawas maupun permintaan direksi. f) Melakukan pembinaan dan pengembangan PD. BPR Kota Bandung seperti menghadiri rapat manajemen pembahasan kinerja dan kesehatan

No	Jabatan	Deskripsi Tugas
		bank atau rapat direksi yang terkait untuk memberikan masukan dan saran, baik diminta maupun tidak.
2	Direksi (DirekturUtama)	a) Memimpin dan mengkoordinir Direktur Operasional, Staf Ahli, SPI dan Manajer terkait secara profesional, guna mewujudkan visi dan misi PD. BPR Kota Bandung. b) Direktur Utama mempunyai tugas menyelenggarakan perencanaan dan koordinasi dalam pelaksanaan tugas Direksi serta melakukan pembinaan dan pengendalian atas Unit Kerja PD. BPR Kota Bandung. c) Menjaga agar pelaksanaan operasional lembaga sesuai dengan ketentuan dan peraturan, baik aturan eksternal maupun internal. d) Menyusun dan menjalankan rencana strategi dan anggaran tahunan yang sudah disetujui oleh dewan pengawas. e) Penetapan kebijaksanaan untuk melaksanakan pengurusan dan pengelolaan PD. BPR Kota Bandung berdasarkan kebijaksanaan umum yang ditetapkan oleh Dewan Pengawas. f) Penyusunan dan penyampaian Rencana Kerja Tahunan dan Anggaran PD. BPR kepada Walikota melalui Dewan Pengawas yang meliputi kebijaksanaan di bidang organisasi, perencanaan. perkreditan, keuangan, kepegawaian umum dan pengawasan untuk

No	Jabatan	Deskripsi Tugas
		mendapatkan pengesahan. g) Penyusunan dan penyampaian laporan perhitungan hasil usaha dan kegiatan PD. BPR Kota Bandung setiap 3 (tiga) bulan sekali kepada Walikota melalui Dewan Pengawas. h) Penyusunan dan penyampaian laporan tahunan yang terdiri atas Neraca dan Laporan Laba Rugi kepada Walikota melalui Dewan Pengawas untuk mendapatkan pengesahan.
3	Direksi (Direktur)	a) Memimpin dan mengkoordinir para Staf ahli, SPI, dan Manajer terkait secara profesional, guna mewujudkan visi dan misi PD. BPR Kota Bandung. b) Menjaga agar pelaksanaan operasional lembaga sesuai dengan ketentuan dan peraturan, baik aturan eksternal maupun internal. c) Bersama Direktur Utama, menyusun dan menjalankan rencana strategi dan anggaran tahunan yang sudah disetujui oleh Dewan Pengawas. d) Bersama Dirut, menetapkan kebijaksanaan untuk melaksanakan pengurusan dan pengelolaan PD. BPR Kota Bandung berdasarkan kebijaksanaan umum yang ditetapkan oleh Dewan Pengawas. e) Bersama Dirut menyusun dan menyampaikan Rencana Kerja Tahunan dan Anggaran PD. BPR Kota Bandung kepada Walikota melalui

No	Jabatan	Deskripsi Tugas
		Dewan Pengawas yang meliputi kebijaksanaan di bidang organisasi, perencanaan. perkreditan, keuangan, kepegawaian umum dan pengawasan untuk mendapatkan pengesahan. f) Penyusunan dan penyampaian laporan perhitungan hasil usaha dan kegiatan PD. BPR Kota Bandung setiap 3 (tiga) bulan sekali kepada Walikota melalui Dewan Pengawas. g) Penyusunan dan penyampaian laporan tahunan yang terdiri atas Neraca dan Laporan Laba Rugi kepada Walikota melalui Dewan Pengawas untuk mendapatkan pengesahan. h) Mengkoordinir dan melakukan kegiatan penggalangan dana dan pemasaran produk baik dana maupun kredit. i) Membina hubungan baik dengan pihak luar baik pemerintah maupun non pemerintah serta organisasi perbankan dalam upaya pengembangan PD BPR Kota Bandung. j) Memberikan tugas, arahan dan bimbingan terhadap staff dibawahnya khususnya untuk bidang operasional dan cabang-cabang.
4	Ketua Satuan Audit Internal (KSAI)	a) Membuat rencana audit periodik tahunan. b) Membuat usulan rencana audit. khusus sesuai permasalahan ataupun atas permintaan direksi. c) Memastikan dan memonitor pelaksanaan audit sesuai dengan jadwal dan prosedur yang berlaku. Audit periodik yang dilaksanakan oleh SPI berkaitan dengan.

No	Jabatan	Deskripsi Tugas
		a) Penyimpangan biaya dan anggaran b) Penyimpangan kebijakan operasional manajemen c) Komplience pelaksanaan SOP dan IT d) Pelaksanaan peraturan perusahaan dan code of conduc perusahaan d) Membahas temuan-temuan ataupun usulan perbaikan yang ditemukan selama audit bersama Direksi dan membuat laporan/ memo rekomendasinya. e) Membuat laporan hasil audit untuk diserahkan kepada direksi dengan tembusan kepada dewan pengawas. f) Mengkoordinir pelaksanaan tugas-tugas audit khusus yang diberikan oleh Direksi untuk menyelesaikan suatu permasalahan. g) Mengkoordinir dan memfasilitasi pertemuan untuk membahas temuan dan menyiapkan laporan yang berkaitan dengan permasalahan tersebut. h) Melakukan pemantauan tindak lanjut pelaksanaan temuan audit yang sudah menjadi komitment untuk perbaikan sesuai dengan jadwal yang disepakati. i) Melakukan peran lebih sebagai konsultan untuk melakukan perbaikan dari hasil temuan tersebut. j) Melakukan koordinasi dengan Auditor ekstern dan auditor pengawas mengenai hasil audit internal maupun hasil audit yang dilakukan

No	Jabatan	Deskripsi Tugas
		oleh pihak eksternal.
5	Staf Ahli	a) Menyusun konsep pemikiran tentang perbaikan pelaksanaan system perbankan yang berlaku pada PD BPR Kota Bandung. b) Memberikan saran dan pertimbangan baik yang diminta ataupun tidak diminta kepada Direksi mengenai langkah-langkah untuk perbaikan / penyempurnaan operasional PD BPR Kota Bandung. c) Memberikan bimbingan teknis mengenai aplikasi system dan prosedur yang ditetapkan pada PD BPR Kota Bandung. d) Memberikan penjelasan dan petunjuk kepada masing-masing petugas pengelola PD BPR Kota Bandung. e) Membuat evaluasi dan laporan secara periodik atas hasil pekerjaan kepada Direksi.
6	Manajer SDM dan Umum	a) Mengelola, dan mengawasi penerapan fungsi operasional yang sesuai dengan ketentuan dan peraturan, baik dari eksternal (Undang-Undang, Peraturan Pemerintah, Keppres, Surat Edaran, Surat Keputusan departemen terkait dan lain-lain) Maupun dari internal (Kebijakan dan Sistim Prosedur Perusahaan). b) Bersama bagian lain menyiapkan perencanaan pelatihan setiap tahun. c) Turut membantu secara aktif atas tugas-tugas harian setiap unit yang berada dibawah tanggung jawabnya.

No	Jabatan	Deskripsi Tugas
		d) Aktif memberikan saran-saran kepada pihak manajemen mengenai masalah-masalah yang berkaitan dengan aspek pengelolaan Sumber Daya Manusia. e) Meningkatkan, membimbing dan melatih staff dibawahnya dalam pencapaian target, maupun usaha-usaha untuk mencapai target yang telah ditetapkan. f) Melaksanakan tugas khusus yang diberikan oleh Direksi diluar tugas rutin. g) Membuat Job Description dan memastikan setiap karyawan mengetahui Job Description sesuai dengan jabatannya. h) Bertanggung jawab terhadap aset Perusahaan termasuk AYIDA. i) Pengadaan barang Inventaris, ATK dan cetakan sesuai dengan kewenangan. j) Proses dan prosedur pengadaan barang Inventaris, ATK dan cetakan.
7	Manajer Operasional	a) Mengkoordinasikan, mengamankan dan mengarahkan kegiatan yang berhubungan dengan sistim dan prosedur untuk operasional PD. BPR Kota Bandung. b) Koordinasi dengan, Supervisor untuk masing-masing bidang (akunting, keuangan dan anggaran, teller, customer services, kas luar, dan administrasi kredit) untuk memastikan proses operasional PD. BPR Kota Bandung berjalan dengan baik.

No	Jabatan	Deskripsi Tugas
		c) Memberikan bantuan sepenuhnya terhadap pelaksanaan audit internekstern. d) Mengawasi / memeriksa / meneliti kebenaran proses perhitungan bunga deposito dan tabungan pada akhir setiap awal hari dan untuk tabungan pada akhir bulan. e) Mengawasi laporan nominatif tabungan dan deposito tepat waktu. f) Menerima laporan transaksi akhir hari. g) Menerima hasil laporan transaksi tabungan dan Deposito dari Supervisor Keuangan dan Anggaran. h) Mengawasi dan memonitor pelaksanaan proses penyetoran / penarikan antar Bank. i) Aktif memberikan saran-saran kepada Direksi mengenai masalah-masalah yang berkaitan dengan operasional.
8	Manajer Penagihan	a) Melakukan Koordinasi monitoring terhadap angsuran nasabah yang telah diberikan Kredit. b) Melakukan review penjadwalan penagihan angsuran Kredit nasabah sesuai dengan perjanjian. c) Mengkoordinir penjemputan angsuran nasabah Kredit dan menyetorkannya kepada Teller dan memastikan seluruh setoran tidak ada yang tertinggal dan tidak terjadi selisih antara catatan dengan uang yang diserahkan. d) Mengkoordinir penyelesaian kredit bermasalah bersama-sama dengan Account Officer dan

No	Jabatan	Deskripsi Tugas
		Bagian Legal. e) Membuat target penyelesaian Kredit bermasalah (kolektibilitas Kurang Lancar, Diragukan dan Macet). f) Melakukan proses sita jaminan dan penjualan jaminan. g) Membuat laporan secara lisan atau tertulis secara periodik kepada Direksi mengenai tugas yang diberikan dan kondisi nasabah yang ditanganinya.
9	Suipervisor Personalia, Pengembangan SDM dan Tata Usaha	a) Memproses permintaan pengadaan tenaga kerja. b) Memproses dokumen atau kelengkapan administrasi untuk karyawan baru, mengundurkan diri, mutasi, dan diberhentikan. c) Mendata pelaksanaan pelatihan untuk pengembangan karyawan. d) Membuat surat peringatan untuk karyawan yang melanggar disiplin kerja. e) Memproses kelengkapan administrasi kenaikan gaji berdasarkan jabatan struktural dan golongan. f) Memproses kenaikan jabatan karyawan secara struktural dan golongan. g) Memonitor status karyawan kontrak. h) Mendata kehadiran dan keterlambatan karyawan dalam bekerja. i) Melakukan proses adm. SDM seperti : membuat surat ijin, surat keterangan, surat

No	Jabatan	Deskripsi Tugas
		kontrak kerja, dll. j) Mendata karyawan yang melanggar disiplin kerja baik yang mendapat surat peringatan atau peneguran secara lisan. k) Membuat daftar gaji dan input ke Tabungan Karyawan masing-masing. l) Korespondensi Internal dan Eksternal serta mengarsipkannya. m) Menbuat Laporan Pajak Pegawai. n) Melaksanakan kerjasama dengan pihak III yang berkaitan dengan kesejahteraan. o) Menyetorkan kewajiban pegawai kepada pihak ketiga. p) Membuat daftar inventaris di tiap-tiap ruangan.
10	Supervisor Umum dan Logistik	a) Memproses permintaan pembelian barang inventaris dan ATK. b) Membuat laporan Pembelian. c) Memonitor proses keluar masuk barang Inventaris dan ATK. d) Pada akhir bulan membuat laporan inventaris kantor dan membuat laporan persediaan ATK. e) Melakukan koordinasi dan memonitoring pekerjaan Satpam untuk memastikan keamanan lingkungan PD. BPR Kota Bandung . f) Melakukan koordinasi dan memonitoring pekerjaan Office Boy dalam lingkungan PD. BPR Kota Bandung.

No	Jabatan	Deskripsi Tugas
		g) Melakukan koordinasi dan memonitoring pekerjaan supir dalam menjalankan tugasnya merawat kendaraan dan mengantar karyawan. h) Melakukan pengelolalan IT yang terkait dengan peralatan, hardware dan jaringan. Sedangkan untuk aplikasi di kelola oleh EDP. i) Membuat laporan biaya operasional kendaraan.
11	Supervisor Akunting, Keuangan dan Anggaran	a) Menyelenggarakan pencatatan mengenai biaya-biaya perusahaan, profit margin dan pendapatan serta mengarsip bukti-buktinya. b) Pembeliah - Memeriksa semua dokumen yang berhubungan dengan pembayaran, yaitu faktur asli, Purchase Order, Bukti barang masuk dan surat jalan yang bersangkutan. c) Pembayaran Hutang - memeriksa semua dokumen yang berhubungan dengan pembayaran, yaitu faktur asli, Purchase order, surat jalan dan bukti barang masuk, bukti potongan / retur pembelian. d) Pendapatan - Memeriksa semua dokumen yang berhubungan dengan pendapatan operasional dan Non operasional. e) Menyelenggarakan dan memeriksa pencatatan mengenai keuangan sehubungan denga Persediaan barang, Hutang-hutang, Komisi, Perbaikan-perbaikan kesalahan, Keuangan dan modal dasar perusahaan. f) Memeriksa daftar inventaris aktiva tetap,

No	Jabatan	Deskripsi Tugas
		perubahannya serta nilai penyusutannya. g) Membuat dan memonitoring aspek kesehatan perbankan (NPL, LDR, ROA, Aset, Efisiensi, dan Pendapatan). h) Melakukan cross check bukti-bukti dari beberapa bagian yang terkait dengan penerimaan dan pengeluaran persediaan ATK. i) Memeriksa laporan persediaan ATK tiap akhir bulan dari data yang diperoleh untuk penyusunan laporan keuangan. j) Menyusun laporan-laporan secara periodik dan teratur kepada pihak eksternal (BI, LPS) dan pihak internal baik secara bulanan, triwulanan/semester, yang terdiri dari : Laporan Neraca, Laporan Rugi/Laba, Laporan Perubahan Modal, Analisa Laporan.
12	Pegawai Umum	a) Melaksanakan kegiatan di PD. BPR sesuai dengan perintah masingmasing manajer sebagai atasannya. b) Mengawasi kerja loket penerangan. c) Memberikan penerangan kepada umum, karena ada hubungannya dengan humas.

2.2 Landasan Teori

Landasan teori memberikan gambaran dan teori terkait dengan membangun sistem (*Survey Kreditor berbasis Android*). Untik membangun sebuah sistem , maka perlu dikemukakan hal-hal atau teori yang berkaitan dengan permasalahan dan ruang lingkup pembahasan sebagai landasan dalam pembuatan laporan tugas akhir ini.

2.2.1 Aplikasi Mobile

Menurut Buyens [2] aplikasi mobile berasal dari kata *application* dan *mobile*. *Application* yang artinya penerapan, lamaran, dan penggunaan. Secara istilah aplikasi adalah program siap pakai yang direka untuk melaksanakan suatu fungsi bagi pengguna atau aplikasi yang lain dan dapat digunakan oleh sasaran yang dituju. Sedangkan *mobile* dapat diartikan sebagai perpindahan dari suatu tempat ke tempat yang lain.

Maka aplikasi *mobile* dapat diartikan sebuah program aplikasi yang dapat dijalankan atau digunakan walaupun pengguna berpindah-pindah dari satu tempat ke tempat yang lain serta mempunyai ukuran yang kecil. Aplikasi *mobile* ini dapat diakses melalui perangkat nirkabel, pager, PDA, telepon seluler, *smartphone*, dan perangkat sejenisnya.

2.2.2 Karakteristik Perangkat Mobile

Perangkat mobile memiliki banyak jenis dalam hal ukuran, desain dan layout, tetapi mereka memiliki karakteristik yang sangat berbeda dari sistem desktop.

Berikut karakteristik perangkat mobile, diantaranya:

a. Ukuran yang kecil

Perangkat mobile memiliki ukuran yang kecil. Konsumen menginginkan perangkat yang terkecil untuk kenyamanan dan mobilitas mereka.

b. Memory yang terbatas

Perangkat mobile juga memiliki memory yang kecil, yaitu primary (RAM) dan secondary (disk). Pembatasan ini adalah salah satu faktor yang mempengaruhi penulisan program untuk berbagai jenis dari perangkat ini. Dengan pembatasan jumlah dari memory, pertimbangan-pertimbangan khusus harus diambil untuk memelihara pemakaian dari sumber daya yang mahal ini.

c. Daya proses yang terbatas

Sistem mobile tidaklah setangguh rekan mereka yaitu desktop. Ukuran, teknologi dan biaya adalah beberapa faktor yang mempengaruhi status dari sumber daya ini. Seperti harddisk dan RAM, anda dapat menemukan mereka dalam ukuran yang pas dengan sebuah kemasan kecil.

d. Mengkonsumsi daya yang rendah

Perangkat mobile menghabiskan sedikit daya dibandingkan dengan mesin desktop. Perangkat ini harus menghemat daya karena mereka berjalan pada keadaan dimana daya yang disediakan dibatasi oleh baterai-baterai.

e. Kuat dan dapat diandalkan

Karena perangkat mobile selalu dibawa kemana saja, mereka harus cukup kuat untuk menghadapi benturan-benturan, gerakan, dan sesekali tetesan-tetesan air.

f. Konektivitas yang terbatas

Perangkat mobile memiliki bandwith rendah, beberapa dari mereka bahkan tidak tersambung. Kebanyakan dari mereka menggunakan koneksi wireless.

g. Masa hidup yang pendek

Perangkat-perangkat konsumen ini menyala dalam hitungan detik, kebanyakan dari mereka selalu menyala. Coba ambil kasus sebuah handphone, mereka booting dalam hitungan detik dan kebanyakan orang tidak mematikan handphone mereka bahkan ketika malam hari. PDA akan menyala jika anda menekan tombol power mereka.

Aplikasi mobile terdapat beberapa tipe, berikut beberapa tipe yang terdapat dalam aplikasi mobile:

A. Berdasarkan teknologi yang digunakan:

1. Server-Request Based

   Mobile Equipment (ME) merequest data dari server, WAP. Menggunakan WML, sebuah markup language khusus berbasil XML.

   Keuntungan :

   a. Platform Independent
   b. Mudah diterapkan

c. Jenis aplikasi beragam

Kerugian :

a. Masih banyak ME yang tidak support WAP
b. Pembayaran untuk data
c. Interface terbatas

Implementasi :

a. Menggunakan Web Server dengan setting tambahan di konfigurasi
b. File WML dibuat selayaknya membuat file HTML

2. SMS based

SMS : Short Message Service. Pengiriman pesan singkat antar ME / dengan sebuah server terbatas ± 160 characters.

Keuntungan :

a. Platform Independent
b. ME sebagian besar sudah dapat menerapkan SMS

Kerugian :

a. Pembayaran untuk setiap aksi
b. Interface hanya text, user harus ingat perintah
c. Panjang text sangat terbatas

Implementasi :

a. Menggunakan interface antara ME dengan server
b. Aplikasi menunggu data dari interface dan mengirim data melalui interface tersebut.

3. SIM-card based

Aplikasi diletakkan di atas SIM-card, aplikasi dapat diakses melalui menu baru yang akan muncul di ponsel. Javacard, khusus untuk javacard aplikasi bersifat downloadable, pengiriman data lewat SMS.

Keuntungan :

a. Platform Independent
b. Data disimpan secara lokal dan dapat digunakan bila berganti ME

Kerugian :

a. Size terbatas
b. Aplikasi harus sangat kecil
c. Sulit diterapkan

Implementasi :

a. Download Javacard
b. Gunakan Simulator
c. Mobile Equipment Based

Aplikasi diletakkan di Mobile Equipment, contoh Pemrograman di atas Symbian OS, pengiriman data SMS.

Keuntungan :

a. Interface sangat mudah
b. Jenis aplikasi sangat beragam, bahkan dapat digunakan untuk aplikasi berat.

Kerugian :

a. Platform Dependent
b. Bila mengubah ME kemungkinan aplikasi dan data tidak dapat digunakan

B. Berdasarkan fungsi :

1. Information

   Pengiriman informasi ke ME user
2. Communications

   Komunikasi antara user dengan satu atau lebih user lain
3. Entertainment

   Aplikasi menghibur seperti Games
4. Productivity Tools

   Aplikasi yang dapat meningkatkan produktivitas walaupun sedang berpindah
5. Commerce and Banking

   Jual beli melalui ME

6. Location Based

   Aplikasi yang mengubah data sesuai dengan lokasi user. Lokasi dapat ditentukan oleh GPS dan sel yang ditempati oleh user.

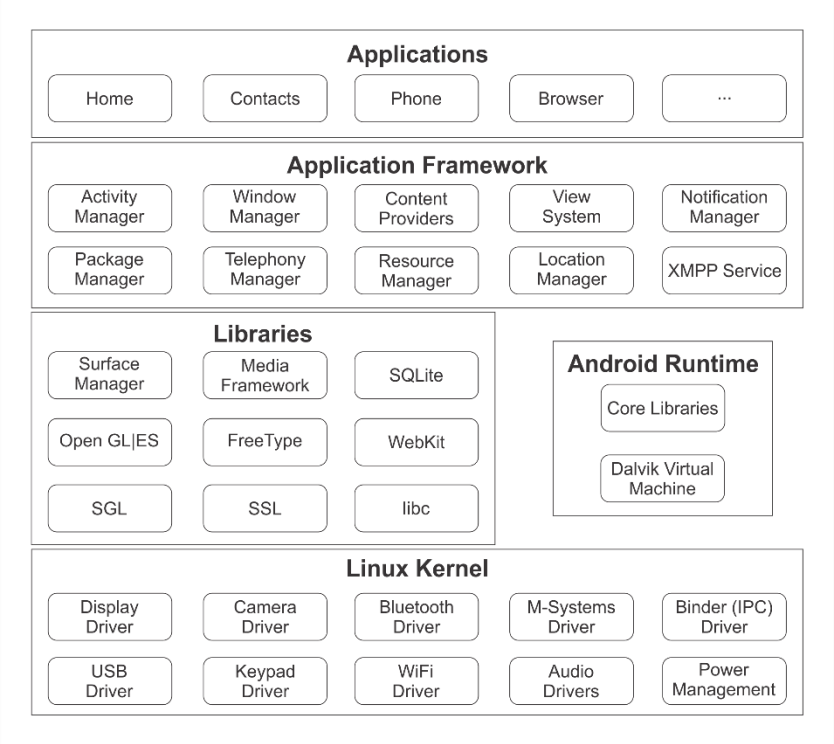
2.2.3 Android

Android merupakan sistem operasi mobile yang tumbuh di tengah sistem operasi lainnya yang berkembang dewasa ini. Sistem operasi lainnya seperti Windows Mobile, i-Phone OS, Symbian dan masih banyak lagi juga menawarkan kekayaan isi dan keoptimalan berjalan diatas perangkat hardware yang ada. Akan tetapi, sistem operasi yang ada ini berjalan dengan memprioritaskan aplikasi inti yang dibangun sendiri tanpa melihat potensi yang cukup besar dari aplikasi pihak ketig. Oleh karena itu, adanya keterbatasan distribusi aplikasi pihak ketiga untuk platform mereka.

Android menawarkan sebuah lingkungan yang berbeda untuk pengembang. Setiap aplikasi memiliki tingkatan yang sama. Android tidak membedakan antara aplikasi inti dengan aplikasi pihak ketiga. Application Programming Interface (API) yang disediakan menawarkan akses ke hardware, maupun data-data ponsel sekalipun, atau data sistem. Bahkan pengguna dapat menghapus aplikasi inti dan menggantikannya dengan aplikasi pihak ketiga. [3]

2.2.3.1 **Arsitektur Android**

Android terdiri dari beberapa stack software yang terdiri dari: *Applications, Application Framework, Libraries, Android Runtime* dan *Kernel Linux*. Arsitekturlengkap platform ini dapat dilihat pada gambar berikut.



Gambar 2.3 Struktur Android **[3]**

1. Aplication

   Application merupakan program yang langsung berhubungan dengan *user*. Baik program yang merupakan bawaan dari *Android* sendiri maupun program yang dibuat oleh *developer* menggunakan bahasa pemrograman *java.* Contoh program bawaan dari platform *Android* sendiri adalah *email client*, program SMS, *calendar, maps, web browser, contact* dan sebagainya.

2. Aplication Framework

   Lapisan ini berisi sekumpulan API yang dapat digunakan oleh *programmer* maupun *core application* dari *android.* Lapisan ini dirancang untuk memudahkan pengguna komponen dari *android* sendiri. Aplikasi maupun dalam *android* dapat berbagi fungsi sehingga aplikasi lain dapat memanfaatkannya. Aplikasi pada Android disusun atas beberapa komponen:

   a. Sekumpulan Views. Digunakan untuk mengatur tampilan pada aplikasi. Contohnya adalah lists, grids, text box, button, bahkan embeddable webbrowser.

b. Content providers. Komponen yang mengatur agar aplikasi dapat mengakses resources dari aplikasi lain (seperti Contacts), atau berbagi data dengan aplikasi lain.
c. Resource Manager. Menyediakan akses ke pada resource non-code seperti localized string, grafik dan file layout.
d. Notification Manager. Memungkinkan agar suatu aplikasi dapat menampilkanperingatan yang dapat di kostumasi pada status bar.
e. Activity Manager. Mengatur siklus aplikasi dan navigasi antar aplikasi yang sedang berjalan.

3. Libraries

Android mendukung beberapa library C/C++ yang digunakan pada berbagai komponen Android. Kemampuan ini dapat diakses oleh developer melalui Android application framework. Beberapa library diantaranya adalah:

a. System C library. Implementasi library C standar (libc).
b. Media Libraries. Mendukung berbagai format multimedia (termasuk MPEG4,H.264, MP3, AAC, AMR, JPG, PNG).
c. Surface Manager. Mengatur akses ke subsistem display.
d. LibWebCore. Engine web browser modern.
e. SGL. Engine grafis 2D.
f. 3D Library. Implementasi OpenGL ES 1.0 yang mendukung akselerasi hardware.
g. FreeType. Rendering untuk bitmap dan vector font.
h. SQLite. Basis data relasional yang kecil namun sangat ampuh.

4. Android Runtime

Tiap aplikasi pada Android memiliki proses-nya masing-masing. Tiap aplikasi tersebut memiliki instans dari Dalvik virtual machine (VM). Dalvik virtual machine dirancang agar suatu devicedapat menjalankan beberapa VM secara efisien. Dalvik VM mengeksekusi file dengan format Dalvik Executable format (.dex) yang dirancang untuk meminimalkan memory footprint . Dalvik VM berbasis register, dan dapat menjalankan kelas-kelas yang

dikompilasi dengan bahasa pemrograman java dan ditransformasikan menjadi format .dex. Dalvik VM sendiri bergantung pada Kernel Linux untuk fungsi dasarnya, seperti threading dan manajemen memori secara low-level.

5. Linux Kernel

Android menggunakan Kernel Linux versi 2.6 sebagai sistem utama. Fungsi kernel yang digunakan antara lain untuk keamanan, manajemen memori, manajemen proses, manajemen jaringan dan driver model. Kernel juga berfungsi sebagai layer abstrak antara hardware dan lapisan lainnya pada software stack.

2.2.3.2 Sejarah Android

Pada Juli 2000, *Google* bekerjasama dengan *Android Inc*., perusahaan yang berada di Palo Alto, California, Amerika Serikat. Para pendiri *Android Inc*. bekerja pada *Google,* diantaranya 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 menunjukan indikasi bahwa *Google* sedang bersiap menghadapi persaingan dalam pasara telepon seluler. Versi *android* terbaru yaitu versi 5.0. (*KitKat*). *Android* juga sudah bergabung dengan beberapa *smart mobile* seperti *LG, Samsung, Sony Ericsson, HTC, Motorola* dan lainnya.

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 Corporotaion* dan tersedia di pasaran pada 5 Januari 2010).

Pada 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 penmbahan fitur baru. [4]

2.2.3.3 Versi Android

Banyak *smartphone* dan PC Tablet menggunakan sistem operasi dengan versi yang berbeda. Semakin tinggi versi, fiturnya semakin canggih dan banyak. Telepon pertama yang memakai sistem operasi *Android* adalah HTC Dream yang dirilis pada tanggal 22 Oktober 2008. Beberapa uraian versi *android* seperti dibawah ini :

1. Android Versi 1.1

Pada 9 Maret 2009, *Google* merilis *Android* versi 1.1. *Android* versi ini dilengkapi dengan pembaruan estetis pada aplikasi, jam alarm, *voice search* (pencarian suara), pengiriman pesan dengan Gmail, dan pemberitahuan Email.

2. Android Versi 1.5 (*Cupcake*)

Pada pertengahan Mei 2009, *Google* kembali merilis telepon seluler dengan menggunakan *Android* dan SDK (*Software Development Kit*) dengan versi 1.5 (*Cupcake*). Terdapat beberapa pembaruan termasuk juga penambahan beberapa fitur dalam seluler versi ini yakni kemampuan merekam dan menonton video dengan modus kamera, mengunggah video ke *Youtube* dan gambar ke Picasa langsung dari telepon, dukungan *Bluetooth* A2DP, kemampuan terhubung secara otomatis ke *headset bluetooth*, animasi layar dan *keyboard* pada layar yang dapat disesuaikan dengan sistem.

3. Android Versi 1.6 (*Donut*)

Donut (versi 1.6) dirilis pada September dengan menampilkan proses pencarian yang lebih baik dibanding sebelumnya, penggunaan baterai indikator dan kontrol *applet* VPN. Fitur lainnya adalah galeri yang memungkinkan pengguna untuk memilih foto yang akan dihapus pada kamera, *camcorder* dan galeri yang diintegrasikan pada CDMA / EVDO, 802.1x, VPN, *Gestures*, dan

Text-to-spech engine. Kemampuan dial kontak teknologi *text to change speech* (tidak tersedia pada semua ponsel).

4. Android Versi 2.0/2.1 (*Éclair*)

Pada 3 Desember 2009 kembali diluncurkan ponsel *Android* dengan versi 2.0/2.1 (*Eclair*), perubahan yang dilakukan adalah mengoptimalkan *hardware*, peningkatan Google Maps 3.1.2, perubahan UI dengan browser baru dan dukungan HTML 5, daftar kontak yang baru, dukungan *flash* untuk kamera 3.2 MP, *digital zoom*, dan *Bluetooth* 2.1. Untuk bergerak cepat dalam persaingan perangkat generasi berikut, Google melakukan investasi dengan mengadakan kompetisi aplikasi *mobile* terbaik (*killer apps*-aplikasi unggulan). Kompetisi ini berhadiah $25,000 bagi setiap pengembang aplikasi terpilih. Kompetisi diadakan selama dua tahap yang tiap tahapnya dipilih 50 aplikasi terbaik. Dengan semakin berkembangnya dan semakin bertambahnya jumlah *handset Android*, semakin banyak pihak ketiga yang berminat untuk menyalurkan aplikasi mereka kepada sistem operasi *Android*. Aplikasi terkenal yang diubah ke dalam sistem operasi *Android* adalah Shazam, *Backgrounds* dan *WeatherBug*. Sistem operasi *Android* dalam situs internet juga dianggap penting untuk menciptakan aplikasi *Android* asli, contohnya oleh *MySpace* dan *Facebook*.

5. Android Versi 2.2 (*Froyo : Frozen Yoghurt*)

Pada 20 Mei 2010, *Android* versi 2.2 (Froyo) diluncurkan. Perubahan-perubahan umumnya terhadap versi-versi sebelumnya antara lain dukungan Adobe Flash 10.1, kecepatan kinerja dan aplikasi 2 sampai 5 kali lebih cepat, integrasi dengan V8 JavaScript *engine* yang dipakai Google Chrome yang mempercepat kemampuan rendering pada browser, pemasangan aplikasi dalam SD Card, kemampuan WiFi Hotspot portabel dan kemampuan *auto update* dalam aplikasi Android Market.

6. Android Versi 2.3 (*Gingerbread*)

Pada 6 Desember 2010, Android versi 2.3 (*Gingerbread*) diluncurkan. Perubahan-perubahan umum yang didapat dari versi ini antara lain peningkatan

kemampuan permainan (*gaming*), peningkatan fungsi *copy paste*, layar antarmuka (*User Interface*) didesain ulang, dukungan format video VP8 dan WebM. efek audio baru (*reverb, equalization, headphone virtualization*, dan *bass boost*), dukungan kemampuan *Near Field Communication* (NFC) dan dukungan jumlah kamera yang lebih dari satu.

7. Android Versi 3.1 (*Honeycomb*)

Android Honeycomb dirancang khusus untuk tablet. *Android* versi ini mendukung ukuran layar yang lebih besar. *User Interface* pada *Honeycomb* juga berbeda karena sudah didesain untuk tablet. *Honeycomb* juga mendukung multi prosesor dan juga akselerasi perangkat keras (*hardware*) untuk grafis. Tablet peratam yang dibuat dengan menjalankan *Honeycomb* adalah Motorola Xoom.

8. Android Versi 4.0 (*Ice Cream Sandwich*)

Diumumkan pada tanggal 19 Oktober 2011, membawa fitur *Honeycomb* untuk *smartphone* dan menambahkan fitur baru termasuk membuka kunci dengan pengenalan wajah, jaringan data pemantauan penggunaan dan kontrol, terpadu kontak jaringan sosial, perangkat tambahan fotografi, mencari email secara *offline* dan berbagi informasi dengan menggunakan NFC.

9. Android Versi 4.1 (*Jelly Bean*)

Pada tanggal 27 Juni 2012 Google mengumumkan *Android* versi 4.1 (*Jelly Bean*) dalam konferensi Google I/O berdasarkan kernel Linux 3.0.31. *Jelly Bean* adalah pembaruan penting yang bertujuan untuk meningkatkan fungsi dan kinerja antarmuak pengguna (*User Interface*). Pembaruan ini diwujudkan dalam proyek *Butter*, perbaikan ini termasuk antisipasi sentuh, *triple buffering*, perpanjangan waktu vsync dan peningkatan *frame rate* hingga 60fps untuk menciptakan UI yang lebih halus. Perangkat pertama yang menggunakan sistem operasi ini adalah tablet Nexus 7, yang dirilis pada 13 Juli 2012.

10. Android Versi 4.4 (*KitKat*)

Android versi 4.4 (KitKat) direncanakan akan dirilis pada bulan Oktober 2013. Setelah sebelumnya beredar rumor bahwa Android versi berikutnya setelah *Jelly Bean* diperkirakan akan diberi nomor 5.0 dan dinamai *Key Lime Pe*. [4]

2.2.3.4 Komponen Aplikasi Android

Fitur penting Android adalah bahwa satu aplikasi dapat menggunakan elemen dari aplikasi lain (untuk aplikasi yang memungkinkan). Sebagai contoh, sebuah aplikasi memerlukan fitur *scroller* dan aplikasi lain telah mengembangkan fitur *scroller* yang baik dan memungkinkan aplikasi lain menggunakannya. Maka pengembang tidak perlu lagi mengembangkan hal serupa untuk aplikasinya, cukup menggunakan *scroller* yang telah ada. Agar fitur tersebut dapat bekerja, sistem harus dapat menjalankan aplikasi ketika setiap bagian aplikasi itu dibutuhkan, dan pemanggilan objek java untuk bagian itu. Oleh karenanya Android berbeda dari sistem-sistem lain, Android tidak memiliki satu tampilan utama program seperti fungsi main() pada aplikasi lain. Sebaliknya, aplikasi memiliki komponen penting yang memungkinkan sistem untuk memanggil dan menjalankan ketika dibutuhkan [4].

1. Activities

Activity merupakan bagian yang paling penting dalam sebuah aplikasi, karena *Activity* menyajikan tampilan visual program yang sedang digunakan oleh pengguna. Setiap *Activity* dideklarasikan dalam sebuah kelas yang bertugas untuk menampilkan antarmuka pengguna yang terdiri dari *Views* dan respon terhadap *Event*. Setiap aplikasi memiliki sebuah *activity* atau lebih. Biasanya pasti akan ada *activity* yang pertama kali tampil ketika aplikasi dijalankan.

Perpindahan antara *activity* dengan *activity* lainnya diatur melalui sistem, dengan memanfaatkan *activity stack*. Keadaan suatu *activity* ditentukan oleh posisinya dalam tumpukan *acitivity*, LIFO (*Last In First Out*) dari semua aplikasi yang sedang berjalan. Bila suatu *activity* baru dimulai, *activity* yang sebelumnya digunakan maka akan dipindahkan ketumpukan paling atas. Jika pengguna ingin menggunakan *activity* sebelumnya, cukup menekan tombol *Back*, atau menutup *activity* yang sedang digunakan, maka *activity* yang berada diatas akan aktif

kembali. *Memory Manager* Android menggunakan tumpukkan ini untuk menentukan prioritas aplikasi berdasarkan *activity*, memutuskan untuk mengakhiri suatu aplikasi dan mengambil sumber daya dari aplikasi tersebut [4].

Ketika *activity* diambil dan disimpan dalam tumpukkan *activity* terdapat 4 kemungkinan kondisi transisi yang akan terjadi :

a. *Active*

Setiap *activity* yang berada ditumpukan paling atas, maka dia akan terlihat, terfokus, dan menerima masukkan dari pengguna. Android akan berusaha untuk membuat *activity* aplikasi ini untuk untuk tetap hidup dengan segala cara, bahkan akan menghentikan *activity* yang berada dibawah tumpukkannya jika diperlukan. Ketika *activity* sedang aktif, maka yang lainnya akan dihentikan sementara [9].

b. *Paused*

Dalam beberapa kasus *activity* akan terlihat tapi tidak terfokus pada kondisi inilah disebut *paused*. Keadaan ini terjadi jika *activity* transparan dan tidak *fullscreen* pada layar. Ketika *activity* dalam keadaan *paused*, dia terlihat *active* namun tidak dapat menerima masukkan dari pengguna. Dalam kasus ekstrim, Android akan menghentikan *activity* dalam keadaan *paused* ini, untuk menunjang sumber daya bagi *activity* yang sedang aktif [4].

c. *Stopped*

Ketika sebuah *activity* tidak terlihat, maka itulah yang disebut *stopped*. *Activity* akan tetap berada dalam memori dengan semua keadaan dan informasi yang ada. Namun akan menjadi kandidat utama untuk dieksekusi oleh sistem ketika membutuhkan sumberdaya lebih. Oleh karenanya ketika suatu *activity* dalam kondisi *stopped* maka perlu disimpan data dan kondisi antarmuka saat itu. Karena ketika *activity* telah keluar atau ditutup, maka dia akan menjadi *inactive* [4].

d. *Inactive*

Kondisi ketika *activity* telah dihentikan dan sebelum dijalankan. *Inactive activity* telah ditiadakan dari tumpukan *activity* sehingga perlu

restart ulang agar dapat tampil dan digunakan kembali. Kondisi transisi ini sepenuhnya ditangani oleh manajer memori Android. Android akan memulai menutup aplikasi yang mengandung *activity inactive*, kemudian *stopped activity*, dan dalam kass luar biasa *paused activity* juga akan ditutup [4].

2. Services

Suatu *service* tidak memiliki tampilan antarmuka, melainkan berjalan di *background* untuk waktu yang tidak terbatas. Komponen *service* diproses tidak terlihat, memperbarui sumber data dan menampilkan notifikasi. *Service* digunakan untuk melakukan pengolahan data yang perlu terus diproses, bahkan ketika *Activity* tidak aktif atau tidak tampak [4].

3. Intents

Intents merupakan sebuah mekanisme untuk menggambarkan tindakan tertentu, seperti memilih foto, menampilkan halaman *web*, dan lain sebagainya. *Intents* tidak selalu dimulai dengan menjalankan aplikasi, namun juga digunakan oleh sistem untuk memberitahukan ke aplikasi bila terjadi suatu hal, misal pesan masuk. *Intents* dapat eksplisit atau implisit, contohnya jika suatu aplikasi ingin menampilkan *URL*, sistem akan menentukan komponen apa yang dibutuhkan oleh *Intents* tersebut [4].

4. Broadcast Receiver

Broadcast Receivers merupakan komponen yang sebenarnya tidak melakukan apa-apa kecuali menerima dan bereaksi menyampaikan pemberitahuan. Sebagian besar *Broadcast* berasal dari sistem misalnya, Batre sudah hampir habis, informasi zona waktu telah berubah, atau pengguna telah merubah bahasa *default* pada perangkat. Sama halnya dengan *service*, *Broadcast Receivers* tidak menampilkan antarmuka pengguna. Namun, *Broadcast Receivers* dapat menggunakan *Notification Manager* untuk memberitahukan sesuatu kepada pengguna [4].

5. Content Providers

Content Providers digunakan untuk mengelola dan berbagi *database*. Data dapat disimpan dalam file sistem, dalam database *SQLite*, atau dengan cara lain

yang pada prinsipnya sama. Dengan adanya *Content Provider* memungkinkan antar aplikasi untuk saling berbagi data. Komponen ini sangat berguna ketika sebuah aplikasi membutuhkan data dari aplikasi lain, sehingga mudah dalam penerapannya [4].

2.2.3.5 Tipe Aplikasi Android

Terdapat tiga kategori aplikasi pada android dapat dilihat pada penjelasan sebagai berikut :

1. *Foreground Activity*

Aplikasi yang hanya dapat dijalankan jika tampil pada layar dan tetap efektif walaupun tidak terlihat. Aplikasi dengan tipe ini mempertimbangkan siklus hidup *activity*, sehingga perpindahan antar *activity* dapat berlangsung dengan lancar [4].

2. *Background Service*

Aplikasi yang memiliki interaksi terbatas dengan user, selain dari pengaturan konfigurasi, semua dari prosesnya tidak tidak tampak pada layar. Contohnya aplikasi penyaringan panggilan atau sms *auto respon* [4].

3. *Intermittent Activity*

Aplikasi yang masih membutuhkan beberapa masukkan dari pengguna, namun sebagian sangat efektif jika dijalankan di *background* dan jika diperlukan akan memberi tahu pengguna tentang kondisi tertentu. Contohnya pemutar musik. Untuk aplikasi yang kompleks akan sulit untuk menentukan kategori aplikasi tersebut apalagi aplikasi memiliki ciri-ciri dari semua kategori. Oleh karenanya perlu pertimbangan bagaimana aplikasi tersebut digunakan dan menentukan kategori aplikasi yang sesuai [4].

2.2.3.6 Kelebihan Android

Sudah banyak *platform* untuk perangkat selular saat ini, termasuk didalamnya *Symbian*, *iPhone*, *Windows Mobile*, *BlackBerry*, *Java Mobile Edition*, *Linux Mobile* (LiM), dan banyak lagi. Namun ada beberapa hal yang menjadi kelebihan Android. Walaupun beberapa fitur-fitur yang ada telah muncul

sebelumnya pada *platform* lain, Android adalah yang pertama menggabungkan hal seperti berikut :

1. Keterbukaan, bebas pengembangan tanpa dikenakan biaya terhadap sistem karena berbasiskan *Linux* dan *open source*. Pembuat perangkat menyukai hal ini karena dapat membangun *platform* yang sesuai yang diinginkan tanpa harus membayar *royality*. Sementara pengembang *software* menyukai karena Android dapat digunakan diperangkat manapun dan tanpa terikat oleh *vendor* manapun.
2. Arsitektur komponen dasar android terinspirasi dari teknologi *internet Mashup*. Bagian dalam sebuah aplikasi dapat digunakan oleh aplikasi lainnya, bahkan dapat diganti dengan komponen lain yang sesuai dengan aplikasi yang dikembangkan.
3. Banyak dukungan *service*, kemudahan dalam menggunakan berbagai macam layanan pada aplikasi seperti penggunaan layanan pencarian lokasi, *database* SQL, *browser* dan penggunaan peta. Semua itu sudah tertanam pada Android sehingga memudahkan dalam pengembangan aplikasi.
4. Siklus hidup aplikasi diatur secara otomatis, setiap program terjaga antara satu sama lain oleh berbagai lapisan keamanan, sehingga kerja sistem menjadi lebih stabil. Pengguna tak perlu khawatir dalam menggunakan aplikasi pada perangkat yang memorinya terbatas.
5. Dukungan grafis dan suara terbaik, dengan adanya dukungan 2D grafis dan animasi yang diilhami oleh Flash menyatu dalam 3D menggunakan *OpenGL* memungkinkan membuat aplikasi maupun *game* yang berbeda.

   Portabilitas aplikasi, aplikasi dapat digunakan pada perangkat yang ada saat ini maupun yang akan datang. Semua program ditulis dengan menggunakan bahas pemrograman Java dan dieksekusi oleh mesin *virtual Dalvik*, sehingga kode program *portable* antara ARM, X86, dan arsitektur lainnya. Sama halnya dengan dukungan masukan seperti penggunaan *Keyboard*, layar sentuh, *trackball* dan resolusi lyr semua dapat disesuaikan dengan program [4].

2.2.4 Basis Data

Sistem basis data adalah sistem terkomputerisasi yang tujuan utamanya adalah memelihara data yang sudah diolah atau informasi dan membuat informasi tersedia saat dibutuhkan. Pada intinya basis data adalah media untuk menyimpan data agar dapat diakses dengan mudah dan cepat [5].

Sistem informasi tidak dapat dipisahkan dengan kebutuhan akan basis data apapun bentuknya, entah berupa file teks ataupun Database Managemenet System (DBMS). Kebutuhan basis data dalam sistem informasi meliputi:

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

2.2.5 DBMS

DBMS (Database Management System) atau dalam bahasa Indonesia sering disebut sebagai Sistem Manajemen Basis Data adalah suatu sistem aplikasi yang digunakan untuk menyimpan, mengelola, dan menampilkan data [5]. Suatu sistem aplikasi disebut DBMS jika memenuhi persyaratan minimal sebagai berikut:

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

2.2.6 Konsep Dasar Berorientasi Object

Pendekatan berorientasi objek merupakan suatu teknik atau cara pendekatan dalam melihat permasalahan dan sistem (sistem perangkat lunak, sistem informasi, atau sistem lainnya). 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 (dihubungkan) 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 [5].

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:

1. 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 (model/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.

2. Objek (*Object*)

Objek adalah abstraksi dari sesuatu yang mewakili dunia nyata 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 makan akan dibuat sebuah objek. Objek dilihat dari segi 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.

3. 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.

4. 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.

5. 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.

6. 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.

7. Pewarisan (*Inheritance*)

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

8. 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.

9. *Reusability*

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

10. 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.

11. Komunikasi Antarobjek

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

12. Polimorfisme (*Polymorphism*)

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

13. *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.

2.2.7 GPS (Global Positioning System)

Global Positioning System adalah sistem navigasi berbasis satelit terdiri dari jaringan 24 satelit ditempatkan ke orbit oleh Departemen Pertahanan Amerika Serikat yang pertama kali diperkenalkan mulai tahun 1978. Layanan GPS dahulu hanya dipergunakan untuk keperluan militer namun mulai terbuka untuk publik.

24 satelit GPS tersebut berada sekitar 12.000 mil di atas bumi bergerak mengelilingi bumi 12 jam dengan kecepatan 7.000 mil per jam. Satelit GPS berkekuatan energi sinar matahari, memiliki baterai cadangan untuk menjaga agar tetap berjalan pada saat gerhana matahari atau pada saat tidak ada energi matahari dan memiliki roket penguat kecil pada masing-masing satelit agar dapat mengorbit tepat pada tempatnya.

Satelit-satelit GPS harus selalu berada pada posisi orbit yang tepat untuk menjaga akurasi data yang dikirim ke GPS *receiver*, sehingga harus selalu dipelihara agar posisinya tepat. Posisi satelit-satelit tersebut selalu dipantau oleh stasiun pengendali. Stasiun-stasiun pengendali di bumi ada di Hawaii, Ascension Islan, Diego Garcia, Kwajalein dan Colorado Spring. Untuk dapat mengetahui posisi seseorang maka diperlukan alat yang diberi nama GPS *receiver* yang berfungsi untuk menerima sinyal yang dikirim dari satelit GPS. GPS *receiver* mengambil informasi tersebut dan melakukan perhitungan triangulation untuk menentukan lokasi pengguna dengan tepat. GPS receiver membandingkan waktu sinyal dikirim dengan waktu sinyal tersebut diterima untuk mengetahui jarak satelit. Dengan mengetahui jarak tersebut, GPS receiver dapat melakukan perhitungan dan menentukan posisi pengguna dan menampilkan dalam peta elektronik. Setelah menentukan posisi pengguna, selanjutnya GPS dapat menghitung informasi lain, seperti kecepatan, arah yang dituju, jalur, tujuan perjalanan, jarak tujuan, matahari terbit dan matahari terbenam dan masih banyak lagi.

Satelit GPS sangat presisi dalam mengirim informasi karena satelit tersebut menggunakan jam atom. Jam atom yang ada pada satelit merupakan partikel atom yang diisolasi, sehingga dapat menghasilkan jam yang akurat dibandingkan dengan jam biasa. Keistimewaan GPS adalah mampu bekerja dalam berbagai kondisi cuaca, siang atau malam. Keakuratan sebuah perangkat GPS bisa mencapai 15 meter, bahkan model terbaru yang dilengkapi teknologi Wide Area Augmentation System (WAAS) keakuratannya sampai 3 meter. Karena GPS bekerja mengandalkan satelit, maka penggunaannya disarankan di tempat terbuka.

Penggunaan di dalam ruangan, atau di tempat yang menghalangi arah satelit (di angkasa), maka GPS tidak akan bekerja secara akurat dan maksimal. Perhitungan waktu yang akurat sangat menentukan akurasi perhitungan untuk menentukan informasi lokasi. Semakin banyak sinyal satelit yang dapat diterima maka akan semakin presisi data posisi yang dihasilkan. Selain itu, ketinggian juga mempengaruhi proses kerja GPS, karena semakin tinggi maka semakin bersih atmosfer, sehingga gangguan semakin sedikit [6].

Sinyal yang diterima oleh alat penerima (receiver) di permukaan, dimana GPS receiver ini akan mengumpulkan informasi dari satelit GPS [6], seperti :

a. Waktu, GPS receiver menerima informasi waktu dari jam atom yang mempunyai keakurasian sangat tinggi.
b. Lokasi, GPS memberikan informasi lokasi dalam tiga dimensi:
    1. Latitude
    2. Longitude
    3. Elevasi
c. Kecepatan, ketika berpindah tempat, GPS dapat menunjukkan informasi kecepatan berpindah tersebut.
d. Arah perjalanan, GPS dapat menunjukkan arah tujuan.
e. Simpan lokasi, tempat-tempat yang sudah pernah atau ingin dikunjungi bisa disimpan oleh GPS receiver.
f. Komulasi data, GPS receiver dapat menyimpan informasi track, seperti total perjalanan yang sudah pernah dilakukan, kecepatan rata-rata, kecepatan paling tinggi, kecepatan paling rendah, waktu/jam sampai tujuan, dan sebagainya.

2.2.8 Latitude dan Longitude

Latitude dan longitude adalah sebuah koordinat yang ada di bumi dan digunakan untuk menentukan kedudukan kita di atas bumi [7]. Berikut penjelasan latitude dan longitude:

1. Latitude

Latitude adalah garis yang melintang dari kutub utara dan kutub selatan. Titik 0 adalah sudut ekuator, tanda + menunjukan arah ke atas menuju kutub utara,

sedangkan tanda minus di koordinat Latitude menuju ke kutub selatan. Titik yang dipakai dari 0 ke 90 derajat arah kutub utara, dan 0 ke -90 derajat ke kutub selatan.

2. Longitude

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

2.2.9 LBS (Location Based-Service)

Salah satu ciri mobile phone atau tablet pc adalah portabilitas, sehingga tidak mengherankan bahwa beberapa fitur Android sangat menarik seperti layanan yang memungkinkan anda menemukan, mencari serta menvisualisasikan posisi kita ke dalam peta lokasi fisik seperti GoogleMaps. Kita dapat membuat peta berbasis GoogleMaps dan menjadikannya sebagai element dalam layout User Interface (UI) yang kita rancang. Kita dapat melakukan akses penuh ke peta GoogleMaps, dan memungkinkan kita untuk mengontrol pengaturan tampilan, mengubah tampilan zoom lokasi, dan memindahkan lokasi tampilan [6].

Location Based Service adalah service yang berfungsi untuk mencari dengan teknologi Global Positioning Service (GPS) dan Google’s cell-based location. Map dan layanan berbasis lokasi menggunakan lintang dan bujur untuk menentukan lokasi geografis, namun sebagai user kita membutuhkan alamat atau posisi realtime kita bukan nilai lintang dan bujur. Android menyediakan geocoder yang mendukung forward dan reverse geocoding. Menggunakan geocoder, anda dapat mengkonversi nilai lintang bujur menjadi alamat dunia nyata atau sebaliknya.

Location based service atau layanan berbasis lokasi adalah istilah umum yang digunakan untuk menggambarkan teknologi yang digunakan untuk menemukan lokasi perangkat yang kita gunakan. Dua unsur utama LBS [6] adalah:

a. Location Manager (API Maps)

Menyediakan tools/resource untuk LBS, Application Programming Interface (API) Maps menyediakan fasilitas untuk menampilkan, memanipulasi maps/peta beserta feature-feature lainnya seperti tampilan satelit, street (jalan), maupun gabungannya. Paket ini berada pada com.google.android.maps.

b. Location Providers (API Location)

Menyediakan teknologi pencarian lokasi yang digunakan oleh device/perangkat. API Location berhubungan dengan data GPS dan data lokasi realtime. API Location berada pada paket android yaitu dalam paket android.location. Dengan Location Manager, kita dapat menentukan lokasi kita saat ini, Track gerakan/perpindahan, serta kedekatan dengan lokasi tertentu dengan mendeteksi perpindahan.

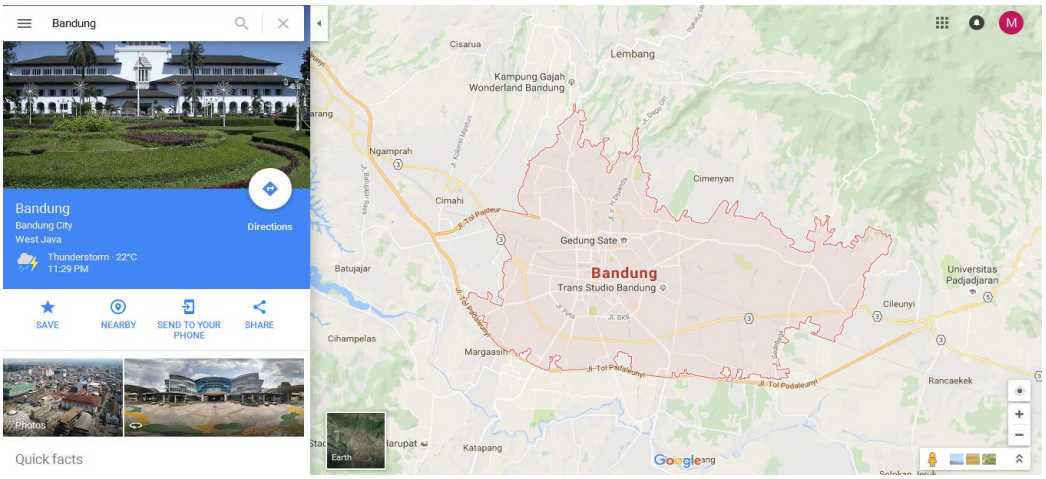
2.2.10 Google Maps

GoogleMaps adalah peta online atau membuka peta secara online, dapat dilakukan secara mudah melalui layanan gratis dari Google. Bahkan layanan ini menyediakan API (Application Programming Interface) yang memungkinkan developer lain untuk memanfaatkan aplikasi ini di aplikasi buatannya. Tampilan GoogleMaps pun dapat dipilih, berdasarkan foto asli atau peta gambar rute saja.

GoogleMaps adalah layanan gratis yang diberikan oleh Google dan sangat popular. GoogleMaps adalah suatu peta dunia yang dapat kita gunakan untuk melihat suatu daerah seperti pada gambar 2.3. Dengan kata lain, GoogleMaps merupakan suatu peta yang dapat dilihat dengan menggunakan suatu browser. Kita dapat menambahkan fitur GoogleMaps dalam web yang telah kita buat atau pada blog kita yang berbayar maupun gratis sekalipun dengan GoogleMaps API. GoogleMaps API adalah suatu library yang berbentuk JavaScript.

Cara membuat GoogleMaps untuk ditampilkan pada suatu web atau blog sangat mudah hanya dengan membutuhkan pengetahuan mengenai HTML serta JavaScript, serta koneksi Internet yang sangat stabil. Dengan menggunakan GoogleMaps API, kita dapat menghemat waktu dan biaya untuk membangun aplikasi peta digital yang handal, sehingga kita dapat fokus hanya pada data-data

yang akan ditampilkan. Dengan kata lain, kita hanya membuat suatu data sedangkan peta yang akan ditampilkan adalah milik Google sehingga kita tidak dipusingkan dengan membuat peta suatu lokasi, bahkan dunia.



Gambar 2.4 Google Maps

Pada GoogleMaps API terdapat 4 jenis pilihan model peta yang disediakan oleh Google, diantaranya adalah :

a. ROADMAP, untuk menampilkan peta biasa 2 dimensi.
b. SATELITE, untuk menampilkan foto satelit.
c. TERRAIN, untuk menunjukkan relief fisik permukaan bumi dan menunjukkan seberapa tingginya suatu lokasi, contohnya akan menunjukkan gunung dan sungai.
d. HYBRID, akan menunjukkan foto satelit yang diatasnya tergambar pula apa yang tampil pada ROADMAP (jalan dan nama kota).

2.2.11 Geofencing

Geofence adalah parameter virtual untuk wilayah geografis di dunia nyata. Parameter virtual yang ditetapkan ini bisa menjadi statis ataupun dinamis. Salah satu contoh khas dari geofence dinamis adalah radius sekitar toko atau titik lokasi. geofence statis dapat di atur dari batas yang telah ditetapkan, seperti zona kehadiran di sekolah atau batas lingkungan. Kostum digital geofencing juga dapat digunakan, jika perangkat yang menyediakan layanan berbasis lokasi (LBS) pengguna masuk atau keluar geofencing, perangkat akan menerima pesan

notifikasi. Notifikasi ini mungkin berisi informasi tentang lokasi dari perangkat. Notifikasi Geo-fence mungkin dikirimkan ke ponsel sebagai contoh meningkatkan setiap bentuk lain dari tindakan.

Kita bisa menggambarkan geo fencing sebagai LBS proaktif. Dalam banyak kasus sistem proaktif jauh lebih nyaman dari yang reaktif, di mana pengguna harus secara eksplisit meminta data untuk lokasi. Ada berbagai jenis lokasi peristiwa untuk memperbaiki posisi GPS yang dapat diuji. Misalnya, apakah pengguna di dekat dengan tempat tujuan (POI) atau ke pengguna lain.

Geofence sekarang ini menjadi salah satu wajah baru di Layanan Berbasis Lokasi. Sekarang pengembangan asli untuk LBS sendiri lebih atau kurang selesai. Kita bisa melihat beberapa standar umum dan aplikasi (berpikir tentang berbagai implementasi layanan Tempat-tempat seperti Foursquare, Twitter places, Google place, Facebook place, dll) Semua dari mereka memudahkan Anda dalam berbagi lokasi info atau mendapatkan beberapa hal yang berada di dekatnya. Dan geo-fence membuka pintu untuk beberapa personalisasi. Anda bisa mendapatkan beberapa kustom disesuaikan dengan data ditempat yang benar, terutama ketika kedua aliran data dan tempat (area) untuk menerima itu dihasilkan secara dinamis. Pada kedua, geo-fence akan terintegrasi dalam waktu terdekat dengan sensor dan aplikasi M2M [8].

2.2.12 Geotagging

Geotagging atau juga dikenal sebagai GPS *photo tagging*, merupakan proses penambahan informasi posisi data GPS (*Latitude*, *Longitude*, *Altitude*) dalam sebuah foto digital. Ponsel-ponsel berkamera yang memiliki GPS *receiver* internal umumnya memiliki fitur ini. Mekanisme GPS *photo tagging* adalah pada saat sebuah foto diambil menggunakan kamera (digital atau ponsel) yang memiliki fitur *geotagging*, kamera atau ponsel tersebut mencatat lebih banyak informasi/data dibandingkan dengan sebuah foto yang diambil dengan kamera biasa. Informasi tersebut termasuk waktu dan data ketika sebuah foto diambil, orientasi dari kamera (*portrait* atau *landscape*), apakah menggunakan lampu flash dan detil kamera lainnya yang digunakan seperti Apertur, Exposure, dan Local

Length. Semua data ini disimpan pada suatu tempat yang disebut EXIF Headers. EXIF (*Exchangeable Image File Format*) headers berisi petunjuk foto dengan data yang dapat dibaca oleh aplikasi manajemen foto atau sebuah situs foto tertentu. Selain itu, EXIF *Headers* juga menyediakan sebuah ruang untuk mengisi koordinat *Longitude*, *Latitude*, dan *Altitude* [9].

2.2.12.1 EXIF Headers

Data EXIF (*Exchangeable Image File Format*) adalah data yang disimpan oleh kamera digital pada sebuah image yang berisi informasi mengenai kondisi dan setting kamera digital pada waktu dilakukan pemotretan. EXIF dikembangkan oleh Japanese Electronics Industry Development Association (JEIDA) sebagai upaya untuk mempermudah dan membuat standar pertukaran data antara perangkat lunak pengolah gambar/citra digital dan perangkat keras seperti kamera, serta didukung oleh hampir seluruh kamera digital.

Informasi umum yang bisa didapatkan dari EXIF sebuah foto digital adalah sebagai berikut [9]:

1. Tanggal dan jam berapa sebuah foto dibuat.
2. Merk, tipe kamera, dan jenis lensa yang dipakai untuk memotret foto.
3. Resolusi dari kamera yang digunakan untuk mengambil gambar, digunakan atau tidaknya fitur flash pada kamera.
4. Lokasi foto tersebut diambil, jika kamera yang digunakan mendukung GPS.

2.2.12.2 EXIF Interface Android Class

Kelas Exif Interface adalah salah satu kelas dalam pemrograman android yang dapat digunakan untuk membaca dan menulis informasi tag Exif pada file yang berformat JPEG. Berikut adalah tabel 2.1 Tag EXIF informasi Exif yang dapat diambil [10].

Tabel 2.2 Tag EXIF [14]

No	Tagging	Keterangan
1	TAG_APERTURE	Type is String
2	TAG_DATETIME	Type is String
3	TAG_DATETIME_DIGITIZED	Type is String

4	TAG_EXPOSURE_TIME	Type is String
5	TAG_FLASH	Type is int
6	TAG_FOCAL_LENGTH	Type is rational
7	TAG_GPS_ALTITUDE	The altitude (in meters) based on the reference in TAG_GPS_ALTITUDE_REF
8	TAG_GPS_ALTITUDE_REF	0 if the altitude is above sea level
9	TAG_GPS_DATESTAMP	Type is String
10	TAG_GPS_LATITUDE	String
11	TAG_GPS_LATITUDE_REF	Type is String
12	TAG_GPS_LONGITUDE	String
13	TAG_GPS_LONGITUDE_REF	Type is String
14	TAG_GPS_PROCESSING_METHOD	Type is String
15	TAG_GPS_TIMESTAMP	Type is String
16	TAG_IMAGE_LENGTH	Type is int
17	TAG_IMAGE_WIDTH	Type is int
18	TAG_ISO	Type is String
19	TAG_MAKE	Type is String
20	TAG_MODEL	Type is String
21	TAG_ORIENTATION	Type is int
22	TAG_SUBSEC_TIME	Type is int
23	TAG_SUBSEC_TIME_DIG	Type is int
24	TAG_SUBSEC_TIME_ORIG	Type is int
25	TAG_WHITE_BALANCE	Type is int

2.2.12.2.1 Google Geocading

Google Maps sudah memiliki fitur yang cukup lengkap, termasuk mencari lokasi kota pada peta (*geocoding*). *Geocoding* adalah proses menemukan koordinat geografis, yaitu bujur dan lintang, dari data geografis lain, misalnya alamat. Fitur *geocoding* dapat dimanfaatkan untuk menambahkan informasi *latitude* dan *longitude* pada foto yang tidak memiliki data *geotagging* [9].

2.2.13 UML (*Unified Modeling Language*)

Unified Modeling Language adalah keluarga notasi grafis yang didukung oleh mera-model tunggal, yang membantu pendeskripsian dan desain sistem

perangkat lunak, khususnya sistem yang dibangun menggunakan pemrograman berorientasi objek (OOP).

Pemodelan visual adalah suatu cara berpikir tentang persoalan menggunakan model-model yang diorganisasikan seputar dunia nyata. Model berguna untuk memahami persoalan, mengkomunikasikan dengan orang-orang yang terlibat dalam proyek (customer, ahli dibidangnya, analis, desainer, dan lain-lain), memodelkan perusahaan, menyiapkan dokumentasi, merancang program, dan merancang basis data.

Pemodelan visual didefinisikan sebagai proses pemodelan sistem informasi menggunakan pengaturan standar elemen grafik. Tujuan utama pemodelan visual adalah untuk memungkinkan adanya komunikasi antara pengguna, pengembang, penganalisis, tester, manager, dan siapapun yang terlibat dalam proyek, menunjukkan interaksi antara pengguna dengan sistem, obyek-obyek dalam sistem, dan antar sistem itu sendiri [11]

2.2.13.1Use Case Diagram

Usecase diagram digunakan untuk memodelkan dan menyatakan unit fungsi/layanan yang disediakan oleh sistem (or bagian sistem: *subsistem* atau *class*) ke pemakai. *Usecase* dapat dilingkupi dengan batasan sistem yang diberi label nama sistem. *Usecase* adalah sesuatu yang menyediakan hasil yang dapat diukur ke pemakai atau sistem eksternal.

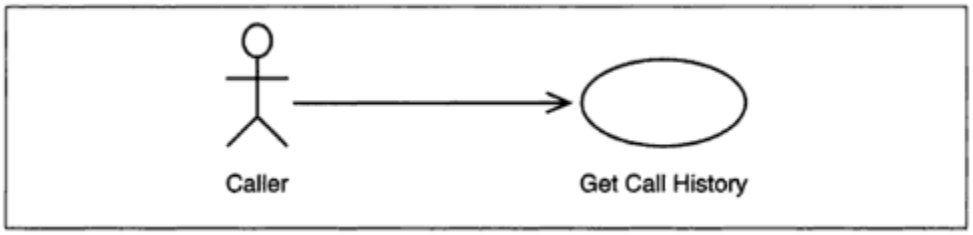
Usecase: aktor, dan asosiasi mereka dapat ditampilkan pada diagram use case. Satu diagram ini, kita melihat seorang aktor yang disebut "pelanggan" dan usecase disebut "*get call history*" arah panah menunjukkan bahwa pelanggan memulai komunikasi.

Use case Diagram memberikan pandangan dan memberikan perpectives yang berbeda pada model usecase tunggal. diagram *usecase* mungkin hanya berisi aktor, hanya usecase, atau kombinasi dari keduanya. umum digunakan usecase diagram adalah sebagai berikut:

1. Diagram gambaran yang menunjukkan semua usecase dan aktor
2. Diagram ringkasan aktor menampilkan serangkaian aktor terkait konseptual

3. Aktor diagram perspektif menunjukkan semua usecase melibatkan aktor
4. Diagram ringkasan penggunaan-kasus yang menunjukkan seperangkat usecase konseptual terkait
5. Usecase diagram perspektif menggambarkan bagaimana usecase yang berhubungan dengan aktor dan kasus pengguna lain yang patut.

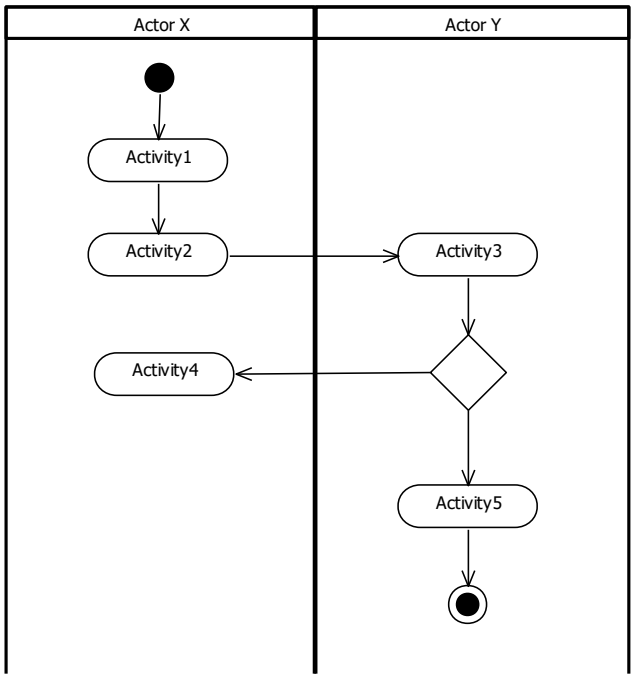
Usecase dibentuk untuk merepresentasikan bagaimana suatu sistem bekerja atau berjalan. Suatu sistem tidak lepas dari bagian-bagian tertentu untuk membentuk sistem tersebut dibentuk [11].



Gambar 2.5 Contoh Usecase diagram sederhana

2.2.13.2Aktivity Diagram

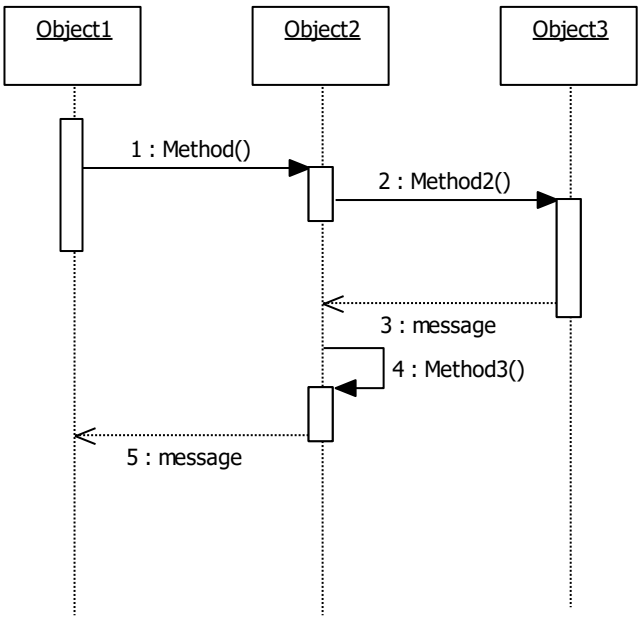
Diagram Aktifitas atau *aktivity diagram* menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis. Yang perlu diperhatikan disini adalah bahwa *diagram* aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan actor, jadi aktivitas yang dapat dilakukan oleh sistem. [12] Dibawah ini merupakan *activity diagram* gambar berikut :



Gambar 2.6 Contoh Bentuk *Activity Diagram*

2.2.13.3Sequence Diagram

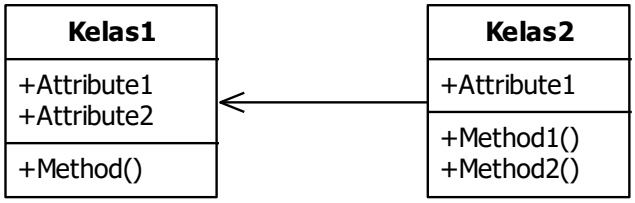
Sequence Diagram menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek. [12] Dibawah ini merupakan *sequence diagram* gambar berikut [11].



Gambar 2.7 Contoh Bentuk *Sequence Diagram*

2.2.13.4 Class Diagram

Class Diagram menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Pada Gambar 2.8 adalah contoh *class diagram* untuk menggambarkan struktur sistem yang akan dibuat untuk membangun sistem [11].



Gambar 2.8 Contoh Bentuk *Class Diagram*

2.2.14 Google Rest Api

Rest Api adalah suatu jenis *web service* yang menerapkan konsep perpindahan antara *state*. *State* disini dapat digambarkan seperti jika *browser* meminta suatu halaman *web,* maka *server* mengirimkan *state* halaman *web* yang sekarang ke *browser.* Bernavigasi melalui *link-link* yang disediakan sama halnya dengan bernavigasi melalui *link-link* HTTP untuk melakukan aktivitas tertentu, seakan-akan terjadi perpindahan *state* satu sama lain. Perpindahan HTTP yang bisa digunakan adalah fungsi GET, POST, PUT atau DELETE. Balasan yang dikirim adalah bentuk XML sederhana tanpa ada protokol pemetaan data, sehingga informasi yang diterima lebih mudah dibaca dan dipasang disisi *client.*

2.3 Tools Pendukung

Dalam pembangunan sistem program terdapat beberapa *tools* perangkat pendukung yang di gunakan untuk membangun sebuah sistem ini ssebagai berikut:

2.3.1 Notepad ++

Notepad++ adalah sebuah text editor yang sangat berguna bagi setiap orang dan khususnya bagi para developer dalam membuat program. Notepad++ menggunakan komponen Scintilla untuk dapat menampilkan dan menyuntingan

teks dan berkas kode sumber berbagai bahasa pemrograman yang berjalan diatas sistem operasi Microsoft Windows.

Selain manfaat dan kemampuannya menangani banyak bahasa pemrograman, Notepad ++ juga dilisensikan sebagai perangkat free. Jadi, setiap orang yang menggunakannya tidak perlu mengeluarkan biaya untuk membeli aplikasi ini karena sourceforge.net sebagai layanan yang memfasilitasi Notepad ++ membebaskannya untuk digunakan. Beberbapa daftar bahasa program yang didukung oleh Notepad++ adalah C, C++, Java, C#, XML, HTML, PHP, Javascript.

2.3.2 Processor Hypertext Protocol (PHP)

PHP adalah suatu bahasa *server-side* yang didesain khusus untuk aplikasi web. PHP dapat disisipkan diantara HTML, dan karena bahasa *server-side*, maka bahasa PHP akan dieksekusi di server, yang dikirimkan ke browser adalah "hasil jadi" bentuk HTML, dan kode PHP tidak akan terlihat. PHP dahulu merupakan proyek pribadi dari Rasmus Lerdorf tahun 1994 (dengan dikeluarkannya PHP versi 1 yang digunakan untuk membuat home page pribadi yang dipublikasikan 8 juni 1995). Versi pertama ini berupa kumpulan script PERL. Untuk versi kedua Ramus menulis ulang script-script PERL tersebut menggunakan bahasa C, kemudian menambahkan fasilitas untuk form HTML dan koneksi MYSQL. Adapun PHP didapat dari singkatan *PHP Hypertext Preprocessor*.

Setelah mengalami perkembangan, Zeev Suraski dan Andi Gutmans (termasuk Rasmus) yang tergabung dalam suatu kelompok open source, maka mereka memulai versi 3 PHP nya, PHP telah menampakkan keunggulan sebagai salah satu bahasa *server scripting* yang handal. Melalui perkembangan yang pesat ini banyak fasilitas yang ditambahkan oleh kelompok ini PHP disebut sebagai "PHP: Hypertext Preprocessor". Pada bulan Mei 2000 dirilis PHP versi 4 yang dikeluarkan oleh Zend Engine yaitu sebuah produk yang diciptakan oleh Suraski dan Gutmans. Kehandalan PHP diperlengkap dengan dirilisnya PHP versi 5 pada 13 Juli 2004 yang mendukung OOP serta berbagai manajemen database yang luas dan penanganan error yang mudah. Sintak yang digunakan berasal dari bahasa C,

Java maupun PERL. Sampai tulisan ini dibuat versi PHP yang terbaru adalah versi 5.2.0 yang dirilis pada November 2006.

PHP merupakan bahasa script yang digunakan untuk membuat halaman web yang dinamis. Dinamis berarti halaman yang akan ditampilkan dibuat saat halaman itu diminta oleh client. Mekanisme ini menyebabkan informasi yang diterima client selalu yang terbaru. Semua script PHP dieksekusi pada server dimana script tersebut dijalankan. Oleh karena itu, spesifikasi server lebih berpengaruh pada eksekusi dari script php daripada spesifikasi client. Namun tetap diperhatikan bahwa web yang dihasilkan tentunya harus dapat dibuka oleh browser pada client. Dalam hal ini versi dari html yang digunakan harus didukung oleh browser client. PHP termasuk dalam *Open Source Product*. Jadi dapat menambah *source code* dan dapat mendistribusikannya secara bebas. PHP juga diedarkan secara gratis. Kita bisa mendapatkannya secara gratis. PHP juga dapat berjalan diberbagai web server semisal IIS, Apache, PWS.

Dengan demikian kode program yang ditulis dalam PHP tidak akan terlihat oleh user sehingga keamanan halaman web lebih terjamin. PHP dirancang untuk membuat halaman web yang dinamis, yaitu halaman web yang dapat membentuk suatu tampilan berdasarkan permintaan terkini, seperti menampilkan isi basis data ke halaman web [13].

```
<?php
        header('Access-Control-Allow-Origin: *');
        require_once("../class/about.php");
        $obj = new About();
        echo json_encode($obj);
?>
```

Gambar 2.9 Contoh Skrip PHP [23]

2.3.3 Database Mysql

Database adalah kumpulan informasi yang disimpan di dalam komputer secara sistematik sehingga dapat diperiksa menggunakan suatu program komputer untuk memperoleh informasi dari basis data tersebut. MySQL adalah sebuah

perangkat lunak system manajemen basis data SQL (DBMS) yang multithread, dan multi-user. MySQL adalah implementasi dari system manajemen basis data relasional (RDBMS). MySQL dibuah oleh TcX dan telah dipercaya mengelola system dengan 40 buah database berisi 10.000 tabel dan 500 di antaranya memiliki 7 juta baris. MySQL AB merupakan perusahaan komersial Swedia yang mensponsori dan yang memiliki MySQL.

MySQL AB adalah dua orang Swedia yang bernama David Axmark, Allan Larsson dan satu orang Finlandia bernama Michael "Monty". Setiap pengguna MySQL dapat menggunakannya secara bebas yang didistribusikan gratis dibawah lisensi GPL (General Public License) namun tidak boleh menjadikan produk turunan yang bersifat komersial.

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

Kepopuleran MySQL antara lain karena MySQL menggunakan SQL sebagai bahasa dasar untuk mengakses databasenya sehingga mudah untuk digunakan, kinerja query cepat, dan mencukupi untuk kebutuhan database perusahaan-perusahaan yang berskala kecil sampai menengah, MySQL juga bersifat open source (tidak berbayar). MySQL merupakan database yang pertama kali didukung oleh bahasa pemrograman script untuk internet (PHP dan Perl). MySQL dan PHP dianggap sebagai pasangan software pembangun aplikasi web yang ideal. MySQL lebih sering digunakan untuk membangun aplikasi berbasis web, umumnya pengembangan aplikasinya menggunakan bahasa pemrograman script PHP [13] .

Pada saat ini MySQL merupakan database server yang sangat terkenal di dunia, semua itu tak lain karena bahasa dasar yang digunakan untuk mengakses

database yaitu SQL. SQL (Structured Query Language) pertama kali diterapkan pada sebuah proyek riset pada laboratorium riset San Jose, IBM yang bernama system R. Kemudian SQL juga dikembangan oleh Oracle, Informix dan Sybase. Dengan menggunakan SQL, proses pengaksesan database lebih user-friendly dibandingan dengan yang lain, misalnya dBase atau Clipper karena mereka masih menggunakan perintah-perintah pemrograman murni. SQL dapat digunakan secara berdiri sendiri maupun di lekatkan pada bahasa pemrograman seperti C, dan Delphi.

Elemen dari SQL yang paling dasar antara lain pernyataan, nama, tipe data, ekspresi, konstanta dan fungsi bawaan. Pernyataan Perintah dari SQL yang digunakan untuk meminta sebuah tindakan kepada DBMS.

Sistem database menyediakan bahasa untuk mendefinisikan database Data Definition Language (DDL) serta bahasa untuk memanipulasi database Data Manipulation Language (DML) untuk melakukan operasi-operasi tertentu pada database dan juga Data Control Language (DCL).

a. Data Definition Language (DDL)

Kumpulan perintah SQL yang digunakan untuk membangun database, bisa juga memiliki arti lain perintah untuk membuat struktur dasar database dan tabel. Perintah-perintah yang termasuk DDL adalah statment connect, create, show, describe, drop, rename, dan alter.

b. Data Manipulation Language (DML)

Kumpulan statement yang digunakan sebagai perintah untuk mengelola data dalam sebuah tabel. Perintah DML digunakan setelah menggunakan DDL antara lain seperti statement insert, update, delete, dan select.

c. Data Control Language (DCL)

Perintah yang digunakan untuk keperluan keamanan (security) database dengan membuat hak akses tertentu bagi setiap user. Statment pada DCL antara lain grant, revoke, set, dan lock table.

Dalam MySQL ada beberapa operasi dasar yang lebih dikenal dengan operasi CRUD sebagai berikut:

1. *Create*

Create adalah operasi penambahan data baru kedalam table. Terdapat 2 Query untuk menambah data, yang pertama adalah penambahan yang tidak menspesifikasikan nama kolom yang akan ditambahkan dan hanya memberikan isi dari tabelnya. Query nya adalah sebagai berikut:

```
INSERT INTO table_name VALUES (value1,value2,value3,...);
```

Untuk yang kedua adalah menginputkan data dengan mencantumkan nama kolom yang akan diisikan dengan isi di dalamnya. Query nya adalah sebagai berikut:

```
INSERT INTO table_name (column1,column2,column3,...)

VALUES (value1,value2,value3,...);
```

2. *Read*

Read adalah operasi untuk menampilkan semua atau sebagian data yang berada di dalam database. Query nya adalah sebagai berikut:

```
SELECT column_name,column_name
FROM table_name;
```

3. *Update*

Update adalah operasi untuk mengubah data yang ada di dalam database. Query nya adalah sebagai berikut:

```
UPDATE table_name SET column1=value1,column2=value2,...
WHERE some_column=some_value;
```

4. *Delete*

Delete adalah operasi untuk menghapus data yang ada di dalam table. Query nya adalah sebagai berikut:

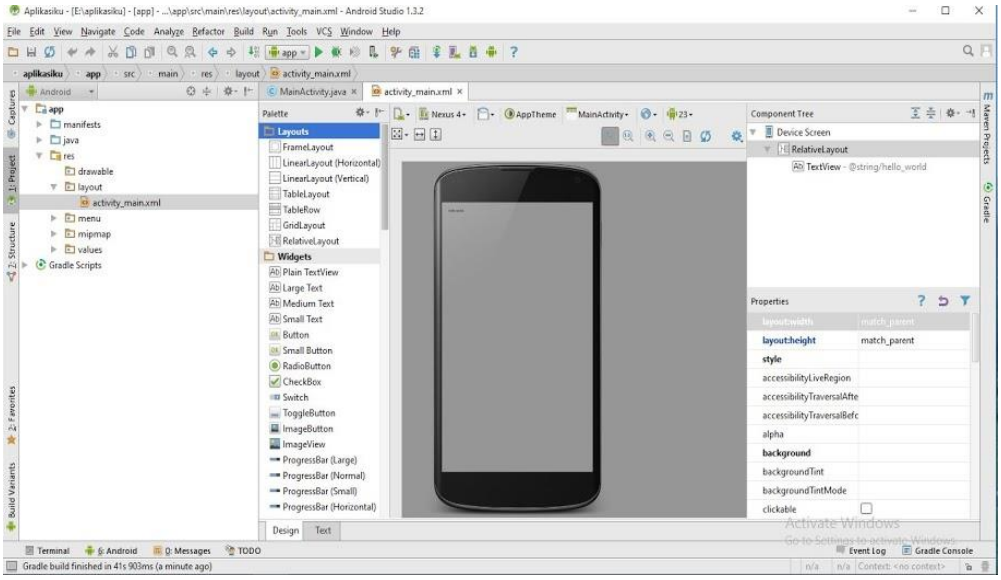
```
DELETE FROM table_name
```

WHERE some_column=some_value;

2.3.4 Android Studio

Android studio adalah lingkungan pengembangan terpadu (*IDE)* resmi untuk pengembangan platform android, hal itu disampaikan pada tanggal 16 mei 2013 Google I/O,android studio sudah tersedia secara bebas dibawah lisensi Apache 2.0.

Android studio pada awalnya tahap *preview* versi 0.1 yang dipakai pada tanggal 1 mei 2013 dan memasuki tahap beta pada bulan juni 2014 dan mulai stabil dirilis pada Desember 2014 dengan versi 1.0 ,berdasarkan *jetBrains* 'IDEA *Intellij Software*, android studio dirancang khusus untuk pengembangan android yang tersedia untuk Windows,Mac OS X dan Linux sebagai pengganti *Eclipse* [14].



Gambar 2.10 Layout Android Studio [14]

2.3.4.1 Struktur Android Studio

Berikut adalah rincian dari struktur android studio dapat dijelaskan sebagai berikut :

1. App : berfungsi untuk menyimpan file file pendukung aplikasi yang akan diolah.
   a. *Manifest*

Tempat dimana konfigurasi konfigurasi pendukung project serta izin (permision) untuk mengakses hardware.

b. *Java*

Tempat berdiamnya *code/scripts* java untuk menjalankan aplikasi yang akan dibuat.

c. *Res*

- *Drawable* : tempat untuk menyimpan gambar atau *icon* pendukung aplikasi.
- *Layout* : tempat berkumpulnya design tampilan yang akan tampak pada tampilan aplikasi anda.
- *Menu* : berisi *code* berupa *file* xml yang didalamnya berisi *array* tentang menu yang akan tampil dalam *project* anda, misalkan *file*, *new*,*open*,*close*,dsb.
- *Mipmap* : tempat berkumpulnya *icon launcher* (icon utama aplikasi) yang ukurannya menyesuaikan *device* yang digunakan.
- *Values* : untuk menyimpan file .xml yang menggambarkan sumber dari berbagai jenis. Sebagai contoh folder berisi *file strings*.xml dengan definisi sumber *string*. Seperti yang akan anda lihat dari menambahkan bagian *color , folder* tata letak juga dapat berisi, misalnya, deskripsi warna. Di dalam file dimens.xml, terdapat *string-string* untuk konfigurasi dimensi aplikasi. Tidak jauh berbeda dengan dimens.xml maupun *strings*.xml, pada file styles.xml terdapat deklarasi variable untuk theme.

2. *Gradle Scripts* : berfungsi sebagai *configurasi* aplikasi anda dengan *device*, serta menghubungkan *library* pendukung aplikasi yang anda buat. Penggunaan android studio ini untuk membuat aplikasi rumah pintar di *smartphone* android yang sebagai perangkat bantuan untuk mengontrol rumah pintar [14].

2.3.5 Java

Java menurut definisi dari Sun adalah nama untuk sekumpulan teknologi untuk membuat dan menjalankan perangkat lunak pada komputer standalone ataupun pada lingkungan jaringan. Java dikembangkan pada bulan Agustus 1991, dengan nama semula Oak. Pada Januari 1995, karena nama Oak dianggap kurang komersial, maka diganti menjadi Java. Pada Desember 1998, Sun memperkenalkan nama "Java 2" (J2) sebagai generasi kedua dari java platform. Konvensi nama baru ini diterapkan untuk semua edisi Java yaitu Standard Edition (J2SE), Enterprise Edition (J2EE), dan Micro Edition (J2ME) [15].

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

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

2.3.5.1 Kelebihan Java

Adapun beberapa kelebihan dari bahasa pemrograman Java adalah sebagai berikut [15] :

1. Sederhana dan Ampuh

Java dirancang untuk mudah dipelajari, terutama bagi programmer yang telah mengenal C/C++ akan mudah sekali untuk berpindah ke Java. Pemakai dapat belajar membuat program dengan Java secara cepat jika telah memahami konsep dasar pemrograman berorientasi objek. Java tidak memiliki hal-hal yang mengejutkan dan aneh. Java memberi anda kemampuan untuk menuangkan semua ide, karena bahasa pemrograman ini bukan merupakan scripting language (bahasa

naskah) yang menghilangkan kemampuan kita untuk berinovasi, tetapi dengan cara berorientasi objek yang mudah dan jelas.

2. Aman

Java dirancang sebagai bahasa pemrograman yang handal dan aman. Aplikasi-aplikasi yang dibangun dengan bahasa Java sangat handal dengan manajemen memori yang bagus. Aplikasi Java juga dikenal sangat secure, yaitu kasus-kasus seperti buffer everflow yang umumnya menjadi lubang keamanan aplikasi-aplikasi berbasis C/C++ tidak terjadi di Java, karena pengaturan securitynya yang bagus.

3. Berorientasi Objek

Paradigma pemrograman berorientasi objek merupakan paradigma pemrograman masa depan. Java merupakan bahasa pemrograman berorientasi objek. Java bukan turunan langsung dari bahasa pemrograman manapun, juga sama sekali tidak kompetibel dengan semuanya. Java memiliki keseimbangan, Java secara cepat jika telah memahami konsep dasar pemrograman berorientasi objek. Java tidak memiliki hal-hal yang mengejutkan dan aneh. Java memberi anda kemampuan untuk menuangkan semua ide, karena bahasa pemrograman menyediakan mekanisme peng-class-an sederhana, dengan model antar muka dinamik yang intuitif hanya jika diperlukan.

4. Kokoh

Java membatasi anda dari beberapa hal kunci supaya anda dapat menemukan kesalahan lebih cepat saat mengembangkan program. Java langsung memeriksa program saat anda menuliskannya, dan sekali lagi ketika program di jalankan. Karena Java adalah bahasa yang sangat ketat dalam hal tipe data dan deklarasi, banyak kesalahan umum terjadi saat kompilasi. Hal ini akan lebih menghemat waktu jika dibandingkan dengan keharusan menjalankan program terlebih dahulu dan memeriksa semua bagian program untuk melihat ketidakcocokan dinamis selama program berjalan. Ini adalah contoh di mana Java lebih luwes dan kokoh dari beberapa bahasa lain, tetapi dengan imbalan yang layak untuk kelebihan itu.

5. Interaktif

Java memiliki beberapa kemampuan yang memungkinkan program melakukan beberapa hal pada saat bersamaan, tanpa harus kesulitan menangani proses yang akan terjadi selanjutnya. Jalinan program-program Java yang mudah digunakan memungkinkan kita untuk memikirkan pembuatan perilaku khusus, tanpa harus mengintegrasikan perilaku tersebut dengan model pemrograman global yang mengatur perulangan kejadian.

6. Netral terhadap Berbagai Arsitektur

Java telah mengambil beberapa keputusan yang sulit dalam pembuatan bahasa Java dan bagaimana program dijalankan, jadi anda dapat sepenuhnya percaya "tulis sekali, jalan di mana saja, kapan saja, selamanya".

7. Terinterpretasi dan Berkinerja Tinggi

Java dilengkapi keajaiban lintas platform yang luar biasa dengan kompilasi ke dalam representasi langsung yang disebut kode-byte Java (Java byte-code), yang dapat diterjemahkan oleh sistem manapun yang memilki program Java didalamnya. Java, bagaimanapun dirancang untuk tetap berkinerja baik pada CPU yang tidak terlalu kuat. Walaupun Java merupakan bahasa terinterpretasi, kodekode Java telah dirancang dengan hati-hati sehingga mudah diterjemahkan ke dalam bahasa asli suatu mesin untuk menghasilkan kinerja yang tinggi.

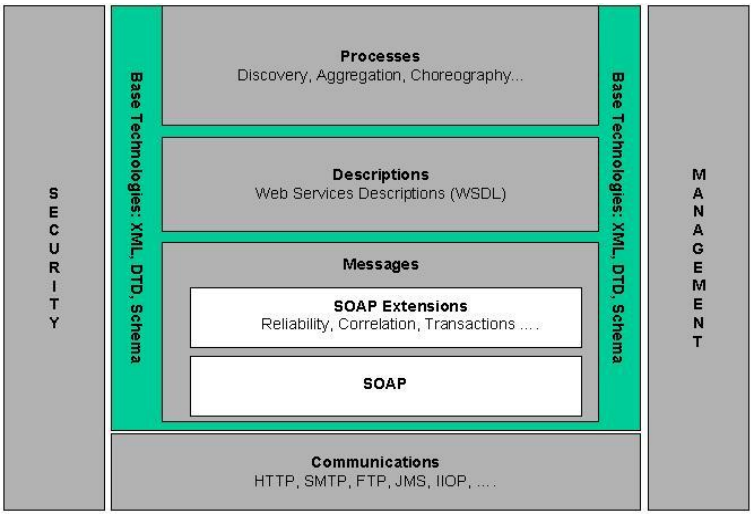
Sistem program Java yang melakukan optimasi tepat waktu tersebut tidak kehilangan keuntungan dari program yang netral terhadap platform. "lintas platform berkinerja tinggi" bukan sekedar omong-kosong. Dalam aplikasi Java (*.class) merupakan Java bytecode yang berjalan di atas jvm (Java Virtual Machine), yang kemudian jvm-lah yang akan menginterpresentasikan kode-kode tersebut ke kode native atau kode mesin dari arsitektur yang bersangkutan. Hal sangat menarik karena urusan arsitektur mesin bukan jadi masalah bagi programmer tapi menjadi urusan kompiler pada bahasa pemrograman Java.

2.3.6 Web Service

Web service adalah sebuah entitas komputasi yang dapat diakses melalui jaringan internet maupun intranet dengan standar protokol tertentu dalam *platform* dan antarmuka bahasa pemograman yang independen. Tujuan pengembangannya

adalah untuk menjembatani komunikasi antar program, sehingga aplikasi yang satu dan aplikasi yang lain yang terdapat pada suatu jaringan yang sama atau pada jaringan berbeda dapat saling berkomunikasi asalkan menggunakan standart protokol yang ditetapkan oleh *web service*. Hal ini bisa terjadi, karena standart protokol itu tidaklah terikat pada suatu *platform* atau bahasa pemograman. Protokol itu sendiri dibangun oleh *Extensible Markup Language* (XML) yang memang kenyataannya telah didukung oleh banyak *paltform*, bahasa pemograman, dan oleh *developer* di seluruh dunia.

Arsitektur *web service* dibangun oleh beberapa layer dan teknologi yang saling berhubungan. Banyak cara untuk memvisualisaskan *service*, sama banyaknya dengan cara untuk membangun dan menggunakan *web service* gambar berikut hanyalah salah satu cara untuk menggambarkan arsitekturnya.



Gambar 2.11 Arsitektur Web Service [16]

Web service disusun oleh tiga komponen standart, yaitu :

1. *Simple Object Access Protocol* (SOAP), yaitu protokol yang bertanggung jawab dalam pertukuran informasi dalam lingkungan jaringan terdistribusi.
2. *Web Service Definition Language* (WSDL), dokumen standart yang dituliskan pada format XML, dan mendefinisikan kehadiran web service dalam suatu jaringan.

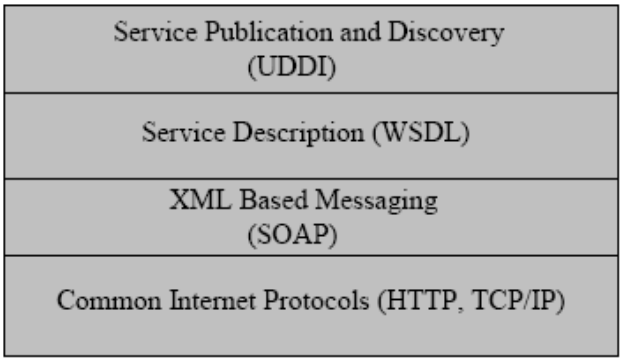
3. *Universal Description, Discovery and Integration* (UDDI), yaitu suatu lokasi direktori yang beriskan *service* (layanan) dan bersifat bebas *platform* (*platform independent*), dituliskan berbasis XML dan dapat diakses oleh entittas yang berada di dalam dan luar jaringan.

Adanya standart tersebut membuat web *service* mudah di akses melalui berbagai antarmuka dan juga memberi peluang dimungkinkannya berbagai sistem yang dibangun pada platform yang berbeda dan bahasa yang berbeda untuk berkolaborasi pada dalam suatu pekerjaan [16].

2.3.6.1 Komponen – Komponen Web Service

Web Service secara keseluruhan memiliki empat layer komponen yaitu sebagai berikut :

1. Layer 1 : Protokol internet standar seperti HTTP, TCP/IP
2. Layer 2 : *Simple Object Access Protocol* (SOAP), merupakan protokol akses objek berbasis XML yang digunakan untuk proses pertukaran data/informasi antar layanan.
3. Layer 3 : *Web Service Definition Language* (WSDL), merupakan suatu standar bahasa dalam format XML yang berfungsi untuk mendeskripsikan seluruh layanan yang tersedia.



Gambar 2.12 Komponen – Komponen Web Service [16]

2.3.7 JavaScript Object Natation (JSON)

JSON (*JavaScript Object Notation*) adalah format pertukaran data yang ringan, mudah dibaca dan ditulis oleh manusia, serta mudah diterjemahkan dan

dibuat (*generate*) oleh komputer. Format ini dibuat berdasarkan bagian dari Bahasa Pemprograman JavaScript, Standar ECMA-262 Edisi ke-3 - Desember 1999. JSON merupakan format teks yang tidak bergantung pada bahasa pemrograman apapun karena menggunakan gaya bahasa yang umum digunakan oleh programmer keluarga C termasuk C, C++, C#, Java, JavaScript, Perl, Python dll. Oleh karena sifat-sifat tersebut, menjadikan JSON ideal sebagai bahasa pertukaran-data [17].

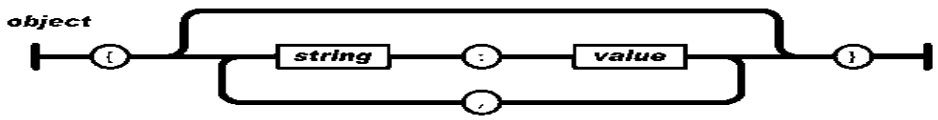
JSON terbuat dari dua struktur [17] :

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 value*). Pada kebanyakan bahasa, hal ini dinyatakan sebagai lirik (*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 [17].

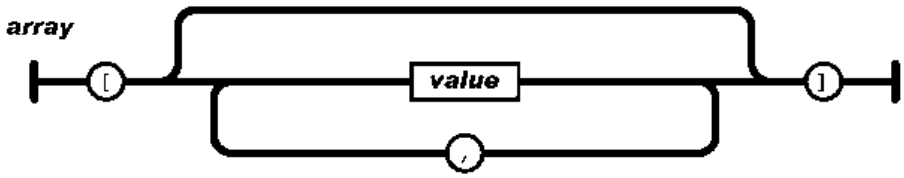
JSON menggunakan bentuk sebagai berikut [17] :

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).



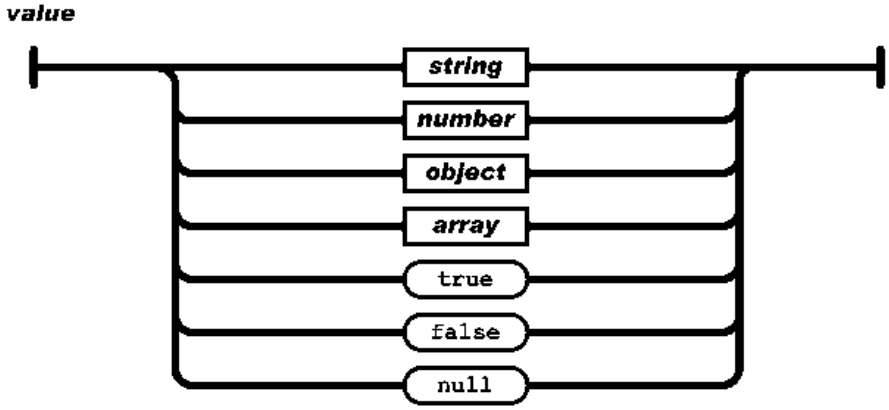
Gambar 2.13 Objek JSON [17]

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



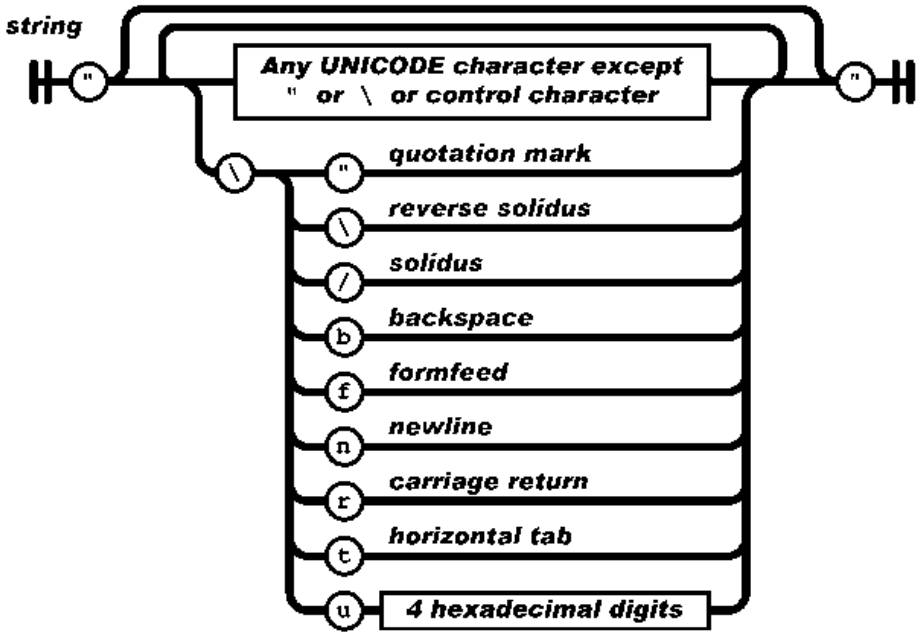
Gambar 2.14 Array JSON [17]

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.



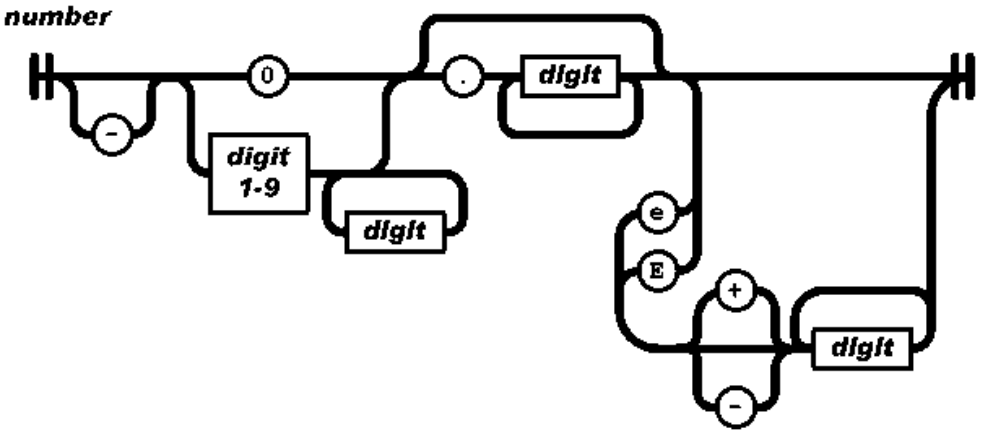
Gambar 2.15 Value JSON [17]

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 2.16 String JSON [17]

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



Gambar 2.17 Number JSON [17]

Spasi kosong (whitespace) dapat disisipkan di antara pasangan tanda-tanda tersebut, kecuali beberapa detil encoding yang secara lengkap dipaparkan oleh bahasa pemrograman yang bersangkutan.

2.3.8 Xampp

XAMPP adalah sebuah *software* yang berfungsi untuk menjalankan *website* berbasis PHP dan menggunakan pengolah data MySQL di komputer lokal. XAMPP berperan sebagai *server web* pada komputer. XAMPP juga dapat disebut juga CPanel server virtual, yang dapat membantu melakukan *priview* sehingga dapat memodifikasi *website* tanpa harus *online* atau terakses dengan *internet.* [18]