BAB 2
TINJAUAN PUSTAKA

2.1. Tinjauan Tempat Pemenelitian

Organisasi Kedaerahan KUMANDANG (Keluarga Mahasiswa Pandeglang - Bandung) berlamat di Jl. Tamansari Bawah No 27 / 59 Bandung 40116 Telpon (022) 426-4050, Website : http://www.kumandang.or.id email : info@kumandang.or.id.

2.1.1. Sejarah Organisasi

KUMANDANG merupakan organisasi kedaerahan yang berdiri sejak tahun 1964. Sebelum menjadi KUMANDANG organisasi ini bernama BKSMB (Badan Kerja Sama Mahasiswa Banten) yang berapda di Bandung, Jakarta, dan Yogyakarta, dimana dalam organisasi terbut merupakan organisasi yang terdiri dari beberapa ke residenan yang ada di Banten pada waktu itu, diantaranya Serang, Cilegon, Lebak dan Pandeglang. Tujuan organisasi ini merupakan untuk mempersatukan semua masyarakat Banten yang berada di luar Banten baik mahasiswa, siswa atau pun masyarakat biasa. Pada awal 1964 perkembangan mahasiswa asal daerah yang ada di Banten semakin berkembang. Maka munculah pergerakan dan kesepakan baru yang berasal dari mahasiswa Bandung asal Banten untuk memecah organisasi BKSM yang sekarang menjadi KMB (Keluarga Mahasiswa Banten) menjadi organisasi kedaerahan, diantaranya KUMALA (Keluarga Mahasiswa Lebak), KAMAYASA (Keluarga Mahasiswa Tirtayasa) dan KUMANDANG (Keluarga Mahasiswa Pandeglang) pada tahun 1964. Hal tersebut dilakukan untuk mempermudah jalur birokrasi antara mahasiswa daerah dengan pemerintah keresidenan (Kabupaten) yang ada di Banten.

Setelah pembentukan KUMANDANG yang di prakarsai oleh bapak H. Ir. Enan Romdani Sahim, M.Sp barulah pergerakan mahasiswa asal Pandeglang semakin terlihat. Diantaranya adalah pengiriman anggota KUMANDANG setiap tahun untuk membantu pembangunan di Pandeglang mulai dari pemerintahan, pendidikan dan lain – lain yang sifatnya untuk membantu pembangunan

Pandeglang. KUMANDANG memiliki fasilitas asrama sekaligus sekretariat yang berada di Tamansari yang merupakan dana dari APBD Pandeglang dan bapak Triyana Sjam'um sebagai donatur terbesar pada saat itu, serta donatur – donatur lain yang berasal dari mahasiswa Pandeglang yang dulunya pernah melanjutkan studinya di Bandung dan mayarakat Pandeglang yang berdomisili di Bandung.

2.1.2. Logo Organisasi

Pada gambar 2.1 merupakan logo dari organisasi KUMANDANG (Keluarga Mahasiswa Pandeglang - Bandung).



Gambar 2.1 Logo KUMANDANG

Arti dari logo KUMANDANG:

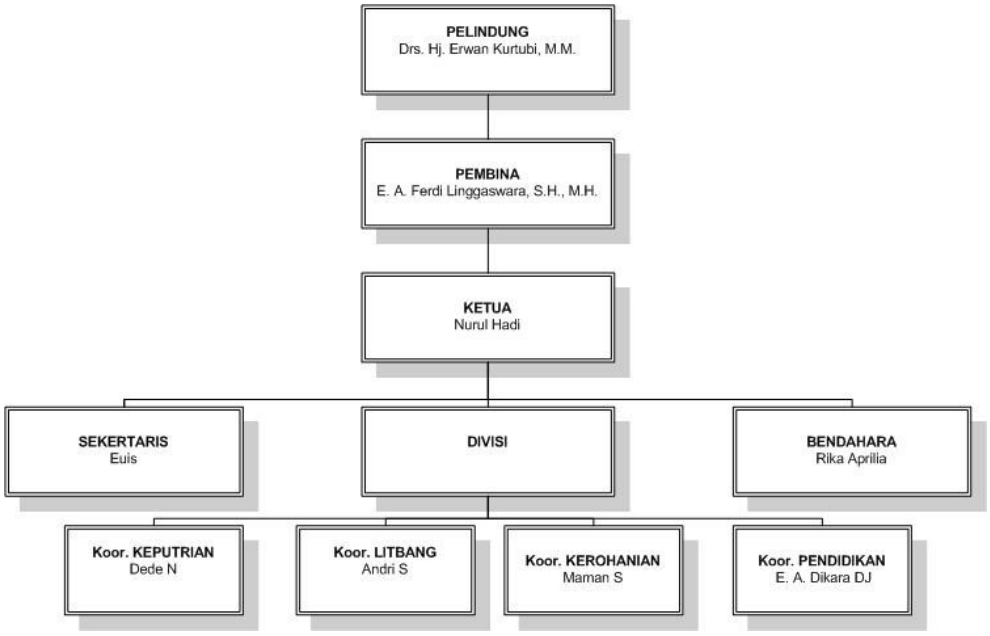
1. Garis diatas gambar yang menyerupai badak merupakan pegunungan karena Kabupaten Pandeglang merupakan daerah dataran tinggi dan memiliki beberapa gunung.
2. Gambar badak cula satu merupakan simbol atau ciri khas dari Kabupaten Pandeglang, karena Badak cula satu merupakan primata purba satu – satu nya di dunia yang berada di Taman Nasional Ujung Kulon – Kabupaten Pandeglang.
3. Bacaan KUMANDANG di samarkan menjadi bentuk seperti badak karena memiliki filosofi dengan sifat dan karakteristik badak itu sendiri. Kuliat tebal yang di miliki oleh badak melambangkan kuatnya mental mahasiswa asal Pandeglang, habitat yang dijaga untuk mengurangi kepunahan

merupakan gambaran mahasiswa Pandeglang yang selalu menjaga budaya daerahnya walaupun mereka hidup di kota lain, prilaku badak yang tidak brutal merupakan sifat mahasiswa asal Pandeglang yang selalu menjaga tingkah laku mereka, cula satu pada badak merupakan sifat mahasiswa Pandeglang yang selalu ingin menjadi yang terdepan dalam segala hal.

4. Tiga garis gelombang merupakan bentuk lautan, karena Pandeglang juga merupakan daerah yang di kelilingi oleh lautan

2.1.3. Struktur Organisasi

Pada gambar 2.2 merupakan struktur organisasi KUMANDANG (Keluarga Mahasiswa Pandeglang - Bandung) Periode 2012 – 2013.



Gambar 2.2 Struktur Organisasi KUMANDANG

Deksripsi dari masing-masing jabatan pada struktur organisasi KUMANDANG adalah sebagai berikut :

1. Pelindung, merupakan jabatan yang menaungi segala sesusautu yang berada di dalam organisasi KUMANDANG seperti kegiatan, permasalah organisasi serta bimbingan emosional terhadap semua elemen yang ada didalamnya yang langsung diduduki oleh bupati Kabupaten Pandeglang.

2. Pembina, merupakan pelindung terdekat yang membimbing setiap segala sesuautu yang terjadi organisasi KUMANDANG, jabatan sebagai pembina sekarang adalah kepala satpol PP Kota Bandung yang merupakan alumni dari organisasi KUMANDANG.
3. Ketua, merupakan jabatan yang memimpin organisasi KUMANDANG serta menaungi semua mahasiswa asal Pandeglang yang melanjutkan studi di Bandung.
4. Sekertaris, jabatan yang mempertanggung jawabkan segala sesuatu yang berkaitan dengan administrasi kepengurusan.
5. Bendahara, jabatan yang mempertanggung jawabkan segala seusuatu yang bersangkutan dengan keuangan organisasi KUMANDANG.
6. Divisi, dalam organisasi KUMANDANG terdapat 4 divisi, dimana setiap divisi memiliki fungsi yang berbeda, diantaranya :
    a. Divisi Keputrian, merupakan divisi yang berfungsi sebagai wadah untuk mahasiswa putri asal Pandeglang yang melanjutkan studi di Bandung.
    b. Divisi Litbang, merupakan divisi yang menaungi bidang penelitian dan pengembangan. Pada divisi ini setiap mahasiswa KUMANDANG dapat mengutarakan gagasan atau masukan yang nantinya dapat memberikan manfaat baik untuk semua mahasiswa asal Pandeglang yang melanjutkan studi di Bandung.
    c. Divisi Kerohaniah, merupakan divisi yang menaungi kegiatan religius seperti pengajian rutin, memberikan fasilitas belajar untuk anak-anak di lingkungan sekretariat KUMANDANG dan segala bentuk kegiatan yang berbau religius.
    d. Divisi Pendidikan, merupakan divisi yang menampung setiap anggota untuk dapat berkumpul, berdiskusi dan memberikan pengajaran terhadap anggota KUMANDANG seputar pendidikan.

2.2. Landasan Teori

Landasan teori menjelaskan beberapa teori yang berkaitan dengan permasalahan yang diabahas sebagai dasar pemahaman dalam sebuah sistem serta metode yang dipakai untuk kegiatan pengembangan terhadap sistem sendiri.

2.2.1. Basis Data

Basis data adalah suatu kumpulan data yang terhubung dan sidimpan secara bersama-sama pada suatu media penyimpan. Data disimpan disimpan sedemikian rupa agar mudah dalam menampilkan, menyimpan, menghubah dan menghapus data serta dapat dengan mudah diakases oleh suatu aplikasi. Berdasarkan pengertian diatas, maka dalam pembuatan sebuah basis data haruslah memiliki kriteria penting yang harus dipenuhi, yaitu : [3]

1. Berorientasi pada data (*data oriented*) dan bukan berorientasi pada program yang akan menggunakannya. Agar memenuhi kriteri ini maka basis data haruslah disimpan secara terpisan dengan program aplikasinya.
2. Data dapat digunakan oleh pengguna yang berbeda-beda atau oleh program yang berbeda-beda tanpa harus mengubah basis data.
3. Data dalam basis data dapat berkembang dengan baik isi maupun strukturnya.
4. Data yang ada dapar memenuhi kebutuhan sistem-sistem baru secara mudah.
5. Data dapat digunakan dengan cara yang berbeda-beda.
6. Data tidak boleh berangkap datu suatu basis data.

Ada beberapa istilah yang digunakan dalam basis data yang bertujuan untuk membermudah dalam sistem manajemen basis data (database manajemen system) diantaranya adalah :

1. Entitas

   Merupakan obyek yang dapat dibedakan dengan obyek lainnya yang dapat diwujudkan di dalam basis data

2. Atribut/Field

   Merupakan sebuah entitas yang terdapat pada suatu basis data atau biasa disebut dengan tabel pada basis data yang memiliki atribut.

3. Isi Data

   Merupakan informasi yang berupa hurup, angkat atau simbol yang disimpan di tiap atribut pada suatu entitas.

4. Record

   Merupakan kumpulan isi elemen data yang saling berhubungan menginformasikan tentang suatu entitas secara lengkap

5. File

   Merupakan kumpulan record sejenis yang mempunyai panjang elemen dan atribut yang sama namum memiliki value/nilai yang berbeda-beda.

2.2.1.1. Sistem Manajemen Basis Data

Sistem Manajemen Basis Data atau biasa disebut *Database Management System* (DBMS) adalah perangkat lunak yang didesain untuk membantu dalam hal pemeliharaan dan utilitas kumpulan data dalam jumlah besar, serta menjadi alternatif penggunaan secara khusus untuk aplikasi, seperti penyimpanan data dalam *file* dan menulis kode aplikasi yang spesifik untuk pengaturannya. Adapun keuntungan dari DBMS adalah sebagai berikut :

1. DBMS mempudah dalam melakukan perawatan data seperti membuat data baru, memperbaharui data, dan menghapus data.
2. DBMS mempermudah dalam melakukan pengaksesan baik dari pengguna manusia ataupun aplikasi.
3. DBMS menyediakan fitur-fitur yang dapat mempermudah dalam mengolah basis data.
4. DBMS memberikan kemudahan dalam permodelah sebuah data.
5. Banyak dukungan media penyimpanan untuk sebuah DBMS

2.2.1.2. Sistem Basis Data Manajemen Relasional

Sistem basis data manajemen relasional atau *relasional database manajement system* (RDBMS) adalah suatu isitilah yang digunakan untuk

menguraikan keseluruhan deretan program untuk mengelola sebuah basis data relasional dan komunikasi mesin basis data. [3]

Beberapa elemen yang terdapat pada basis data relasional antara lain adalah sebagai berikut : [3]

1. Tabel

   Merupakan kumpulan informasi secara login yang terkait dan diperlakukan sebagai unit.

2. Baris

   Suatu tabel terdiri adari baris (record). Suatu baris adalah kejadian tunggal yang berisi data di dalam tabel. Setiap baris diperlakukan sebagai unit tunggal.

3. Kolom

   Beris diorganisasikan sebagai sekumpulan kolom (field-field) yang ada pada tabel dalam suatu basis data.

4. Kunci

   Ada dua jenis kunci, yakni utama (*primary*) dan tamu (*foreign*). Kunci utama adalah suatu kolom (atau kelompok kolom) dimana nilai unik dihunakan untun mengidentifikasi setiap baris di dalam tabel.

2.2.2. Sistem

Sistem merupakan sekumpulan komponen benda nyata atau abstrak yang mempunyai kriteria masing-masing, berkumpul dalam suatu lingkungan atau ruang lingkup yang sama dan saling berkaitan agar dapat mencapai tujuan yang sama. Seperti contoh, dalam suatu perlombaan cerdas cermat ada beberapa kelompok A, B dan C. Kelompok merupakan sebuah sistem, dan anggota kelompok merupakan satu kesatuan komponen yang memiliki masing-masing kriteria, tujuannya mereka sama adalah untuk menjawab pertanyaan agar mendapatkan juara dalam perlombaan cerdas cermat itu.

Beberapa karakteristik dalam sebuah sistem diantaranya adalah sebagai berikut :

1. Komponen

Suatu sistem mepunyai komponen atau subsistem, dimana dalam masing-masing subsistem memiliki fungsi yang berbeda-beda. Walaupun berbeda-beda tetapi antara subsistem satu dengan yang lainnya saling ketergantungan dan saling mempengaruhi satu sama lain.

2. Batasan sistem

   Batasan sistem merupakan ruang lingkup dalam sebuah sistem. Hal ini menunjukan bahwa sebuah sistem haruslah memiki tujuan yang terarah.

3. Lingkungan luar sistem

   Lingkungan luar sistem merupakan karakteristik yang memberikan keuntungan ataupun merugikan. Lingkungan luar sistem yang memeberikan keuntungan dan harus dijaga. Pada lingkungan luar sistem yang bersifat merugikan haruslah ditahan, jika tidak dijaga makan akan menghambat kelangsungan sebuah sistem.

4. Penghubung sistem

   Penghubung sistem berupa subsistem yang saling berhubungan satu sama lain agar sebuah sistem dapat berjalan sebagaimana mestinya, sehingga tujuan yang berada pada sistem dapat tercapai.

2.2.3. Informasi

Informasi merupakan sekumpulan data yang bersifat pengetahuan lalu diolah untuk menghasilkan pengetahuan baru atau informasi yang dapat berguna untuk masa mendatang. Selain itu juga, informasi yang dihasil haruslah memiliki kualitas yang dapat dipercaya kebenarannya. Berukut ini merupakan karakteristik informasi yang memiliki kualitas :

1. Relevan

   Informasi yang disajikan harusnya bersifat keputusan yang nantinya akan diambil oleh pengguna informasi. Sebagai contoh, seorang nsabah yang mengajukan pembuatan kartu kredit kepada pihak bank tidak akan langsung disetujui perlu peninjauan terhadap riwayat nasabah terkait kredir. Apakah nasabah tersebut pernah memiki masalah atau tidak dalam ruang lingkup pengkreditan.

2. Akurat

   Kecocokan informasi dengan kejdiannya nyata haruslah tepat. Sebagai contoh, satu informasi menyatakan bahwa seorang nasabah tidak pernah mempunyai kasus dalam kredit macet pada suatu bank, akan tetapi setelah dilakukan peninjauan terhadap pihak bank terkait ternyata nasabah tersebut pernah memiliki kasus kredit macet.

3. Lengkap

   Sebuah informasi haruslah jelas dan saling berhubungan agar informasi yang dihasilkan berurut, dari awal sampai akhir. Seperti contoh, seorang siswa menunggu angkutan umum di sebuah halte, lalu siswa tersebut tiba di sekolah pukul 07.00 pagi. Pada penggal informasi tersebut tidak dijelaskan siswa tersebut menaiki angkutan apa.

4. Tepat waktu

   Informasi yang dihasilkan haruslah yang terbaru agar dapat digunakan sebagai bahan pengambil keputusan. Seperti contoh, seorang pegawai harus masuk kerja pukul 8.00 dan pegawai tersebut masuk pukul 08.30. Pemberitahuan jadwal masuk kerja diberika pada saat pegawai tersebut sampai kantor pukul 08.30 hal tersebut membuat informasi tidak berguna.

5. Dapat dipahami

   Cara penyampaian sebuah informasi haruslah jelas baik dari urutan dan tata bahasa yang dipakai. Sehingga dapat dibaca dan difahami oleh siapapun.

6. Dapat dibandingkan

   Sebuah informasi yang dapat memungkinkan pengguna dapat membanding dua buah objek yang isinya sama dengan dengan objek lain.

2.2.4. Sistem Informasi

Sistem informasi merupakan seuatu sistem dalam suatu organisasi yang mempertemukan kebutuhan pengolahan pengolahan transaksi sehari-hari, dapat mendukung fungsi operasi yang bersifat manajerial dengan kegiatan strategi dari

suatu organisai sehingga dapat menyediakan laporan-laporan yang diperlukan pihak lain. Beberapa komponen pada sebuah sistem informasi, diantaranya adalah:

1. Komponen input

   Merupakan data yang masuk kedalam sistem informasi, media yang bersifat untuk memasukan data dari sebuah dokumen ataupun sumber informasi lain.

2. Komponen model

   Komponen model merupakan gabudang dari prosedur, logika dan model matematik yang bertujuan untuk memanipulasi data yang masuk ataupun data yang tersimpam dalam basis data agar dapat dengan mudah mengatur keluaran sesuai dengan yang diinginkan.

3. Komponen output

   Merupakan keluaran sistem informasi yang berupa informasi berkualitas dan sebuah dokumentasi yang dapat berguna untuk semua pemakai informasi.

4. Komponen teknologi

   Teknologi merupakan alat dalam sebuah sistem informasi, dimana teknologi ini digunakan untuk menerima input, menjalankan model, menyimpan dan mengakses data yang bertujuan untuk mengeluarkan sebuah informasi yang berkualitas, serta membantu dalam pengendalian dari sistem secara keluruhan.

5. Komponen hardware

   Sebuah komponen yang sangat penting dalam pembangunan sistem informasi, karena komponen ini yang menampung basis data atau lebih mudak dikatan sebagai sumber infomasi agar mempermudah kerja dari sistem informasi.

6. Komponen software

   Software berfungsi sebagai sarana untuk mengolah, menghitung dan memanipulasi data yang diambil dari harware. Tujuannya untuk menghasilkan sebuah informasi.

7. Komponen basis data

   Basis data merupakan kumpulan data yang salin berkaitan antara data satu dengan yang lainnya. Sebuah data perlu disimpan dalam basis data untuk keperluan penyediaan informasi yang lebih lanjut.

8. Komponen kontrol

   Perencanaan yang harus dilakukan ketika akan membuat sebuah sistem informasi. Hal ini dilakukan agar kejadian yang diluar dugaan dapat diatasi denga cepat ketika kejadian yang kita tidak inginkan terjadi pada sistem yang kita bangun. Seperti bencana alam, ancaman peretas, *malware* serta anacaman lain yang dapat menggangu data yang berisi informasi-informasi yang diperlukan.

Sistem informasi dibuat untuk mendukung kepada pemakai agar mendapatkan informasi sesuai dengan apa yang diinginkan. Maka, dari kebutuhan pemakai itulah sistem informasi dibagi menjadi beberapa bagian, antara lain : [6]

1. Sistem Pemrosesan Transaksi (*Transaction Processing System* atau TPS)
2. Sistem Informasi Manajemen (*Management Information System* atau MIS)
3. Sistem Otomasi Perkantoran (*Office Automation System* atau OAS)
4. Sistem Pendukung Keputusan (*Decision Support System* atau DSS)
5. Sistem Informasi Eksekutif (*Executive Information System* atau EIS)
6. Sistem Pendukung Kelompok (*Group Support System* atau GSS)
7. Sistem Pendukung Cerdas (*Intelegent Support System* atau ISS)

2.2.5. Sistem Pendukung Keputusan

SPK merupaka sistem informasi interaktif yang menyediakan informasi, pemodelan, dan manipulasi data. Sistem ini digunakan ketika menghadapi permasalahan yang semiterstruktur dan permasalahan yang tidak terstruktur disaat tidak seorangpun yang dapat memberikan solusi seperti apa keputusan yang harus diambil. SPK biasanya dibangun untuk mendapatkan solusi dari sebuah permsalahan atau untuk mengevaluasi suatu peluang. Aplikasi SPK dibangung menggunakan *Computer Based Information System* (CBIS) yang fleksibel, interaktif dan dapat beradaptasi dengan mudah.

SPK lebih ditunjukan untuk mendukung manajemen dalam melakukan pekerjaan yang sifatnya analitis dalam kondisi yang kurang terstruktur dan memiliki kriteria yang tidak jelas.

2.2.5.1. Tujuan Sistem Pendukung Keputusan

Sama dengan sistem lainnya yang mempunyai tujuan dalam pembangunaannya, SPK sendiri mempunya tujuan tersendiri diantaranya :

1. Membantu seorang manajer dalam mengambil sebuah keputusan yang berasal dari permasalahan yang semiterstruktur.
2. Memberikan pilihan keputusan untuk seorang manajer, bukan untuk menggantikan fungsi seorang manajer.
3. Meningkatkan efektivitas dalam menentukan sebuah keputusan agar lebih efiesien dalam pengambilan sebuah keputusan
4. Kecepatan komputasi memungkinkan para pengambil keputusan untuk melakukan banyak komputasi secara cepat dan mudah dengan biaya yang lebih murah.
5. Meningkatkan produktivitas suatu kelompok dalam mengambil sebuah keputusan, terutama para pakar pengambil keputusan yang mengharuskan untuk membayar mereka dengan mahal.
6. Dukungan kualitas komputer dapat membantu meningkatkan kualitas keputusan yang dibuat.
7. Berdaya saing menyebabkan tugas pengambilan keputusan menjadi sulit.
8. Mengatasi keterbatasan dalam menyimpan data, karena setiap manusia memiliki kemampuan daya ingat yang berbeda-beda.

Pengembangan sebuah SPK dalam teknologi sangat pesat. Maka dari itu sebuah teknologi SPK sibagi menjadi 3, yaitu :

1. SPK spesifik

   SPK mempunyai fungsi untuk memecahkan suatu masalah dengan karakteristik tertentu.
2. Pembangkit SPK

   Sebuah software yang khusu digunakan dalam pengembangan SPK sehinggal pengembang SPK dapat dengan mudah sebuah sistem SPK yang lebih spesifik
3. Perlengkapan SPK

Software dan harfware yang digunakan dalam pembangunan sebuah SPK spesifik atau pembangkin SPK

2.2.5.2. Arsitektur Sistem Pendukung Keputusan

Ada beberapa komponen atau subsistem yang terdapat dalam arsitektur pembangunan sebuah SPK, agar informasi yang dihasilkan dapat sesuai dengan harapan. Berikut adalah komponen dalam pembangunan sebuah SPK :

1. *Data Management*

   Subsistem ini berfungsi untuk memasukan suatu *database* yang berisi data yang relevan dan data diolah oleh sebuah perangkat lunak yang biasa disebut DBMS (*Databse management system*).

2. *Model Management*

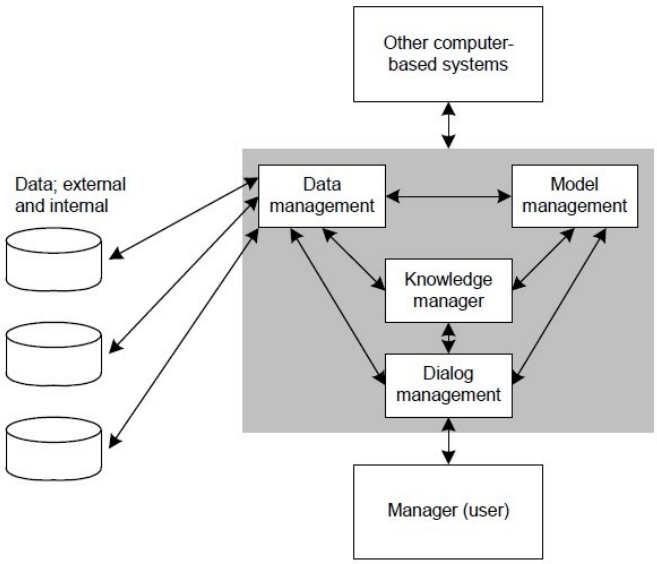
   Subsistem ini berfungsi sebagai model yang kuantitatif yang memberikan kapabilitas analitik juga sebagai manajemen perangkat yang lebih tepat.

3. *Dialog Management*

   Subsistem ini berfungsi untuk melakukan interaksi antara sistem dan pengguna. Sebuah keputusan tidak terlepas dari konstribusi dari pengguna yang secara insentif berinteraksi dengan komputer dan pembuat keputusan.

4. *Knowledge Management*

   Subsistem ini merupakan sistem yang mendukung dari semua subsistem yang ada, memiliki intelegensi untuk membantu dalam penentuan sebuah keputusan.



Gambar 2.3 Model Konseptual SPK [4]

Keputusan yang diambil untuk menyelesaikan suatu masalah dilihat dari keterstrukturnya yang bisa dibagi menjadi 3 bagian :

1. Keputusan terstruktur (*structured decision*)
   Merupakan keputusan yang dilakukan secara berulang-ulang dan bersifat rutin seperti keputusan penagihan piutang.
2. Keputusan semiterstruktur (*semistructured decision*)
   Merupakan keputusan yang memiliki dua sifat yang dilakukan oleh manajer tingkat menengah seperti pengevaluasian kredit.s
3. Keputusan tak terstruktur (*unstrukctured decision*)
   Merupakan keputusan yang penanganannya rutin karena tidak terjadi berulang-ulang atau tidak selalu terjad seperti keputusan untuk pengembangan teknologi baru.

2.2.5.3. Metode Sistem Pendukung Keputusan

Beberapa metode yang dapat digunakan untuk membuat sebuah sesitem penduku keputusan yang dapat kita gunakan untuk mempermudah mendapat sebuah keputusan dari sebuah masalah yang dihadapai. Metode juga digunakan tergantung kebutuhan atau jenis masalah yang dihadapi, untuk lebih jelasnya dapat dilihat pada penejelasan dibawah ini :

1. Profile Matching

Merupakan suatu proses yang sangat penting dalam manahemen SDM dimana terlebih dahulu mendefinisikan kriteria yang harus atau kemampuan yang diperlukan oleh suatu jabatan. Kemampuan tersebut haruslah dimiliki oleh seorang calon yang akan menjabat, dengan kata lain profile matching adalah metode yang mengedepannkan suatu kriteria yang akan dicapat terlebih dahulu sebelum masuk terhadap masalah dalam penentuan keputusan.

2. Metode B/C Ratio (Benefit Cost Ratio)

Metode ini digunakan untuk mengetahui besaran keuntungan atau kerugian serta kelayakan suatu proyek. Dalam perghitungannya analisis ini menghitungkan biaya serta manfaat yang akan diperoleh dari pelaksanaan suatu program. Biasanya metode ini digunakan untuk permasalah dalam ruang lingkup akuntansi.

3. Metode AHP

AHP merupakan suatu model pendukung keputusan yang dikembangkan oleh Thomas L. Saaty. Model pendukung keputusan ini akan menguraikan multi daktor atau multi kriteria yang kompleks menjadi suatu hirarki. Sebuah hirarki mendefinisikan sebuah permasalah yang kompleks dalam suatu struktur multi level dimana level pertama adalah tujuan, yang diikuti level faktor, kriteria, sub kriteria, dan seterusnya. Sehingga permasalahan akan tampak lebih terstruktur dan sistematis. Metide AHP sering digunakan untuk memecahkan suatu masalah yang kompleks dan semi terstrukstur.

4. Metode IRR

Metode ini untuk membuat peringkat usulan investasi dengan menggunakan tingkat pengembalian atas investasi yang dihitung dengan mencari tingkat diskonto yang menyamarkan nilai sekarang dari arus kas masuk proyek yang diharapkan terhadap nilai sekarang.

5. Metode NPV

Metode NPV merupakan selisih antara pengeluaran dan pemasukan yang telah didiskon dengan menggunakan social opportunity cost of capital sebagai diskon faktor, atau dengan kata lain merupakan arus kas yang diperkirakan pada masa yang akan datang. Dalam menghitung NPV diperlukan data tentang perkiraan biaya inversitasi, biaya operasi, dan pemeliharaan serta perkiraan manfaat atau keuntungan dari proyek yang direncanakan.

2.2.6. *Analytical Hierarchy Process* (AHP)

AHP merupakan suatu model pendukung keputusan yang dikembangkan oleh Thomas L. Saaty. Model pendukung keputusan ini akan menguraikan masalah multi faktor atau multi kriteria yang akan menguraikan masalah multi faktor atau multi kriteria yang kompleks menjadi suatu hierarki, menurut Saaty, hierarki didefinisikan sebagai suatu representasi dari sebuah permasalahan yang kompleks dalam suatu struktur multi level dimana level pertama adalah tujuan yang diikuti level faktor, kriteria, sub kriterian, dan seterusnya kebawah hingga level terakhi dari alternatif. Dengan hierarki, suatu masalah yang kompleks dapat diuraikan ke dalam kelompok-kelompoknya yang kemudian diatur menjadi suau bentuk hierarki sehingga permasalahan akan tampak lebih terstruktur dan sistematis.

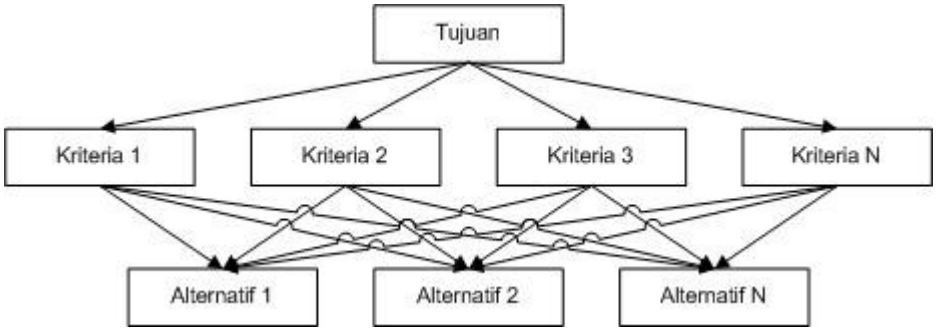
Komponen AHP adalah sebuah hierarki fungsional dengan input utamanya berupa prinsip manusia. AHP memiliki banyak keunggulan salah satunya dara dapat digambarkan secara grafis sehingga mudah difahami oleh semua pihak yang terlibat dalam sebuah pengambilan keputusan.

Beberapa prinsip ketika permasalahan SPK menggunakan metode AHP yang harus dipahami, diantaranya [6]:

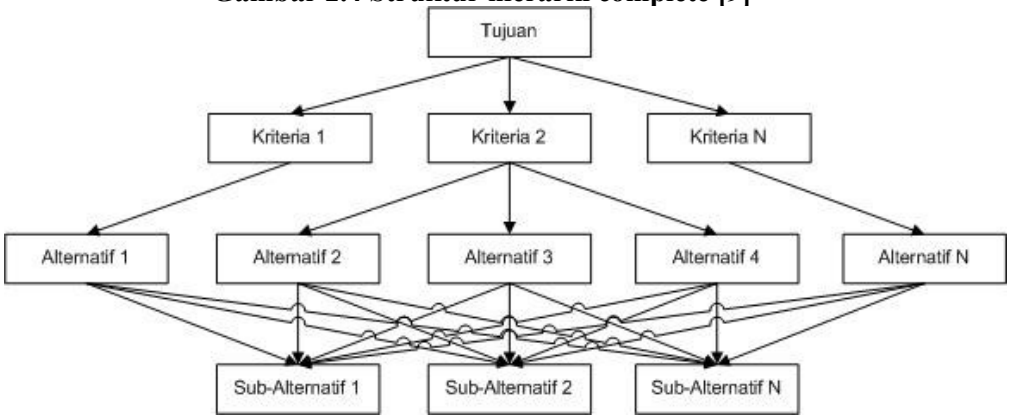
1. Membuat hierarki

   Sistem yang kompleks bisa difahami dengan memecahnya menjadi elemen-elemen pendukung, menyusun elemen secara hierarki, dan

menggabungkannya atau mensintesisnya. *Decomposition* adalah berfungsi untuk memecahkan atau membagi problema yang utuh menjadi unsur-unsur lalu menjadi bentuk hierarki proses pengambilan keputusan, dimana setiap unsur saling berhubungan satu dengan yang lainnya. Terbagi kedalam dua unsur hierarki keputusan yaitu sebagai *complete* dan *incomplete.* Terlihat pada gambar 2.4 dan gambar 2.5.



Gambar 2.4 Struktur hierarki complete [9]



Gambar 2.5 Struktur hierarki incomplete [9]

2. Penilaian kriteria dan alternatif

   Kriteria dan alternatif dilakukan dengan perbandingan berpasangan. Menurut Saaty (1998), untuk berbagai persoalan, skala 1 sampai 9 adalah skala terbaik untuk mendeskripsikan pendapat. Terlihat pada tabel 2.1. [6]

Tabel 2.1 Skala nilai perbandingan pasangan

Intensitas Kepentingan	Keterangan
1	Kedua elemen sama pentingnya

3	Elemen yang satu sedikit lebih penting daripada elemen yang lainnya
5	Elemen yang satu lebih penting daripada elemen lainnya
7	Satu elemen jelas lebih mutlak penting daripada elemen lainnya
9	Sudut elemen mutlak penting daripada elemen lainnya
2,4,6,8	Nilai-nilai antara dua nilai pertimbangan yang berdekatan
Kebalikan	Jika aktivitas i mendapat satu angka dibandingkan dengan aktivitas j, maka i memiliki nilai kebalikannya dibandingkan dengan i

3. *Systhesis of priority* (Menentukan prioritas)

   Setiap kriteria dan alternatif perludilakukan perbandingan berpasangan (*pairwise comprasions*). Nilai-nilai dari semua perbandingan dapat disesuaikan dengan *judgement* yang telah ditentuk untuk menghasilkan sebuah bobot dan prioritas. Bobot dan prioritas dilakukan perhitungan dengan melakukan manipulasi matriks atau bisa juga dengan melakukan persamaan matematika.

4. *Logical Consistency* (Konsistensi Logis)

   Konsistensi memiliki dua makna. Pertama, objek-objek yang sama/serupa dapat dikelompokan berdasarkan keseragaman dan relevansi. Kedua, menyangkut tingkat hubungan antar objek yang didasarkan pada kriteria tertentu.

AHP memiliki prosedur dalam memproses data untuk mendapatkan keputusan yang sesuai dengan hierarki yang telah ditentukan. Prosedur yang ada pada AHP adalah sebagai berikut : [6]

1. Mendefinisikan masalah dan menentukan solusi yang diinginkan. Setelah itu baru menyusun sebuah hierarki.
2. Menentukan prioritas elemen sebagai langkah pertama. Merupakan sebagai perbandingan pasangan yang sesuai dengan kriteria yang diberikan. Matriks perbandingan diisi menggunakan bilangan agar dapat merepresentasikan kepentingan relatif dari satu elemen dengan elemen lainnya.
3. Membuat matriks perbandingan sesuai dengan kriterian maupun subkriteria, lalu membuat rata-rata dari setiap kriteria ataupun subkriteria:

a. Menentukan matriks perbandingan berpasangan

Berikut ini adalah gambaran matrik perbandingan berpasangan dalam bentuk tabel yang dapat dilihat pada tabel 2.2.

Tabel 2.2 Matriks perbandingan berpasangan

	K1	K2	..	Kn
K1	Skala penilaian perbandingan	K1/K2	..	..
K2	..	..	..	..
..	..	..	..	..
Kn	..	..	..	..
Jumlah				

Dalam mengisi tabel 2.2 ada beberapa syarat yang harus dilakukan diantaranya adalah :

1) Setiap nilai pada kolom 2 dengan baris ke 2 sampa ke-n mengacu pada skala penilaian yang ada pada tabel 2.1. Penentukan nilai harus berdasarkan skala penilaian jika kriteria kolom dan baris sama maka memiliki nilai satu karena merupakan elemen yang sama-sama penting. Pada baris kedua dan seterusnya membandingkan apakah kolom 2 dengan baris 3 memiliki nilai yang sama atau tidak, jika tidak maka tentukan nilai skala yang harus di tentukan apakah sama pentih, penting atau yang lainnya yang sesuai dengan keterangan pada tabel 2.1 seperti contoh tabel dibawah ini :

Tabel 2.3 Penilaia kriteria beasiswa (1)

	K1	**K2**	**K3**
K1	1	..	..
K2	2	1	..
K3	.3	..	1
Jumlan	$\sum K1$	$\sum K2$	$\sum K3$

2) Meliha dari contoh tabel 2.3, maka pada Kolom 3 baris 2 diisi dengan nilai yang diambil dari kolom 2 baris 3, kolom 4 baris 2 diisi dengan nilai yang diambil dari kolom 2 baris 4, dan kolom 4 baris 3 diisi

dengan nilai yang diambil dari kolom 3 baris 3. Seperti contoh dibawah ini :

Tabel 2.4 Penilaian kriteria beasiswa (2)

	K1	K2	K3
K1	1	1/2	1/3
K2	2	1	1/2
K3	.3	..	1
Jumlah	$\sum K1$	$\sum K2$	$\sum K3$

3) Berdasarkan contoh tabel 2.3 dan tabel 2.4 maka pengisian nilai terakhir adalah untuk kolom 3 baris 4. Cara mengambil nilainya adalah sama dengan proses no 1 yaitu berdasarkan tabel 2.1 dengan membandingkan dari kolom yang memiliki kriteria yang sama. Dalam tabel 2.3 dan 2.4 kolom 3 dan kolom 4 memiliki kriteria yang sama yaitu K2 dan K2 makan skala nilai yang diambil adalah 1 dan untuk kolom 3 baris 4 dengan kriteria K2 dan K3 maka diambil skala nilai 2 karena urutan penilaian yang dilakukan pada kolom 2. Hal ini dilakukan untuk mempermudah dalam penentukan kriteria. Sebagai contoh hasil akhir dapat dilihat pada tabel 2.5.

Tabel 2.5 Penilaia kriteria beasiswa (3)

	K1	K2	K3
K1	1	1/2	1/3
K2	2	1	1/2
K3	3	2	1
Jumlah	$\sum K1$	$\sum K2$	$\sum K3$

4) Agar lebih mudah perhitungan, semua nilai dijadikan bentuk desimal, seperti di bawah ini :

Tabel 2.6 Penilaian kriteria beasiswa (4)

	K1	K2	K3
K1	1,00	0,50	0,33
K2	2,00	1,00	0,50
K3	3,00	2,00	1,00
Jumlah	6,00	3,50	1,83

b. Menentukan matriks nilai kriteria

Berikut ini adalah gambaran matriks nilai kriteria dalam bentuk tabel yang dapat dilihat pada tabel 2.7.

Tabel 2.7 Matriks Nilai Kriteria

	K1	K2	..	Kn	Jumlah	Prioritas
K1	Nilai baris kolom baru	..	..	..	$\sum K$	$\sum_{i=0}^{n} K$
K2	..	..	..	..	..	..
..	..	..	..	..	..	..
Kn	..	..	..	..	..	..

Dalam mengisi tabel 2.6 mengacu pada rumus sebagai berikut :

Nilai baris kolom baru = nilai baris kolom lama/ jumlah baris (2.1)

Keterangan :

Nilai baris kolom baru merupakan hasil dari pembagian nilai baris kolom lama dengan jumlah baris. Sebagai contoh, untuk nilai baris kolom lama dan jumlah baris diambil dari contoh kasus yang ada pada tabel 2.6. Supaya lebih jelas maka diterapkan pada contoh tabel 2.8 yang merupakan hasil pembagian nilai baris kolom lama dengan jumlah baris.

Tabel 2.8 Matriks nilai kriteria beasiswa

	K1	**K2**	**K3**	**Jumlah**	**Prioritas**
K1	0,16	0,14	0,18	0,48	0,16
K2	0,33	0,28	0,27	0,88	0,29
K3	0,50	0,57	0,54	1,61	0,53

1) Nilai 0,16 pada kolom 2 K1 dengan baris 2 K1 pada tabel 2.8 merupakan hasil pembagian dari nilai yang ada pada kolom 2 K1 baris 2 K2 dengan jumlah yang ada pada kolom 2 baris 5 pada tabel 2.6.
2) Niai 0,48 pada kolom 5 Jumlah dengan bari 2 K1 merupakan hasil penjumlahan dari kolom 2, 3 dan 4 pada baris 1 K1.
3) Nilai 0,16 pada kolom 6 dengan baris 2 K1 merupakan nilai rata-rata yang didapat dari hasil perhitungan kolom 5 jumlah dengan jumlah kriteri yaitu sebanyak 3 (K1, K2 dan K3).

c. Membuat matriks penjumlahan setiap baris

Berikut ini adalah gambaran matriks penjumlahan setiap baris dalam bentuk tabel yang dapat dilihat pada tabel 2.9.

Tabel 2.9 Matriks Penjumlahan Setiap Baris

	K1	K2	..	Kn	Jumlah
K1	(Nilai perbandingan x (prioritas per baris)	..	..	..	
K2	..	..	..	..	
..	..	..	..	..	
Kn	..	..	..	..	

Keterangan :

Dalam mengisi nilai pada setiap baris kolom yang ada pada tabel 2.9 mengacu pada rumus sebagai berikut :

Nilai baris kolom baru = Nilai perbandingan berpasangan x prioritas per baris (2.2)

Agar lebih jelas, berikut ini merupakan contoh tabel yang dapat dilihat pada tabel 2.10 dan data diambil dari contoh tabel 2.8. Tabel 2.10 merupaka hasil dari rumus (2.2).

Tabel 2.10 Matriks penjumlahan setiap baris untuk kriteria beaiswa

	K1	**K2**	**K3**	**Jumlah**
K1	0,16	0,08	0,02	0,26
K2	0,58	0,29	0,07	0,94
K3	1,59	1,06	0,53	3,18

Nilai 0,16 pada kolom 2 K1 baris 2 K2 merupakan hasil pengkalian dari nilai matriks perbandingan tabel 2.6 pada kolom 2 K1 baris 2 K1 dengan nilai prioritas perbaris tabel 2.8 pada kolom 6 Prioritas baris 2 K1.

Nilai 0,58 pada kolom 2 K1 baris 3 K2 merupakan hasil pengkalian dari nilai matriks perbandingan tabel 2.6 pada kolom 2 K1 baris 3 K2 dengan nilai prioritas perbaris pada tabel 2.8 pada kolom 6 Prioritas baris 3 K2.

Nilai 0,26 pada kolom 5 Jumlah baris 2 K1 merupakan hasil penjumlahan K1 + K2 + K3 pada baris 2.

d. Menghitung rasio konsistensi

Menghitung rasio konsistensi karena dalam tahapan ini kita dapat mengetahui seberapa baik konsistensi sebuah keputusan. Karena tidak mungkin menentukan sebuah keputusan dengan tingkat konsistensi yang rendah. Pada tabel 2.11 merupakan gambaran hasil perhitungan dari masing-masing kriteria yang akan diuji konsistensinya.

Tabel 2.11 Perhitungan Rasio Konsistensi

	Jumlah per baris	Prioritas	Hasil
K1	..	..	..
K2	..	..	..
..	..	..	..
Kn	..	..	..
Jumlah			

Sebelum menghitung sebuah konsistensi, lakukan terlebih dahulu tahapan sebagai berikut :

Hasil (pada Tabel 2.11) = Jumlah perbaris + prioritas (2.3)

Jumlah (pada Tabel 2.11) = K1+K2+K3+Kn (2.4)

λ maks = Jumlah (pada Tabel 2.11) / jumlah kriteria (2.5)

λ maks adalah hasil pembagian Jumlah (Tabel 2.11) dengan jumlah kriteria

Hitung *Consistency Index* (CI) dengan rumus :

CI = (λ maks-n)/n (2.6)

Dimana n = banyaknya elemen

Hitung rasio konsistensi/*Consistency Ratio* (CR) dengan rumus :

CR = CI/IR (2.7)

Dimana CR = *Consistency Ratio*

CI = *Consistency Index*

IR = *Index Random Concsitency*

Menentukan konsistensi sebuah keputusan. Jika nilai lebih dari 10% maka penilaian *judgement* (skala penilaian) harus diperbaiki, jika sebuah rasiso konsistensi (CI/IR) kurang atau sama dengan 0,1 maka dinyatakan benar.

Tabel 2.12 Daftar Indeks Random Konsistensi [6]

Ukuran Matriks	Nilai IR
1,2	0,00
3	0,58
4	0,90
5	1,12
6	1,24
7	1,32
8	1,41
9	1,45
10	1,49
11	1,51
12	1,48
13	1,56
14	1,57
15	1,59

Supaya lebih jelas dalam pemahasan proses akhir, maka dapat diterapkan pada contoh tabel 2.13 yang merupakan hasil dari perhitngan rumus (2.3) untuk kolom hasil dan (2.3) untuk baris 5 jumlah kolom 4 Hasil. Data sebelumnya di ambil dari hasil perhitungan tabel 2.8 dan tabel 2.10.

Tabel 2.13 Perhitungan rasio konsistensi kriteria beasiswa

	Jumlah per baris	**Prioritas**	**Hasil**
K1	0,26	0,16	0,42
K2	0,94	0,29	1,23
K3	3,18	0,53	3,71
Jumlah			4,82

Nilai 0,26 pada kolom 2 Jumlah per baris dengan baris 2 K1 diambil dari tabel 2.10 pada kolom 4 Jumlah baris 2 K1.

Nilai 0,16 pada kolom 3 Prioritas baris 2 K1 diambil dari tabel 2.8 pada kolom 6 prioritas baris 2 K1.

Mencari nilai λ maks menggunakan rumus (2.6)

λ maks = 4,82 / 3

= 1,60

Hitung *Consistency Index* (CI) dengan rumus menggunakan rumus (2.6) :

CI = ((1,60 – 3) / 3)

= - 0,46

Hitung rasio konsistensi/*Consistency Ratio* (CR) dengan rumus (2.7) :

CR = - 0,46 / 0,58

= - 0,79

Pada rumus CR terdapat RI yang merupakan *Index Random Consistency*. Dalam menentukan niai RI dapat dilihat seberapa banyak kriteria yang terdapat dalam suatu masalah atau seberapa banyak sub kriteria dalam suatu kriteria. Karena pada kasus diatas memiliki 3 kriteria maka RI yang diambil adalah 0,58 yang bisa dilihat pada tabel 2.12.

Melihat dari hasil CR adalah – 0,79 maka dapat diambil kesimpulan bahwa konsistensi yang didapat sangat baik karena memenuhi syarat yaitu kurang atau sama dengan 0,1. Kolom hasil yang ada pada tabel 2.13 dapat diambil sebagai nilai pertibangan dalam menentukan keputusan.

2.2.7. Pemrograman Berorientasi Objek

Pemrograman Berorientasi Objek adalah pemrograman yang memiliki teknik dengan cara melihat objek-objek dari suatu permasalahan dan Sistem,

sebagai contoh jika dibuat program tentang hewan, *programmer* harus berfikir apa nama hewan, bagaimana hewan berkembang, bagaimana hewan bergerak dan seterusnya. Jika dibuat program tentang sebuah benda, maka *programmer* harus berfikir tentang data yang berhubungan dengan benda tersebut dan apa yang dapat dikerjakan oleh maupun terhadap benda tersebut. Konsep dasar yang harus difahami tentang pemrograman berorientasi objek, diantaranya adalah :

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 (metode/operasi), hubungan (*relationship*) dan arti. Suatu kelas dapat diturunkan dan kelas yang lain, dimana atribut dan kelas semula dapat diwariskan ke kelas yang baru. Secara tekniks 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, satua organisasi, tempat, kejadian, struktur, status, atau hal-hal lain yang bersifat abstra. 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.

3. Metode (*Method*)

   Method pada sebuah kelas hampir sama dengan fungsi atau prosedur pada metodologi struktural. Sebuah kelas boleh memiliki lebih dari satu metode atau operasi.

4. Atribut (*Attribute*)

   Atribut dari sebuah kelas adalah variabel global yang dimiliki sebuah kelas. Atribut dapat beruoa 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. Abstrak (*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*)

   enkapsulasi merupakan dasar pertama bagi OOP, enkapsulasi berarti menyembunyikan (membungkus) detail *class* dari *object* yang mengirimkan pesam kepadanya. *Class* adalah sebuah tipe data yang *user-defined,* berisi karakteristik abstrak dari sebuah *object* (benda), seperti atribut, properti, serta aksi yang bisa dilakukan oleh benda tersebut, atau sering dikenal dengan istilah *method*. *Object* adalah *instance* dari *class*, atau dengan kata lain *object* adalah implementasi dari *class* dengan atau parameter yang lebih konkrit.

7. Pewarisan (*Inheritance*)

Pewarisan adalah sebuah usaha untuk menurunkan *class* turunan (*sub class*) dari *class* induk yang memiliki sifat – sifat khusu, namun memiliki pula sifat – sifat yang dimiliki oleh *class* induk. *Sub class* ini biasanya memiliki karakter yang lebih khusus bila dibandingkan dengan *class* induk. Selain *single inheritance*, dikenal pula *multiple inheritance. Multiple inheritance* adalah sebuah usaha *inheritance* dengan *class* induk lebih dari satu, antara *class* induk yang satu dengan yang lain tidak terdapat hubungan *inheritance.*

8. Polimorpisme (*Polimorphisme*)

   Polimorpisme memungkinkan *object* memiliki karakter yang berbeda – beda saat program dijalankan, tergantung dari *class* induk yang dirujuk oleh *object*. Dengan *polimorfisme* ini, *object* mampu mengeksekusi *method* yang berbeda – beda dan menghasilkan keluaran yang berbeda – beda pula meskipun menggunakan nama *Object* yang sama.

9. 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.

10. *Reusability*

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

11. 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.

12. Komunikasi Antar objek

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

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.

2.2.8. *Design Pattern*

Merupakan bagian dari teknologi pemrograman berorientasi objek. Pola desain ini merupakan solusi yang sudah teruji dalam menyelesaikan permasalahan pemrograman. Cara membuat sebuah pola dasain komponen yang baik adalah dengan menperhatikan kemungkinan perubahan sistem yang mungkin terjadi di masa mendatang. Pola desain sebaiknya dipelajari ketika sudah memahami konsep pemrograman berorientasi objek karena pola desain memiliki sifat *reusable* (dapat digunakan kembali berulang-ulang) dan *framework* merupakan rancangan arsiterktur perangkat lunak yang digunakan sebagai standar pembuatan program dan memudahkan pengguna *framework*.

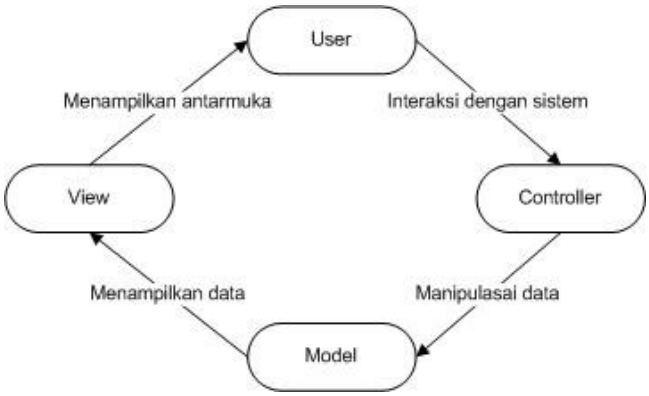
Framework merupakan rangkaian kerja yang memudahkan programmer untuk membuat sebuah aplikasi dimana dalam sebuah *framework* terdapat modul-modul yang biasanya dipergunakan oleh seoranga programmer dalam membuat sebuah aplikasi. *Model View Contreoller* (MVC) merupakan salah satu contoh dari *Architectural Design Pattern* dan sekarang banyak *framework* yang didedikasikan pemrograman berorientasi objek. Contoh *framework* yang sering digunakan antara lain :

a. PHP : CodeIgniter, Smarty, CakePHP, Zend, Symfony, Yii, dll
b. Java : Spring, Hibernate, Struts, dll
c. Ruby : Ruby on Rails, dll
d. Phyton : Django, dll
e. .NET : ASP.Net MVC framework

2.2.9. *Model View Controller* (MVC)

MVC merupakan teknik dalam pembuatan sebuah aplikasi yang populer saat ini. Biasanya konsep MVC diterapkan pada sebuah *framework* yang tujuannya agar seorang programmer dapat dengan memudah memperbaharui

aplikasi yang telah dibangun, dan juga dapat memudahkan programmer lain ketika memodifikasi sebuah aplikasi. Ada 3 komponen inti dalam konsep ini yang merupakan nama dari teknik MVC tersebut, diantaranya :



Gambar 2.6 Skema MVC

1). *Model*, berfungsi sebagai metode untuk berinteraksi langsung dengan basis data.
2). *View,* berfungsi untuk memanajemen sebuah antar muka pada sebuah aplikasi.
3). *Controller,* berfungsi sebagai media komunikasi antara *view* dan *model.*

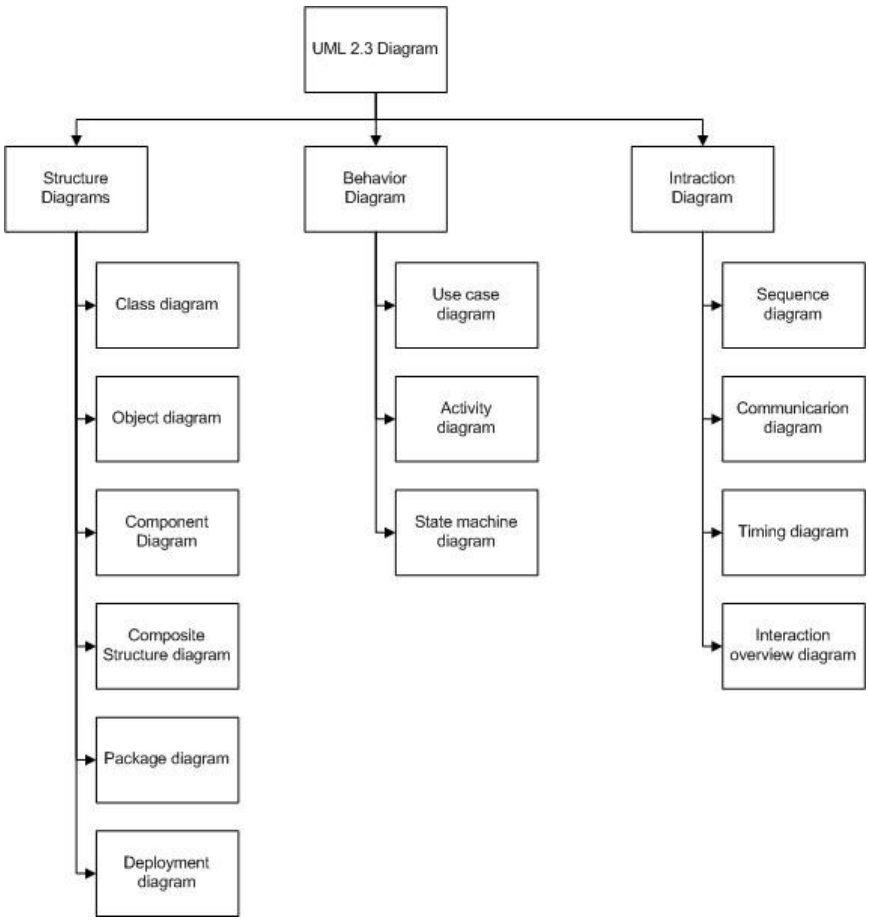
2.2.10. *Unified Modeling Language* (UML)

Unified Modeling Language (UML) adalah sebuah bahasa yang berdasarkan grafik/gambar untuk memvisualisasi, menspesifikasikan, membangun, dan pendokumentasian dari sebua sistem pengembangan software berbasis OO (*Object-Oriented*). UML sendiri juga memberikan standar penulisan sebuah sistem blue print, yang meliputi konsep bisnis proses, penulisan kelas – kelas dalam bahasa pemrograman yang spesifik, skema *database*, dan komponen – komponen yang diperlukan dalam sistem *software*.

UML menyediakan beberapa notasi dan *artifact* standar yang bisa digunakan sebagai alat komunikasi bagi para pelaku dalam proses analisis dan desain *Artifact* didalam UML didefinisikan sebagai informasi dalam bentuk yang

digunakan atau dihasilkan dalam proses pengembangan perangkat. Contohnya adalah *source code* yang dihasilkan oleh proses pemrograman.

Yang harus diperhatikan untuk menjaga konsistensi antar *artifact* selama proses analisis dan desain adalah bahwa setiap perubahan yang terjadi pada satu *artifact* harus juga dilakukan pada *artifact* sebelumnya.



Gambar 2.7 Struktur diagram UML [1]

Penjelasan dari 3 kategori diagram yang ada pada UML 2.3 dijelaskan sebagai berikut :

1. *Structure Diagram*

   Digunakan untuk menggambarkan suatu struktur statis dari sistem yang dimodelkan.

2. *Behavior Diagram*

Merupakan kumpulan diagram yang digunakan untuk menggambarkan aktifitas atau rangkaian perbuhanan kegiatan pada sebuah sistem

3. *Intraction Diagram*

   Merupakan kumpulan diagram yang digunakan untuk menggambarkan interaksi sistem dengan sistem lain maupun interaksi antar subsistem pada suatu sistem.

Untuk membuat suatu model, UML memiliki diagram grafis yang dapat menggambarkan rancangan sistem dan proses pada sebuah sistem. Beberpa diagram yang gunakan adalah sebagai berikut:

1. *Use Case Diagaram*

   Use case atau diagram *use case* merupakan pemodelan untuk kelakuan (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Secara kasar, *use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi itu. Nama pada *use case* haruslah dibuat sederhana dan harus mudah dimengerti.

2. *Class Diagram*

   Menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Sebuah kelas memiliki atribut dan metode atau operasi.

3. *Activity Diagram*

   Merupakan rangkaian kerja pada sebuah sistem bukan menggambarkan rangkaian kerja seorang aktor atau pengguna pada sebuah sistem yang dijalankan.

4. *Sequence Diagram*

   Merupakan gambaran aktifitas objek pada *use case* dengan mendeskripsikan waktu proses. Untuk menggambar *sequence diagram* maka harus diketahui terlebih dahulu objek-obejk yang terlibat dalam sebuah *use case* beserta metode-metode yang dimiliki oleh sebuah kelas.

Diagram – diagram tersebut dibuat agar model yang dibuat semakin mendekati realitas. Berikut merupakan penjelasan tetang fungsi pada komponen yang berada di UML, diantaranya adalah sebagai berikut :

1. Actor

Actor menggambarkan segala pengguna *software* aplikasi (*user*). *Actor* memberikan suatu gambaran jelas tentang apa yang harus dikerjakan *software* aplikasi. Sebagai contoh sebuah *actor* dapat memberikan input kedalam dan menerima informasi dan *software* aplikasi, perlu dicatat bahwa sebuah *actor* berinteraksi dengan *use case,* tetapi tidak memiliki kontrol atas *use case*. Sebuah *actor* mungkin seorang manusia, satu *device, hardware* atau sistem informasi lainnya.

2. Use Case

Use case menjelaskan urutan kegiatan yang dilakukan *actor* dan sistem untuk mencapai suatu tujuan tertentu. Walaupun menjelaskan kegiatan, namum *use case* hanya menjelaskan apa yang dilakukan oleh *actor* dan sistem bukan bagaimana *actor* dan sistem melakukan kegiatan tersebut.

a. *Use-case* Konkret adalah *use case* yang dibuat langsung karena keperluan *actor*. *Actor* dapat melihat dan berinisiatif terhadapnya.
b. *Use-case* Abstrak adalah *use case* yang tidak pernah berdiri sendiri. *Use case* abstrak senantiasa termasuk didalam (*include*), diperluas dari (*extend*) atau memperumum (*generalize*) *use case* lainnya.

Untuk menggambarkannya dalam *use case* model biasanya digunakan *association relationship* yang memiliki *stereotype include*, *extend* atau *generalization relationship*. Hubungan dari *use case* lainnya. Hubungan *extend* antar *use case* yang lain jika kondisi atau syarat tertentu terpenuhi.

3. Class

Class merupakan pembentuk utama dari sistem berorientasi obyek, karena *class* menunjukkan kumpulan *objek* yang memiliki atribut dan operasi yang sama. *Class* digunakan untuk mengimplementasikan *interface*.

Class digunakan untuk mengabstraksikan elemen – elemen dari sistem yang sedang dibangun. Class bisa merepresentasikan baik perangkat lunak maupun perangkat keras, baik konsep maupun benda nyata.

Notasi *class* berbentuk persegi panjang berisi 3 bagian persegi panjang paling atas untuk nama *class*, persegi panjang paling bawah untuk operasi, dan persegi panjang ditengah untuk atribut.

Atribut digunakan untuk menyimpan informasi. Nama atribut menggunakan kata benda yang bisa dengan jelas merepresentasikan informasi yang tersimpan didalamnya. Operasi menunjukan sesuatu yang bisa dilakukan oleh obyek dan menggunakan kata kerja.

4. Inteface

Inteface merupakan kumpulan operasi tanpa implementasi dari suatu *class*. Implementasi operasi dalam *interface* dijabarkan oleh operasi didalam *class*. Oleh karena itu keberadaan *interface* selalu disertai oleh *class* yang mengimplementasikan operasinya. *Interface* ini merupakan salah satu cara mewujudkan prinsip enkapsulasi dalam obyek.

5. Interaction

Interaction digunakan untuk menunjukkan baik aliran pesan atau informasi antar obyek maupun huungan antar obyek. Biasanya *interaction* ini dilengkapin juga dengan teks bernama operation *singature* yang tersusun dari mana operasi, parameter yang dikirim dan tipe parameter yang dikembalikan.

6. Note

Note digunakan untuk memberikan keterangan atau komentar tambahan dari suatu elemet sehingga bisa langusng terlampir dalam model. *Note* ini bisa disertakan ke semua elemen notasi yang lain.

7. Dependency

Dependency merupakan relasi yang menunjukan bahwa perubahan pada salah satu elemen memberi pengaruh pada elemen lain. Elemen yang ada di bagian tanda panah adalah yang tergantung pada elemen yang ada dibagian tanpa tanda panah.

Terdapat 2 *stereotype* dar *dependency,* yaitu *include* dan *extend. Include* menunjukkan bahwa suatu bagian dari elemen (yang ada digaris tanpa panah) memicu eksekusi bagain dari elemen lain (yang ada di garis dengan panah). *Extend* menunjukkan bahwa suatu bagian dari elemen di garis tanpa panah bisa disiapkan kedalam elemen yang ada di garis dengan panah.

8. Association

Association menggambarkan navigasi antar *class* (*navigation*), berapa banyak obyek lain yang bisa berhubungan dengan satu obyek (*multiplicity* antar *class*) dan apakan suatu class menjadi bagian dari class lainnya (*aggregation*).

9. Navigation

Navigation dilambangkan dengan penambahan tanda panah di akhir garis. *Bidirectional navigiation* menunjukkan bahwa dengan mengetahui salah satu *class* bisa didapatkan informasi dari *class* lainnya. Sementara *Uni Directional navigation* hanya dengan mengetahui *class* diujung garis *association* tanpa panah kita bisa mendapakan informasi dari *class* di ujung dengan panah, tetapi tidak sebaliknya.

10. *Aggregation*

Aggregation mengacu pada hubungan "*has-a*", yaitu bahwa suatu *class* memiliki *class* lain, misalnya Rumah memiliki *class* Kamar.

11. Generalization

Generalization menunjukkan hubungan antara elemen yang lebih umum ke elemen yang lebih spesifik. Dengan *generalization, class* yang lebih spesifik (*subclass*) akan menurunkan atribut dan operasi dari *class* yang lebih umum (*superclass*) atau "*subclass is superclass*". Dengan menggunakan notasi *generalization* ini, konsep *inheritance* dari prinsip hierarki dapat dimodelkan.

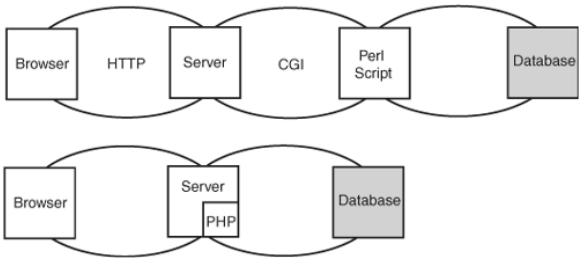
12. *Realization*

Realization menunjukan hubungan bahwa elemen yang ada dibagian tanpa pernah akan merealisasikan apa yang dinyatakan oleh elemen yang ada di bagian dengan panah. Misalnya *class* merealisasikan *package*, *component* merealisasikan *class* atau *interface*.

2.2.11. *Hypertext Preprocessor* (PHP)

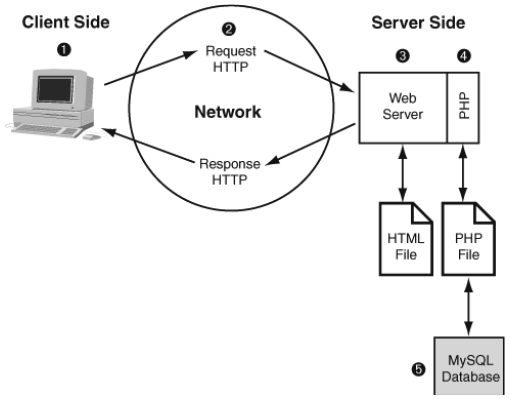
PHP adalah bahasa pemrograman untuk mebuah *website* dinamis dan *website* interaktif. Pada sebuat auturan umum, pemrograman PHP berjalan di atas sebuah *Web Server*, dan halaman sebuah *web* dapat ditampilkan ketika pengunjung *webiste* meminta. Salah satu fitur dari PHP adalah kamu dapat menyisipkan kode PHP dalam hlaman *web* HTML yang dapat mempermudah para developer untuk membuat sebuah konten yang dinamik secara cepat.

Apa yang dimaksud dengan kata "dinamis, dan halaman *web* interaktif" ? sebuah *website* dinamis adalah sebuah halam *website* dapat berubah sesuai secara otomatis setiap kali halaman ditampilkan. Bahkan dengan halaman statis sekalipun, seperti halaman yang berisi tag HTML dan dapat ditampilkan dengan halaman yang sama sampai ada proses pengeditan. Dan yang dimaksud dengan *website* interaktif adalah sebuah *website* yang merespon inputan dari pengunjung sebuah *website*, seperti *website* yang menyediakan pengisian formulir yang dapat diisi pengunjung.



Gambar 2.8 Proses pembuatan website dinamis [8]

PHP adalah sebuah *server-side scripting language*, yang artinya sebuah kode PHP dapat berjalan di atas *web server* sebagai contoh sebuah kode *client-side* adalah *JavaScript* dimana kode tersebut dapat berjalan tanpa bantuan *web server*. Cukup menggunakan *browser*, kode tersebut dapat terekseskusi. Sedangkan PHP dolah terlebih dahulu oleh mesing pengolahan PHP yang terdapat pada *web server*. Dengan menggunakan PHP, pengembang *website* dapat membuat sebuah *web* yang dinamis seperti CMS (*Content Manageman System*), blog bahkan dengan inovasi para pengembang, PHP menjadi pemrograman berorientasi objek melalui framework.



Gambar 2.9 Siklus dari bahasa pemrograman server-side [8]

2.2.12. *Framework* CodeIgniter

CodeIgniter merupaka *framework* berbasis PHP, memiliki konsep MVC sama seperti dengan *framework* PHP lainnya. Salah satu *framework* yang mudah difamai dan mudah digunakan. Kelebihan yang dimiliki *framework* CodeIgniter adalah sebagai berikut :

1). Mudah untuk dipelajari karena hanya memerlukan sedikit konfigurasi.
2). Mudah digunakan untuk pembanguna aplikasi berskala kecil sampai memengah.
3). Sistem yang dimiliki CodeIgniter sangat ringan dibanding dengan *framework* PHP lainnya.
4). Memiliki banyak komunitas untuk berbagi permasalah dan memecahkan solusi secara bersama-sama

Selainnya kelebih, CodeIgniter juga memiliki kelemahan. Diantaranya adalah :

1). Sedikitnya dukungan plugin JavaScript yang tidak dapat berjalan dengan baik pada CodeIgniter
2). Dukungan module pengolahan database yang lebih sedikit dibanding *framework* lain.
3). Tidak dianjurkan untuk membuat sebuah aplikasi berskala besar.

2.2.13. MySQL

MySQL adalah sebuah implementasi dari sistem manajemen basisdata relational (RDBMS) yang didistribusikan secara gratis dibawah lisensi GPL (*General Public Lisense*). Setiap pengguna dapat secara bebas menggunakan MySQL, namun dengan batasan perangkat lunak tersebut tidak boleh dijadikan produk terutama yang sifatnya komersial. MySQL sebenarnya merupakan turunan salah satu konsep utama dalam basisdata yang telah ada sebelumnya, SQL (*Structured Query Language*). SQL adalah sebuah konsep pengoperasian basisdata, terutama untuk pemilihan atau seleksi dan pemasukan data, yang memungkinkan pengoperasian data dikerjakan dengan mudah secara otomatis.

Kehandalan suatu basisdata (DBMS) dapat diketahui dari cara kerja pengoptimasinya dalam melakukan proses perintah – perintah SQL yang dibuat oleh pengguna maupun program – program aplikasi yang memanfaatkannya. Sebagai peladen basisdata, MySQL mendukung operasi basisdata transaksional maupun operasi basisdata nontransaksional. Pada modus operasi nontransaksional, MySQL dapat dikatakan unggul dalam hal unjuk kerja dibandingkan perangkat lunak peladen basisdata kompetitor lainnya. Namum demikian pada modus nontransaksional tidak ada jaminan atas reliabilitas terhadap data yang tersimpan, karenanya modus nontransaksional hanya cocok untuk jenis aplikasi yang tidak membutuhkan reliabilitas data seperti aplikasi blogging berbasis web (*wordpress*), CMS dan sejenisnya. Untuk kebutuhan sistem yang ditunjukan untuk bisnis sangat disarankan untuk menggunakan modus basisdata transaksional, hanya saja sebagai konsekuensunya unjuk kerja MySQL pada modus transaksional tidak secepat unjuk kerja pada modus nontransaksional**.** MySQL memiliki beberapa keistimewaan, antara lain :

1. Portabilitas, MySQL dapat berjalan stabil pada berbagai sistem operasi seperti Windows, Linux, FreeBSD, MacOS X Server, Solaris, Amiga dan masih banyak lagi.
2. Perangkat lunak sumber terbuka, MySQL didistribusikan sebagai perangkat lunak sumber terbuka, dibawah lisensi GPL sehingga dapat digunakan secara gratis.

3. Multiuser, MySQL dapat digunakan oleh beberapa pengguna dalam waktu yang bersamaan tanpa mengalami masalah atau konflik.
4. Performance tuning, MySQL memiliki kecepatan yang menakjubkan dalam menangani *query* sederhana, dengan kata lain dapat memproses lebih banyak SQL per satuan waktu.
5. Ragam tiper data, MySQL memiliki ragam tipe data yang sangat kaya, seperti *signed / unsigned integer, float, double, char, text, date, timestamp* dan lain – lain.
6. Perintah dan Funsi, MySQL memiliki operator dan fungsi secara penuh yang mendukung perintah *Select* dan *Where* dalam perintah (*query*).
7. Keamanan, MySQl memiliki beberapa lapisan kemanan seperti lever subnetmask, nama *host*, dan izin akses user dengan sistem perizinan yang mendetail serta sandi terenkripsi.
8. Skalabilitas dan Pembatasan, MySQL mampu menangani basis data dalam skala besar, dengan jumlah rekaman (*record*) lebih dari 50 juta dan 60 ribu tabel serta 5 milyar baris. Selain itu batas indeks yang dapat ditampung mencapat 32 indeks pada tiap tabelnya.
9. Konektivitas, MySQL dapat melakukan koneksi dengan client menggunakan protokol TCP/IP, Unix soket (UNIX), atau Named Pipes (NT).
10. Lokalisasi, MySQL dapat mendeteksi pesan kesalah pada clients dengan menggunakan lebih dari 2 puluh bahasa. Meski pun demikian, bahasa Indoneisa belum termasuk kedalamnya.
11. Antar Muka, MySQL memiliki antar muka (*interface*) terhadap berbagai aplikasi dan bahasa pemrograman dengan menggunakan fungsi API (*Application Programming Interface*).
12. Clients dan Peralatan, MySQL dilengkapi dengan berbagai peralatan (*tool*) yang dapat digunakan untuk administrasi basisdata, dan pada setiap peralatan yang ada disertakan petunjuk online.
13. Struktur tabel, MySQL memiliki struktur tabel yang lebih fleksibel dalam menangani ALTER TABLE, dibandingkan basisdata lainnya semacam PostageSQL ataupun Oracle.