BAB 2
TINJAUAN PUSTAKA

2.1 Profil Perusahaan

CV. Jatiluhur Mas merupakan perusahaan yang bergerak dalam bidang peternakan ikan, pengurusan mitra petani, serta penjualan pakan yang berlokasi di kecamatan Jatiluhur, Purwakarta.

2.1.1 Sejarah Perusahaan

CV. Jatiluhur Mas yang mulai dirintis sejak tahun 1993 merupakan perusahaan yang bergerak dibidang peternakan ikan. Pemiliknya Yanto Sugianto awalnya memulai bisnisnya dengan membuka 2 petak tambak ikan dan berkembang menjadi 16 petak pada tahun 1996. Seiring berjalannya waktu tambak ikan bapak Yanto pun semakin berkembang. CV. Jatiluhur Mas memiliki beberapa bagian diantara nya bagian penjualan pakan ikan, bagian pengelolaan tambak ikan, dan bagian pengelolaan tambak milik mitra petani.

2.1.2 Logo Perusahaan

Logo perusahaan merupakan identitas diri suatu perusahaan. Logo dari CV. Jatiluhur Mas dapat dilihat pada Gambar 2-1.



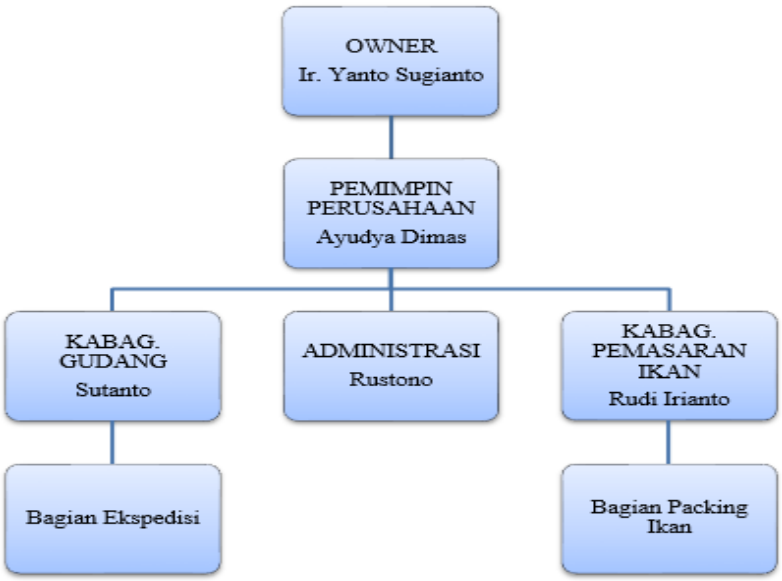
Gambar 2-1 Logo CV. Jatiluhur Mas

2.1.3 Struktur Organisasi Perusahaan

Struktur organisasi yang diterapkan oleh CV. Jatiluhur Mas adalah tipe organisasi fungsional dimana jenis struktur organisasi ini mengelompokkan orang

berdasarkan fungsi yang mereka lakukan dalam kehidupan professional atau menurut fungsi yang dilakukan dalam organisasi.

Struktur organisasi CV. Jatiluhur Mas dipimpin oleh pemiliknya sendiri yang yaitu bapak Yanto Sugianto. Namun pada 2 tahun terakhir kepemimpinan CV. Jatiluhur Mas di berikan kepada anaknya yaitu Ayudya Dimas. Sebagai pemimpin perusahaan saat ini Ayudya Dimas membawahi beberapa bagian yang dipimpin oleh seorang kepala bagian. Secara umum struktur organisasi CV. Jatiluhur Mas dapat dilihat pada Gambar 2-2.



Gambar 2-2 Struktur Organisasi CV. Jatiluhur Mas

Adapun tugas dan wewenang dari struktur organisasi CV. Jatiluhur Mas adalah sebagai berikut:

a. *Owner*

Tugas dari *owner* adalah sebagai berikut:

i. Sebagai pemilik perusahaan.

ii. Memantau perkembangan dan kinerja perusahaan

b. Pemimpin Perusahaan

Tugas dari pemimpin perusahaan adalah sebagai berikut:

i. Bertanggung jawab terhadap aktivitas perusahaan secara umum

ii. Mengawasi dan memonitor kegiatan seluruh perusahaan.

iii. Bertanggung jawab atas kelangsungan hidup perusahaan.
iv. Mengevaluasi hasil kerja yang dicapai dan mengambil tindakan yang diperulukan.
v. Menetapkan kebijakan perusahaan dengan menentukan rencana dan tujuan perusahaan baik jangka pendek maupun jangka panjang.
vi. Memperkirakan kolam siap panen untuk memenuhi permintaan pasar.
vii. Menangani pembayaran pakan mitra petani dengan cara transfer.

c. Kepala Bagian Gudang

Tugas dari kepala bagian gudang adalah sebagai berikut:

i. Menerima pemesanan pakan
ii. Mengelola pakan yang akan dikirim
iii. Mendata stok pakan dan sisa pakan di gudang.
iv. Menangani pembayaran pakan tunai mitra petani.

d. Administrasi

Tugas dari bagian administrasi adalah sebagai berikut:

i. Mengolah data calon mitra petani.

e. Kepala Bagian Pemasaran Ikan

Tugas dari kepala bagian pemasaran ikan adalah sebagai berikut:

i. Mendata panen ikan milik mitra petani.
ii. Mendata penjualan hasil pemasaran panen ikan.

f. Bagian Ekspedisi

Tugas dari bagian ekspedisi adalah:

i. Melakukan pengiriman pakan.
ii. Mendata jumlah pakan yang dikirim ke setiap kolam milik mitra petani.

g. Bagian Packing ikan

Tugas dari bagian packing ikan adalah:

i. Mem-*packing* ikan untuk dijual.

ii. Mengirim hasil panen yang sudah di *packing* ke pasar yang melakukan pemesanan ikan.

2.2 Landasan Teori

Pada landasan teori akan diterangkan teori-teori yang berhubungan dengan judul penulisan ini diantaranya pembahasan mengenai konsep dasar sistem informasi, sistem informasi manajemen, pengadaan pakan ikan, perencanaan pengiriman pakan ikan, monitoring dan evaluasi pengiriman pakan, analisis perancangan OOAD *(Object Oriented Analysis and Design)*, bahasa pemrograman, visual studio, *database*, metode TOPSIS *(Technique for Order Performance by Similiarity to Ideal Solution)* dan lain sebagainya. Berikut ini akan dibahas mengenai teori-teori yang akan menjadi sumber kajian dari sistem informasi manajemen yang akan dibangun.

2.2.1 Konsep Dasar Sistem Informasi

Informasi adalah data yang telah diolah menjadi sebuah bentuk yang berarti berarti bagi penerimanya dan bermanfaat dalam mengambil keputusan saat ini atau mendatang [1].

Hubungan data dengan informasi adalah seperti bahan baku sampai barang jadi. Dengan perkataan lain, sistem pengolahan informasi mengolah data menjadi informasi. Atau lebih tepatnya, sistem pengolahan yang mengolah data yang memiliki arti menjadi berguna bagi penerimanya.

Nilai informasi berhubungan dengan keputusan. Bila tidak ada pilihan atau keputusan, informasi menjadi tidak diperlukan. Keputusan dapat dapat berkisar dari keputusan berulang sederhana hingga keputusan strategis jangka panjang. Informasi dalam lingkup sistem informasi, memiliki beberapa ciri:

1. Benar atau salah. Ini dapat berhubungan dengan realitas atau tidak. Bila penerima informasi yang salah mempercayainya, akibatnya sama seperti yang benar.
2. Baru. Informasi dapat sama sekali baru dan segar bari penerimanya.
3. Tambahan. Informasi dapat memperbaharui atau memberikan tambahan baru pada informasi yang telah ada.

4. Korektif. Informasi dapat menjadi suatu koreksi atas informasi salah atau palsu sebelumnya.
5. Penegas. Informasi dapat mempertegas informasi yang telah ada. Ini masih berguna karena meningkatkan persepsi penerimanya ataskebenaran informasi tersebut.

2.2.1.1 Komponen Sistem Informasi

Sistem informasi terdiri dari komponen-komponen yang disebut dengan istilah blok bangunan (*building block*), yang teridiri dari blok masukan (*input block*), blok model (*model block*), blok keluaran *(output block*), blok teknologi (*technology block*), blok basis data (*database block*), dan blok kendali (*controls block*). Sebagai suatu sistem, keenam blok tersebut masing-masing berinteraksi satu dengan yang lainnya membentuk dengan satu kesatuan untuk mencapai sasaran. [5]

1. Blok Masukan

   Input mewakili data yang masuk ke dalam sistem informasi. Input disini termasuk metode-metode dan media untuk menangkap data yang akan dimasukkan, yang dapat berupa dokumen-dokumen dasar.

2. Blok Model

   Blok ini terdiri dari kombinasi prosedur, logika dan model matematik yang akan memanipulasi data input dan data yang tersimpan di basis data dengan cara yang sudah tertentu untuk menghasilkan keluaran yang diinginkan.

3. Blok Keluaran

   Produk dari sistem informasi adalah keluaran yang merupakan informasi yang berkualitas dan dokumentasi yang berguna untuk semua tingkatan manajemen serta semua pemakai sistem.

4. Blok Teknologi

   Teknologi merupakan “kotak alat” (*tool-box*) dalam sistem informasi. Teknologi digunakan untuk menerima input, menjalankan model, menyimpan dan mengakses data, menghasilkan dan mengirimkan keluaran dan membantu pengendalian dari sistem cara keseluruhan. Teknologi terdiri dari 3 bagian

utama, yaitu teknisi (*humanware atau brainware*), perangkat lunak (*software*) dan perangakat keras (*hardware*). Teknisi dapat berupa orang-orang yang mengetahui teknologi dan membuatnya dapat beroperasi. Misalnya teknisi adalah operator komputer, pemrograman, operator pengolah kata, spesialis telekomunikasi, analis sistem, penyimpan data dan sebagainya.

5. Blok Basis Data

   Basis data (*database*) merupakan kumpulan dari data yang saling berhubungan satu dengan yang lainnya, tersimpan di perangkat keras komputer dan digunakan perangkat lunak untuk memanipulasinya. Data perlu di simpan didalam basis data untuk keperluan penyediaan informasi lebih lanjut. Data didalam basis data perlu di organisasikan sedeimikian rupa, supaya informasi yang dihasilkan berkualitas. Organisasi basis data yang baik juga berguna untuk efisiensi kapasitas penyimpannya. Basis data diakses atau dimanipulasi dengan menggunakan perangkat lunak paket yang disebut dengan DBMS (*Database Management Systems*).

6. Blok Kendali

   Banyak hal yang dapat merusak informasi, seperti misalnya bencana alam, api, temperatur, air, debu, kecurangan-kecurangan, kegagalan-kegagalan sistem itu sendiri, kesalahan-kesalahan, ketidak efisienan, sabutase dan lain sebagainya. Beberapa pengendalian perlu dirancang dan diterapkan untuk meyakinkan bahwa hal-hal yang dapat merusak sistem dapat dicegah ataupun bila terlanjur terjadi kesalahan-kesalahan dapat langsung cepat diatasi.

2.2.1.2 Mutu Informasi

Mutu informasi dalam sebuah telaah yang dibuat oleh Adam mengenai sikap manajemen terhadap sistem informasi, 75% pemimpin perusahaan menilai peningkatan kuantitas dan mitu hampir identik dampaknya terhadap prestasi kerja. Tetapi bila diminta memilih, maka 90% lebih menyukai peningkatan dalam mutu informasi dibandingkan terhadap kuantitasnya. Informasi bervariasi dalam mutunya karena adanya bias atau kesalahan [2].

Kebanyakan sistem informasi, penerima informasi tidak memiliki pengetahuan tentang bias atau kesalahan yang dapat mempengahruhi mutu informasi tersebut. Proses pengukuran yang menghasilkan laporan dan ketepatan data di dalam lapoan secara tak langsung menyatakan bahwa ketepatannya tidak terjamin.

Kesulitan terhadap bias dapat ditangani dalam pengolahan informasi melalui prosedur-prosedur untuk mencari dan menguku bias dan kemudian menyesuaikannya. Kesulitan terhadap kesalahan dapat diatasi dengan:

1. Pengendalian intern untuk mengthui kesalahan.
2. Audit intern dan ekstern.
3. Menambahkan “batas-batas kepercayaan” pada data.
4. Intruksi pemakai dalam prosedur pengukuran dan pengolahan agar pemakai dapat meniai kesalahan yang mungkin terjadi.

Ada perbedaan antara dua cara pertama dengan kedua cara terakhir dalam mengatasi kesalahan. Kedua cara terakhir memberi batas kepercayaan pada pemakai, sedangkan dua cara pertama berusaha mengurangi ketidakpastian data dan karena itu meningkatkan kandungan informasi. Pengendalian intern dan pengauditan dalam konteks ini dapat dianggap menambahkan nilai informasi yang diberikan oleh sistem informasi dengan mengurangi keraguan akan kemungkinan adanya banyak kesalahan. Prosedur pengendalian dan audit tidak cenderung mempengaruhi bias maupun kesalahan yang disebabkan oleh metode pengukuran dan pengumpulan data.

2.2.2 Jenis-jenis Sistem

Ada beberapa cara untuk memandang sistem. Satu klasifikasi telah dikemukakan: sistem fisik dan abstrak. Klasifikasi lain adalah sistem deterministik dan probabilistik serta sistem tertutup dan terbuka.

2.2.2.1 Sistem Deterministik dan Probabilistik

Sistem deterministik beroperasi dalam cara yang dapat diramalkan secara tepat. Interaksi antara bagian-bagian diketahui dengan pasti. Bila seseorang

memiliki uraian keadaan sistem pada saat tertentu beserta uraian operasinya, maka keadaan sistem selanjutnya dapat disebutkan secara tepat tanpa kesalahan [2].

Sistem probabilistik dapat diuraikan dalam istilah perilaku yang mungkin, tetapi selalu ada sedikit kesalahan atas ramalan terhadap jalannya sistem.

2.2.2.2 Sistem Tertutup dan Terbuka

Sebuah sistem tertutup dalam fisika didefinisikan sebagai sebuah sistem yang mandiri (*self-contained*). Sistem ini tidak bertukar materi, informasi atau energi dengan lingkungannya [2].

Pada organisasi dan pengolahan informasi, ada sistem yang relatif terisolasi dari lingkungannya tetapi tidak sama sekali tertutup dalam arti fisik. Ini akan disebut sistem tertutup atau sistem yang secara relatif tertutup.

Sistem terbuka mengadakan pertukaran informasi, materi atau energi dengan lingkungannya. Pertukaran dapat meliputi masukan acak dan tak tentu. Sistem terbuka cenderung memiliki sifat adaptasi, berarti sistem dapat menyesuaikan terhadap perubahan dlaam lingkungannya sedemikian hingga dapat meneruskan eksistensinya. Sistem ini mengorganisasi diri dan mengubah organisasinya sebagai tanggapan atas perubahan keadaan.

2.2.3 Konsep Dasar Manajemen

Manajemen telah didefinisikan dalam berbagai cara. Akan tetapi, untuk maksud manajemen kita pahami sebagai proses atau kegiatan yang menjelaskna apa yang dilakukan pemimpin perusahaan pada operasional organisasi mereka untuk merencanakan, meng organisasikan, memprakarsai dan mengandalikan operasi [1]. Mereka merencanakan dengan menetapkan strategi, tujuan, dan memilih arah pindahan yang terbaik untuk mencapai aa yang direncanakan. Mereka mengorganisasikan tugas-tugas yang diperlukan untuk rencana operasional, menyusun tugas ini dalam kelompok yang homogen, dan menetapkan pendelegasian wewenang. Mereka mengendalikan prestasi kerja (*performance)* dengan menentukan norma-norma prestasi kerja dan menghindari penyimpanan terhadap norma atau standar tersebut. Karena pengambilan keputusan merupakan persyaratan yang mendasar bagis setiap proses terdulu, maka tugas SIM menjadi

sesuatu yang memberikan kemudahan pengambilan keputusan untuk perencanaan, pengorganisasian, pengendalian pekerjaan, dan fungsi bisnis. Ini berkaitan erat dengan keptusan terprogram dan peraturan pengambilan keputusan yang rumit karena teknik ilmu manajemen yang digabung dengan kemampuan komputer untuk menyajikan pemecahan masalah sangat membantu pemimpin perusahaan dalam membuat keputusan yang rumit. Herbert A. Simon mengemukakan teknik-teknik tradisional dan modern dalam pembuatan keputusan yang diprogram dan tidak diprogram, seperti yang ditunjukkan pada Tabel 2-1.

Tabel 2-1 Teknik-teknik pembuatan keputusan tradisional dan modern

Tipe-tipe keputusan	Teknik-teknik pembuatan keputusan	
	Tradisional	**Modern**
Diprogram: keputusan-keputusan rutin dan berulang-ulang. Organisasi mengembangkan proses-proses khusus bagi penanganannya.	1. Kebiasaan 2. Kegiatan rutin: Prosedur-prosedur pengoperasian standar. 3. Struktur organisasi - Pengharapan umum - Sistem tujuan - Saluran-saluran informasi yang disusun dengan baik	1. teknik-teknik riset operasi Analisa Matematika model-model simulasi komputer. 2. Pengolahan data elektronik
Tidak diprogram: keputusan-keputusan sekali pakai disusun tidak sehat. Kebijaksanaan ditangani dengan proses pemecahan masalah umum.	1. Kebijakan intuisi dan kreatifitas 2. Coba-coba 3. Seleksi & pelatihan para pelaksana	Teknik pemecahan masalah yang diterapkan pada: 1. Latihan membuat keputusan 2. Penyusunan program 2 komputer “heuristic”

2.2.3.1 Bentuk Kegiatan Manajemen

Kegiatan manajemen dihubungkan dengan tingkatannya didalam organisasi. Kegiatan menejemen tingkat atas, menengah, dan bawah adalah berbeda. Kegiatan-kegiatan manajemen mempengaruhi pengolahan informasi karena informasi yang dibutuhkan berbeda untuk masing-masing tingkatan. Kebutuhan informasi yang berbeda ini dapat diketahui dari masing-masing kegiatan manajemen tersebut [1]. Kegiatan manajemen untuk masing-masing tingkatan dapat dikategorikan sebagai berikut:

- Perencanaan strategis

  Merupakan kegiatan menajemen tingkat atas *(top manager)*. Frederick H. Wu dalam bukunya mendefinisikan perencanaan strategis.

“Perencanaan Strategis” adalah proses evaluasi lingkungan organisasi, penetapan tujuan organisasi, dan penentuan strategi-strategi.

a. Proses evaluasi lingkungan luar organisasi

Lingkugan luar organisasi selalu berubah secara konstan dan perubahan ini mungkin dapat mengakibatkna perubahan strategi yang sudah ditetapkan. Manajemen atas harus cepat tanggap tehadap tekanan-tekanan lingkungan luar yang dapat merugikan organisasi dan sedapat mungkin mengubah tekanan menjadi kesempatan.

b. Penentuan tujuan

Tujuan adalah apa yang ingin dicapai oleh organisasi. Tujuan ditetapkan oleh manajemen tingkat atas dalam proses perencanaan strategis yang bersifat jangka panjang.

c. Penetuan strategis

Manajemen tingkat atas menentukan tindakan-tindakan yang harus dilakukan organisasi dengan maksud mencapai tujuannya. Inilah yang deisebut strategi. Dengan strategi, semua kemampuan yang berupa sumber daya yang terlihat seperti material, modal, personil, dan kesempatan lingkungan luar organisasi.

- Pengendalian manajemen

  Pengendalian manajemen adalah proses meyakinkan bahwa organisasi telah menjalankan strategi yang sudah ditetapkan dengan efektif dan efisien. Pengendalian manajemen merupakan tingkatan taktik, yaitu bagaimana manjemen tingkat menengah menjalankan taktis supaya perencanaan strategis dapat dilakukan dengan berhasil. Taktik yang dijalankan biasanya besifat jangka pendek. Proses dalam pengendalian manajemen menyangkut komunikasi informal dan formal. Komunikasi informal terjadi dalam pertemuan-pertemuan dan diskusi. Sebagai tambahan informasi informal, beberapa perusahaan juga mempunyai proses pengendalian menajemen yang formal, yang terdiri dari tahap pemrograman, penyusunan anggaran, pelaksanaan, dan pengukuran serta pelaporan dan analisis.

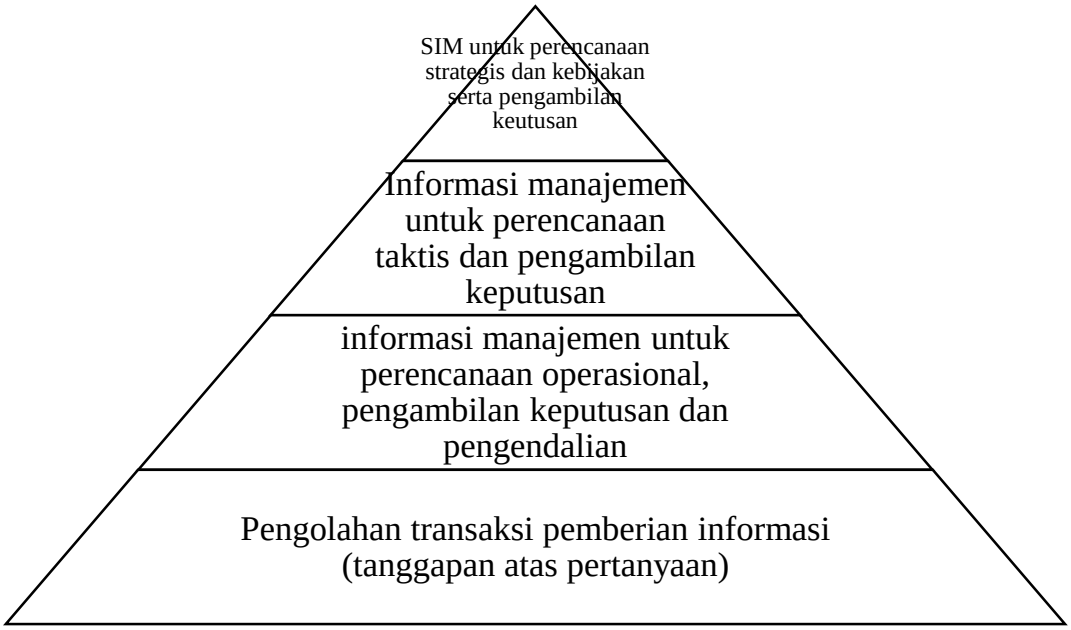
- Pengendalian Operasional

Pengendalian operasional adalah proses meyakinkan bahwa setiap tugas tertentu telah dilaksanakan secara efektif dan efisien. Pengendalian operasi ini merupakan proses penerapan program yang telah ditetapkan dalam pengendalian manajemen [7]. Pengendalian operasi dilakukan di bawah pedoman proses pengendalian manajemen dan difokuskan pada tugas manajemen tingat bawah. Dengan demikian, dapat diambil kesimpulan bahwa kegiatan-kegiatan dalam manajemen tingkat atas lebih menjurus ke perencanaan jangka panjang dan penentuan strategis. Lebih bawah tingkatannya, kegiatan manajemen lebih menjurus ke hal-hal yang sifatnya operasional.

2.2.4 Sistem Informasi Manajemen

Sebuah sistem informasi manajemen atau SIM adalah sebuah sistem informasi yang selain melakukan semua pengolahan transaksi yang perlu untuk sebuah organisasi, juga memberi dukungan informasi dan pengolahan fungsi manajemen dan pengambilan keputusannya [2]. Organisasi selalu membutuhkan sistem-sistem untuk mengumpulkan, mengolah, menyimpan, melihat kembali, dan menyalurkan informasi. SIM merupakan sistem pengolahan informasi berdasarkan komputer yang dirancang untuk fungsi operasi, manajemen, dan keputusan sebuah organisasi.

Sebuah organisasi mengadakan transaksi-transaksi yang harus diolah agar bisa menjalankan kegiatannya sehari-hari. Daftar gaji harus disiapkan, penjualan dan pembayaran semua ini dan hal-hal lainnya adalah kegiatan pengolahan data yang mengikuti prosedur standar tertentu. Komputer bermanfaat untuk tugas-tugas pengolahan data semacam ini, tetapi sebuah sistem informasi manajemen melaksanakan pula tugas-tugas lain dan lebih daripada sekedar sistem pengolahan data, yaitu sistem pengolahan informasi yang menerapkan kemampuan komputer untuk menyajikan informasi bagi manajemen dan bagi pengambilan keputusan. Sistem informasi manajemen digambarkan sebagai sebuah piramida yang dapat dilihat pada Gambar 2-3.

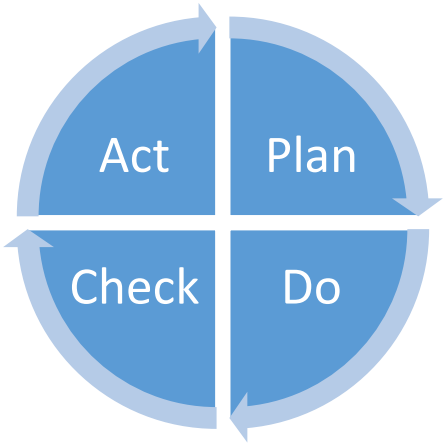


Gambar 2-3 Piramida Sistem Informasi Manajemen

Dimana lapisan dasarnya terdiri dari informasi untuk pengolahan transaksi, penjelasan status, lapisan berikutnya terdiri dari sumber-sumber informasi dalam mendukung operasi manajemen sehari-hari, lapisan ketiga terdiri dari sumber daya sistem informasi untuk membantu perencanaan taktis dan pengambilan keputusan untuk pengendalian manajemen, dan lapisan puncak terdiri dari sumberdata informasi untuk mendukung perencanaan dan perumusan kebijakan oleh tingkat puncak manajemen.

Definisi sebuah sistem informasi manajemen, istilah yang umum dikenal orang, adalah sistem yang terpadu, untuk menyaikan informasi guna mendukung fungsi operasi, manajemen, dan pengambilan keputusan dalam sebuah organisasi. Sistem ini menggunakan perangkat keras dan perangkat lunak komputer, prosedur pedoman, model manajemen dan keputusan, dan sebuah *database.*

Adapun tahapan dalam sistem informasi manajemen (SIM) dengan menggunakan siklus PDCA seperti pada Gambar 2-4.



Gambar 2-4 Siklus PDCA

Siklus PDCA adalah metode sistematis untuk perbaikan proses terus-menerus didasarkan pada prinsip bahwa kita perlu mengerti situasi atau proses sebelum memperbaikinya.

Proses utama, hasil sebenarnya dari suatu tindakan dibandingkan dengan target atau tujuan awal, lalu dilakukan langkah-langkah perbaikan jika jarak perbedaan terlalu besar. Sifat berulang dan perbaikan terus-menerus merupakan karakteristik siklus PDCA (Plan, Do, Check, Act). Hal ini disebut juga sebgai Deming Circle, dinamai oleh W.E. Deming. Variasi lain dari PDCA adalah PDSA (Plan, Do, Study, Act) [6].

Berdasarkan Gambar 2-4 siklus PDCA diatas terdapat 4 kegiatan yaitu *Plan, Do, Check, Action.* Berikut adalah penjelasan dari masing-masing kegiatan siklus PDCA:

1. *Plan,* tahap *plan* adalah tahap untuk menetapkan target atau sasaran yang ingin dicapai dalam pengingkatan proses ataupun permasalahan yang ingin dipecahkan, kemudian menentukan metode yang akan digunakan untuk mencapai target atau sasaran yang telah diteteapkan.
2. *Do,* tahap *do* adalah tahap penerapan atau melaksanakan semua, yang telah direncanakan di tahap *plan* termasuk menjalankan proses-nya, serta melakukan pengumpulan data *(data collection)* yang kemudian akan digunakan untuk tahap *check* dan *act*.
3. *Check,* tahap *check* adalah tahap pemeriksaan dan peninjauan ulang serta mempelajari hasil-hasil dari penerapan di tahap *do*. Melakukan

perbandingan antara hasil sebenarnya yang telah dicapai dengan target yang ditetapkan dan juga ketepatan jadwal yang telah ditentukan.

4. *Act*, tahap *act* adalah tahap untuk mengambil tindakan yang seperlunta terhadap hasil-hasil dari tahap *check*. Terdapat 2 jenis tindakan yang harus dilakukan berdasarkan hasil yang dicapainya, antara lain:
    a. Tindakan perbaikan *(Coreective Action)* yang berupa solusi terhadap masalah yang dihadapi dalam pencapaian target. Tindakan perbaikan ini perlu diambil jika hasilnya tidak mencapai apa yang telah ditargetkan.
    b. Tindakan standarisasi *(Standardization Action)* yaitu tindakan untuk men-standarisasi-kan cara ataupun praktek terbaik yang telah dilakukan, tindakan standarisasi ini dilakukan jika hasilnya mencapai target yang telah ditetapkan.

Siklus tersebut akan kembali lagi ke tahap *plan* untuk melakukan peningkatan proses selanjutnya sehingga terjadi siklus peningkatan proses yang terus menerus *(Continuous Process Improvement).*

2.2.5 Analisis Perancangan OOAD *(Object Oriented Analysis and Design)*

OOAD *(Object Oriented Analysis Design)* adalah metode analisis yang memeriksa *requirements* pada suatu kelas dan objek yang ditemui dalam ruang lingkup permasalahan yang mengarah pada arsitektur perangkat lunak yang didasarkan pada manipulasi objek-objek sistem atau subsistem. Diman terdapat beberapa konsep dalam OOAD, yaitu: kelas, *object*, metode, atribut, abstraksi, dan lain-lain.

2.2.6 UML

Unified Modelling Language (UML) merupakan sebuah notasi grafik untuk menggambarkan diagram konsep perangkat lunak. Dapat digunakan untuk menggambar diagram dari domain masalah, membuat desain perangkat lunak, atau implementasi perangkat lunak yang telah lengkap [9]. *Fowler* menjelaskan terdapat tiga perbedaan yaitu Konseptual, Spesifikasi, dan Implementasi.

Pada tahap Spesifikasi dan Impelementasi memiliki hubungan yang kuat dengan kode program. Itu merupakan kebutuhannya mengubah tingkat *Spesifikasi* menjadi kode sumber. Dengan demikian ada aturan dan semantik diagram pada tingkat tersebut harus diikuti. Diagram tersebut memiliki sedikit ambiguitas dan banyak formalitas.

UML merupakan kesatuan dari bahasa pemodelan yang dikembangkan oleh Booch, *Object Modelling Technique* (OMT) dan *Object Oriented Software Engineering* (OOSE) [9]. Metode Booch dari Grady Booch sangat terkenal dengan nama metode *Design Object Oriented*. Metode ini menjadikan proses analisis dan desain ke dalam empat tahapan iteratif, yaitu: identifikasi kelas – kelas dan obyek – obyek, identifikasi semantic dari hubungan obyek dan kelas terbut, perincian interface dan implementasi. Keunggulan metode Booch adalah pada detil dan kayanya dengan notasi dan elemen. Pemodelan OMT yang dikembangkan oleh Rumbaugh didasarkan pada analisis terstruktur dan pemodelan entity-relationship. Tahapan utama dalam metodologi ini adalah analisis, desain sistem, desain obyek dan implementasi. Keunggulan metode ini adalah dalam penotasian yang mendukung semua konsep OO. Metode OOSE dari Jacobson lebih memberi penekanan pada use case. OOSE memiliki tiga tahapan yaitu membuat model requirement dan analisis, desain, implementasi, dan model pengujian (test model). Keunggulan metode ini adalah mudah dipelajari karena memiliki notasi yang sederhana namun mencakup seluruh tahapan dalam rekayasa perangkat lunak.

Dengan UML, metode Booch, OMT dan OOSE digabungkan dengan membuang elemen – elemen yang tidak praktis ditambah dengan elemen – elemen dari metode lain yang lebih efektif dan elemen – elemen baru yang belum ada pada metode terdahulu sehingga UML lebih ekspresif dan seragam daripada metode lainnya. Unsur – unsur yang membentuk UML dapat dilihat pada Gambar 2-5.



Gambar 2-5 Unsur Pembentuk UML

Komponen – komponen UML

UML mendefinisikan diagram – diagram berikut ini:

a. *Use Case Diagram*

Use Case Diagram menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Yang ditekankan adalah "apa" yang diperbuat sistem, dan bukan "bagaimana". Sebuah use case merepresentasikan sebuah interaksi antara aktor dengan sistem.

b. *Class Diagram*

1) Class adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek. Class menggunakan keadaan (atribut/properti) suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (metoda/fungsi).
2) Class diagram menggambarkan struktur dan deskripsi class, package dan objek beserta hubungan satu sama lain seperti containment, pewarisan, asosiasi, dan lain – lain.

c. *Object Diagram*

Object diagram serupa dengan diagram kelas, tetapi dari pada menggambarkan kelas objek, lebih baik menggunakan diagram objek yang memodelkan *instance* objek aktual dengan menunjukkan nilai – nilai saat ini dari *attribute instance*.

d. *Statechart Diagram*

Statechart diagram menggambarkan transisi dan perubahan keadaan (dari satu state ke state lainnya) suatu objek pada sistem sebagai akibat dari stimuli yang diterima.

e. *Activity Diagram*

1) Activity diagram menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing – masing alir berawal, keputusan mungkin terjadi, dan bagaimana mereka berakhir. Activity diagram juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi.
2) Activity diagram merupakan state diagram khusus, yang sebagian besar state adalah action dan sebagian besar transisi di-*trigger* oleh selesainya state sebelumnya (*internal processing*). Oleh karena itu, activity diagram tidak menggambarkan perilaku internal sebuah sistem dan interaksi antar subsistem secara eksak, tetapi lebih menggambarkan proses – proses dan jalur – jalur aktivitas.

f. *Sequence Diagram*

Sequence diagram secara grafis menggambarkan bagaimana objek berinteraksi antara satu sama lain melalui pesan pada sebuah use case atau operasi.

g. *Collaboration Diagram*

Collaboration diagram juga menggambarkan interaksi antar objek seperti sequence diagram, tetapi lebih menekankan pada peran masing – masing objek dan bukan pada waktu penyampaian *message*. Setiap *message* memiliki *sequence number*, dimana *message* dari level tertinggi memiliki nomor satu. *Messages* dari level yang sama memiliki prefix yang sama.

h. *Component Diagram*

1) Component diagram menggambarkan struktur dan hubungan antar komponen perangkat lunak, termasuk *dependency* diantaranya.
2) Komponen perangkat lunak adalah modul berisi kode, baik berisi kode sumber maupun *binary code*, baik *library* maupun *executable*, baik yang muncul pada *compile time, link time,* maupun *run time*. Umumnya

komponen terbentuk dari beberapa *class* dan/atau *package*, tapi dapat juga dari komponen – komponen yang lebih kecil. Komponen dapat juga berupa *interface*, yaitu kumpulan layanan disediakan sebuah komponen untuk komponen lain.

i. *Deployment Diagram*

1) Deployment/physical diagram menggambarkan detail bagaimana komponen di-*deploy* dalam infrastruktur sistem, dimana komponen akan terletak pada mesin, server atau perangkat keras apapun, bagaimana kemampuan jaringan pada lokasi tersebut, spesifikasi server, dan hal – hal lain yang bersifat fisikal.
2) Sebuah node adalah server, *workstation*, atau perangkat keras lain yang digunakan untuk me-*deploy* komponen dalam lingkungan sebenarnya. Hubungan antar node misalnya TCP/IP dan *requirement* dapat juga didefinisikan dalam diagram ini.

2.2.7 Visual Paradigm

Visual Paradigm sebuah software model dengan system visualisasi memungkinkan model yang telah dibuat dapat digunakan sebagai representasi proyek-proyek lain dilengkapi dengan beberapa fitur yang ada didalamnya sampai pada menganalisa sebuah proyek yang akan dikerjakan. Diagram dapat disusun sedemikian rupa sehingga dapat dipustakakan menjadi proyek per proyek yang saling berkaitan. Hal ini dapat juga membantu memisahkan terhadap pekerjaan proyek sampai level terkecil.

Visual Paradigm dapat membandingkan perubahan antara diagram yang satu dengan yang lain yang saling berkaitan dan dapat digunakan untuk melacak perbedaan antara diagram-digram yang ada. Dengan dukungan *design* layaknya *Freehand* dapat menambahkan ke diagram untuk penjelasan lebih detail apa yang menjadi tujuan. Fasilitas lain dapat menggunakan *mouse pen* atau pena untuk *sketching* bentuk jalan, kita juga dapat menambahkan bentuk solid seperti bintang, dan panah *rectangles*.

Visual Paradigm, bentuk nama dengan model horizontal, dan menengah atas atau vertikal, tergantung pada karakteristik dari bentuk. Hal ini dimungkinkan untuk menyetel kembali bentuk nama, yang berguna untuk bahasa yang ditulis dari kanan ke kiri, seperti Ibrani modern. Gaya dukungan untuk bentuk dan konektor. Tampilan visual bentuk sebagian besar tergantung pada format properti seperti latar belakang, depan, pilihan font dan anak panah presentasi. Dengan gaya mendukung, pengguna dapat menentukan gaya dengan format properti khusus, dan kembali pada bentuk lainnya tanpa perlu mendefinisikan ulang format lagi dan lagi.

Diagram Layer adalah fitur untuk membantu dalam berbagai bentuk *categorizing* lapisan berbeda. Kita dapat menggunakan lapisan yang terpisah untuk penjelasan, kami dapat juga tempat alternatif jalan yang mengalir di lapisan yang terpisah. Sejak lapisan dapat ditampilkan atau menyembunyikan, memberikan solusi yang ideal untuk mengendalikan porsi diagram untuk muncul di diagram.

2.2.8 Database

Semua sistem informasi membuat, membaca, memperbarui, dan menghapus (kadang-kadang disingkat CRUD) data. Data disimpan di dalam *file* dan *database*. *File* adalah sebuah kumpulan *record* yang serupa. Contoh-contohnya mencakup *FILE COSTUMER, FILE ORDER,* dan *FILE PRODUK. Database* adalah kumpulan *file* yang saling terkait. Kata kuncinya adalah "saling terkait". *Database* tidak hanya berupa kumpulan *file*. *Record* pada setiap *file* juga harus memperbolehkan hubungan-hubungan (anggaplah sebagai *"pointer"*) untuk menyimpan *files* lainnya [4].

2.2.8.1 Database Management System

Database management system adalah perangkat lunak komputer khusus yang disediakan dari vendor-vendor komputer yang digunakan untuk membuat, mengakses, mengontrol, dan mengelola *database*. DBMS sering disebut *database engine.* Mesin ini merepon perintah-perintah khusus untuk membuat struktur *database* kemudian membuat, membaca, memperbarui, dan menghapus *records* pada sebuah *database.* Sistem manajemen *database* dibeli dari sebuah vendor teknologi *database* seperti Oracle, IBM, Microsoft, atau Sybase [4].

Analisis *database*, mendesain struktur data dalam tipe hal *record, fields* yang diisikan pada *record* tersebut, dan hubungan yang ada pada tipe-tipe *record*. Struktur ini didefinisikan untuk sistem manajemen *database* dengan menggunakan bahasa definisi data milik sistem tersebut. *Data Definition Language (DDL)* digunakan oelh DBMS untuk menetapkan secara fisik tipe, *record, field*, dan hubungan struktural. Selain itu DDL menentukan *view database*. *View* tersebut membatasi bagian dari sebuah *database* yang dapat digunakan atau diakses oleh para pengguna dan program yang berbeda.

Sebagian besar sistem menajemen *database* menyimpan baik *user data* maupun metadata – data (atau spesifikasi) mengenai data seperti *record* dan *field,* sinonim-sinonim, hubungan data, aturan-aturan, validasi, *help message,* dan seterusnya.

Sistem manajemen *database* juga menyediakan sebuah bahasa manipulasi data untuk mengakses dan menggunakan data yang disimpan pada aplikasi-aplikasi. *Data Manipulation Language (DML)* digunakan untuk membuat, membaca, memperbarui, dan menghapus *records* pada sebuah database dan untuk menjelajahi diantara *records* dan tipe-tipe *record* yang berbeda. DBMS dan DML menyembunyikan rincian-rincian yang terkait dengan bagaimana *record* diorganisasikan dan dialokasikan pada sebuah *disk.* Pada umumnya DML sangat fleksibel dalam hal ia dapat digunakan untuk membuat, membaca, memperbarui, dan menghapus *records* atau perintah-perintah DML yang dapat "dipanggil" dari sebuah bahasa *host programming* terpisah seperti COBOL, *Visual Basic,* atau *Java.*

2.2.8.2 MySql

MySQL adalah sebuah perangkat lunak system manajemen basis data SQL (DBMS) yang multithread, dan multi-user. MySQL adalah implementasi dari system manajemen basisdata 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. Pendiri 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.

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 pemograman seperti C, dan Delphi.

A. Elemen SQL

Elemen dari SQL yang paling dasar antara lain pernyataan, nama, tipe data, ekspresi, konstanta dan fungsi bawaan.

b. Pernyataan

Perintah dari SQL yang digunakan untuk meminta sebuah tindakan kepada DBMS.

Pernyataan dasar SQL antara lain:

1. ALTER : Merubah struktur tabel
2. COMMIT : Mengakhiri eksekusi transaksi
3. CREATE : Membuat tabel, indeks
4. DELETE : Menghapus baris pada sebuah tabel
5. DROP : Menghapus tabel, indeks
6. GRANT : Menugaskan hak terhadap basis data kepada user
7. INSERT : Menambah baris pada tabel
8. REVOKE : Membatalkan hak kepada basis data

9. ROLLBACK : Mengembalikan pada keadaan semula apabila transaksi gagal
   dilaksanakan
10. SELECT : Memilih baris dan kolom pada sebuah tabel
11. UPDATE : Mengubah value pada baris sebuah tabel

c. Tipe data

Tipe data numerik antara lain:

1. TINYINT : Nilai integer yang sangat kecil
2. SMALLINT : Nilai integer yang kecil
3. MEDIUMINT : Nilai integer yang sedang
4. INT : Nilai integer dengan nilai standar
5. BEGINT : Nilai integer dengan nilai besar
6. FLOAT : Bilangan decimal dengan single-precission
7. DOUBLE : Bilangan decimal dengan double-precission
8. DECIMAL(M,D) : Bilangan float yang dinyatakan sebagai string. M : jumlah digit yang disimpan, D : jumlah angka dibelakang koma

Tipe data String antara lain :

1. CHAR : Karakter yang memiliki panjang tetap yaitu sebanyak n
2. VARCHAR : Karakter yang memiliki panjang tidak tetap yaitu maksimum n
3. TINYBLOB : BLOB dengan ukuran sangat kecil
4. BLOB : BLOB yang memiliki ukuran kecil
5. MEDIUMBLOB : BLOB yang memiliki ukuran sedang
6. LONGBLOB : BLOB yang memiliki ukuran besar
7. TINYTEXT : teks dengan ukuran sangat kecil
8. TEXT : teks yang memiliki ukuran kecil
9. MEDIUMTEXT : teks yang memiliki ukuran sedang
10. LONGTEXT : teks yang memiliki ukuran besar

11. ENUM : kolom diisi dengan satu member enumerasi
12. SET : Kolom dapat diisi dengan beberapa nilai anggota himpunan

Tipe data tunggal dan jam:

1. DATE : date memiliki format tahun-bulan-tanggal
2. TIME : time memiliki format jam-menit-detik
3. DATETIME : gabungan dari format date dan time

a. Ekspresi

Ekspresi digunakan untuk menghasilkan/menghitung nilai.
Misalnya : jumlah=harga-diskon
Ekspresi aritmatika antara lain:

1. + : tambah
2. – : kurang
3. / : bagi
4. * : kali

b. Konstanta
Nilai yang tetap

a. Fungsi bawaan

Fungsi adalah subprogram yang dapat menghasilkan suatu nilai apabila fungsi tersebut dipanggil. Fungsi Agregat adalah fungsi yang digunakan untuk melakukan summary, statistik yang dilakukan pada suatu tabel/query.

1. AVG(ekspresi) : digunakan untuk mencari nilai rata-rata dalam kolom dari tabel.
2. COUNT(x) : digunakan untuk menghitung jumlah baris dari sebuah kolom dari tabel
3. MAX(ekspresi) : digunakan untuk mencari nilai yang paling besar dari suatu kolom dari tabel

4. MIN(ekspresi) : digunakan untuk mencari nilai yang paling kecil dari suatu kolom dari tabel
5. SUM(ekspresi) : digunakan untuk mengitung jumlah keseluruhan dari suatu kolom dari tabel

B. Kelompok Penyataan SQL

Pernyataan SQL dapat dikelompokan menjadi 3 kelompok, yaitu : DDL, DML dan DCL.

a. Pernyataan SQL kelompok DDL (Data Defination Language)

DDL berfungsi untuk mendefinisikan atribut basis data, table, atribut(kolom), batasan-batasan terhadap suatu atribut, serta hubungan antar tabel. Perintah yang digunakan biasanya : CREATE, ALTER, dan DROP

1. Membuat Database
   Syntax yang digunakan : CREATE DATABASE namadatabase;
2. Menghapus Database
   Syntax yang digunakan : DROP DATABASE namadatabase;
3. Membuat Tabel
   Syntax yang digunakan : CREATE TABLE namatabel;
4. Menghapus Database
   Syntax yang digunakan : DROP DATABASE namadatabase;
5. Menghapus Tabel
   Syntax yang digunakan : DROP TABEL namatabel;
6. Mendefinisikan null/not null
   Syntax yang digunakan : CREATE TABLE namatabel
   (Field1 TipeData1 NOT NULL,
   Field2 TipeData2
   );
7. Mendefinisikan Nilai Default
   Syntax yang digunakan : CREATE TABLE namatabel
   (Field1 TipeData1,
   Field2 TipeData2 DEFAULT nilai

);

8. Mendefinisikan Primary Key pada Tabel

   Dapat dilakukan dengan 3 Syntax :

   1) CREATE TABLE namatabel
      (Field1 TipeData1 NOT NULL PRIMARY KEY,
        Field2 TipeData2
      );
   2) CREATE TABLE namatabel
      (Field1 TipeData1,
        Field2 TipeData2,
        PRIMARY KEY(Field1)
      );
   3) ALTER TABEL namatabel ADD CONSTRAINT namaconstraint PRIMARY KEY (NAMAKOLOM)

9. Menghapus Primary Key pada Tabel

   Syntax yang digunakan ada 2 :

   1) ALTER TABLE namatabel DROP CONSTRAINT NAMACONSTRAINT;
   2) ALTER TABLE NAMATABEL DROP PRIMARY KEY;

10. Menambah Kolom Baru pada Tabel

    Syntax yang digunakan : ALTER TABEL NAMATABEL ADD newfield tipe;

11. Mengubah Tipe Data atau Lebar Kolom pada Tabel

    Syntax yang digunakan : ALTER TABEL namatabel MODIFY COLUMN field tipe;

12. Mengubah Nama Kolom

    Syntax yang digunakan : ALTER TABEL namatabel CHANGE COLUMN namakolomlama namakolombaru tipe;

13. Menghapus Kolom pada Tabel

    Syntax yang digunakan : ALTER TABEL namatabel DROP COLUMN namakolom;

14. Mendefinisikan Foreign Key pada Tabel

    Dapat dilakukan dengan 2 Syntax :

    1) CREATE TABLE namatabel
       (Field1 TipeData1,
       Field2 TipeData2,
       FOREIGN KEY (Field2) REFERENCES namatabelinduk
       (namakolominduk) ON UPDATE CASCADE ON DELETE NO ACTION)
    2) ALTER TABEL namatabel ADD CONSTRAINT namaconstraint FOREIGN KEY (namakolom) REFERENCES namatabelinduk (namakolominduk) ON UPDATE CASCADE ON DELETE NO ACTION;

15. Menghapus Foreign Key

    Syntak yang digunakan: ALTER TABEL namatabel DROP FOREIGN KEY namaconstraint;

b. DML (Data Manipulation Language)

DML berfungsi untuk memanipuladi data yang ada di dalam basis data, contohnya untuk pengambilan data, penyisipan data, pengubahan data dan penghapusan data.

Perintah yang digunakan biasanya adalah: INSERT, DELETE, UPDATE, dan SELECT.

1. INSERT

   menambah baris pada tabel. Syntax yang paling sering digunakan: INSERT INTO namatabel VALUES (nilai1, nilai2, nilai-n);

2. DELETE

   Menghapus baris pada tabel. Syntax: DELETE FROM namatabel [where kondisi];

3. UPDATE

   Mengubah isi beberapa kolom pada tabel. Syntax: UPDATE namatabel SET kolom1=nilai1, kolom2=nilai2 [where kondisi];

4. SELECT

   Menampilkan isi dari suatu tabel yang bisa dihubungkan dengan tabel yang lainnya.

2.2.9 Bahasa Pemrograman

Bahasa pemrograman adalah teknik komando/instruksi standar untuk memerintah komputer yang merupakan suatu himpunan dari aturan sintaks dan semantik yang dipakai untuk mendefinisikan program komputer. Bahasa ini memungkinkan seorang programmer dapat menentukan secara persis data mana yang akan diolah oleh komputer, bagaimana data ini akan disimpan/diteruskan, dan jenis langkah apa secara persis yang akan diambil dalam berbagai situasi.

Menurut tingkat kedekatannya dengan mesin komputer, bahasa pemrograman terdiri dari:

1. Bahasa mesin. Yaitu memberikan perintah kepada komputer dengan memakai kode bahasa biner, contohnya 0110110101.
2. Bahasi tingkat rendah, atau dikenal dengan istilah bahasa rakitan (bahasa inggris *assembly*), yaitu memberikan perintah kepada komputer dengan memakai kode-kode singkat (kode *mnemonic*), contohnya MOV, SUB, CMP, JMP, JGE, JL, LOOP, dan sebagainya.
3. Bahasa tingkat menengah, yaitu bahasa komputer yang memakai campuran instruksi dalam kata-kata manusia dan instruksi yang bersifat simbolik, contohnya {,},?,<<,>>,&&,||, dan sebagainya.

Sebagian besar bahasa pemrograman digolongkan sebagai bahasa tingkat tinggi, hanya bahasa C yang digolongkan sebagai bahasa tingkat menengah dan *Assembly* yang merupakan bahasa tingkat rendah.

2.2.9.1 C Sharp (C#)

Bahasa C# (dibaca C sharp) adalah suatu aset yang terdapat dalam .NET *Framework*. Anda dapat menggunakan C# untuk membuat *file* eksekusi yang berdiri sendiri atau untuk membuat isi atau konten dinamis dalam sebuah *web form.* Untuk memulai pembuatan program atau aplikasi, akan dibutuhkan *text editor*, seperti notepad, untuk menghasilkan *source code* (kode sumber). Tipe *file* yang

sangat dipertahankan dari kode C# adalah sebuah file class. Suatu class C# adalah sebuah file teks yang disimpan dengan ekstensi file *.cs. hasil pengkodingan dapat dikompilasi dengan menggunakan *compiler* c# (csc.exe) yang terdapat pada command prompt yang mana membuat file eksekusi berekstensi *.exe.

2.2.10 Visual Studio

Microsoft Visual Studio merupakan sebuah perangkat lunak lengkap (*suite*) yang dapat digunakan untuk melakukan pengembangan aplikasi, baik itu aplikasi bisnis, aplikasi personal, ataupun komponen aplikasinya, dalam bentuk aplikasi console, aplikasi windows, ataupun aplikasi Web. Visual studio mencakup kompiler, SDK, Integrated Development Environment (IDE), dan dokumentasi (umumnya berupa MSDN Library). Kompiler yang dimasukkan ke dalam paket Visual Studio antara lain Visual C++, Visual C#, Visual Basic, Visual Basic .NET, Visual InterDev, Visual J++, Visual FoxPro, dan Visual SourceSafe.

Microsoft Visual Studio dapat digunakan untuk mengembangkan aplikasi dalam native code (dalam bentuk bahasa mesin yang berjalan diatas Windows) ataupun managed code (dalam bentuk Microsoft Intermediate Language diatas .NET Framework). Selain itu, Visual Studio juga dapat digunakan untuk mengembangkan aplikasi Silverlight, aplikasi Windows Mobile (yang berjalan di atas .NET Compact Framework).

2.2.11 TOPSIS *(Technique for Order Performance by Similiarity to Ideal Solution)*

TOPSIS adalah salah satu metode pengambilan keputusan multikriteria yang pertama kali diperkenalkan oleh Yoon dan Hwang (1981). TOPSIS menggunakan prinsip bahwa alternatif yang terpilih harus mempunyai jarak terdekat dasi solusi ideal positif dan terjauh dari solusi ideal negatif dari sudut pandang geometris dengan menggunakan jarak Euclidean untuk menentukan kedekatan relatif dari suatu alternatif dengan solusi optimal. Solusi ideal positif didefinisikan sebagai jumlah dari sleuruh nilai terbaik yang dapat dicapai untuk setiap atribut, sedangkan solusi negatif-ideal terdiri dari seluruh nilai terburuk yang dicapai untuk setiap atribut [3].

TOPSIS mempertimbangkan keduanya, jarak terhadap solusi ideal positif dan jarak tehadap solusi ideal negatif dengan mengambil kedekatan relatif terhadap solusi ideal positif. Berdasarkan perbandingan terhadap jarak relatifnya, susunan prioritas alternatif bisa dicapai. metode ini banyak digunakan untuk menyelesaikan pengambilan keputusan. Hal ini disebabkan konsepnya sederhana, mudah dipahami, komputasinya efisien, dan memiliki kemampuan mengukur kinerja relatif dari alternatif-alternatif keputusan.

Langkah-langkah penyelesaian masalah dengan TOPSIS, adalah sebagai berikut:

- Membuat matriks keputusan yang ternormalisasi
- Membuat matriks keputusan yang ternormalisasi terbobot
- Menentukan matriks solusi ideal positif dan matriks solusi ideal negatif
- Menentukan jarak antara nilai setiap alternatif dengan matriks solusi ideal positif dan solusi ideal negatif
- Menentukan nilai preferensi untuk setiap alternatif
- Merangking alternatif

Algoritma TOPSIS ini juga tentunya mempunyai kelebihan maupun kekurangan. Untuk detailnya dapat dilihat pada Tabel 2-2.

Tabel 2-2 Kelebihan dan kekurangan algoritma TOPSIS

Kelebihan	**Kekurangan**
Konsepnya sederhana dan mudah dipahami	Harus adanya bobot yang dihitung untuk melanjutkan hitungan data selanjutnya dengan memakai TOPSIS.
Komputasinya efisien	
Memiliki kemampuan untuk mengukur kinerja relatif dari alternatif-alternatif keputusan dalam bentuk matematis sederhana	

Adapun langkah-langkah algoritma dari TOPSIS ini adalah sebagai berikut:

1. Membuat matriks keputusan yang ternormalisasi

   Elemen r_{ij} hasil dari normalisasi *decision matrix R* dengan metode *Euclidean length of a vector* adalah:

$$R_{ij} = \frac{X_{ij}}{\sqrt{\sum_i^m X_{ij}^2}}$$

Dimana:

R_{ij} = Hasil dari normalisasi matriks keputusan R

i = 1,2,3,…,m;

j = 1,2,3,…,n;

2. Membuat matriks keputusan yang ternormalisasi terbobot

   Dengan bobot *W* = (*w1, w2,…,wn*), maka normalisasi bobot matriks Y adalah:

$$Y = \begin{bmatrix} W_{11}R_{11} & \cdots & W_{1n}R_{1n} \\ \vdots & \ddots & \vdots \\ W_{m1}R_{m1} & \cdots & W_{nm}R_{nm} \end{bmatrix}$$

3. Menentukan matriks solusi ideal positif dan matriks solusi ideal negatif

   Solusi ideal positif dinotasikan dengan Y^+ dan solusi ideal negatif dinotasikan dengan Y^-, sebagai berikut:

   Menentukan solusi ideal (+) dan (-)

   $Y^+ = \{(\max y_{ij})\ (\min y_{ij} \mid j \in J'),\ i = 1,2,3,\ldots,\ m\} = \{y_1^+, y_2^+, \ldots, y_m^+\}$

   $Y^- = \{(\max y_{ij})\ (\min y_{ij} \mid j \in J'),\ i = 1,2,3,\ldots,\ m\} = \{y_1^-, y_2^-, \ldots, y_m^-\}$

   Dimana:

   y_{ij} = elemen matriks Y baris ke-*I* dan kolom ke-*j*

   j = {j=1,2,3,…,n dan j berhubungan dengan *benefit criteria*}

   J' = {j=1,2,3,…,n dan j berhubungan dengan *cost criteria*}

4. Meghitung jarak antara nilai setiap alternatif dengan matriks solusi ideal positif dan solusi ideal negatif

   Separation measure ini merupakan engukuran jarak dari suatu alternatif ke solusi ideal positif dan solusi ideal negatif. Perhitungan matematisnya adalah sebagai berikut:

   Separation measure untuk solusi ideal positif

   $D_i^+ = \sqrt{\sum_{j=1}^n \left(y_{ij} - y_j^+\right)^2}$, dengan *i* = 1,2,3,…,m

   Dimana:

   J = {j=1,2,3,…,n dan j berhubungan dengan *benefit criteria*}

J' = {j=1,2,3,…,n dan j berhubungan dengan *cost criteria*}

Separation measure untuk solusi ideal negatif

$D_i^- = \sqrt{\sum_{j=1}^{n}\left(y_{ij} - y_j^-\right)^2}$, dengan *i* = 1,2,3,…,m

Dimana:

J = {j=1,2,3,…,n dan j berhubungan dengan *benefit criteria*}

J' = {j=1,2,3,…,n dan j berhubungan dengan *cost criteria*}

5. Menghitung nilai preferensi untuk setiap alternatif

   Kedekatan relatif dari alternatif Y^+ dengan solusi ideal Y^- direpresentasikan dengan:

   $C_i \frac{D_i^-}{D_i^- + D_i^+}$, dengan $0 < C_i^+ < 1$ dan *i* = 1,2,3,…,m

6. Merangking alternatif

   Alternatif dapat diranking berdasarkan urutan C_i^*. Maka dari itu, alternatif terbaik adalah salah satu yang berjarak terpendek terhadap solusi ideal dan berjarak terjauh dengan solusi ideal negatif.

2.2.12 Menghitung Jarak Antar Titik Koordinat

Euclidean distance adalah perhitungan jarak dari 2 buah titik dalam *Euclidean space*. *Euclidean space* diperkenalkan oleh Euclid, seorang matematikawan dar Yunani. Rumus mencari jarak menggunakan *euclidean distance,* adalah sebagai berikut:

$$d = \sqrt{(x1 - x2)^2 + (y1 - y2)^2}$$

Dimana:

d = Jarak

x1 = Koordinat Latitude titik awal

y1 = Koordinat Longitude titik awal

x2 = Koordinat Latitude titik tujuan

y2 = Koordinat Longitude titik tujuan

Dari perhitungan tersebut didapat hasil jarak namun masih dalam satuan *decimal degree* (sesuai dengan format longitude dan latitude yang dipakai)

sehingga untuk menyesuaikannya perlu dikalikan dengan 111.319 km (1 derajat bumi = 111.319 km).

2.2.13 Metode SMART

Metode *SMART* merupakan metode yang dilakukan untuk menentukan sasaran atau target daipada suatu proyek. Prinsip *SMART GOAL* ini pertama kali diperkenalkan oleh George T. Doran pada tahun 1981 dalam Majalah *Management Review* edisi November 1981.

Penentuan sasaran ataupun target yang tepat merupakan hal yang sangat penting dalam memotivasi dan meningkatkan kinerja kerja suatu tim karena adanya fokus yang jelas terhadap apa yang akan dicapainya.

Kata "*SMART*" merupakan kumpulan dari 5 huruf pertama kata dalam Bahasa Inggris yaitu *spesific, measureable, attainable, realistic* dan *time bound.* Berikut ini adalah penjelasan singkatan mengenai *SMART Goals*:

1. Spesific

Target suatu proyek harus ditetapkan secara spesifik dan jelas. Suatu target yang ditentukan dengan spesifik akan memiliki pencapaian yang lebih tinggi dibandingkan dengan target yang ditentukan secara umum dan luas.

2. Measurebale

Target proyek yang ditentukan harus dapat diukur dengan menggunakan indikator yang tepat sehingga dapat melakukan tindakan-tindakan perbaikan yang seperlunya. Pengukuran harus berupa nilai-nilai fluktuatif yang berbentuk angka-angka berdasarkan fakta-faktanya.

3. Attainable

Target yang ditentukan harus dapat dicapai melalui usaha-usaha yang menantang dan harus berasarkan kemampuan yang dimiliki. Tim harus mengetahui letak kemampuannya dan mempertimbangkan kinerja sekarang dengan kinerja yang sifatnya sempurna. Dari kinerja sekarang sampai ke kinerja sempurna harus dilakukan secara bertahap dan target yang ingin dicapainya juga harus ditetapkan secara bertahap pula. Pada versi *SMART GOAL* lainnya, *attainable* juga disebut *achievable.*

4. Realistic

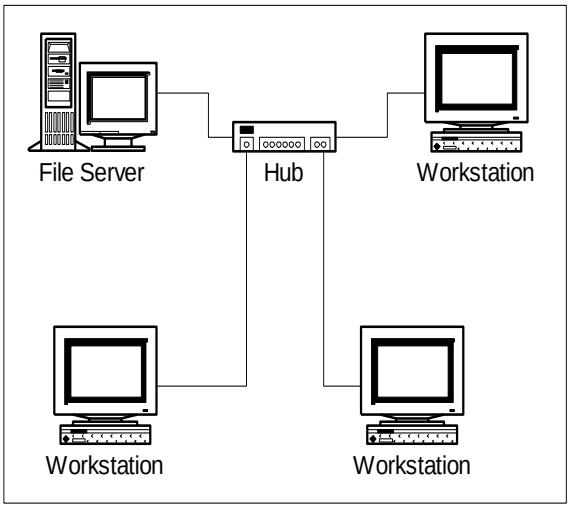
   Target proyek yang ditentukan harus bersifat realistis, jangan menentukan target yang terlalu tinggi dalam waktu yang singkat. Harus mengetahui batas kemampuan dari tim untuk mencapai target proyek yang ditentukan.

5. Timebound

   Harus menetapkan batas waktu dalam mencapai target proyek. Tanpa adanya batas waktu, tim akan bekerja lambat dan tidak ada perasaan *urgency* (mendesak) sehingga sangat sulit untuk mencapai target yang diinginkan.

2.2.14 Topologi *Star*

Dalam topologi star, sebuah terminal pusat bertindak sebagai pengatur dan pengendali semua komunikasi data yang terjadi. Terminal-terminal yang lain terhubung padanya dan pengiriman data dari satu terminal ke terminal yang lainnya melalui terminal pusat. Terminal pusat akan menyediakan jalur komunikasi khusus pada dua terminal yang berkomunikasi dapat dilihat pada Gambar 2-6.



Gambar 2-6 Topologi *Star*

1. Keuntungan:

   Keterandalan terbesar diantara topologi yang lain, mudah dikembangkan, keamanan data tinggi, dan kemudahan akses ke jaringan LAN lain.

2. Kerugian:

   Lalu lintas yang pada dapat menyebabkan jaringan lambat dan jaringan tergantung pada terminal pusat (dapat berupa komputer PC, mini atau

mainframe) yang merupakan bagian paling bertanggung jawab terhadap pengaturan arah semua informasi ke terminal yang dikehendaki.

2.2.15 Arsitektur Jaringan Komputer *client-server*

Arsitektur jaringan *Client Server* merupakan model konektivitas pada jaringan yang membedakan fungsi komputer sebagai *Client* dan *Server*. Arsitektur ini menempatkan sebuah komputer sebagai Server. Server yang bertugas memberikan pelayanan kepada terminal-terminal lainnya tang terhubung dalam *system* jaringan atau yang dsebut *Client*. *Server* juga dapat bertugas untuk memberikan layanan berbagi pakai berkas (*file server*), *printer* (*printer server*), jalur komunikasi (*server* komunikasi).

Pada model arsitektur ini, *Client* tidak dapat berfungsi sebagai *Server*, tetapi *Server* dapat berfungsi menjadi *Client* (*server non-dedicated*). Prinsip kerja pada arsitektur ini sangat sederhana, dimana *Server* akan menunggu permintaan dari *Client*, memproses dan memberikan hasil kepada *Client*, sedangkan *Client* akan mengirimkan permintaan ke *Server*, menunggu proses dan melihat visualisasi hasil prosesnya.

Sistem *Client Server* ini tidak hanya diperuntukkan bagi pembangunan jaringan komputer skala luas. Sistem ini menggunakan protokol utama *Transmission Control Protocol/Internet Protocol (TCP/IP)*, sedangkan sistem operasi yang digunakan antara lain Unix, Linux dan Windows NT. Lingkungan *Database Client/Server* dapat menggunakan menggunakan LAN untuk mendukung jaringan PC. Masing-masing PC memiliki penyimpan tersendiri.

Pada dasarnya *Client Server* terdiri dari 3 komponen pembentuk dasar, yaitu *Client, Middleware, dan Server*. Gabungan dari ketiganya dapat dilihat pada Gambar 2-7.



Gambar 2-7 Komponen *Client-Server*

Aktivitas pada *server*:

a) Semua pemrosesan dilakukan pada sisi *workstation*
b) Satu atau beberapa *server* terhubungkan dalam jaringan
c) *Server* bertindak sebagai *file server*
d) *File server* bertindak sebagai pengelola *file* dan memungkinkan klien mengakses *file* tersebut
e) Setiap klien dilengkapi DBMS (*Database Management System)* tersendiri
f) DBMS berinteraksi dengan data yang tersimpan dalam bentuk *file* pada server

Aktivitas pada klien:

a) Meminta data
b) Meminta penguncian data
c) Tanggapan dari klien
d) Memberikan data
e) Mengunci data dan memberikan statusnya

2.2.16 Pengujian Perangkat Lunak

Pengujian perangkat lunak merupakan proses menelusuri dan mempelajari sebuah program dalm rangka menemukan kesalahan pada perangkat lunak sebelum diserahkan kepada pengguna.

Tujuan pengujian perangkat lunak:

1. Pengujian adalah proses menjalankan program dengan maksud untuk mencari kesalahan (*error*).
2. Kasus uji yang baik adalah kasus yang memiliki peluang untuk mendapatkan kesalahan yang belum diketahui sebelumnya.
3. Pengujian dikatakan berhasil bila dapat memunculkan kesalahan yang susah untuk ditemukan.
4. Pengujian yang baik nukan untuk memastikan tidak ada kesalahan tetapi untuk mencari sebanyak mungkin kesalahan yang ada diprogram.

Pelaku pengujian perangkat lunak sebaiknya *tester independent* karena jika pelaku pengujian merupakan bagian dari tim *developer* akan lebih memahami sistem, dan melakukan pengujian secara subjektif. *Tester Independent* merupakan pelaku pengujian yang tidak memahami sistem, kreatif mencari kesalahan, mencari kesalahan secara objektif.

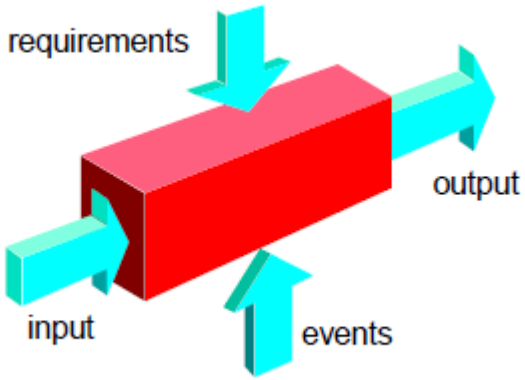
Selain menentukan pelaku pengujian sistem juga dibutuhkan strategi pengujian perangkat lunak. terdapat 2 strategi pengujian perangkat lunak, yaitu:

1. Pengujian *Alpha*
    - Pengujian dilakukan dilingkungan *developer*
    - *User* didampingi *developer*
    - Hasil pengujin diisi *developer*
    - *End user* dipilih oleh *developer*
2. Pengujian *Beta*
    - Pelaku pengujian berasal dari lingkunan *user*
    - *User* menguji sendiri
    - *Feedback online*
    - Perangkat lunak uji disebar

Selain strategi pengujian, ada juga metode pengujian. *Blackbox* merupakan salah satu metode pengujian. Pengujian *blackbox* dilakukan untuk menghindari:

1. Fungsi tidak benar atau hilang
2. Kesalahan antarmuka
3. Kesalahan pada pengaksesan basis data
4. Kesalahan inisialisasi dan akhir program
5. Kesalaham performasi

Ilustrasi pengujian *blackbox* dapat dilihat pada Gambar 2-8.



Gambar 2-8 Ilustrasi pengujian *Black Box*