BAB II

TINJAUAN PUSTAKA

II.1. Sistem

Secara sederhana, suatu sistem dapat diartikan sebagai suatu kumpulan atau himpunan dari unsur, komponen, atau *variable* yang terorganisir, saling berinteraksi, saling tergantung satu sama lain dan terpadu. Selain itu suatu sistem tidak bisa lepas dari lingkungan sekitarnya maka umpan balik (*feed back*) dapat berasal dari lingkungan sistem yang dimaksud. (Tata Sutabri : 2012 : 10).

II.1.1. Karakteristik Sistem

Adapun karakteristik yang dimaksud adalah sebagai berikut :

1. Komponen Sistem *(Components).*

   Suatu sistem terdiri dari sejumlah komponen yang saling berinteraksi, artinya saling bekerja sama membentuk satu kesatuan.

2. Batasan Sistem *(Boundary)*

   Ruang lingkup sistem merupakan daerah yang membatasi antara sistem dengan sistem yang lain atau sistem dengan lingkungan luarnya.

3. Lingkungan Luar Sistem *(Environtment)*

   Bentuk apapun yang ada diluar ruang lingkup atau batasan sistem yang mempengaruhi operasi sistem tersebut disebut lingkungan luar sistem.

4. Penghubung Sistem *(Interface)*

   Media yang menghubungkan sistem dengan subsistem lain disebut penghubung sistem atau interface. Penghubung ini memungkinkan sumber – sumber daya mengalir dari satu subsistem ke subsistem lain.

5. Masukan Sistem *(Input)*

   Energi yang dimasukan ke dalam sistem disebut masukan sistem, yang dapat berupa pemeliharaan *(maintenance input)* dan *signal (signal input).*

6. Keluaran Sistem *(output)*

   Hasil energi yang diolah dan diklasifikasikan menjadi keluaran yang berguna. Keluaran ini merupakan masukan bagi subsistem yang lain.

7. Pengolah sistem *(Proses)*

   Sudah sistem dapat mempunyai suatu proses yang akan mengubah masukan menjadi kelauran.

8. Sasaran Sistem *(Objective)*

   Suatu sistem memiliki tujuan dan sasaran yang pasti dan bersifat determinisik. Kalau suatu sistem tidak memiliki sasaran, maka operasi sistem tidak ada gunanya. Suatu sistem diaktakan berhasil bila mengenai sasaran atau tujuan yang telah direncanakan. (Tata Sutabri : 2012 : 21).

II.1.2.Informasi

Informasi adalah data yang telah diklasifikasikan atau diolah atau diinterpretasikan untuk digunakan dalam proses pengambilan keputusan. Informasi dapat mengenai data mentah, data tersusun, kapasitas sebuah saluran komunikasi dan lain sebagainya. (Tata Sutabri : 2012 : 29).

Kualitas informasi tergantung dari 3 hal yaitu :

1. Akurat *(Accurate)*

   Informasi harus bebas dari kesalahan – kesalahan dan tidak menyesatkan.

2. Tepat waktu *(Timeline)*

   Informasi yang datang pada si penerima tidak boleh terlambat. Informasi yang sudah usang tidak akan mempunyai nilai lagi karena informasi merupakan landasan dalam pengambilan keputusan.

3. Relevan *(Relevance)*

   Informasi tersebut mempunyai manfaat untuk pemakainya. Relevansi informasi utnuk orang satu dengan yang lain berbeda, misalnya informasi sebab kerusakan mesin produksi kepada akuntan perusahaan adalah kurang relevan dan akan lebih relevan dan akan lebih relevan bila ditunjukan ahli teknik perusahaan. (Tata Sutabri : 2012 : 41).

II.1.3. Sistem Informasi

Sistem informasi adalah suatu sistem didalam suatu organisasi yang mempertemukan kebutuhan pengolahan transaksi harian yang mendukung fungsi operasi organisasi yang bersifat manajerial dengan kegiatan strategi dari suatu organisasi untuk dapat menyediakan kepada pihak luar dengan laporan-laporan yang diperlukan (Tata Sutabri : 2012 : 46). Sistem informasi terdiri dari komponen – komponen yang disebut blok bangunan *(building block)* yaitu :

1. Blok masukan (*input block*)

   Input mewakili data yang masuk kedalam sistem informasi, *input* disini termasuk metode dan media untuk menangkap data yang akan dimasukkan, yang dapat berupa dokumen – dokumen dasar.

2. Blok model (*model block*)

   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 di inginkan.

3. Blok keluaran (*output block*)

   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 (*technology block*)

   Teknologi merupakan "*tool box*" dalam sistem informasi. Teknologi digunakan untuk menerima *input*, menjalankan model, menyimpan dan mengakses data, menghasilkan dan mengirimkan keluaran, teknologi terdiri dari 3 bagian utama, yaitu teknisi (*brainware*), Perangkat lunak (*software*), perangkat keras (*hardware*).

5. Blok basis data (*database block)*

   Basis data atau *database* merupakan kumpulan data yang saling berkaitan dan berhubungan satu dengan yang lain, tersimpan di perangkat keras komputer an menggunakan perangkat lunak untuk memanipulasinya.

6. Blok kendali *(control block)*

   Banyak hal yang dapat merusak sistem informasi, seperti bencana alam, api, temperatur, air ,debu, kecurangan – kecurangan, kegagalan – kegagalan. (Tata Sutabri : 2012 : 47-48).

II.2. Sistem Penggajian

Analisa sistem pengolahan data gaji pegawai terdiri dari analisa *input*, analisa proses, dan analisa *output*. Adapun uraian masing-masingnya adalah sebagai berikut :

1. Analisa *Input*

   Input dari pengolahan data gaji pegawai adalah Surat Kerja (SK) dan data identitas pegawai. Semua data yang ada pada Surat Kerja dan data identitas pegawai akan dicatat oleh bendahara gaji untuk dijadikan data pegawai yang akan di olah untuk informasi data pembayaran gaji pegawai.

2. Analisa Proses

   Proses perhitungan gaji, Jumlah gaji kotor di dapat dari :

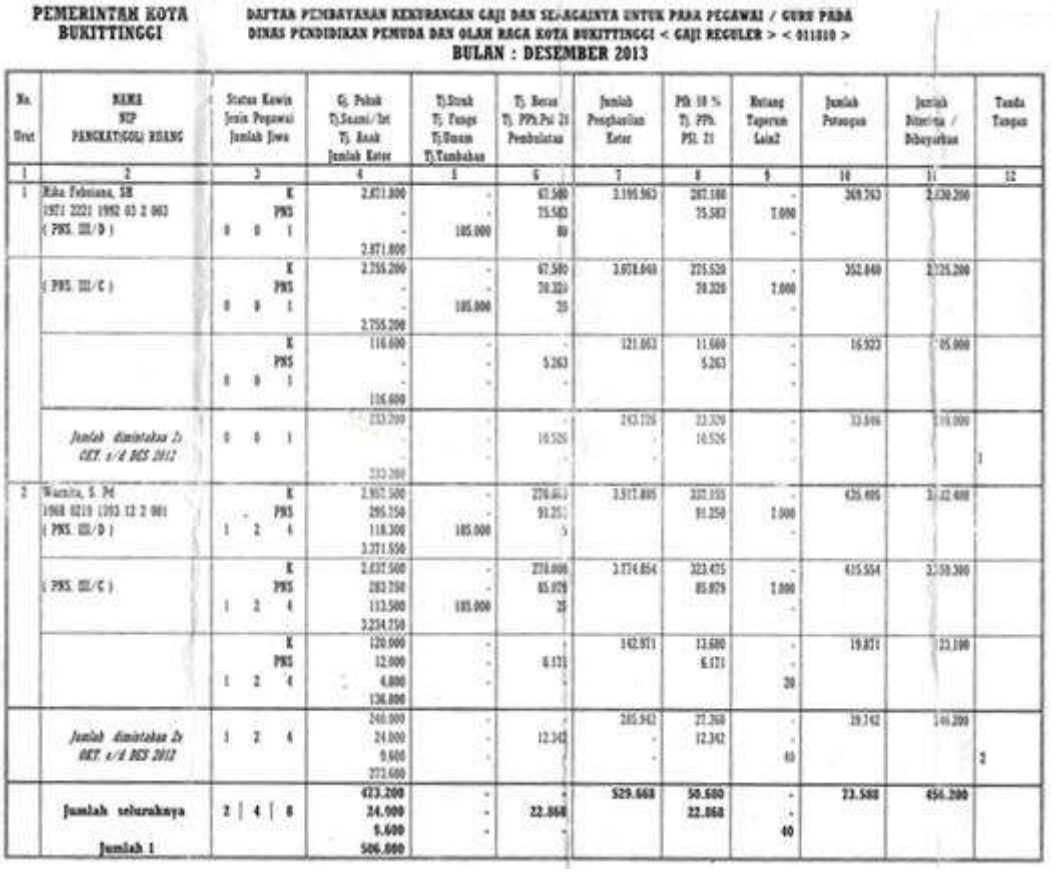
   Gaji pokok + tunjangan Istri/ suami + Tunjangan anak + tunjangan PPH + Tunjangan umum + Tunjangan Beras + tunjangan Struktural + tunjangan Fungsional + Tunjangan tambahan + pembulatan. Dimana gaji pokok di dapat berdasarkan lama atau masa kerja kemudian tunjangan istri/suami 10% dari gaji pokok, tunjangan anak 2% dari gaji pokok sedangkan PPh di dapat berdasarkan ketentuan pemerintah. Perhitungan tersebut merupakan perhitungan yang sebenarnya yang di ambil waktu obserfasi.

3. Analisa *Output*

   Sebagaimana yang telah di jelaskan pada aliran sistem informasi yang lama, pengolahan data gaji pegawai terdapat beberapa *output* yang diantaranya adalah slip gaji yang di terima oleh pegawai. Berdasarkan slip gaji, perhitungan atau proses yang harus dilakukan harus baik sehingga didapatkan hasil yang lebih baik dan waktu yang cepat untuk mendapatkan slip tersebut.

Pada saat ini untuk mengisi slip gaji tersebut masih dilakukan dengan cara manual sehingga membutuhkan waktu yang lama.

Selain itu bagian keuangan mencetak amprah gaji yang akan diberikan nantinya kepada kepala dinas. (Jurnal Rafki Hamdani:2010:4-5).



PEMERINTAH KOTA BUKITTINGGI

DAFTAR PEMBAYARAN KEKURANGAN GAJI DAN SEBAGAINYA UNTUK PARA PEGAWAI / GURU PADA DINAS PENDIDIKAN PEMUDA DAN OLAH RAGA KOTA BUKITTINGGI < GAJI REGULER > < 011810 >

BULAN : DESEMBER 2013

No. Urut	NAMA NIP PANGKAT/GOL RUANG	Status Kawin Jenis Pegawai Jumlah Jiwa	Gj. Pokok Tj.Suami/Ist Tj. Anak Jumlah Kotor	Tj.Struk Tj. Fungs Tj.Umum Tj.Tambahan	Tj. Beras Tj. PPh.Pst 21 Pembulatan	Jumlah Penghasilan Kotor	Pth 10 % Tj. PPh. PSL 21	Hutang Taperum Lain2	Jumlah Potongan	Jumlah Diterima / Dibayarkan	Tanda Tangan
1	2	3	4	5	6	7	8	9	10	11	12
1	Rika Febriana, SE 1971 2221 1992 03 2 003 (PNS. III/D)	K PNS 0 0 1	2.871.800 - - 2.871.800	- - 185.000	83.580 75.580 80	3.199.563	287.180 75.580 -	- 7.000 -	369.763	2.829.800	
	(PNS. III/C)	K PNS 0 0 1	2.755.200 - - 2.755.200	- - 185.000	67.580 70.320 20	3.078.840	276.520 70.320 -	- 7.000 -	352.840	2.726.200	
		K PNS 0 0 1	116.600 - - 116.600	- - -	- 5.263 -	121.863	11.660 5.263 -	- - -	16.923	95.000	
	Jumlah dimintakan Dr GET. s/d BES 2012	0 0 1	232.200 - - 232.200	- - -	- 10.526 -	243.726	23.320 10.526 -	- - -	33.846	191.000	1
2	Wanita, S. Pd 1968 0219 1993 12 2 001 (PNS. III/D)	K PNS 1 2 4	2.907.500 296.250 118.300 3.371.550	- - 185.000	276.663 91.251 5	3.917.895	392.555 91.250 -	- 7.000 -	476.805	3.442.600	
	(PNS. III/C)	K PNS 1 2 4	2.637.500 282.250 113.500 3.234.250	- - 185.000	270.000 85.979 25	3.774.854	323.425 85.979 -	- 7.000 -	415.554	3.359.300	
		K PNS 1 2 4	120.000 12.000 4.800 136.800	- - -	- 6.171 -	142.971	12.680 6.171 -	- - 20	19.871	123.100	
	Jumlah dimintakan Dr GET. s/d BES 2012	1 2 4	240.000 24.000 9.600 273.600	- - -	- 12.342 -	285.942	27.360 12.342 -	- - 40	39.742	246.200	2
	Jumlah seluruhnya **Jumlah 1**	2 | 4 | 8	**423.200** **24.000** **9.600** **506.800**	- - -	- **22.868** -	**529.668**	**50.680** **22.868** -	- - 40	**73.588**	**456.200**	

Gambar II.1. Amprah Gaji
Sumber:(Jurnal Rafki Hamdani:2010:5).

II.3. Java

II.3.1.Tentang Java

Bahasa pemograman Java pertama, yang dinamakan juga dengan Oak, lahir dari *The Green Project*, yang berjalan mulai awal tahun 1991 hingga musim panas 1992. Proyek ini dimotori oleh Patrick Naughton, Mike Sheridan, James Gosling dan Bill Joy berserta sembilan programmer lainnya dari *Sun*

Microsystems. Proyek ini ditutup dengan menghasilkan sebuah program *Java Oak* pertama, yang diperuntukkan sebagai pengkontrol sebuah peralatan dengan teknologi *touch screen*, seperti pada *PDA* sekarang ini. Teknologi baru ini dinamakan *Star Seven* (*7). Era setelah *Star Seven* selesai, sebuah anak perusahaan TV kabel bekerjasama dengan beberapa orang dari *The Green Project*. Dalam rentang waktu yang singkat ditetapkan bahwa Internet merupakan medium yang menjembatani kerja dan ide diantara mereka. Mereka menjadikan *webbrowser Mosaic*, sebagai landasan awal untuk membuat *Java browser* pertama yang dinamakan ”*Web Runner*”. Perkembangan *release* pertama, *Web Runner* berganti nama menjadi *Hot Java*. Kesuksesan mereka diikuti dengan untuk pertama kali diberitakan dalam surat kabar *Sun Jose Mercury News* pada tanggal 23 Mei 1995. Sayangnya terjadi perpecahan diantara mereka, tiga dari pimpinan utama proyek Eric Schmidt dan George Paolini dari Sun Microsystems bersama Marc Andreessen membentuk *Netscape*. Nama *Oak*, diambil dari pohon *oak* yang tumbuh di depan jendela ruangan kerja ”bapak *Java*”, James Gosling. Nama oak ini tidak dipakai untuk versi *release Java* karena sudah terdaftar *softwar*e dengan merek dagang itu, sehingga diambil nama penggantinya menjadi ”*Java*”, diambil dari nama kopi tubruk kesukaan bapak Gosling.

Java merupakan pemrograman berorientasi objek. Oleh karena itu, setiap konsep yang akan diimplementasikan dalam *Java* berbentuk dalam kelas. Kelas ini mendefinisikan objek-objek yang memiliki kesamaan perilaku dan keadaan. Pada *Java* terdapat kumpulan kelas standar yang dikenal dengan *Application Programming Interface* (API) *Java*, selain itu dapat juga dideskripsikan kelas

sendiri sesuai kebutuhan. *Java* mengadopsi konsep *smart client,* konsep ini berbicara tentang kemampuan *client* untuk meminta dan menangkap pesan dari *server*. Dalam proses meminta dan menangkap itu. *Client* melakukan proses validasi dan verifikasi data.

Salah satu kelebihan *Java* yang paling signifikan adalah *run everywhere*. Dengan kelebihan ini, para *developer* yang sudah terbiasa mengembangkan aplikasi dalam bingkai kerja *J2SE* dan *J2EE*, akan mampu bermigrasi dengan mudah untuk mengembangkan aplikasi *J2ME*. (Tri Mardiono:2006:9-10).

Pada dasarnya, terdapat tiga garis besar kelompok program yang dapat dibuat menggunakan bahasa *java (Standard Edition),* yaitu:

1. *Applet* sering disebut sebagai aplikasi mini. *Appplet* dijalankan oleh *browser* WWW (*Word Wide Web*).
2. Apikasi *java* (biasa disebut "aplikasi"). Aplikasi ini dapat dibedakan menjadi:
    a. Aplikasi *GUI (Graphical User Interface)* yaitu aplikasi yang tampilannya memakai grafik.
    b. Aplikasi *command-line.* Yaitu aplikasi yang tampilannya sebatas teks, yang dijalankan dan ditampilkan di dalam *command-prompt.*
3. *Package* atau disebut juga *library java. Package* sebdiri tidak untuk dijalankan, ia hanya menyediakan sekumpulan *class-class java* yang dianggap berguna dan umum untuk dipakai kembali.

II.3.2.Membuat Apliksi *Java*

Berikut adalah langkah-langkah tipikal untuk membuat aplikasi *java:*

1. Membuat *teks* program (*source code*) dengan memakai *syntax* bahasa *java*.
2. Meng-*compile* teks program menjadi aplikasi *java*.

3. Menjalankan aplikasi *java* dengan *interpreter* atau dengan *applet-viewer* jika berbentuk *applet.*

Selain langkah-langkah tersebut, kita juga dapat melakukan hal berikut:

1. Menambahkan dokumentasi ke dalam aplikasi *java.*
2. Melakukan *dubbing* untuk melihat jalannya aplikasi secara *step-by-step* ataupun untuk mencari letak kesalahan program.
3. Mengumpulkan beberapa *file output* ***(.class)*** menjadi satu *file java-archive* ***(.jar)***. (Matius Soesilo:2005:13).

II.3.3 Struktur Leksikal dari Bahasa *Java*

Dalam pemrograman *Java* terdapat beberapa macam struktur leksikal dari bahasa *Java* yaitu :

1. Statement

Statement atau pernyataan adalah baris-baris instruksi dalamprogram *java.* Setiap *statement* diakhiri dengan tanda titik koma (;).

2. *Variable* atau *Field*

Field adalah atribut dari sebual *class*. Sedangkan istilah *variable* (dalam *java*) mencakup *field, local-variable, componen array,* dan *parameter.* Variabel merupakan sebuah lokasi dalam memori yang nilainya dapat diubah oleh program, sebuah variabel mempunyai nama, tipe data, dan nilai.

3. Tipe *Variabel*

Adapun tipe *variabel* diantaranya tipe *primitive (tipe numeric, boolean),* dan tipe *reference (*tipe *class,* tipe *interface* dan tipe *array*).

4. Type-Casting

Type-casting adalah proses konversi dari suatu tipe ke tipe lainnya. Misalnya dari tipe *int* ke tipe *long.*

5. *Input-Element* dan *Token*

Element dasar dari sebuah program *java* disebut *input-element*. *Input-element* terdiri dari *white-space, comment* dan *token.*

6. Comment

Comment dipakai dalam teks *source-code* untuk memberikan keterangan atau catatan untuk baris program atau blok program tertentu. *Comment* sangat penting untuk memberikan penjelasan mengenai cara kerja program.

7. Operator

Operator adalah *token* yang dipakai untuk melakukan berbagai macam operasi dalam program *java.* Ada berbagai jenis operator dalam *java,* diantaranya operator aritmatika. (Matius Soesilo:2005:117-131).

Tabel.II.1 Daftar Operator Aritmatika dalam *Java*
Sumber:(Matius Soesilo:2005:131).

Operator	Jenis Operasi	Contoh
+	Addition (penjumlahan)	a+b
-	Subtraction (pengurangan)	a-b
*	Multiplication (perkalian)	a*b
/	Division (pembagian)	a/b
%	Modulus (sisa pembagian)	a%b

II.3.4. *Java 2 Standard Edition* (J2SE)

Menurut M. Shalahuddin dan Rosa A.S. (2006, p4), *Java 2 Standard Edition* (J2SE) adalah inti dari bahasa pemograman Java. *Java Development Kit* (JDK) adalah salah satu *tool* dari J2SE untuk mengkompilasi dan menjalankan

program *Java*. Di dalamnya terdapat *tool* untuk mengkompilasi program *Java* dan JRE.

II.3.5 *Java 2 Micro Edition* (J2ME)

Java 2 Micro Edition (*J2ME*) adalah lingkungan pengembangan yang didesain untuk meletakkan perangkat lunak *Java* pada barang elektronik beserta perangkat pendukungnya. *J2ME* membawa *Java* ke dunia informasi, komunikasi dan perangkat komputasi selain perangkat komputer *desktop*, yang biasanya lebih kecil. *J2ME* bisa digunakan pada *handphone, peger, Personal Digital Assistants* (*PDA*) dan sejenisnya. Selain itu, *J2ME* mengadopsi konsep *smart client*. Konsep ini berbicara tentang kemampuan *client* untuk meminta dan menangkap pesan dari *server*. Dalam proses meminta dan menangkap itu, client melakukan proses-proses validasi dan verifikasi data. (Tri Mardiono:2006:9).

2.3.6 *Java 2 Enterprise Edition* (*J2EE*)

J2EE adalah kombinasi dari beberapa teknologi yang menawarkan suatu keterpaduan untuk menyatukan sistem di sisi *server*. Berikut ini adalah beberapa fungsi dari *J2EE*, yaitu :

1. Mendukung *HTML*, baik sebagai *Java Applet* maupun sebagai aplikasi, yang disokong oleh *Java Server Page* dan kode *Servlet* untuk membentuk *HTML* dan data terformat lainnya untuk *client*.
2. *Servlet* merupakan sebuah program sederhana yang berjalan di *server*. Istilah s*ervlet* juga ditujukan untuk aplikasi yang berjalan di dalam lingkungan *Web Server*, yang dapat disamakan dengan *Java Applet* yang berjalan di dalam lingkungan *Web Browser*.

3. *Enterprise Javabeans (EJBs) EJB server* dapat berfungsi sebagai keamanan dan manajemen memori.
4. *Java Database Connectivity* (JDBC) adalah *interface* untuk database *Java*.
5. *Java Servlet API : java servlet API* dapat meningkatkan konsistensi para pengembang tanpa memerlukan grafis *interface* pemakai.

II.4. MySQL

MySQL adalah salah satu jenis *database server* yang sangat terkenal. Kepopulerannya desebabkan *MySQL* menggunakan *SQL* sebagai bahasa dasar untuk mengakses *database*nya. Selain itu, ia bersifat *Open Source* (Anda tidak perlu membayar untuk menggunakannya. Database merupakan kumpulan data yang didalamnya terdapat tabel-tabel. Jika kita berbicara mengenai database, sebenarnya mengacu pada bentuk data relational yang terdiri dari baris (row/record) dan kolom (colomn/field). Perangkat lunak PHP mendukung perangkat lunak database konvensional hingga database modern. Database MYSQL tergolong database server, PHP sangat serasi denga server web apache dan database MYSQL. (Ir.Yuniar Supardi :2010:154).

Database digunakan untuk penyimpanan data, demikian pula dengan *MYSQL*. Kita akan memanggil data pada *MYSQL* melalui *PHP*, kemudian hasilnya dikirim kekomputer *klien* untuk ditampilkan pada *browser*. Data pada *MYSQL* dapat dipanggil, dihapus atau di tambah melalui *query* .(Dadan Sutisna : 2008 : 46).

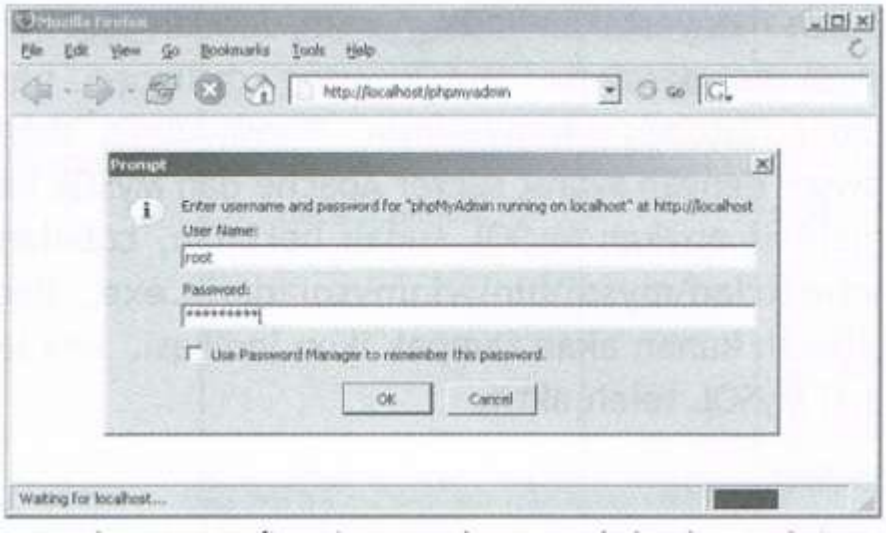
Langkah untuk menjalankan *webserver local* adalah :

1. Buka program *browser*, misalnya ***Mozilla Firefox*** atau ***Internet Explorer*.**
2. Dalam jendela *browser*, ketil ***localhost*** dibagian ***Address***, kemudian tekan ***Enter***. Jika proses instalasi benar maka akan tampil halaman *localhost* dapat dilihat pada gambar II.2.

Gambar II.2. Halaman Awal Localhost

Sumber : (Dadan Sutisna : 2008 : 57)

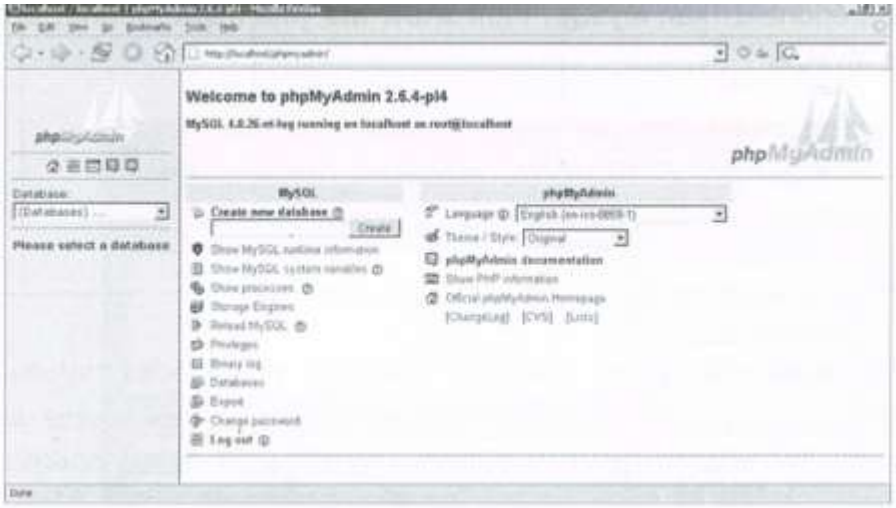
3. Gambar II.2 juga menampilkan informasi 4 komponen pendukung *webserver* yang telah terinstal pada komputer sekaligus menampilkan versi yang digunakan.
4. Berikutnya adalah membuka halaman *phpMyAdmin*, klik *link* ***phpMyAdmin Database Manager Version 2.10.2.***
5. Berikutnya program akan meminta memasukkan *user* dan *password* dari s*erver MySQL*. Ketik *user* nama ***root*** pada kotak ***User Name,*** dan masukkan *password*-nya pada kotak ***Password*** sesuai dengan *password* pada saat penginstalan. Tampilan konfirmasi permintaan *username* dan *password server MySQ*L dapat dilihat pada gambar II.3.



Gambar II.3. Konfirmasi Permintaan Username dan Password Server MySQL

Sumber : (Dadan Sutisna : 2008 : 58)

Jika *username* dan *password* yang dimasukkan benar, maka tampil halaman *phpMyAdmin* tampak pada gambar II.4.



Gambar II.4. Tampilan Halaman PhpMyAdmin

Sumber : (Dadan Sutisna : 2008 : 58)

II.5. Sistem Basis Data

Basis adalah markas, gudang, tempat bersarang tau berkumpul. Sedangkan data adalah representasi fakta dunia nyata yang mewakili suatu objek seperti manusia (dosen, mahasiswa, pembeli, pelanggan), barang, hewan, peristiwa, konsep, keadaan dan sebagainya yang direkam dalam bentuk angka, huruf, simbol, teks, gambar, bunyi atau kombinasinya. Basis data adalah kumpulan data yang saling berhubungan yang disimpan/diorganisasi secara bersama, dalam bentuk sedemikian rupa, dan tanpa redudansi (pengulangan) yang tidak perlu supaya dapat dimanfaatkan kembali dengan cepat dan mudah untuk memenuhi berbagai kebutuhan. (Ema Utami dan Anggit Dwi Hartanto ; 2012 : 3).

Komponen-komponen pada sebuah sistem basis data antara lain:

1. Perangkat keras
2. Sistem operasi
3. Basis data
4. DBMS *(Database Management System)*
5. Pemakai
6. Aplikasi lain

DBMS merupakan perangkat lunak yang dirancang untuk dapat melakukan utilisasi dan mengelola koleksi data dalam jumlah yang besar. DBMS juga dirancang untuk dapat melakukan manipulasi data secara lebih mudah. DBMS merupakan antarmuka pengguna basis data (baik merupakan pengguna langsung maupun aplikasi) dengan data yang tersimpan. Penyimpanan data oleh DBMS disesuaikan dengan bentuk model datanya, beberapa contoh DBMS adalah *PostgreSQL, MYSQL, DB2, Oracle, SQLserver* dan lain lain. Sebelum ada

DBMS, data pada umumnya disimpan dalam bentuk *flat file*, yaitu *file teks* yang ada pada sistem operasi. Sampai sekarang pun masih banyak aplikasi yang menyimpan dalam bentuk *flat file* secara langsung. Contoh yang sederhana adalah misalnya dalam sistem operasi linux terdapat *file password* yang digunakan untuk menyimpan nama penggunanya. (Ema Utami dan Anggit Dwi Hartanto ; 2012 : 5).

II.6. Alat Bantu Pengembangan Sistem

Alat bantu pengembangan sistem yang digunakan dalam penulisan tugas akhir ini adalah sebagai berikut :

II.6.1. Konsep UML (*Unified Modelling Language*)

Unified Modelling Language (UML) merupakan alat komunikasi yang konsisten dalam mensupport para pengembang sistem saat ini. Sebagai perancang sistem, mau tidak mau pasti akan menjumpai *UML*, baik kita sendiri yang membuat atau sekedar membaca diagram *UML* buatan orang lain. (Prabowo Pudjo Widodo ; 2011 : 7). Sebelum ada *UML*, para pengembang bahasa pemrograman berorientasi objek sulit untuk berkomunikasi satu sama lain. Ada kira-kira 50 jenis notasi dan grafik yang menggambarkan bahasa pemrograman berorientasi objek pada waktu itu. Para pengguna notasi yang berlainan ini saling berebut pengaruh agar notasi yang mereka gunakan jadi notasi standar. (Prabowo Pudjo Widodo ; 2011 : 8).

Sejak tahun 1997, divisi *Revision Task Force (RTF)* milik *OMG* beberapa kali merevisi *UML*. Revisi dimaksudkan untuk memperkuat konsistensi notasi, meningkatkan kekompakan antara *user* dan pengembang perangkat lunak. Akan tetapi *UML* terpaksa mengikuti perkembangan *software-software* berbasisi objek

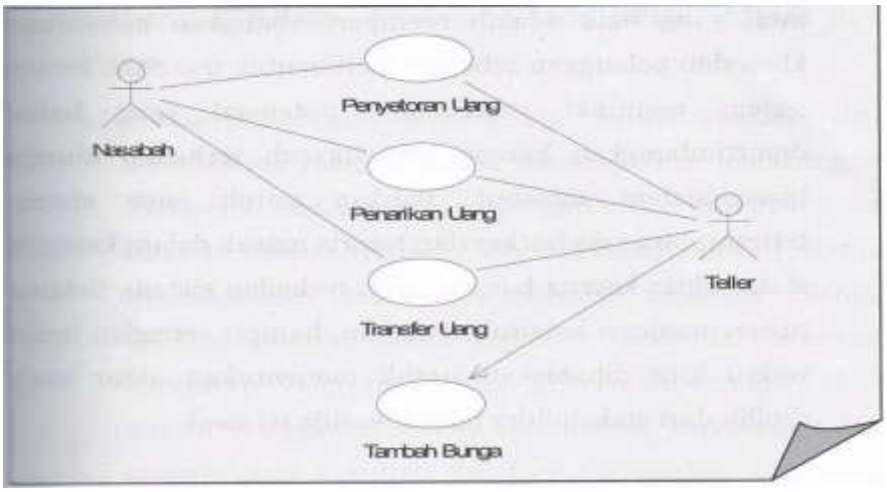
yang ada. Setelah dilakukan perubahan secara sistematik, akhirnya dihasilkan *UML 2.0* pada tahun 2003. (Prabowo Pudjo Widodo ; 2011 : 9).

II.6.2 Diagram – diagram Pada Metode UML

1. Use Case Diagram

   Menurut (Philone:2005:bab 7.1) *use case* adalah menggambarkan fungsi tertentu dalam suatu sistem berupa komponen, kejadian atau kelas. Sedangkan (Whitten:2004:258) mengartikan use case sebagai urutan langkah-langkah yang secara tindakan saling terkait, baik terotomatisasi maupun secara manual, untuk tujuan melengkapi satu tugas bisnis tunggal. (Prabowo Pudjo Widodo ; 2011 : 21).

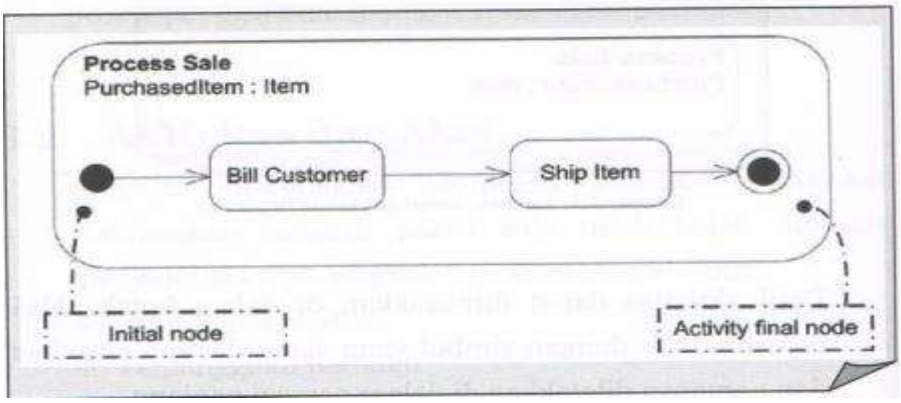
   Notasi *use case* menunjukkan 3 aspek dari sistem yaitu *actor use case* dan *system / sub system boundary*. *Actor* mewakili peran orang, *system* yang lain atau alat ketika berkomunikasi dengan *use case*. Ilustrasi *actor, use case* dan *system* ditunjukkan pada gambar II.5.



Gambar II.5. *Use Case* Diagram
Sumber : (Prabowo Pudjo Widodo ; 2011 : 17)

2. Activity diagram

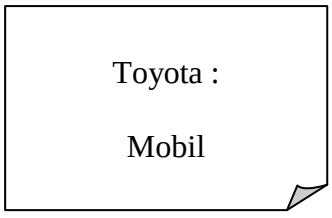
Activity diagram lebih memfokuskan diri pada eksekusi dan alur sistem dari pada bagaimana sistem itu dirakit. Diagram ini tidak hanya memodelkan *software* melainkan memodelkan bisnis juga. Diagram aktivitas menunjukkan aktivitas dalam bentuk kumpulan aksi-aksi. (Prabowo Pudjo Widodo ; 2011 : 143). Berikut gambar dari sederhana dari *activity diagram*.



Gambar II.6. Contoh Activity Diagram Sederhana
Sumber : (Prabowo Pudjo Widodo ; 2011 : 146)

3. Class Diagram

Diagram kelas atau class diagram adalah inti dari proses pemodelan objek baik *forward engineering* maupun *reverse engineering* memanfaatkan diagram ini. *Forward engineering* adalah proses perubahan model menjadi kode program sedangkan *reverse engineering* sebaliknya merubah kode program menjadi model. (Prabowo Pudjo Widodo :2011 : 37).



Gambar II.7. Contoh Activity Diagram Sederhana
Sumber : (Prabowo Pudjo Widodo ; 2011 : 41)

4. *Squence Diagram*

   Squence Diagram digunakan untuk menggambarkan perilaku pada sebuah sekenario. Diagram ini menunjukkan sebuah contoh objek dan pesan yang diletakkan diantara objek-objek ini didalam *use case*.

   Komponen utama *Squence diagram* terdiri dari atas objek yang dituliskan dengan kotak segiempat bernama. *Messege* diwakili oleh garis dengan tanda panah dan waktu yang ditunjukkan dengan *progress vertical.*

II.7. Normalisasi

Normalisasi merupakan proses pengelompokan elemen data menjadi tabel yang menunjukkan entitas sekaligus relasinya. (Ema Utami dan Anggit Dwi Hartanto ; 2012 : 40).

Tahap normalisasi terdiri dari beberapa bentuk :

1. Bentuk Tidak Normal (*Unnormalized Form*)

   Bentuk ini merupakan kumpulan data yang akan direkam, tidak ada keharusan mengikuti suatu format tertentu, dapat saja data tidak lengkap atau terduplikasi. Data dikumpulkan apa adanya sesuai dengan kedatangannya.

2. Bentuk Normal Kesatu (1NF/ *First Normal Form*)

   Bentuk normal kesatu mempunya ciri : setiap data dibentuk dalam *file file* (*file* datar/rata), data dibentuk dalam satu *record* demi *record* dan nilai dari *field* berupa ”atomic value”. Tidak ada set atribut yang berulang atau atribut bernilai ganda (*multivalue*). Tiap *field* hanya satu pengertian, bukan merupakan kumpulan kata yang mempunyai arti mendua, hanya satu arti saja

dan juga bukan pecahan kata sehingga artinya lain. Atom adalah zat terkecil yang masih memiliki sifat induknya, bila dipecah lagi, maka ia tidak memiliki sifat induknya.

3. Bentuk Normal Kedua (2NF/ *Second Normal Form*)

   Bentuk normal kedua memiliki syarat : bentuk data telah memenuhi kriteria bentuk normal kesatu. Atribut bukan kunci haruslah bergantung fungsi pada kunci utama/*primary key* sehingga untuk membentuk normal kedua haruslah sudah ditentukan kunci *field*. Kunci *field* haruslah unik dan dapat mewakili atribut lain yang menjadi anggotanya. Tahap normalisasi terdiri dari beberapa bentuk:

4. Bentuk Normal Ketiga (3NF/ *Third Normal Form*)

   Untuk menjadi bentuk normal ketiga, relasi haruslah dalam bentuk normal kedua dan semua atriut dalam primer tidak punya hubungan yang transif. Dengan kata lain, setiap atribut bukan kunci haruslah bergantung hanya pada *primary key* dan pada *primary key* secara menyeluruh. Contoh pada bentuk kedua di atas termasuk juga bentuk normal ketiga seluruh atribut yang ada disitu bergantung penuh pada kunci primernya.

5. Bentuk Normal *Boyce Codd* (BCNF/ *Boyce Codd Normal Form*)

   Boyce Codd Normal *Form* mempunyai paksaan yang lebih kuat dari pada bentuk normal ketiga. Untuk menjadi BCNF, relasi harus dalam bentuk normal kesatu dan setiap atribut harus bergantung fungsi pada atribut *superkey*. (Ema Utami dan Anggit Dwi Hartanto ; 2012 : 40).

II.8. *Entity Relationship Diagram* (ERD)

ERD merupakan notasi grafis dalam pemodelan data *konseptual* yang mendeskripsikan hubungan antar penyimpanan. *ERD* juga merupakan gambaran yang menghubungkan antara objek satu dengan objek yang lainnya dalam dunia nyata. (Ema Utami dan Anggit Dwi Hartanto ; 2012 : 18)

ERD menggunakan sejumlah notasi dan simbol untuk menggambarkan struktur dan hubungan antar dua data. Pada dasarnya ada 3 macam simbol yang digunakan, yaitu :

1. *Entity*

   Entity adalah suatu objek yang dapat diidentifikasikan dalam lingkungan pemakai, sesuatu yang penting bagi pemakai dalam konteks sistem yang akan dibuat. Sebagai contoh adalah barang, pemasok, pekerja dan lain-lain.

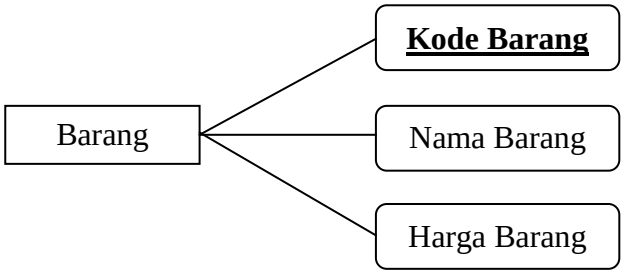
   Seandainya adalah A maka barang A adalah isi dari barang, sedangkan jika B adalah seorang pelanggan maka B adalah isi dari pelanggan. Karena itu harus dibedakan antara entitas sebagai bentuk umum dari deskripsi tertentu dan isi entitas seperti A dan B dalam contoh diatas. Entitas dapat digambarkan dalam bentuk persegi empat.

Gambar II.8. Entitas
Sumber : (Ema Utami dan Anggit Dwi Hartanto ; 2012 : 19)

2. *Atribut*

   Entitas mempunyai elemen yang disebut *atribut* dan berfungsi mendeskripsikan karakter *entitas*, misalnya atribut nama barang dari *entitas*

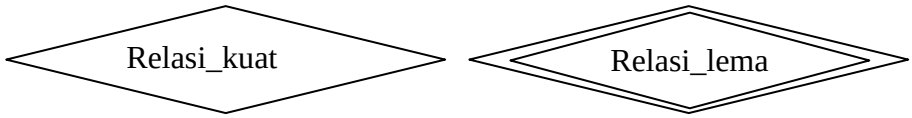
barang. Setiap *ERD* bisa berisi lebih dari satu *atribut*. *Entitas* digambarkan dalam bentuk elips.



Gambar II.9. Atribut
Sumber : (Ema Utami dan Anggit Dwi Hartanto ; 2012 : 20)

3. Hubungan / *relationship*

Belah ketupat merupakan penggambaran hubungan (relasi) antarentitas atau sering disebut kerelasian. Ada dua macam penggambaran relasi yaitu relasi kuat dan relasi lemah. Relasi kuat biasanya menghubungkan antarentitas kuat, sedangkan relasi lemah untuk menghubungkan antara entitas kuat dengan entitas lemah. *Relationship* digambarkan sebagai berikut :



Gambar II.10. *Relationship*
Sumber : ((Ema Utami dan Anggit Dwi Hartanto ; 2012 : 24)

Untuk menghubungkan entitas-kerelasian-entitas digunakan garis lurus eperti gambar berikut ini :

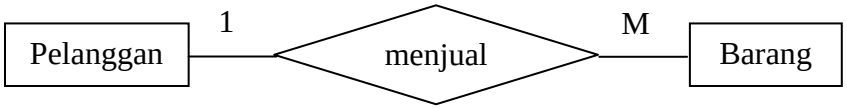
Gambar II.11. Kerelasian antarentitas
Sumber : ((Ema Utami dan Anggit Dwi Hartanto ; 2012 : 24)

Jenis-jenis hubungan:

a. Satu ke satu, misalnya suatu perusahaan mempunyai aturan satu supir hanya boleh menangani satu kendaraan karena alasan tertentu.

Gambar II.12. *Relational* 1 to 1
Sumber : ((Ema Utami dan Anggit Dwi Hartanto ; 2012 : 24)

b. Satu ke banyak atau banyak ke satu, misalya suatu perusahaan selalu berasumsi bahwa satu pelanggan dapat membeli banyak barang



Gambar II.13. *Relational* 1 to *Many*
Sumber : (Ema Utami dan Anggit Dwi Hartanto ; 2012 : 25)