BAB II

LANDASAN TEORI

2.1 Konsep Dasar Program

2.1.1 Program

Menurut Junaedi dalam (Mulyanto & Khasanah, 2018) mengemukakan bahwa "Program adalah serangkaian kode yang dituliskan dalam bahasa pemrograman tertentu, seperti bahasa C, C++, Pascal, Fortran, Java, CGI, Perl, Cobol, ASP (*Active Server Pages*), PHP dan sebagainya, yang umumnya merupakan penjabaran algoritma yang telah dibuat".

Menurut Kadir dalam (Mulyanto & Khasanah, 2018) menyatakan bahwa "Program adalah kumpulan instruksi yang digunakan untuk mengatur komputer agar melakukan suatu tindakan tertentu". Itulah sebabnya sering dikatakan bahwa komputer mencakup tiga aspek penting, berupa perangkat keras (*hardware*), perangkat lunak (*software*) yang dalam hal ini berupa program, dan perangkat akal (*brainware*) atau orang yang berperan terhadap operasi komputer maupun pengembangan perangkat lunak.

Menurut Raharjo dalam (Yulia, 2017) mengemukakan bahwa "Program adalah perangkat lunak (*software*) yang sebenarnya merupakan tuntunan instruksi yang ditulis dalam bentuk kode–kode menggunakan bahasa pemrograman tertentu dan telah dikompilasi dengan menggunakan *compiler* yang sesuai".

Proses program komputer bukan saja menulis suatu urutan instruksi yang harus dikerjakan oleh komputer akan tetapi bertujuan untuk memecahkan masalah serta membuat mudah pengguna komputer (user).

Penulis menyimpulkan bahwa “Program adalah kumpulan instruksi yang ditulis dengan bahasa pemrograman seperti C, C++, Java, dan sebagainya yang digunakan untuk mengatur komputer dalam melakukan tindakan tertentu dan mempermudah pengguna komputer.”

2.1.2 Netbeans IDE 8.2

Menurut (Suryono & Kardian, 2017) mengemukakan bahwa “Netbeans adalah sebuah aplikasi Integrated Development Environment (IDE) yang berbasiskan Java dari Sun Microsystem yang berjalan di atas swing.”

Swing merupakan sebuah teknologi Java untuk pengembangan aplikasi desktop yang berjalan pada berbagai macam platform seperti Windows, Linux, Mac OS X, dan Solaris. Netbeans juga dapat digunakan programmer untuk menulis, mengcompile, mencari kesalahan dan menyebarkan program netbeans yang ditulis dalam bahasa pemrograman java, namun selain itu dapat juga mendukung bahasa pemrograman lainnnya dan program ini pun bebas untuk digunakan dan untuk membuat professional desktop, enterprise, web, dan mobile applications dengan Java language, C/C++, dan bahkan dynamic languages seperti PHP, JavaScript, Groovy, dan Ruby.

Setiap aplikasi memiliki kekurangan dan kelebihan, bagitu pula dengan Netbeans. Berikut kelebihan dari Netbeans, antara lain:

1. Netbeans GUI Builder gratis dengan ribuan plug in yang bisa didownload langsung di website resminya maupun pihak ketiga.
2. Netbeans GUI Builder sangat kompatibel dengan swing karena memang langsung dikembangkan oleh Sun Microsystem yang notabennya sebagai pengembang swing.

3. Netbeans GUI Builder tidak hanya dapat digunakan untuk Java saja, karena Netbeans dapat digunakan untuk bahasa pemrograman lain seperti C, C++, Ruby, dan PHP.
4. Netbeans GUI Builder sangat cocok untuk digunakan dalam pengembangan sistem berskala enterprise.

Kekurangan yang terdapat pada Netbeans, antara lain:

1. Netbeans GUI Builder hanya mensupport 1 pengembangan Java GUI, yaitu Swing. Padahal ada Java GUI yang dikembangkan oleh eclipse bernama SWT dan JFace yang sudah cukup popular.
2. Netbeans GUI Builder mempatenkan source untuk Java GUI yang sedang dikerjakan dalam sebuah Generated Code, sehingga programmer tidak dapat mengeditnya secara manual.
3. Netbeans GUI Builder memerlukan sumber daya yang besar, seperti memori dan ruang hard disk.
4. Netbeans GUI Builder memerlukan dukungan prosesor yang cukup handal untuk mendapatkan performa yang maksimal.

2.1.3 Bahasa Pemrograman

Menurut Kadir dalam (Mulyanto & Khasanah, 2018) mengemukakan bahwa "Bahasa pemrograman dapat dianalogikan dengan bahasa yang digunakan manusia (bahasa manusia)".

Secara garis besar, bahasa-bahasa pemrograman dapat dikelompokkan menjadi:

1. Bahasa beraras-tinggi (*high-level language*)

   Bahasa beraras-tinggi adalah bahasa pemrograman yang berorientasi kepada bahasa manusia. Program dibuat menggunakan bahasa pemrograman yang mudah

dipahami manusia. Biasanya menggunakan kata-kata bahasa inggris; misalnya IF untuk menyatakan “jika” dan AND untuk menyatakan “dan”. Termasuk dalam kelompok bahasa ini yaitu Java, C++, Pascal, dan BASIC.

2. Bahasa beraras-rendah (*low-level language*)

   Bahasa beraras-rendah adalah bahasa pemrograman yang berorientasi kepada mesin. Bahasa ini menggunakan kode biner (yang hanya mengenal kode 0 dan 1), atau suatu kode sederhana untuk menggantikan kode-kode tertentu dalam sistem biner. Yang tergolong dalam kelompok bahasa ini adalah bahasa mesin dan bahasa rakitan. Bahasa-bahasa tersebut sangat sulit dipahami orang awam dan sangat membosankan bagi pemrogram. Pemrogram harus benar-benar menguasai operasi komputer secara teknis. Namun, bahasa generasi ini memberikan eksekusi program yang sangat cepat. Selain itu, bahasa mesin bersifat sangat bergantung pada mesin (*machine dependent*); Artinya, bahasa mesin antara satu mesin dengan mesin yang lain jauh berbeda.

Menurut (Bachtiar dan Fakhrul, 2018) mengemukakan bahwa “Java merupakan bahasa pemrograman tingkat tinggi yang dipelopori oleh James Gosling yang merupakan engineer di Sun Microsystem.”

Java mulai dibangun pada tahun 1991. Versi alpha dan beta dari Java dirilis pada tahun 1995, 4 tahun setelah project Java diinisiasi. Pada tahun 2010, Sun Microsystem diakuisisi oleh Oracle dan menjadikan Java dikembangkan di bawah kuasa Oracle. Per Januari 2018, versi stabil terakhir dari Java yaitu versi Java SE 9. Sebelum Java SE 9, terdapat beberapa versi dari Java yang telah dirilis.

Java merupakan salah satu bahasa pemrograman yang populer saat ini. Hal itu dikarenakan Java dapat berjalan di berbagai platform sistem operasi. Java pun dikenal

dengan istilah *Write Once, Run Anywhere* (menulis sekali, berjalan dimana saja) karena kompabilitasnya tersebut. Menurut Sun Microsystem, Java itu:

1. Simple

   Java merupakan bahasa yang simple sehingga mudah dipahami. Bermodalkan pengetahuan programming dasar, fundamental dari pemrograman java akan cepat dipahami.

2. Berorientasi Objek

   Java menggunakan konsep pemrograman berorientasi objek sehingga pembuatan aplikasi bisa dilakukan lebih modular.

3. Kuat

   Java didesain untuk membuat aplikasi yang memiliki reliabilitas tinggi. Salah satu cara dari Java untuk membuat program yang memiliki realibilitas tinggi yaitu dengan megeliminasi situasi error dengan memeriksanya pada compile time dan runtime.

4. Aman

   Java didesain untuk digunakan pada lingkungan yang terdistribusi. Keamanan merupakan hal yang terpenting dalam lingkungan terdistribusi. Fitur keamanan yang dimiliki Java membuat perangkat lunak yang dibuat tidak bisa diserang dari luar atau disisipi virus.

5. Arsitektur Netral

   Aplikasi yang dibuat menggunakan Java merupakan aplikasi yang platform independent. Aplikasi hanya perlu satu buah versi yang bisa dijalankan pada platform sistem operasi yang berbeda.

6. Portable

   Java dengan karakteristik arsitektur netralnya membuat Java tidak bergantung pada mesin tertentu atau dengan kata lain Java sangat portable.

7. Performa Tinggi

   Java sangat memperhatikan performa. Dengan pengenalan Just In Time compilation, proses kompilasi Java menjadi lebih cepat.

8. Multithreaded

   Java memungkinkan pembuatan aplikasi yang bisa melakukan beberapa pekerjaan secara bersamaan.

9. Dinamis

   Java lebih dinamis dibandingkan bahasa C atau C++ karena didesain untuk dijalankan pada lingkungan yang dinamis.

2.1.4 Basis Data

Menurut Anhar dalam (Yulia, 2017) mengemukakan bahwa “Database adalah sekumpulan tabel-tabel yang berisi data dan merupakan kumpulan dari field atau kolom. Struktur file yang menyusun sebuah database adalah Data Record dan Field”.

Menurut Fathansyah dalam (Mulyanto & Khasanah, 2018) menyatakan bahwa “Basis data terdiri atas 2 kata, yaitu Basis dan Data. Basis kurang lebih dapat diartikan sebagai markas atau gudang, tempat bersarang/berkumpul. Sedangkan data adalah representasi fakta dunia nyata yang mewakili suatu objek yang direkam dalam bentuk angka, huruf, simbol, teks, gambar, bunyi, atau kombinasinya.”

Basis data juga disebut dengan jantung sebuah sistem informasi atau komponen yang sangat penting di dalam sebuah sistem informasi, karena bagus atau tidaknya sebuah sistem tergantung dari basis datanya. Basis data juga bisa dipakai untuk

menentukan kualitas informasi yang dinilai dari keakuratan, relevan, dan ketepatan waktu.

Menurut Nugroho dalam (Mulyanto & Khasanah, 2018) mengemukakan bahwa "MySQL (*My Structured Query Languange*) adalah sebuah program pembuat dan pengelola database atau yang sering disebut dengan DBMS (*Database Management System*), sifat dari DBMS ini adalah *Open Source*".

MySQL merupakan program pengakses database yang bersifat jaringan, sehingga dapat digunakan untuk aplikasi Multi User (banyak pengguna). Kelebihan lain dari MySQL adalah menggunakan bahasa *query* (permintaan) standar SQL (*Structured Query Languange*), SQL adalah suatu bahasa permintaan yang terstruktur. Program-program aplikasi yang mendukung MySQL adalah PHP (*Page Hipertext Preprosesor*), Borland Delphi, Borland C++ Builder, Visual Basic 5.0/6.0 dan .Net, Visual FoxPro, Cold Fusion.

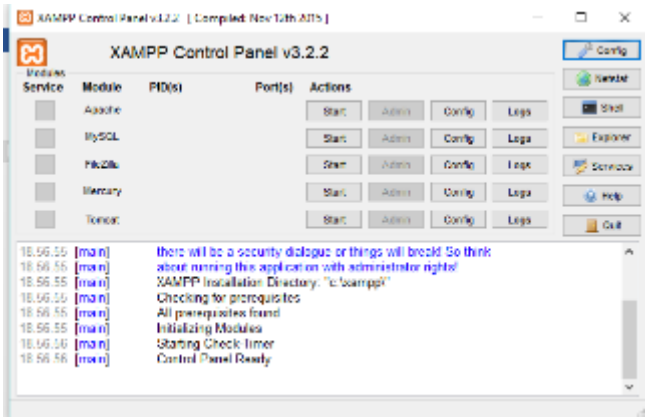
Menurut Anhar dalam (Agus & Safitri, 2015) mengemukakan bahwa "MySQL (*My Structure Query Language*) adalah sebuah perangkat lunak sistem manajemen basis data SQL *Database Management System* atau DBMS dari sekian banyak DBMS seperti Oracle, MS SQL, Postagre SQL dan lainnya".

2.1.5 XAMPP

Menurut (Wardana, 2016) mengemukakan bahwa "Xampp adalah paket *software* yang didalamnya sudah terkandung Web Server Apache, database MySql, dan PHP Interpreter".

Menurut Wahana dalam (Agus & Safitri, 2015) mengemukakan bahwa "XAMPP adalah salah satu paket instalasi apache, PHP, dan MySQL secara instant yang dapat digunakan untuk membantu proses instalasi ketiga produk tersebut".

Berikut ini tampilan Control Panel Xampp.



Gambar II.1

Tampilan Control Panel XAMPP

Keterangan:

(a) Apache, web server

(b) MySQL, database mysql

(c) Filezilla, untuk transfer antar komputer

(d) Mercury, berhubungan dengan fitur email

XAMPP mendukung banyak sistem operasi, yang merupakan campuran dari beberapa program. Yang mempunyai fungsi sebagai server yang berdiri sendiri (localhost), yang terdiri dari program MySQL database, Apache HTTP Server, dan penerjemah ditulis dalam bahasa pemrograman PHP dan Perl.

2.1.6 Model Pengembangan Perangkat Lunak

Model pengembangan yang digunakan pada perancangan program pengelolaan data laundry ini adalah menggunakan prosedur pengembangan model *Waterfall* (Air Terjun). Model ini melakukan pendekatan secara sistematis dan urut mulai dari level kebutuhan sistem lalu menuju ke tahap analisis kebutuhan, desain, pembuatan kode program, pengujian, dan pendukung atau pemeliharaan.

Menurut Sommerville dalam (Ghozali, 2016) mengemukakan bahwa “Model *waterfall* (Air Terjun) adalah proses pengembangan perangkat lunak dengan tahap-tahap utama dari model ini memetakan kegiatan-kegiatan pengembangan dasar yaitu analisis dan definisi persyaratan, perancangan sistem/perangkat lunak, implementasi dan pengujian unit, integrasi dan pengujian sistem, serta operasi dan pemeliharaan.”

Menurut Sukamto & Salahuddin dalam (Achyani & Arviana, 2018) menyatakan bahwa “Model *Waterfall* (Air Terjun) adalah sebuah proses hidup perangkat lunak memiliki sebuah proses yang linear dan sekuensial”.

Tahapan – tahapan yang ada pada model *waterfall* (Air Terjun) secara umum menurut Sukamto & Salahuddin dalam (Achyani & Arviana, 2018) adalah:

a. Analisis Kebutuhan

   Adalah proses pengumpulan kebutuhan yang dilakukan secara intensif untuk menspesifikasikan kebutuhan perangkat lunak agar dapat dipahami seperti apa yang dibutuhkan oleh user.

b. Desain

   Adalah proses multi langkah yang fokus pada desain pembuatan program perangkat lunak termasuk strutur data, arsitektur perangkat lunak, representasi antar muka dan prosedur pengodean.

c. Pembuatan Kode Program

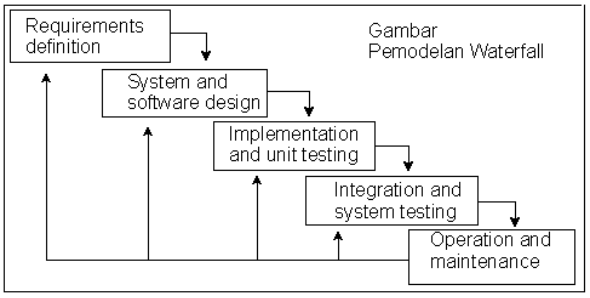
   Adalah desain harus ditranslasikan ke dalam program perangkat lunak.

d. Pengujian

   Adalah fokus pada perangkat lunak secara dari segi lojik dan fungsional dan memastikan bahwa semua bagian sudah diuji. Hal ini dilakukan untuk meminimalisir kesalahan dan memastikan keluaran yang dihasilkan sesuai dengan yang diinginkan.

e. Pendukung atau Pemeliharaan

Adalah tahapan yang dapat mengulangi proses pengembangan mulai dari analisis spesifikasi untuk perubahan perangkat lunak yang sudah ada, tapi tidak untuk membuat perangkat lunak baru.



Gambar
Pemodelan Waterfall

Sumber: Pascapraharastyan, Supriyanto, & Sudarmaningtyas (2014)

Gambar II.2

Tahap-tahap Pemodelan *Waterfall*

Menurut Pressman dalam (Pascapraharastyan, Supriyanto, & Sudarmaningtyas, 2014) menjelaskan tahap – tahap yang dilakukan di dalam model *waterfall*, antara lain:

a. Requirements definition

Proses pencarian kebutuhan diintensifkan dan difokuskan pada *software*.

b. System And Software Design

Desain harus dapat mengimplementasikan kebutuhan yang telah disebutkan pada tahap sebelumnya.

c. Implementation And Unit Testing

Tahap ini merupakan implementasi dari tahap design yang secara teknis nantinya dikerjakan oleh *programmer*.

d. Integration And System Testing

Semua fungsi-fungsi *software* harus diuji coba agar *software* bebas dari error, dan hasilnya harus benar- benar sesuai dengan kebutuhan yang sudah didefinisikan sebelumnya.

e. Operation And Maintenance

Pemeliharaan suatu *software* diperlukan, termasuk di dalamnya adalah pengembangan, karena *software* yang dibuat tidak selamanya hanya seperti itu.

2.2 Tools Program

2.2.1 *Enterprise Relationship Diagram* (ERD)

Menurut James A. Hall dalam (Pascapraharastyan et al., 2014) mengemukakan bahwa "ERD adalah suatu teknik dokumentasi yang digunakan untuk menyajikan relasi antar entitas dalam sebuah sistem".

Sedangkan menurut Hanif dalam (Kamil & Duhani, 2016) menyatakan bahwa "ERD adalah gambar atau diagram yang menunjukkan informasi dibuat, disimpan, dan digunakan dalam sistem bisnis."

Entitas biasanya menggambarkan jenis informasi yang sama. Dalam entitas digunakan untuk menghubungkan antar entitas yang sekaligus menunjukkan hubungan antar data. Pada akhirnya ERD bisa juga digunakan untuk menunjukkan aturan-aturan bisnis yang ada pada sistem informasi yang akan dibangun.

Menurut Rosa dan M. Shalahuddin dalam (Yulia, 2017) "ERD adalah bentuk paling awal dalam melakukan perancangan basis data rasional".

Derajat Relationship menurut Widianti dalam (Yulia, 2017) adalah:

a) *Unary* (Derajat Satu)

Adalah satu buah relationship menghubungkan satu buah entity.

b) *Binary* (Derajat Dua)

Adalah satu buah relationship menghubungkan dua buah entity.

c) *Ternary* (Derajat Tiga)

Adalah satu buah relationship menghubungkan tiga buah entity.

Menurut (Lubis, 2016) menjelaskan bahwa dalam ERD hubungan antara entitas dapat dipetakan menjadi beberapa pembatas, yaitu:

1) Satu-ke-satu atau *one-to-one* (1-1)

   Pembacaan pemetaan satu-ke-satu dalam ER-D, berarti bahwa setiap entitas akan berhubungan dengan paling banyak satu entitas yang lain, demikian sebaliknya.

2) Satu-ke-banyak atau *one-to-many* (1-M/N)

   Pembacaan pemetaan satu-ke banyak adalah satu atribut dapat berhubungan dengan lebih dari satu (banyak) atribut yang lain, tetapi tidak sebaliknya lebih dari satu entitas B.

3) Banyak-ke-satu atau many-to-one (M/N-1)

   Hubungan banyak-ke-satu merupakan kebalikan dari hubungan satu ke banyak, yaitu banyak (lebih dari satu) entitas yang satu akan berhubungan dengan hanya satu pada entitas yang lain, namun tidak sebaliknya.

4) Banyak-ke-banyak atau *many to many* (M-M)

   Pembacaan pemetaan banyak-kebanyak, ini berarti banyak entitas dapat dihubungkan dengan banyak entitas yang lain, demikian sebaliknya

2.2.2 *Logical Record Structure* (LRS)

Menurut Tabrani dalam (Mulyanto & Khasanah, 2018) mengemukakan bahwa "*Logical Record Structure* dibentuk dengan nomor dari tipe *record* digambarkan oleh kotak empat persegi panjang dan dengan nama yang unik. Perbedaan LRS dengan E-R diagram adalah nama tipe *record* berada diluar kotak *field* tipe *record* ditempatkan."

Logical Record Structure terdiri dari link-link diantara tipe *record*. Link ini menunjukkan arah dari satu tipe record lainnya. Banyak link dari LRS yang diberi tanda *field-field* yang kelihatan pada kedua link tipe *record*. Penggambaran LRS mulai

dengan menggunakan model yang dimengerti. Dua metode yang digunakan, dimulai dengan hubungan kedua model yang dapat dikonverensikan ke LRS, metode yang lain dimulai dengan ERD dan langsung dikonversikan ke LRS.

2.2.3 Pengkodean

Pengkodean digunakan untuk mengklasifikasikan data yang dimasukkan kedalam komputer ataupun untuk mengambil bermacam-macam informasi. Kode dapat terbentuk dari kumpulan angka, huruf atau simbol lainnya.

Menurut Kusrini dan Koniyo dalam (Mulyanto & Khasanah, 2018) "Kode akun adalah pemberian tanda/nomor tertentu dengan memakai angka, huruf, atau kombinasi angka dan huruf pada setiap akun atau rekening". Kode akun meliputi kode numerikal, desimal, mnemonik, serta kode kombinasi huruf dan angka. Berikut penjelasan mengenai beberapa kode:

1. Kode Numerikal

   Kode numerikal adalah cara pengkodean akun berdasarkan nomor urut, yang dapat dimulai dari angka 1,2,3 dan seterusnya.

2. Kode Desimal

   Kode desimal adalah cara pemberian kode akun dengan menggunakan lebih dari satu angka. Setiap angka mempunyai makna atau karakter sendiri. Kode desimal dapat dibedakan atas kode kelompok, kode blok, kode stelse akun desimal.

3. Kode Mnemonik

   Kode mnemonik adalah cara pengkodean akun dengan menggunakan huruf tertentu, misalkan harta dengan kode 'H', akun hutang dengan huruf 'U', dan akun modal dengan huruf 'M'.

4. Kode Akun dengan Sistem kombinasi Huruf dan Angka

   Sistem kombinasi huruf dan angka adalah cara pengkodean dengan kombinasi antara huruf dan angka.

2.2.4 *Hierarky Input Proccess Output* (HIPO)

1. Pengertian

Menurut Fatta dalam (Mulyanto & Khasanah, 2018) mengemukakan bahwa "HIPO merupakan teknik untuk mendokumentasikan pengembangan suatu sistem yang dikembangkan oleh IBM." HIPO dapat digunakan untuk memenuhi kebutuhan beberapa pengguna untuk kepentingan berbeda-beda, antara lain:

1) Seorang manajer dapat menggunakan dokumentasi HIPO untuk memperoleh gambaran umum sistem.
2) Seorang programer menggunakan HIPO untuk menentukan fungsi-fungsi dalam program yang dibuatnya.
3) Programer juga dapat menggunakan HIPO untuk mencari fungsi-fungsi yang dimodifikasi dengan cepat.

Teknik ini mempunyai beberapa tujuan utama. *Pertama*, dapat dibuat sebuah struktur yang menggambarkan hubungan antar fungsi dalam program secara hierarkis. *Kedua*, untuk menentukan fungsi-fungsi apa saja yang harus ada dalam sistem yang dikembangkan. *Ketiga*, untuk mendapatkan gambaran input dari fungsi dan output apa yang dihasilkan.

2. Tingkatan Diagram HIPO

Tingkatan HIPO terdiri dari 3 diagram, yaitu diagram daftar isi visual (*visual table of content*), diagram ringkas (*overview diagram*), dan diagram rinci (*detail diagram*).

Berikut penjelasannya, antara lain:

1) Daftar Isi Visual (DIV)

   Diagram ini membuat semua modul yang ada dalam sistem berikut nama dan nomornya, yang nantinya akan diperinci dalam diagram ringkas dan diagram rinci. Dalam DIV juga bisa dilihat fungsi-fungsi utama yang menyusun sebuah sistem dan hubungan antar fungsi tersebut.

2) Diagram Ringkas

   Diagram ringkas menerangkan *input*, proses, dan *output* dari sistem. Diagram ringkas menggambarkan input dan output dari fungsi-fungsi yang telah didefinisikan dalam daftar isi *visual*.

3) Diagram Rinci

   Diagram rinci HIPO digunakan untuk memperinci *input*, proses, dan *output* yang telah digambarkan dalam diagram ringkas. Dalam *input* data dijelaskan *field-field* datanya secara detail. Untuk fungsi, juga dideskripsikan proses apa yang dilakukan oleh fungsi-fungsi tersebut. Rincian *field-field* data *output* juga dengan lebih detail.

2.2.5 Diagram Alir Program (Flowchart)

1. Pengertian

Flowchart merupakan alat yang digunakan untuk menggambarkan sebuah algoritma. Menurut (Sitorus, 2015) mengemukakan bahwa "*Flowchart* menggambarkan urutan logika dari suatu prosedur masalah, sehingga *flowchart* merupakan langkah-langkah penyelesaian masalah yang dituliskan dalam simbol-simbol tertentu.

Menurut Rachmat dalam (Mulyanto & Khasanah, 2018) menyatakan bahwa "*Flowchart* merupakan alur pemikiran yang dituangkan kedalam bentuk gambar/simbol".

Dengan menggunakan *flowchart* (diagram alir) maka seorang *programmer* dapat memberikan idenya secara tertulis sehingga dapat dipahami oleh *programmer* lain, oleh klien, atau oleh tim kerjanya.

Menurut Jogiyanto dalam (Yulia, 2017) mengemukakan bahwa "*Flowchart* adalah bagan (*chart*) yang menunjukkan alir (*flow*) di dalam program atau prosedur sistem secara logika".

Diagram alur dapat menunjukan secara jelas arus pengendalian suatu algoritma, yakni bagaimana pelaksanaan suatu rangkaian secara logis dan sistematis suatu diagram alur dapat memberikan gambaran dua dimensi yang berupa simbol-simbol grafis. Masing-masing simbol telah ditetapkan terlebih dahulu fungsi dan artinya. Sedangkan arti khusus dari *flowchart* itu sendiri adalah simbol-simbol yang digunakan untuk menggambarkan urutan proses yang terjadi dalam sebuah program atau suatu diagram yang menggambarkan susunan logika suatu program.

Flowchart sendiri terdiri dari tiga struktur, yaitu:

a) Struktur Sederhana (*Squence Structure*).

   Diagram yang alurnya mengalir secara berurutan dari atas ke bawah atau dengan kata lain tidak adanya percabangan ataupun perulangan.

b) Struktur Percabangan (*Branching Structure*)

   Diagram yang alurnya ada atau banyak terjadi alih kontrol berupa percabangan dan terjadi apabila kita dihadapkan pada suatu kondisi dengan dua pilihan benar atau salah.

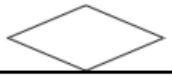
c) Struktur Perulangan (*Looping Structure*)

Pemutaran kembali, terjadi kendali mengalihkan arus diagram alur kembali keatas, sehingga beberapa alur berulang beberapa kali.

2. Bentuk *Flowchart*

a. Program *Flowchart*

Program *flowchart* aitu simbol-simbol *flowchart* yang digunakan untuk menggambarkan logika dari pemrosesan terhadap data. Menurut Suarga dalam (Mulyanto & Khasanah, 2018) menyatakan bahwa simbol yang digunakan dalam program *flowchart* yaitu:

Simbol	Keterangan
	Terminator, mulai atau selesai.
	Proses, menyatakan proses terhadap data.
	Input/Output, menerima input atau menampilkan output.
	Seleksi/Pilihan, memilih aliran berdasarkan syarat.
	Predefined-Data, definisi awal dari variabel atau data.
	Predefined-Process, lambang fungsi atau sub-program.
	Connector, penghubung.
	Off-page Connector, penghubung penghubung pada halaman yang berbeda

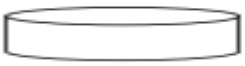
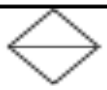
Sumber: Mulyanto & Khasanah (2018)

Gambar II.3

Simbol Program *Flowchart*

b. Sistem *Flowchart*

Sistem *flowchart* merupakan simbol-simbol peralatan sistem komputer yang digunakan untuk menyatakan proses pengolahan data. Menurut Suarga dalam (Mulyanto & Khasanah, 2018) menyatakan bahwa simbol yang digunakan dalam *system flowchart* yaitu:

Simbol	Keterangan
	Keyboard
	Printer
	File/Storage
	Display/Monitor
	Magnetic Tape
	Magnetic Disk
	Sorting
	Extract
	Merge

Sumber: Mulyanto & Khasanah (2018)

Gambar II.4

Simbol Sistem *Flowchart*

3. Teknik Pembuatan

Adapun teknik pembuatan program *flowchart* ini dibagi menjadi dua bagian yaitu sebagai berikut:

a) *General Way*

Teknik pembuatan *flowchart* dengan cara ini biasanya dipakai didalam menyusun logika suatu program, yang menggunakan pengulangan proses secara tidak langsung (*Non-DirectLoop).*

b) *Iteration Way*

Teknik pembuatan *flowchart* dengan cara ini biasanya dipakai untuk logika program yang cepat serta bentuk permasalahan yang kompleks. Dimana pengulangan proses yang terjadi bersifat langsung (*Direct-Loop).*

2.2.6 Implementasi dan Pengujian Unit

Implementasi merupakan rangkaian kegiatan yang dilakukan setelah tahap perancangan selesai dilaksanakan. Tujuan implementasi diantaranya adalah sebagai berikut:

1. Mengkaji mengenai rangkaian sistem, baik software maupun hardware dalam bentuk sistem informasi terpusat (*integrated information system*).
2. Melakukan uji coba mengenai *software* dan *hardware* sebagai sarana pengolah data dan sekaligus penyaji informasi yang dibutuhkan.
3. Melakukan penerapan serta peralihan sistem yang lama ke sistem baru sebagai keputusan terakhir di dalam tahap pengembangan sistem.
4. Pemeliharaan sistem.

Pengujian adalah proses yang bertujuan untuk memastikan apakah semua fungsi sistem bekerja dengan baik dan mencari kesalahan yang mungkin terjadi pada sistem. Tujuan dari pengujian adalah untuk:

1. Mendeteksi kesalahan bahasa (*language error*), diakibatkan karena kesalahan penulisan sintaks.
2. Mendeteksi kesalahan waktu proses (*runtime error*), kesalahan yang terjadi ketika program dijalankan. Kesalahan ini akan menyebabkan proses program terhenti sebelum waktunya berhenti.
3. Mendeteksi kesalahan logika (*logical error*), kesalahan yang disebabkan oleh logika program yang dibuat. Kesalahan ini sulit ditemukan karena tidak ada pemberitahuan letak kesalahannya.