

BAB II

LANDASAN TEORI

2.1. Konsep Dasar Program

2.1.1. Definisi Program

Program adalah kumpulan intruksi yang digunakan untuk mengatur komputer agar melakukan suatu tindakan tertentu. Tanpa adanya program komputer hanyalah perangkat keras (*hardware*) yang tidak bisa melakukan apa-apa. (Sukmaindrayana & Sidik, 2017)

2.1.2. Definisi Bahasa Pemrograman

Bahasa pemrograman merupakan notasi untuk memberikan perintah secara tepat program komputer. Berbeda dengan bahasa, misalkan Bahasa Indonesia dan Inggris yang merupakan bahasa alamiah (*natural language*), sintaksis dan semantik bahasa pemrograman komputer ditentukan secara jelas dan terstruktur, sehingga bahasa pemrograman juga disebut sebagai bahasa formal (*formal language*). (Fridayanthie, 2015).

1. Java

Menurut Sugiarti (2018:1) Java merupakan bahasa pemrograman tingkat tinggi (*high level*). Namun demikian, pemrograman ini bahasanya mudah dipahami karena menggunakan bahasa sehari-hari. Java dibuat oleh perusahaan *Sun Microsystem*, oleh James Gosling, Patrick Naughton dan Mike Sheridan pada 1991. Salah satu alasan yang membuat Java begitu populer karena mempunyai konsep bahasa OOP (*Object Oriented Programming*). Oleh karena itu, untuk

mengembangkannya jauh lebih mudah, serta menjaga sistem tetap modular, fleksibel, dan ekstensibel.

2. *Netbeans IDE*

Menurut Sugiarti (2018:6) *Netbeans* merupakan IDE yang ditujukan untuk memudahkan pemrograman Java. Dalam *Netbeans*, pemrograman dilakukan berbasis visual. Persis seperti IDE lain, misalnya *Borland Delphi* dan *Microsoft Visual Studio*. Untuk membuat dialog atau *user-interface*, kita tidak perlu membuat teks program secara manual baris per baris, tetapi cukup klik pada *component-pallete*.

2.1.3. Basis Data

Menurut Sukanto & Shalahuddin (2015:43) Sistem basis data adalah sistem terkomputerisasi yang tujuan utamanya adalah memelihara data yang sudah diolah atau informasi dan membuat informasi tersedia saat dibutuhkan. Pada intinya basis data adalah media untuk menyimpan data agar dapat diakses dengan mudah dan cepat.

Menurut Mardiani, Rahmansyah, Kurniawan, Sensuse, & Jayanta (2016:9) *database* adalah suatu susunan/kalimat data operasional lengkap dari suatu organisasi/perusahaan yang diorganisir/dikelola dan disimpan secara terintegrasi menggunakan metode tertentu, dengan menggunakan komputer sehingga mampu menyediakan informasi secara optimal yang diperlukan pemakainya.

1. *PhpMyAdmin*

PhpMyadmin adalah perangkat lunak bebas yang ditulis dalam bahasa pemrograman PHP yang digunakan untuk menangani administrasi *MySQL* melalui Jejaring Jagat Jembar (*World Wide Web*). *PhpMyAdmin* mendukung berbagai operasi *MySQL*, diantaranya (mengelola basis data, tabel- tabel, bidang (*fields*), relasi

(*relations*), indeks, pengguna (*users*), perijinan (*permissions*), dan lain-lain). (Barri, Lumenta, & Wowor, 2015)

Menurut Haqi (2017:9) *PhpMyAdmin* merupakan bagian untuk mengelola basis data *MySQL* yang ada di komputer. Untuk membukanya, buka *browser* lalu ketikkan alamat *http://localhost/phpmyadmin*, akan muncul halaman *phpMyAdmin*. Nantinya seseorang bisa membuat (*create*) basis data baru, dan mengelolanya.

2. *Xampp*

XAMPP adalah perangkat lunak bebas, yang mendukung banyak sistem operasi, merupakan kompilasi dari beberapa program. Fungsinya adalah sebagai *server* yang berdiri sendiri (*localhost*), yang terdiri atas program *Apache HTTP Server*, *MySQL database*, dan penerjemah bahasa yang ditulis dengan bahasa pemrograman *PHP* dan *Perl*. Nama *XAMPP* merupakan singkatan dari X (empat sistem operasi apapun), *Apache*, *MySQL*, *PHP* dan *Perl* (Palit, Rindengan, & Lumenta, 2015).

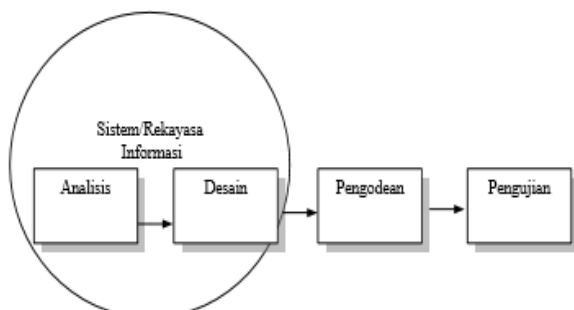
2.1.4. Model Pengembangan Perangkat Lunak

Menurut Sukamto & Shalahuddin (2015:26) mengemukakan bahwa SDLC atau *Software Development Life Cycle* atau sering disebut juga *System Development Life Cycle* adalah proses mengembangkan atau mengubah suatu sistem perangkat lunak dengan menggunakan model-model dan metodologi yang digunakan orang untuk mengembangkan sistem-sistem perangkat lunak sebelumnya, berdasarkan *best practice* atau cara-cara yang sudah teruji baik.

1. Metode Air Terjun (*Waterfall*)

Menurut Sukamto & Shalahuddin (2015:28) di jelaskan bahwa model SDLC air terjun (*waterfall*) sering juga disebut model sekuensi linear (*sequential linear*) atau alur hidup klasik (*classic life cycle*). Model air terjun menyediakan pendekatan alur

hidup perangkat lunak secara sekuensial atau terurut dimulai dari analisis, desain, pengodean, pengujian, dan tahap pendukung (*support*).



Sumber: Sukamto & Shalahuddin (2015:29)

Gambar. II.1
Ilustrasi Model Waterfall

2.2. Tools Program

2.2.1. Entity Relationship Diagram (ERD)

ERD atau diagram ER adalah model jaringan data yang menekankan pada struktur-struktur dan *relationship* data.

Menurut Sukamto & Shalahuddin (2015:50) Pemodelan awal basis data yang paling banyak digunakan adalah *Entity Relationship Diagram* (ERD). ERD dikembangkan berdasarkan teori himpunan dalam bidang matematika. ERD digunakan untuk pemodelan basis data rasional.

Menurut Trisyanto (2017:89) 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.

2.2.2. Logical Relationship Structural (LRS)

Logical Record Structure (LRS) digambarkan kotak persegi panjang dan dengan nama yang unik. *File record* pada LRS ditempatkan dalam kotak. LRS terdiri dari *link* dari LRS yang diberi nama oleh *field-field* yang kelihatan pada kedua *link tipe record*. (Sukmaindrayana & Sidik, 2017)

2.2.3. Pengkodean

Menurut Kristanto (2018:106) Pengkodean adalah suatu tahap dari analisa kebutuhan sistem dan desain sistem yang dituliskan dalam suatu bahasa pemrograman komputer tertentu yang biasanya oleh pabrik komputer sudah ditentukan spesifikasinya.

Dalam (Azis & Sarmidi, 2018) disebutkan bahwa angka merupakan simbol yang banyak digunakan pada sistem kode akan tetapi kode yang berbentuk angka lebih dari 6 digit akan sangat sulit untuk di ingat kode numerik (*numeric code*) menggunakan 10 macam kombinasi angka di dalam kode. Kode alphabetik (*alphabetic code*) menggunakan 26 kombinasi huruf untuk kodenya. Kode alphanumerik (*alphanumeric code*) merupakan kode yang menggunakan gabungan angka, huruf, dan karakter-karakter khusus meskipun kode numerik, alphabetik dan alphanumerik merupakan kode yang paling banyak digunakan di dalam sistem informasi, tetapi kode yang lain juga mulai banyak digunakan, seperti misalnya kode batang (*bar code*). Ada beberapa macam tipe dari kode yang dapat digunakan di dalam sistem informasi, diantaranya adalah kode mnemonik (*mnemonic code*), kodeurut (*sequential code*), kode blok (*block code*), kode grup (*group code*), dan kode desimal (*decimal code*), masing-masing tipe dari kode tersebut mempunyai kebaikan dan kelemahannya tersendiri. Dalam praktek, tipe-tipe kode yang ada dapat dikombinasikan:

1. Kode Mnemonik (*mnemonic code*)

Digunakan untuk tujuan supaya mudah di ingat. Kode mnemonik dibuat dengan dasar singkatan atau mengambil sebagian karakter dari item yang akan di wakili dengan kode ini. Sebagai contoh kode “P” untuk mewakili pria dan kode “W” untuk wanita akrab untuk mudah diingat. Umumnya kode mnemonic menggunakan huruf, akan tetapi dapat juga menggunakan gabungan huruf dan angka misalnya barang dagangan komputer IBM PC dengan ukuran memori 640 Kb, *colour monitor*, dapat dikodekan menjadi K-IBM- PC-640-CO supaya lebih mudah diingat. Kebaikan dari kode ini adalah mudah diingat dan kelemahannya adalah kode dapat menjadi terlalu panjang.

2. Kode Urut (*Sequential Code*)

Kode yang ini disebut juga kode seri (*serial code*) merupakan kode yang nilainya urut antara satu kode dengan kode berikutnya. Contoh kode urut adalah sebagai berikut:

001 Kass

002 Piutang Dagang

003 Persediaan Produk Selesai

004 Persediaan Produk Dalam Proses

005 Persediaan Bahan Baku

006 Biaya Dibayar Dimuka

3. Kode Blok (*block code*)

Kode blok (*block code*) mengklasifikasikan *item* ke dalam kelompok blok tertentu yang mencerminkan suatu klasifikasi tertentu atas dasar pemakaian maksimum yang diharapkan. Contoh kode blok adalah sebagai berikut:



Rekening-rekening dalam buku besar dapat diberi kode dengan mengklasifikasikannya ke dalam kelompok rekening utama sebagai berikut:

BLOK KELOMPOK

1000-1999 AKTIVA LANCAR

2000-2999 AKTIVA TETAP

3000-3999 HUTANG LANCAR

3500-3999 HUTANG JANGKA PANJANG

4000-4999 MODAL

4. Kode Desimal (*desimal code*)

Kode desimal (*desimal code*) mengklasifikasikan kode atas dasar 10 unit angka desimal dimulai dari angka 0 sampai dengan angka 9 atau dari 00 sampai dengan 99 tergantung dari banyaknya kelompok.

5. Kode Grup (*group code*)

Kode grup (*group code*) merupakan kode yang berdasarkan *field-field*, dan tiap *field-field*nya mempunyai arti.

6. Kode Batang (*barcode*)

Sebagai kumpulan kode yang berbentuk garis, dimana masing-masing ketebalan setiap garis berbeda sesuai dengan isi kodenya.

2.2.4. *Hierarchy Input Process Output* (HIPO)

1. Definisi HIPO

Menurut Trisyanto (2017:82) HIPO (*Hierarchy Plus Input-Proses-Output*) merupakan metodologi yang dikembangkan dan didukung oleh IBM. HIPO sebenarnya adalah alat dokumentasi program, akan tetapi sekarang HIPO juga banyak digunakan sebagai alat desain dan teknik dokumentasi dalam siklus

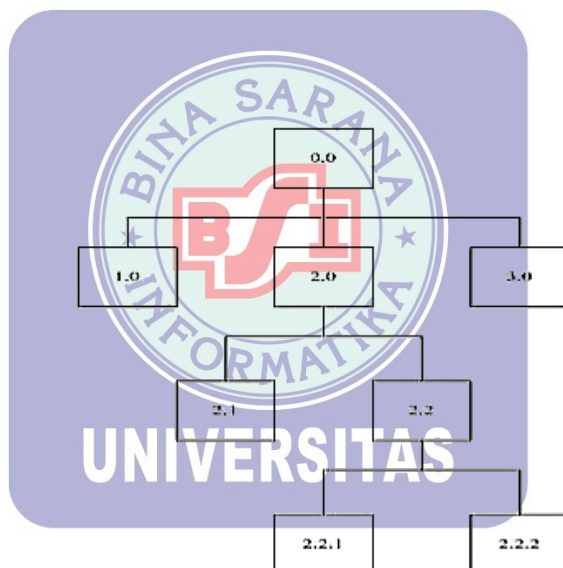
pengembangan sistem. HIPO berbasis pada fungsi, yaitu tiap-tiap modul di dalam sistem digambarkan oleh fungsi utamanya.

2. Tingkatan diagram HIPO

Fungsi-fungsi dari sistem digambarkan oleh HIPO dalam tiga tingkatan. Untuk masing-masing tingkatan digambarkan dalam bentuk diagram tersendiri. Dengan demikian HIPO menggunakan tiga macam diagram untuk masing-masing tingkatannya, yaitu sebagai berikut ini:

a. *Visual Table Of Contents (VTOC)*

Diagram ini menggambarkan hubungan dari fungsi-fungsi di sistem secara berjenjang.



Sumber: Trisyanto (2017:84)

Gambar II.2
Ilustrasi *Visual Table Of Contents*

b. *Overview Diagram*

Overview diagram menunjukkan secara garis besar hubungan dari *input*, proses dan *output*. Bagian *input* menunjukkan *item-item* data yang akan digunakan oleh bagian proses, bagian proses berisi sejumlah langkah-langkah yang menggambarkan kerja dari fungsi, bagian *output* berisi dengan *item-item* data yang dihasilkan atau dimodifikasi oleh langkah-langkah proses.

c. Detail Diagrams

Detail diagrams merupakan diagram tingkatan yang paling rendah di diagram HIPO.

Diagram ini berisi dengan elemen-elemen dasar dari paket yang menggambarkan secara rinci kerja dari fungsi.

2.2.5. Diagram Alir Program (*Flowchart*)

1. Definisi *Flowchart*

Menurut Sukamto (2018:843) *Flowchart* dapat dimengerti dari asal katanya adalah sebuah *flow* atau aliran dan *chart* atau bagan, sehingga didapat jika dari asal katanya *flowchart* adalah sebuah bagan aliran dari sesuatu, dan sesuatu itu dapat juga berupa aliran proses.

2. Bentuk *Flowchart*

Dalam (Yulia, 2017) disebutkan bahwa bentuk-bentuk *flowchart* atau diagram alur yang sering digunakan didalam proses pembuatan suatu program adalah sebagai berikut:

a. Program *Flowchart*

Flowchart Program dihasilkan dari *flowchart* sistem. *Flowchart* Program merupakan keterangan yang lebih rinci tentang bagaimana setiap langkah program atau prosedur sesungguhnya dilaksanakan.

b. Sistem *Flowchart*

Flowchart Sistem merupakan bagan yang menunjukkan alur kerja atau apa yang sedang dikerjakan di dalam sistem secara keseluruhan dan menjelaskan urutan dari prosedur-prosedur yang ada di dalam sistem.

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

Menurut Sukamto & Shalahuddin (2015:272) pengujian adalah satu set aktifitas yang direncanakan dan sistematis untuk menguji atau mengevaluasi kebenaran yang diinginkan. Aktifitas pengujian terdiri dari satu set atau sekumpulan langkah dimana dapat menempatkan desain kasus uji yang spesifik dan metode pengujian.

Menurut Sukamto & Shalahuddin (2015:275) *Black Box Testing* (pengujian kotak hitam) yaitu menguji perangkat lunak dari segi spesifikasi fungsional tanpa menguji desain dan kode program. Pengujian dimaksudkan untuk mengetahui apakah fungsi-fungsi, masukan, dan keluaran dari perangkat lunak sesuai dengan spesifikasi yang dibutuhkan. Pengujian kotak hitam dilakukan dengan membuat kasus uji yang bersifat mencoba semua fungsi dengan memakai perangkat lunak apakah sesuai dengan spesifikasi yang dibutuhkan. Kasus uji yang dibuat untuk melakukan pengujian kotak hitam harus dibuat dengan kasus benar dan kasus salah, misalkan untuk kasus proses *login* maka kasus uji yang dibuat adalah:

1. Jika *user* memasukkan nama pemakai (*username*) dan kata sandi (*password*) yang benar.

2. Jika *user* memasukkan nama pemakai (*username*) dan kata sandi (*password*) yang salah, misalnya nama pemakai benar tapi kata sandi salah, atau sebaliknya, atau keduanya salah.

