BAB II
LANDASAN TEORI

Pembuatan tugas akhir ini tidak terlepas dari teori-teori yang mendukung kemudahan dalam mempelajari serta merancang program yang di harapkan berfungsi secara maksimal. Kemudahan dalam menggunakan suatu program aplikasi bagi setiap pengguna akan sangat membantu dalam menyelesaikan setiap pekerjaan.

Keuntungan lain dari suatu program aplikasi yang mudah di gunakan adalah akan memperkecil kemungkinan terjadinya kesalahan yang dilakukan oleh pengguna pada saat menjalankan program aplikasi tersebut. Berikut ini adalah teori pendukung yang memperkuat penulisan tugas akhir ini.

2.1 Konsep Dasar Program

Bentuk dari berbagai macam jenis aplikasi yang dipergunakan dalam bidang bisnis ataupun dalam bidang ilmiah yang berguna untuk menghasilkan suatu laporan, informasi atau tujuan yang diinginkan berupa rangkaian instruksi dalam bahasa komputer yang disusun secara logis dan sistematis biasa disebut dengan program.

Pemrograman adalah “kegiatan menulis kode yang akan dieksekusi oleh komputer”. Program juga bisa diartikan sebagai suatu proses untuk mengimplementasikan algoritma dengan menggunakan suatu bahasa pemrograman. Bahasa pemrograman adalah perintah-perintah yang dapat dimengerti.

A. Program

Menurut Yulikuspartono (2009:29), ”Program merupakan sederetan instruksi atau *statement* dalam bahasa yang dimengerti oleh komputer yang bersangkutan”. Proses pemrograman komputer bukan hanya sekedar menulis suatu urutan instruksi yang harus dikerjakan oleh komputer, akan tetapi bertujuan untuk memecahkan suatu masalah serta membuat mudah pekerjaan yang diinginkan oleh pemakai.

Salah satu dari tahap pembangunan suatu program adalah menerjemahkan atau mengkodekan rancangan terinci yang telah dibuat menjadi suatu program komputer yang siap pakai. Dengan menerjemahkan berarti penulisan program dengan menggunakan satu bahasa pemrograman yang kita kuasai. Bahasa pemrograman merupakan prosedur atau tata cara penulisan program. Pada bahasa pemrograman terdapat dua faktor penting yaitu sintaks dan simatik.

Sintaks (*syntax*) adalah aturan-aturan gramatikal yang mengatur tata cara penulisan kata, ekspresi dan pernyataan, sedangkan simatik adalah aturan-aturan untuk menyatakan suatu arti. Bahasa pemrograman merupakan suatu proses guna mengimplementasikan algoritma dengan menggunakan suatu bahasa program, satu hal yang cukup penting sebelum seorang program mermulai menyusun program adalah memilih bahasa pemrograman yang akan digunakan. Fungsi bahasa pemrograman adalah sebagai media untuk menyusun dan memahami serta sebagai alat komunikasi antara programer dangan komputer, meskipun dapat juga digunakan sebagai alat komunikasi antara orang yang satu dengan yang lain.

Secara umum bahasa pemrograman dapat dibagi dalam empat kelompok yaitu:

1. Bahasa Tingkat Rendah (*Low-Level-Language*)

   Bahasa tingkat rendah merupakan bahasa pemrograman yang berorientasi pada mesin. Pemrograman yang menggunakan bahasa ini harus dapat berfikir berdasarkan logika mesin komputer, sehingga bahasa ini dimulai kurang fleksibel dan sulit untuk dipahami oleh pemula.

2. Bahasa Tingkat Menengah (*Middle-Level-Language*)

   Bahasa tingkat menengah merupakan bahasa pemrograman yang menggunakan aturan-aturan gramatikal dalam penulisan ekspresi atau pernyataan dengan standar bahasa yang mudah dipahami oleh manusia serta memiliki instruksi-instruksi tertentu yang dapat langsung diakses oleh komputer.

3. Bahasa Tingkat Tinggi (*High-Level-Language*)

   Bahasa tingkat tinggi merupakan bahasa pemrograman yang memiliki aturan-aturan gramatikal dalam penulisan ekspresi atau pernyataan dengan standar bahasa dapat dipahami secara langsung oleh manusia.

4. Bahasa Berorientasi Obyek *(Object-Oriented-Language)*

   Bahasa berorientasi pada obyek merupakan bahasa pemrograman yang mengandung “kapsul-kapsul” yang berisi fungsi-fungsi untuk menyelesaikan satu masalah. Dengan bahasa ini pemrograman tidak lagi harus menuliskan secara detail semua pernyataan dan ekspresi seperti pada bahasa tinggkat tinggi, melainkan cukup.

B. Konsep DasarPemrograman *Visual Basic*

"Definisi visual basic menurut Andi sunyoto (2007 : 1), *Visual Basic 6.0* merupakan salah satu *software* pembuat program aplikasi yang sangat handal.". *Visual Basic* menyediakan *tool* untuk membuat aplikasi yang sederhana sampai aplikasi kompleks atau rumit baik untuk keperluan pribadi maupun untuk keperluan perusahaan/instansi dengan sistem yang lebih besar. "*Visual*" dalam hal ini merupakan bahasa pemrograman yang menyerahkan berbagai macam desain dengan model GUI (*Graphical User Interface*). Hanya dengan mengetikan sedikit kode programan dan sudah dapat menikmati program dengan tampilan yang menarik. "*Basic*" menunjukan bahasa pemrograman *BASIC (Biginner All-Purpose Symbolic Instruction Code)*. *VisualBasic* dikembangkan dari bahasa *BASIC* yang ditambah ratusan perintah tambahan, *function, keyword*, dan banyak berhubungan langsung dengan GUI *(Graphical User Interface) Windows.*

Visual Basic merupakan *event-driven programming* (Pemrograman Terkendali Kejadian) artinya program menunggu sampaidan merespon dari pemakai berupa *event* atau kejadian tertentu seperti tombol diklik, pilih *menu* dan lain-lain.

Dalam pengembangan aplikasi, *visual basic* menggunakan pendekatan *visual* untuk merancang *user interface* dalam bentuk *form*, sedangkan untuk kodingnya menggunakan bahasa *basic* yang cenderung mudah dipelajari. Pada pemrograman *visual,* pengembangan aplikasi dimulai dengan pembentukan *user interface*, kemudian mengatur *property* dari objek-objek yang akan digunakan dalam *user*

interface, dan baru dilakukan penulisan kode program untuk menangani kejadian-kejadian atau *event.*

Visual Basic berbasiskan prinsip *Object Oriented Programming* (OOP) dan dikembangkan dengan basis *Visual* yang berarti menggunakan sarana grafis untuk mengembangkannya. *Visual Basic* berorientasi pada objek-objek yang dipisah-pisah, sehingga disebut pemrograman *Object Oriented Programming* (OOP). *Visual Basic* juga bersifat *modular programming* karena kode-kode program letaknya tersebar di dalam modul-modul (objek-objek) yang terpisah-pisah.

C. Pengertian Basis Data (*Database*)

Basis dapat diartikan sebagaimana, tempat bersarang atau berkumpul. Sedangkan Data merupakan presentasi fakta dunia nyata yang mewakili suatu objek. *Database* menurut Hartono (2005 : 217) merupakan "kumpulan dari data yang saling berhubungan satu dengan yang lainnya, tersimpan disimpan luar komputer dan digunakan perangkat lunak tertentu untuk memanipulasinya". *Database* merupakan salah satu komponen yang penting disistem informasi, karena berfungsi sebagai basis penyedia informasi bagi para penggunanya *(user)*.

Basis Data *(Database)* terdiri atas dua kata, yaitu Basis dan Data. Basis kurang lebih dapat diartikan sebagai markas atau gudang, tempat bersarang atau berkumpul. Sedangkan Data adalah representasi fakta dunia nyata yang mewakili suatu objek bagi suatu manusia, barang, hewan, peristiwa, konsep, keadaan, dan sebagainya.

Yang direkam dalam bentuk angka, huruf, simbol, teks, gambar, bunyi atau kombinasinya.

Basis Data sendiri dapat didefinisikan dalam sejumlah sudut pandang seperti:

1. Himpunan kelompok data atau arsip yang saling berhubungan dengan yang diorganisasikan sedemikian rupa agar kelak dapat dimanfaatkan kembali dengan cepat dan mudah.
2. Kumpulan data yang saling berhubungan yang disimpan secara bersama sedemikian rupa dan tanpa pengulangan *(Redudancy)* yang tidak perlu untuk memenuhi berbagai kebutuhan.
3. Kumpulan *file* atau tabel yang saling berhubungan yang disimpan dalam media penyimpanan elektronik.

Basis Data dan lemari arsip sesungguhnya memeiliki prinsip kerja dan tujuan yang sama. Prinsip utamanya adalah pengaturan data/arsip. Sedangkan tujuan utamanya adalah kemudahan dan kecepatan dalam penggambilan kembali data/arsip. Perbedaanya hanya terletak pada media penyimpanan yang digunakan. Basis data hanya sekedar penyimpanan data secara elektronik. Penulis menjelaskan aplikasi basis data yang digunakan pada program yang dibangun, Yaitu:

1. XAMPP

XAMPP adalah sebuah *software webserver apache* yang didalamnya sudah tersedia *database server mysql* dan *support php programming*. *XAMPP* merupakan *software* yang mudah digunakan, gratis dan mendukung instalasi di *Linux* dan

Windows. Keuntungan lainnya adalah cuma menginstal satu kali sudah tersedia *Apache Web Server, MySQL Database Server, PHP Support (PHP 4 dan PHP 5)* dan beberapa *module* lainnya. Hanya bedanya kalau yang versi untuk *Windows* sudah dalam bentuk instalasi grafis dan yang *Linux* dalam bentuk *file*ter kompresi tar.gz. Kelebihan lain yang berbeda dari versi untuk *Windows* adalah memiliki fitur untuk mengaktifkan sebuah *server* secara grafis, sedangkan *Linux* masih berupa perintah-perintah di dalam *console*. Oleh karena itu yang versi untuk *Linux* sulit untuk dioperasikan.Dulu *XAMPP* untuk *Linux* dinamakan *LAMPP*, sekarang diganti namanya menjadi *XAMPP FOR LINUX*.

2. MySQL

MySQL adalah suatu perangkat lunak *database* relasi (*Relational Database Management System* atau *RDBMS*), seperti halnya *ORACLE, Postgresql, MS SQL*, dan sebagainya.*MySQL AB* menyebut produknya sebagai *database open source* terpopuler di dunia. Berdasarkan riset dinyatakan bahwa bahwa di *platform Web*, dan baik untuk kategori *open source* maupun umum, *MySQL* adalah *database* yang paling banyak dipakai. Menurut perusahaan pengembangnya, *MySQL* telah terpasang di sekitar 3 juta komputer. Puluhan hingga ratusan ribu situs mengandalkan *MySQL* bekerja siang malam memompa data bagi para pengunjungnya.

D. Model Pengembangan Perangkat Lunak

Menurut Pressman (2010), model *waterfall* adalah model klasik yang bersifat sistematis, berurutan dalam membangun *software*. Nama model ini sebenarnya adalah "*Linear Sequential Model*". Model ini sering disebut dengan "*classic life cycle*" atau model *waterfall*. Model ini termasuk kedalam model *generic* pada rekayasa perangkat lunak dan pertama kali diperkenalkan oleh Winston Royce sekitar tahun 1970 sehingga sering dianggap kuno, tetapi merupakan model yang paling banyak dipakai didalam *Software Engineering* (SE).

Model ini melakukan pendekatan secara sistematis dan berurutan. Disebut dengan *waterfall* karena tahap demi tahap yang dilalui harus menunggu selesainya tahap sebelumnya dan berjalan berurutan.

Waterfall adalah suatu metodologi pengembangan perangkat lunak yang mengusulkan pendekatan kepada perangkat lunak sistematik dan sekuensial yang mulai pada tingkat kemajuan sistem pada seluruh analisis, *design*, kode, pengujian dan pemeliharaan. Masalah dengan *waterfall*:

1. Perubahan sulit dilakukan karena sifatnya yang kaku.
2. Karena sifat kakunya, model ini cocok ketika kebutuhan dikumpulkan secara lengkap sehinga perubahan bisa ditekan sekecil mungkin. Tapi pada kenyataannya jarang sekali konsumen/pengguna yang bisa memberikan kebutuhan secara lengkap, perubahan kebutuhan adalah sesuatu yang wajar terjadi.
3. *Waterfall* pada umumnya digunakan untuk rekayasa sistem yang besar yaitu dengan proyek yang dikerjakan di beberapa tempat berbeda, dan dibagi menjadi beberapa bagian sub-proyek.

2.2 *Tools Program*

Merupakan alat yang digunakan untuk menggambarkan bentuk logika model dari suatu sistem dengan menggunakan simbol, lambang, diagram yang menunjukan secara tepat arti dan fungsinya. Fungsunya sendiri adalah untuk menjelaskan kepada *user* bagaimana fungsi dari *system* dapat bekerja dengan bentuk *logical model* dan *physical model.* Adapun peralatan pendukung yang penulis gunakan dalam pembuatan perancangan program ini sebagai berikut:

A. *Entity Relantionship Diagram (ERD)*

Menurut salah satu para ahli, Brady dan Loonam (2010), *Entity Relationship Diagram* (ERD) merupakan teknik yang digunakan untuk memodelkan kebutuhan data dari suatu organisasi, biasanya oleh *System Analys* dalam tahap analisis persyaratan proyek pengembangan *system*. Hubungan *atribute* yang ada di dalam satu atau dua file, yaitu sebagai berikut:

1. *One to one relationship*

   Hubungan antara file pertama dengan file kedua adalah satu berbanding satu.

2. *One to many relationship*

   Hubungan antar file pertama dengan file kedua adalah satu berbanding banyak atau dapat pula dibalik banyak lawan satu.

3. *Many to many relationship*

   Hubungan antara file pertama dengan file kedua adalah banyak berbanding banyak.

4. Relasi *one to one*

Hubungan antara satu atribut dengan atribut yang lain dalam satu file yang sama mempunyai hubungan satu lawan satu.

5. Relasi *many to one*

   Hubungan antara satu atribut dengan atribut lainnya dalam satu file yang sama mempunyai hubungan satu lawan banyak.

6. Relasi *many to many*

   Hubungan antara satu atribut dengan atribut yang lain dalam satu file yang sama mempunyai hubungan banyak lawan banyak.

B. Pengkodean

Menurut Hartono (2005 : 384) mengemukakan bahwa pengkodean adalah suatu bentuk struktur yang berfungsi untuk mengklasifikasikan data, memasukkan data ke dalam komputer dan untuk mengambil bermacam-macam informasi yang berhubungan dengannya

Bertujuan untuk mengklasifikasikan data, memasukan data kedalam komputer dan untuk mengambil bermacam-macam informasi yang berhubungan dengan data tersebut. Kode dapat dibentuk dari kumpulan angka, huruf dan karakter-karakter khusus. Dalam merancang kode yang baik ada beberapa hal yang harus diperhatikan, yaitu sebagai berikut:

1. Harus mudah diingat

   Agar kode mudah diingat, maka dapat dilakukan dengan cara menghubungkan kode tersebut dengan obyek yang diwakili dengan kodenya.

2. Harus unik

   Kode harus unik untuk masing-masing *item* yang diwakili. Unik berarti tidak ada kode yang kembar.

3. Harus fleksibel

   Kode harus fleksibel sehingga memungkinkan perubahan-perubahan atau penambahan *item* baru tetap dapat diwakili oleh kode.

4. Harus efisien

   Kode harus sependek mungkin, sehingga mudah diingat dan juga akan efisien bila direkam atau disimpan didalam komputer.

5. Harus konsisten

   Kode harus konsisten dengan kode yang telah digunakan.

6. Harus distandarisasi

   Kode harus distandarisasi untuk seluruh tingkatan dan departemen dalam organisasi. Kode yang tidak standarakan mengakibatkan kebingungan, salah pengertian dan cenderung dapat kerja dikesalahan pemakai begitu juga dengan yang menggun akan kode tersebut.

7. Hindari spasi

   Spasi dalam kode sebaiknya dihindari, karena dapat menyebabkan kesalahan dalam menggunakannya.

8. Hindari karakter yang mirip

   Karakter-karakter yang hampir serupa bentuk dan bunyi pengucapannya sebaiknya tidak digunakan dalam kode.

Ada beberapa macam tipe kode yang dapat digunakan dalam sistem informasi, antara lain:

a. Kode Mnemonik (*Mnemonic Code*)

   Bertujuan supaya kode mudah diingat, dibuat dengan dasar singkatan atau mengambil sebagian karakter dari *item* yang akan diwakili dengan kode ini.

b. KodeUrut (*Sequential Code*)

   Disebut juga dengan kode seri, merupakan kode yang nilainya urutan antara satu kode dengan kode berikutnya.

c. Kode Blok (*Block Code*)

   Mengklasifikasikan *item* kedalam kelompok blok tertentu yang mencerminkan satu klasifikasi tertentu atas dasar pemakaian maksimum yang diharapkan.

d. Kode Grup (*Group Code*)

   Kode yang berdasarkan *field-field* dan tiap-tiap *field* kode mempunyai arti tertentu.

e. Kode Desimal (*Decimal 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.

C. HIPO (*Hierarchy Plus Input Process Output*)

Menurut Hartono (2005 : 787) HIPO (*Hierarchy Plus Input Process Output*) merupakan "metodologi yang dikembangkan dan didukung oleh IBM". HIPO adalah paket yang berisi sebuah set diagram secara grafis menjelaskan fungsi sebuah

sistem dari tingkat umum ke tingkat khusus. Saat ini HIPO juga telah digunakan sebagai alat bantu untuk merancang dan mendokumentasikan siklus pengembangan sistem. Bagian-bagian HIPO adalah sebagai berikut:

1. *Index* Program

   Merupakan nomor acuan yang menunjukan nomor layar dialog.

2. Nama Program

   Merupakan nama layar dialog atau nama suatu program.

3. *Escape* Program

   Merupakan nomor layar dialog sebelumnya yang akan dituju balik.

Sasaran HIPO:

a. Untuk menyediakan suatu struktur guna memahami fungsi dari sistem.
b. Untuk lebih menekankan fungsi-fungsi yang harus diselesaikan oleh program.
c. Untuk menyediakan penjelasan yang jelas dari *input* dan *output* pada masing-masing tingkatan dari HIPO.
d. Untuk menyediakan *output* yang tepat dan sesuai dengan kebutuhan pemakai.

Ada tiga macam diagram dalam tingkatan HIPO yaitu:

1. *Visual Table Of Contents (VTOC)*

   Diagram ini menggambarkan hubungan dari fungsi-fungsi dalam sistem berjenjang.

2. *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.

3. *Detail* Diagram

   Detail diagram merupakan diagram tingkatan yang paling rendah di diagram HIPO. Diagram ini berisi elemen-elemen dasar dari paket yang menggambarkan secara rinci kerja dari fungsi.

D. Diagram Alur Data (*Flowchart*)

Flowchart menurut Landjamudin (2005 : 263) adalah "Bagan-bagannya yang mempunyai arus yang menggambarkan langkah-langkah penyelesaian suatu masalah". *Flowchart* digunakan terutama untuk alat bantu komunikasi dan untuk dokumentasi dalam membuat suatu algoritma.

Diagram alur dapat menunjukan secara jelas arus pengendalian suatu algoritma, yakni bagaimana pelaksanaan suatu rangkaian secara logis dan sistematis.

Flowchart sendiri terdiri dari tigastruktur,yaitu:

1. Struktur Sederhana (*SquenceStructure*)

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

2. Struktur Percabangan (*Branching Structure*)

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

3. Struktur Perulangan (*Looping Structure*)

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

Jenis *flowchart* atau bentuk-bentuk diagram alur yang sering digunakan dalam proses pembuatan suatu program komputer adalah sebagai berikut:

a. Sistem *Flowchart*

   Bagan alur yang menggambarkan urutan prosedur secara *detail* didalam suatu sistem komputerisasi dan bersifat fisik.

b. Program *Flowchart*

   Simbol-simbol yang menggambarkan proses secara rinci dan *detail* diantaranya instruksi yang lainnya didalam suatu program komputer yang bersifat logis

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

1. *General way*

   Yaitu teknik pembuatan dengan cara ini, lazim digunakan dalam menyusun logika suatu program, yang menggunakan pengulangan proses secara tidak langsung atau *non direct loop*.

2. *Iteration way*

   Yaitu teknik pembuatan yang dipakai untuk logika program yang cepat serta bentuk permasalahan yang kompleks. Dimana pengulangan proses yang terjadi bersifat langsung atau *direct loop*.