BAB II

LANDASAN TEORI

2.1. Konsep Dasar Progam

> konsep pemograman terstruktur merupakan peranan terpenting dalam merancang, menyusun, memelihara dan mengembangkan suatu program, khususnya program aplikasi yang benar dan komplek. Pendekatan terstruktur (*structured design*) dilakukan dengan cara memecah–mecah suatu masalah yang besar dan rumit menjadi beberapa masalah yang lebih kecil dalam bentuk modul–modul sehingga menjadi cukup mudah ditangani,sebagaimana diinginkan (*user*)(yuswanto, 2009:7).

Aplikasi menurut Sutarto (2010:105), "Merupakan perangkat lunak (*software*) yang digunakan untuk tujuan tertentu, sepertimengolah dokumen, mengatur *windows* 7, permainan (*game*) dan sebagai.

Menurut Susilawati (2009:63), "Data kependudukan adalah kumpulan elemen data penduduk yang terstruktur yang diperoleh dari hasil pendaftaran penduduk.

Dari teori diatas dapat disimpulkan, konsep dasar program adalah teori-teori yang membahas secara detail dari judul yang diangkat.

2.1.1. Program

Menurut Yuswanto (2009:8), "Progam merupakan kata, ekspresi, pernyataan dan kombinasi yang disusun dan dirangkai menjadi satu kesatuan prosedur,berupa urutan langkah untuk menyelesaikan masalah yang diimplementasikan dengan mengunakan bahasa pemograman sehingga dapat dieksekusi oleh progam".

Progam menurut Sutarman (2009:3), "Adalah barisan perintah atau intruksi yang disusun sehingga dapat dipahami oleh komputer dan kemudian dijalankan

sehingga perhitungan numerik, dimana barisan perintah tersebut berhingga, berahiran dan menghasilkan *output*".

Berdasarkan uraian definisi diatas, program merupakan barisan perintah yang hanya dibaca oleh komputer dan menghasilkan sebuah keluaran yang berupa tulisan, gambar dan lain-lain.

2.1.2. Bahasa Pemrogaman

Menurut Yuswanto (2009:8), "Bahasa Pemrogaman merupakan prosedur atau tata cara penulisan program".

Bahasa Pemrogaman menurut Shelly dan Vermaat (2012:664), "merupakan sejumlah kata, kode dan simbol yang membuat pemrogaman dapat menyampaikan perintah kepada Komputer".

Sedangkan bahasa *pemrograman* menurut Putra dan Utomo (2011:13), "adalah tatanan aturan standarisasi tentang bagaimana kita menyusun program".*MicrosoftVisual Basic* adalah bahasa program yang digunakan untuk membuat aplikasi *windows* yang berbasis GUI (*Grafhical User Interface*) *visual basic* merupakan konsep *event-driven prograamming*, artinya program menungu sampai adanya respon dari *user* berupa *event* kejadian tertentu (tombol diklik, menu dipilih, dan sebagainya). Ketika *event* terdeteksi, *event* yang berhubungan akan melakukan aksi sesuai dengan kode yang diberikan (Putra dan Utomo, 2011:16).

Dalam penulisan ini bahasa pemograman yang digunakan adalah bahasa pemograman *visual basic* yang mengunakan *microsoft visual basic* sebagai *editor* pembuatan program.

2.1.3 Basis Data

1. Definisi Basis Data

Database menurut Asmiranda dan Fadlisyah (2008:1), "Adalah sekumpulan tabel-tabel yang saling berelasi, relasi tersebut bisa ditunjukan dengan kunci dari tiap tabel yang ada".

Menurut Priyadi (2014:2), "Basis Data adalah sekumpulan fakta berupa reperensi tabel yang saling berhubungan dan disimpan dalam media penyimpanan digital",

Sedangkan menurut Shelly dan Vermaat (2012:514), "Basis Data adalah kumpulan data yang ditata dengan cara yang memungkinkan untuk diakses, dicari dan digunakan datanya",

Berdasarkan uraian diatas, penulis menyimpulkan bahwa Basis Data adalah tempat penampungan data–data yang telah diolah dan siap untuk digunakan.

2. MySQL

MySQL menurut Raharjo (2011:21), "*Software* RDBMS (atau *server database*") yang dapat mengelola *Database* dengan sangat cepat, dapat menampung data dalam jumblah besar, dapat diakses oleh banyak *user* (*multi user*), dan dapat melakukan suatu proses secara sinkron atau berbarengan (*multi-threaded*)".

Menurut Wahana Komputer (2010:16), "MySQL adalah salah satu *software system management database* (DBMS) *Structured Query Language* (SQL) yang bersifat open *source* ".

Menurut Anhar (2010:45), "MySQL (*My Struture Query Langue*) adalah salah satu *Database management System* (DBMS) dari sekian banyak DBMS seperti *Oracle, MS SQL* dan lainnya".

MySQL merupakan *Database Managenent System* (DBMS) yang digunakan penulis sebagai media penyimpanan data.

2.1.4. Model Pengembanan Perangkat Lunak

Model pengembangan perangkat lunak yang digunakan adalah *software development life cycle* (SDLC) dengan model proses *Waterfall* yang dikemukakan oleh Rosa dan Shalahuddin (2015:28). Tahap – tahapannya seperti berikut ini:

1. Analisa kebutuhan perangkat lunak

   Proses pengumpulan kebutuhan dilakukan secara intensif untuk menspefikasikan kebutuhan perangkat lunak agar dapat dipahami perangkat lunak seperti yang dibutuhkan oleh *user*. Pada tahap analisa ini penulis melakukan pengamatan dan wawancara secara langsung kepada petugas yang melayani pembuatan KK untuk memdapatkan imformasi mengenai tahap-tahap pembuatan kartu keluarga.

2. Desain

   Desain perangkat lunak adalah proses multi langkah yang fokus pada pembuatan program perangkat lunak termaksud struktur data, arsitektur perangkat lunak, reperensi antar muka dan proses pengkodean. Tahap ini mentranslasi kebutuhan perangkat lunak dari tahap analisa kebutuhan ke reperensi desain agar dapat diimplementasikan menjadi program pada tahap selanjutnya. Tahapandesain ini dirancang sesuai hasil dari analisa kebutuhan perangkat lunak yang diperlukan oleh petugas yang melayani pembuatan KK

Pada Kantor Kecamatan Menjalin. Serta penulis akan menjelaskan tentang program yang akan dibuat untuk instansi tersebut dengan mengunakan aplikasi *microsoft visual Basic* 6.0.

3. Pembuatan Kode Program

   Desain harus Ditransalasikan kedalam program perangkat lunak. hasil dari tahap ini adalah program komputer sesuai dengan desain telah dibuat pada tahap desain. Untuk pengkodean penulis mengunakan bahasa pemograman penulis *Visual Basic*, sedangkan untuk *Database editor* penulis mengunakan PhpMyAdmin dan MySQL sebagai media penyimpanan data (*Database*).

4. Pengujian

   Pengujian fokus pada perangkat lunak secara dari segi lojik dan fungsional dan memastikan bahwa semua bagian telah diuji. hal ini digunakan untuk meminimalisir kesalahan (*error*) dan memastikan keluaran yang disesuaikan dengan yang diinginkan. Pada tahap pengujian dilakukan dengan cara memeriksa alur kerja program yang dilakukan dengan *flowchart.*

5. Pendukung (*support*) atau pemeliharaan (*maintenance*)

   Tahap pendukung atau tahap pemeliharaan mengulangi proses pengulangan mulai dari analisis spesifikasi untuk perubahan perangkat lunak yang sudah ada, tapi tidak untuk membuat perangkat lunak baru. Tahapan pendukungan dan pemeliharaan pada instansi Kantor Kecamatan menjalin dilakukan untuk memberikan pelayanan yang baik bagi masyarakat demi menciptakan program yang selalu ter*update*dan terjaga keamanannya.

2.2. Peralatan Pendukung (*Tools Program*)

Adapunperalatan pendukung (*Tools Program*) yang digunakan dalam pembuatan program ini dijelaskan sub–sub bab berikut.

A. *Enterprise Relationship Diagram* (ERD)

1. Definisi *Enterprise Relationship Diagram* (ERD)

Sering kita ketahui bahwa ERD atau biasa disebut dengan *entity relationship* diagram merupakan aplikasi pendukung yang nantinya membantu dalam proses pembuatan kode program.

ERD menurut Ladjamudin (2013:142), "adalah suatu model jaringan jaringan yang mengunakan susunan data yang disimpan dalam sistem secara abstrak",

Menurut Simarmata dan Paryudi (2010:67) "*Entity Relationship Diagram* adalah alat permodelan data utama dan akan membantu mengorganisasi data dalam suatu proyek kedalam entitas–entitas dan menentukan hubungan antar entitas",

Sedangkan menurut Raharjo (2011:11), "*Entity Relationship Diagram* (ERD) merupakan salah satu alat bantu (berupa gambar) dalam model *database* relational yang berguna untuk menjelaskan hubungan atau relasi antar tabel yang terdapat didalam *database*",

Dari uraian diatas *Entity Relationship Diagram* (ERD) adalah sebuah alat bantu untuk menjelaskan hubungan atau relasi antar tabel dalam *database*.

2. Komponen *Entity Relationship Diagram* (ERD)

Adapun komponen *Entity Relationship Diagram* (ERD) menurut Simarmata dan Paryudi (2010:67), yaitu:

a) Entitas (*entity*)

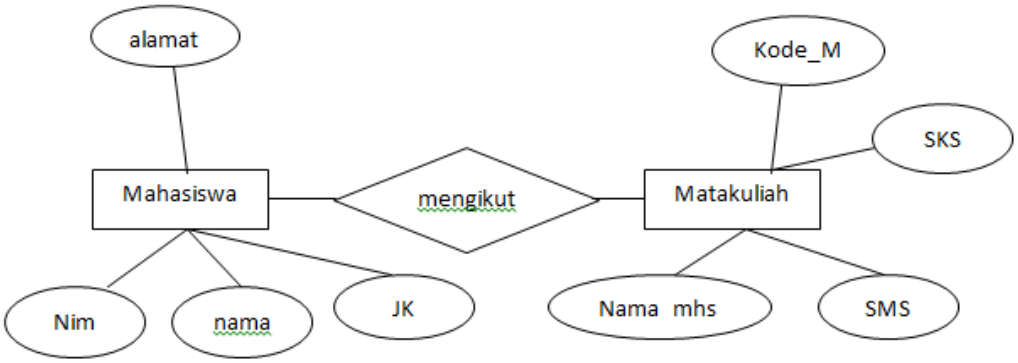
Entitas adalah sesuatu yang nyata dan abstrak dimana kita akan menyimpan data.

b) Relasi (*relationship*)

Relasi adalah hubungan alamiah yang terjadi antara satu atau lebih entitas, misalnya pembayaran pegawai.

c) Atribut (*Attribute*)

Atribut adalah ciri umum semua atau sebagian besar instansi pada entitas tertentu.



Sumber: Supardi (2013:15)

Gambar II.1*Enterprise Relationship Diagram*

3. *Logical Record Structure* (LRS)

Menurut Hasugian dan Shiddiq, (2012:608) mengungkapkan: sebuah model yang digambarkan dengan sebuah *diagram-ER* akan mengikuti pola/aturan permodelan tertentu dalam kaitannya dengan konversi ke LRS, maka perubahan yang terjadi adalah mengikuti aturan–aturan berikut: setiap setiap entitas yang dirubah kebentuk kotak,sebuah atribut relasi disatukan dalam sebuah kotak bersama entitas jika hubungan yang terjadi pada *diagram-ER.*

Aturan–aturan dalam melakukan transformasi E-R diagram ke *logical record structure* menurut Ladjamudin (2013:159) adalah sebagai berikut:

a) setiap *entity* akan diubah kebentuk sebuah kotak dengan nama *entity* berada diluar kotak dan atribut berada didalam kotak.
b) Sebuah relasi kadang disatukan dalam sebuah kotak bersama *entity*, kadang dipisah dalam sebuah tersendiri.

Aturan pokok diatas akan sangat dipengaruhi oleh elemen yang menjadi titik perhatian utama pada langkah transformasi yaitu *cardinality*/kardinalitas.

a) 1:1 (*one to one*)

   pada kardinalitas *one to one*, sebaiknya panah diarahkan ke *entity* dengan jumlah atribut yang lebih sedikit.

b) 1:M (*one to many*)

   Pada kardinalitas relasi*oneto many*, maka relasi harus digabungkan dengan *entity* pada pihak yang *many*, dan tidak perlu melihat sedikit banyaknya atribut pada *entity* tersebut.

c) M:M (*many to many*)

   Pada kardinalitas *many to many*, maka *relationship* berubah status menjadi *file* konektor (yang akan mungubah kardinalitas*many to many* seolah–olah menjadi *one to many*), sehingga baik *entity* maupun relasi akan menjadi struktur *record* tersendiri.

2.2.3. Pengkodean

Dalam proses pembuatan suatu program kita sering mendengar tentang kata pengkodean yang dimana pengkodean ini merupakan *script*yang membantu

dalam pembuatan program, karena tanpa *script* kita tidak akan bisa membuat program.

Menurut Riyanto (2014:3) coding adalah "mengelola file agar program dapat berjalan dengan baik dan terstruktur".

Sedangkan menurut Djiwandono (2015:110) "pengkodean adalah melakukan analisis data dengan menelurkan kode-kode dan memetakan hubungan antara kode-kode tersebut".ada dua jenis pengkodean, yaitu(djiwandono,2015:110):

1. *Open coding*

   Memberikan tanda (dengan garis bawah, lingkaran atau penanda yang lain)pada kata-kata atau prasa yang dianggap mewakili suatu konsep pentingdalam suatu gugus data.

2. *Axial coding*

   Menetapkan beberapa tema/kategori yang mewadahi beberapa kode yang sudah dibuat dalam *open coding.*

Dari uraian diatas pengkodean adalah suatu kegiatan pemberian kode atau simbol pada keterangan-keterangan tertentu, kalau pengolahan akan dilakukan dengan komputer elektronik".

2.2.4. *Hierachy plus Input-Proses-Output* (HIPO)

1. Pengertian HIPO

Menurut Kusbianto (2010:16) "*Hierachy Plus Input-Process-Output* (HIPO) adalah alat dokumentasi program yang berbasis pada fungsi, yaitu tiap-tiap modul didalam sistem digambarkan oleh fungsi utama nya".

Hierachy plus Input-Proses-Output (HIPO) menurut Ladjamudin (2013:211), “merupakan tehnik untuk mendokumentasikan sistem pemograman”.

Berdasarkan kutipan diatas, dapat disimpulkan bahwa HIPO adalah prosedur–prosedur yang nantinya akan dikembangkan seorang *programmer* supaya bisa menjadi progam baku.

2. Tingkat Diagram HIPO

Tingkat Diagram HIPO terdiri atas tiga jenis diagram menurut Ladjamudin. (2013:213), yaitu:

a) Daftar Isi Visual (DIV)

DIV merupakan diagram pertama HIPO yang terdiri satu atau lebih Diagram Hirarki.

b) Diagram Ringkas

Diagram Ringkas ini merupakan diagram kedua pada paket HIPO yang menjelaskan pungsi dan referensi utama yang diperlukan dalam program detail untuk memperluas fungsi sehingga cukup rinci.

c) Diagram Rinci

Diagram rinci HIPO berisikan elemen–elemen dasar sistem, menerangkan fungsi–fungsi khusus, menampilkan item–item *input* dan *output* secara rinci (yaitu nama *fieldinput* yang digunakan dan *output* yang dihasikan), dan memberikan diagram HIPO yang lain seperti *flowchart* dan tabel keputusan dari logika yang rumit.

2.2.5. *Crystal Report*

Menurut MADCOM (2010:10) mengemukakan "*Crystal Report* merupakan program yang terpisah dengan program *microsoft Visual Basic 6.1,* tetapi keduanya dapat dihubungkan (*lingage*) membuat laporan dengan *Crystal Report* banyak tersedia objek–objek maupun komponen yang mudah digunakan".

Penulis menarik kesimpulan bahwa *Crystal Report* adalah aplikasi yang digunakan untuk membuat laporan yang akan digunakan pada progam yang akan dibuat.

2.2.6. ODBC konektor

Menurut wahana komputer (2010:126), "ODBC merupakan sebuah konektor yang fungsi menghubungkan/koneksi *database*mengunakan API (*Aplication Programming Interface*) ODBC disemua *platform Microsoft Windows* dan *unix*".

Penulis menarik kesimpulan bahwa ODBC *Connector* adalah sebuah penghubung antara *Database* dan tampilan antar muka yang sangat berperan penting dalam sebuah aplikasi.

2.2.7. Diagram Alir Data (*flowchart*)

1. Pengertian *flowchart*

Menurut Ladjamudin (2013:263) "*flowchart* adalah bagan–bagan yang mempunyai arus yang mengambarkan langkah–langkah penyelesaian suatu masalah".

Menurut Supardi (2013:51), "*Flowchart* merupakan bagan (*chart*) yang menunjukan alir (*flow*) didalam program atau prosedur sistem secara logika".

Berdasarkan dari penjelasan diatas *flowchart* adalah gambaran umum mengenai sistem berjalan suatu rancangan analisa program untuk menyajikan kegiatan.

2. Bentuk *flowchart*

a) Diagram *flowchart*

Menurut Ladjamudin (2013:263) "program *flowchart* adalah bagan yang memperlihatkan urutan instruksi yang digambarkan dengan simbol tertentu untuk memecahkan suatu masalah dalam suatu progam".

Menurut Supardi (2013:58), "progam *flowchart* merupakan bagan yang menjelaskan secara rinci langkah – langkah dari proses program".

Berdasarkan penjelasan diatas program *flowchart* adalah gambaran dengan simbol tertentu untuk memecahkan masalah dan baga yang menjelaskan secara rinci langkah-langkah dari proses program

b). Sistem *flowchart*

Menurut Ladjamudin (2013:263) sistem *flowchart* adalah "Bagan yang memperlihatkan urutan proses dalam sistem yang menunjukan alat media *input, output* serta jenis media penyimpanan dalam proses pengolahan data".

Menurut supardi (2013:58) "sistem *flowchart* merupakanbagan alir sistem yang menunjukan pekerjaan arus pekerjaan secara keseluruhan dari sistem".

Dari penjabaran diatas Diagram Alir Data (*flowchart*) "adalah prosedur sistem berjalan dalam menjelaskan secara rinci langkah-langkah dari proses pembuatan program secara detail".

3. Tehnik pembuatan

Dalam penulisan Tugas Akhir, penulis mengunakan program *flowchart* sebagai alat pendukung. Adapun tehnik pembuatan progam *flowchart*, sebagai berikut:

a) *General way*

tehnik pembuatan *flowchart* dengan cara ini lazim digunakan dalam menyusun logika suatu program yang mengunakan proses secara tidak langsung (*Non-Direct-loop*).

b) *Interation way*

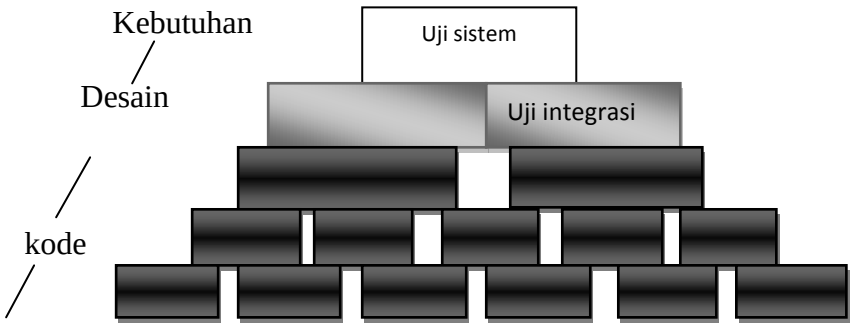
Tehnik pembuatan *flowchart* dengan cara ini biasanya dipakai untuk logika program yang cepat dan juga bentuk permasalahan yang kompleks.

2.2.8. Pengujian Program

Menurut Rosa dan Shalahuddin (2015:272) "Pengujian adalah satu set aktifitas yang direncanakan dan sistematis untuk menguji atau mengevaluasi kebenaran yang diinginkan". Secara umum pola pengujian pada perangkat lunak adalah sebagai berikut:

1. Pengujian dimulai dari *level* komponen hingga integrasi antar komponen menjadi sebuah sistem.

2. Teknik pengujian berbeda-beda sesuai dengan berbagai sisi atau unit uji dalam waktu yang berbeda-beda pula bergantung pada pengujian pada bagian mana yang dibutuhkan.
3. Pengujian dilakukan oleh pengembang perangkat lunak, dan jika untuk proyek besar, pengujian bisa dilakukan oleh tim uji yang tidak terkait dengan tim pengembang perangkat lunak (*independent test group (ITG)*).
4. Pengujian dan penirkutuaan (*debugging*) merupakan aktifitas yang berbeda, tapi penirkutuan (*debugging*) harus diakomodasi pada berbagai strategi pengujian.Pengujian untuk verifikasi dilakukan mulai dari lingkup yang kecil naik ke lingkup yang besar seperti pada gambar berikut.

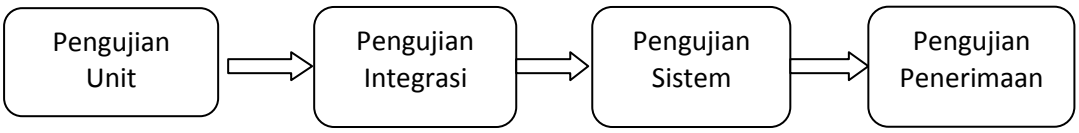


Arah pengujian

Sumber: Rosa dan Shalahudin (2015:274)

Gambar II.2 Hirarki pengujian sistem

Gambar di atas menunjukan tahap pengujian pada *level* program di tangan pengembang perangkat lunak. Tahapan pengujian yang secara keseluruhan adalah sebagai berikut:



Sumber: Rosa dan Shalahudin (2015:274)

Gambar II.3 Pengujian perangkat lunak

Black-box testing (pengujian kotak hitam) yaitu menguji perangkat lunak dari segi spesifikasi fungsional tampa menguji dan desain dan kode program.pengujian dimaksudkan untuk mengetahui