

APLIKASI KONVERSI *FLOWCHART* KE KODE PROGRAM BAHASA PEMROGRAMAN *PL/SQL* *MYSQL*

I Made Sukarsa

Staff Pengajar Teknik Elektro, Fakultas Teknik, Universitas Udayana

Kampus Bukit Jimbaran, Bali, 80361

Email : e_arsa@yahoo.com

Abstrak

Flowchart merupakan bagan yang menjelaskan secara rinci langkah-langkah dari proses program. Orang dengan mudah dapat memahami urutan proses dari program yang dibuat dengan melihat *flowchart* saja. Setelah *flowchart* selesai dibuat barulah dilakukan proses menulis program (*coding*) sesuai dengan *flowchart* yang telah dibuat menggunakan bahasa pemrograman yang diinginkan.

Proses menulis program (*coding*) memerlukan keahlian khusus untuk melakukannya. Orang yang akan melakukan *coding* haruslah mengetahui struktur dari bahasa pemrograman yang digunakan dan paham dengan bahasa pemrograman tersebut, sehingga banyak orang yang tidak bisa melakukannya atau orang yang sedang belajar pemrograman kesulitan dalam melakukannya. Salah satunya terjadi pada saat pemrograman *PL/SQL* pada Mesin *MySQL*. Oleh sebab itu, dibuat aplikasi konversi *flowchart* ke dalam bentuk kode program bahasa pemrograman *PL/SQL* khusus untuk database *MySQL*.

Pada Aplikasi ini, pengguna hanya perlu menggambar *flowchart* pada aplikasi dan kode program *PL/SQL* *MySQL* yang bersesuaian dengan *flowchart* tersebut akan tercipta dan telah diuji dapat berjalan sesuai ketentuan.

Kata kunci : *Flowchart*, *coding*, *database* , *PL/SQL*, *MySQL*.

1 PENDAHULUAN

Program adalah intruksi-intruksi yang diberikan kepada komputer agar komputer dapat melaksanakan tugas-tugas tertentu. Sebuah program berisi sekumpulan kode. Kode-kode yang digunakan dapat bermacam-macam dan tergantung bahasa pemrograman yang digunakan. Adapun upaya untuk membuat program dikenal dengan sebutan pemrograman atau pengkodean. Langkah-langkah dalam pemrograman adalah identifikasi permasalahan, perancangan algoritma, menulis program (*coding*), *testing* dan *debugging*, *implementation*, dan *maintenance*.

Pada tahap perancangan algoritma umumnya dilakukan dengan cara membuat *flowchart* (diagram alir). *Flowchart* merupakan bagan yang menjelaskan secara rinci langkah-langkah dari proses program. Orang dengan mudah dapat memahami urutan proses dari program yang dibuat dengan melihat *flowchart* saja. Setelah *flowchart* selesai dibuat barulah dilakukan proses menulis program (*coding*) sesuai dengan *flowchart* yang telah dibuat menggunakan bahasa pemrograman yang diinginkan.

Proses menulis program (*coding*) memerlukan keahlian khusus untuk melakukannya. Orang yang akan melakukan *coding* haruslah mengetahui struktur dari bahasa pemrograman yang akan digunakan dan paham dengan bahasa pemrograman yang akan digunakan tersebut, sehingga banyak orang yang tidak bisa melakukannya atau orang yang sedang belajar pemrograman kesulitan dalam melakukannya. Banyak orang lebih bisa membuat algoritma (*flowchart*) daripada *coding* karena dapat dilakukan

tanpa pemahaman khusus pada suatu bahasa pemrograman tertentu. Salah satu bahasa pemrograman database yang populer saat ini adalah *PL/SQL*. *PL/SQL* adalah bahasa prosedural yang digunakan untuk mengoptimalkan pembuatan aplikasi *database* relational (*MySQL* atau *Oracle*). *PL/SQL* menjadi kebutuhan yang penting saat ini dalam pemrograman *database* karena kinerja dari eksekusi operasi-operasi *SQL* lebih baik dari bahasa pemrograman lainnya.

Berdasarkan permasalahan tersebut dalam penelitian ini akan dibuat suatu aplikasi konversi *flowchart* ke dalam bentuk kode program bahasa pemrograman *PL/SQL* khusus untuk database *MySQL*.

2 DASAR TEORI

2.1 Bagan Alir Program (*Flowchart*)

Bagan alir program (*flowchart*) merupakan bagan yang menjelaskan secara rinci langkah-langkah dari proses program. Bagan alir program dibuat dari verifikasi bagan alir sistem.

2.2 *PL/SQL* *MySQL*

PL/SQL *MySQL* adalah bahasa prosedural yang digunakan untuk mengoptimalkan pembuatan aplikasi *database* yang menggunakan *database* *MySQL*. Kata *PL* pada *PL/SQL* merupakan singkatan dari *Procedural Language*. Dalam *PL/SQL* dapat digunakan perintah untuk memanipulasi data yang ada dalam *database* *MySQL*. *PL/SQL* *MySQL* membentuk pemrograman terstruktur dalam memproses data. Pada *PL/SQL* ditambahkan

beberapa hal yang dikenal pada dunia pemrograman, seperti variabel, *loop*, pemrosesan kondisi, operasi *cursor*, modularisasai, dan hal-hal lainnya. Semua tambahan itu bertujuan untuk meningkatkan kinerja operasi-operasi SQL pada database MySQL sehingga manfaat dari keahliannya menjadi maksimal.

2.2.1 Struktur PL/SQL MySQL

..

1. Procedure

```
CREATE PROCEDURE procedure1          /* name */
(IN|OUT|INOUT parameter1 INT)        /* parameters */
BEGIN                                /*start of block*/
    DECLARE variable1 CHAR(10);      /* variables */
    ...                               /* statement */
    SET variabel1 = 1;                 /* statement */
    INSERT INTO table1 VALUES (variable1); /* statement */
    ...                               /* statement */
END                                  /* end of block */
```

2. Function

```
CREATE FUNCTION function1            /* name */
(parameter1 INTEGER)                 /* parameters */
RETURNS CHAR(10)                     /* returns value*/
BEGIN                                /*start of block*/
    DECLARE variable1 CHAR(10);      /* variables */
    ...                               /* statement */
    SET variabel1 = "satu";            /* statement */
    INSERT INTO table1 VALUES (variable1); /* statement */
    ...                               /* statement */
END                                  /* end of block */
```

3. Trigger

```
CREATE TRIGGER triggername BEFORE|AFTER INSERT|UPDATE|DELETE
ON tbl_name FOR EACH ROW             /* header */
BEGIN                                /*start of block*/
    DECLARE variable1 CHAR(10);      /* variables */
    ...                               /* statement */
    SET variabel1 = "satu";            /* statement */
    INSERT INTO table1 VALUES (variable1); /* statement */
    ...                               /* statement */
END                                  /* end of block */
```

..

2.3 Finite Automata

Finite automata adalah model matematika sistem dengan masukan dan keluaran diskrit (Bambang Hariyanto,Ir.,MT.,2004). Sistem dapat berada di salah satu dari sejumlah berhingga konfigurasi internal disebut *state*. *State* sistem merupakan ringkasan informasi yang berkaitan dengan masukan-masukan sebelumnya yang diperlukan untuk menentukan perilaku sistem pada masukan-masukan berikutnya.

3 METODE DAN PERANCANGAN SISTEM

3.1 Bahasa Pemrograman

Pada penelitian ini *Database Management System* (DBMS) yang digunakan dalam penyusunan manajemen basisdata adalah MySQL 5. Bahasa pemrograman yang digunakan untuk pembuatan aplikasi ini dan antarmukanya adalah dengan

PL/SQL dapat dibagi menjadi tiga, yaitu: prosedur, fungsi, dan *trigger*. Prosedur dan fungsi bekerja berdasarkan eksekusi langsung dari *user/program*, sedangkan *trigger* akan bekerja secara otomatis apabila terjadi aktivitas *insert*, *update* atau *delete* data.

Struktur dari bahasa pemrograman PL/SQL MySQL (MySQL AB, 2005) adalah sebagai berikut:

menggunakan bahasa pemrograman *Microsoft Visual Basic .NET* dan untuk pembuatan simbol-simbol flowchart menggunakan *Microsoft Visio 11.0 Drawing Control*.

3.2 Daftar Kejadian

Kejadian-kejadian yang akan terjadi pada sistem yang dirancang adalah sebagai berikut :

1. *User* membuka aplikasi dan sebuah jendela akan muncul sebagai pemulaan.
2. *User* memilih menu *New* kemudian menekan salah satu pilihan jenis *flowchart* (*procedure/function/trigger*) untuk memulai penggambaran baru. Untuk penggambaran *flowchart procedure*, *user* akan diberikan jendela baru yang isinya pemberian nama prosedur dan pemilihan database. Untuk penggambaran *flowchart function*, *user* akan diberikan jendela baru yang isinya pemberian nama fungsi,

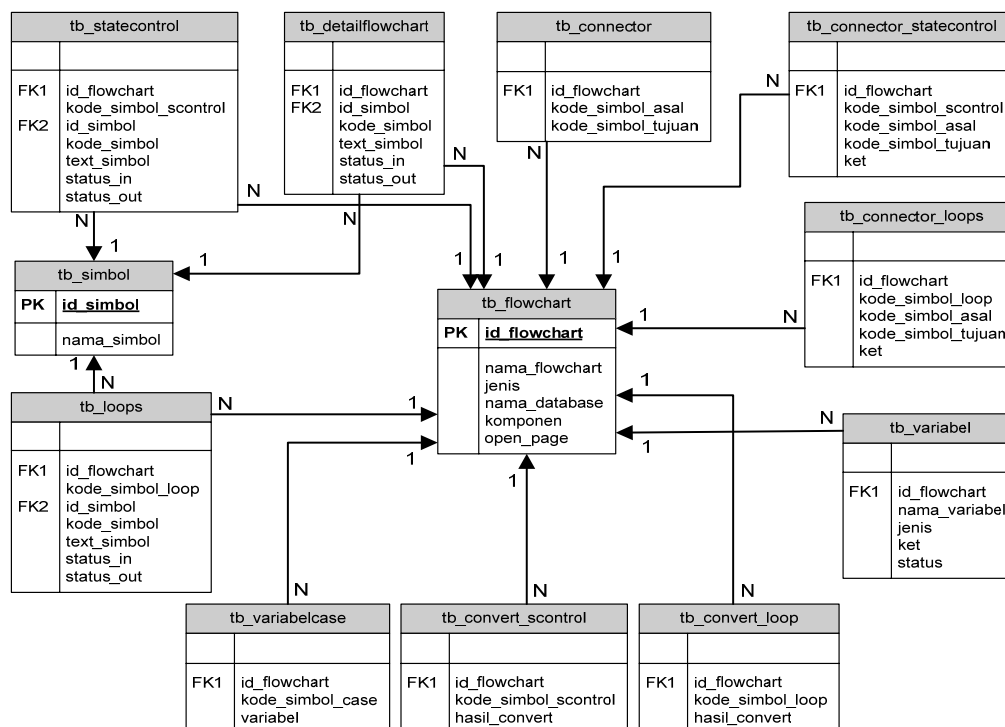
- pemilihan tipe data *Return* dan pemilihan database. Untuk penggambaran *flowchart trigger*, *user* akan diberikan jendela baru yang isinya pemberian nama *trigger*, penentuan waktu dieksekusinya *trigger*, penentuan *event* dieksekusinya *trigger*, tabel *trigger* dan pemilihan database. Setelah itu *user* akan diberikan dokumen baru dengan lahan kerja kosong dan tombol-tombol yang nantinya akan dapat digunakan dalam penggambaran *flowchart*.
- User* menggambar simbol dengan menekan salah satu tombol, untuk beberapa simbol, *user* akan diberikan antar muka yang berfungsi untuk memberikan perintah/informasi dari simbol yang digambar. Setelah informasi diberikan, *user* mendapati lahan kerja telah berisi simbol *flowchart* yang digambar serta informasi yang dimasukkan sebelumnya akan ada didalam simbol tersebut.
 - Untuk penggambaran statemen perulangan dan statemen kontrol, *user* akan diberikan lahan kerja baru yang terpisah dari lahan kerja utama. Pada lahan kerja utama hanya terdapat simbol dari statemen perulangan/ kontrol tersebut, sedangkan gambar detail dari statemen tersebut akan terdapat pada lahan kerja baru tersebut.
 - Koneksi antar simbol dilakukan dengan menekan tombol *connector*, *user* akan diberikan jendela baru yang isinya penentuan simbol asal dan simbol tujuan koneksi. Koneksi antar simbol

harus dilakukan menurut aturan *flowchart*, baik urutan aliran dan arah koneksi ditentukan oleh *user*.

- User* akan melakukan konversi setelah semua *shape* yang ada dalam lahan kerja telah terkoneksi. Hasil dari konversi akan muncul dalam sebuah jendela baru.
- User* dapat mengeksekusi hasil konversi dengan menekan tombol *execute*, hasil dari eksekusi dapat dilihat pada database *MySQL* (berupa prosedur/fungsi/*trigger* sesuai dengan yang dibuat).
- User* dapat menyimpan gambar yang dibuat, gambar *flowchart* akan disimpan dalam bentuk file visio dan informasinya akan disimpan dalam database. Untuk membuka kembali gambar *flowchart* yang sebelumnya dibuat dengan membuka aplikasi dan memilih menu *open*.
- User* dapat melakukan proses *edit* pada gambar *flowchart* yang dibuat. Proses *edit* akan mengubah struktur *flowchart* dan hasil konversinya, jika *user* menghapus salah satu simbol atau mengubah urutan koneksi maka koneksi yang terputus harus disambungkan kembali agar proses konversi dapat dilakukan.

3.3 Gambaran Tabel

Gambaran dari tabel yang digunakan dalam aplikasi yang dibuat adalah seperti gambar 1.



Gambar 1. Gambaran Tabel dari Aplikasi Konversi *Flowchart* ke Kode Program Bahasa Pemrograman *PL/SQL MySQL*

3.4 Flowchart Sistem

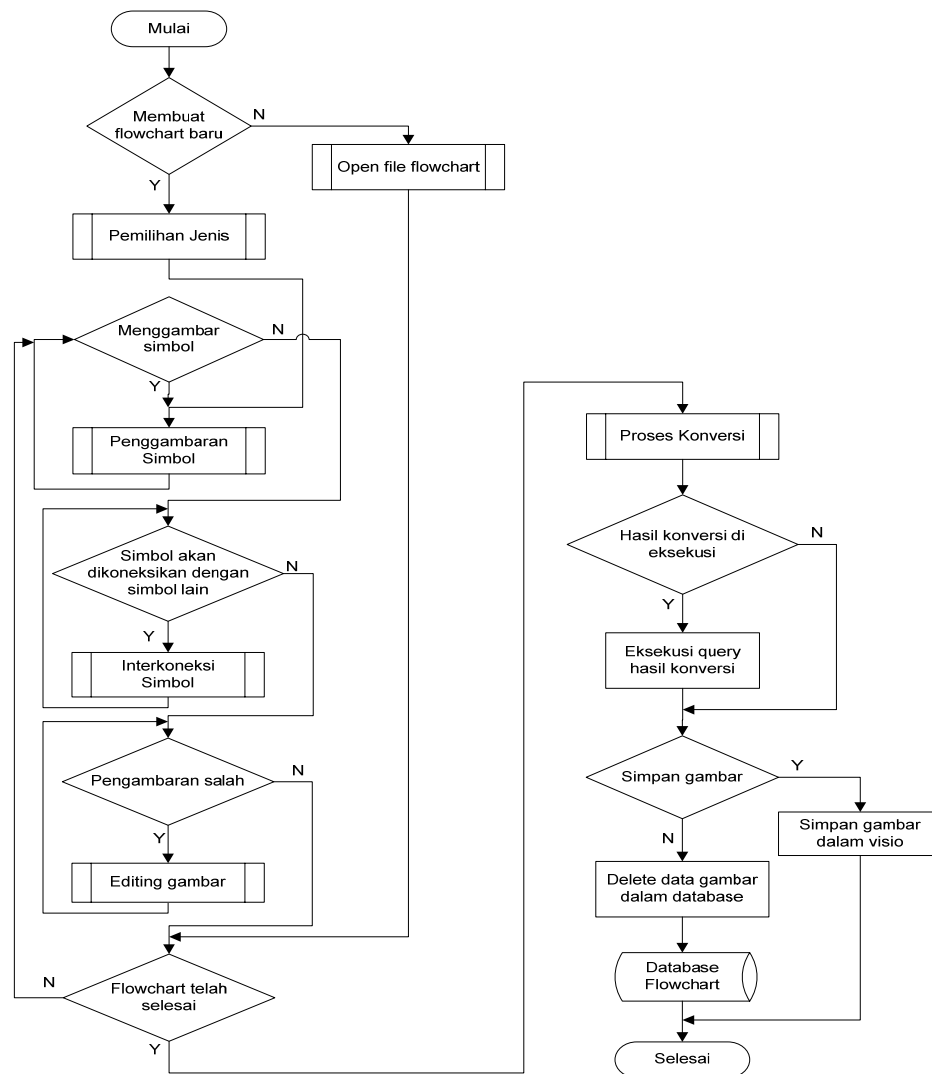
Flowchart atau bagan alir sistem menunjukkan arus dari prosedur pada sistem yang dirancang.

3.4.1 Flowchart Sistem Keseluruhan

Pada proses sistem keseluruhan jika akan membuat *flowchart* baru maka akan dilakukan proses pemilihan jenis dari *flowchart* yang akan dibuat, kemudian dilakukan proses penggambaran simbol *flowchart*. Setelah itu, jika ingin dilakukan interkoneksi antar simbol maka dilakukan proses interkoneksi simbol. Kemudian, jika penggambaran salah dilakukan proses *editing* gambar yang dibuat. Semua proses diatas kecuali proses pemilihan jenis *flowchart* diulang sampai gambar *flowchart* selesai.

..

Jika tidak membuat *flowchart* baru maka akan dilakukan proses *open file flowchart*, kemudian dilakukan proses seperti diatas kecuali proses pemilihan jenis *flowchart* (jika penggambaran *flowchart* belum selesai). Setelah *flowchart* selesai digambarkan maka dilakukan proses konversi. Jika hasil konversi ingin dieksekusi maka dilakukan proses eksekusi *query* hasil konversi, hasil konversi tersebut yang berupa prosedur/fungsi/*trigger* akan tersimpan dalam *database MySQL*. Setelah itu, jika gambar *flowchart* ingin disimpan maka dilakukan proses penyimpanan gambar dalam bentuk file *Visio*, jika tidak ingin disimpan maka data gambar *flowchart* dalam *database* dihapus. Gambar 2 adalah *flowchart* dari proses sistem keseluruhan.

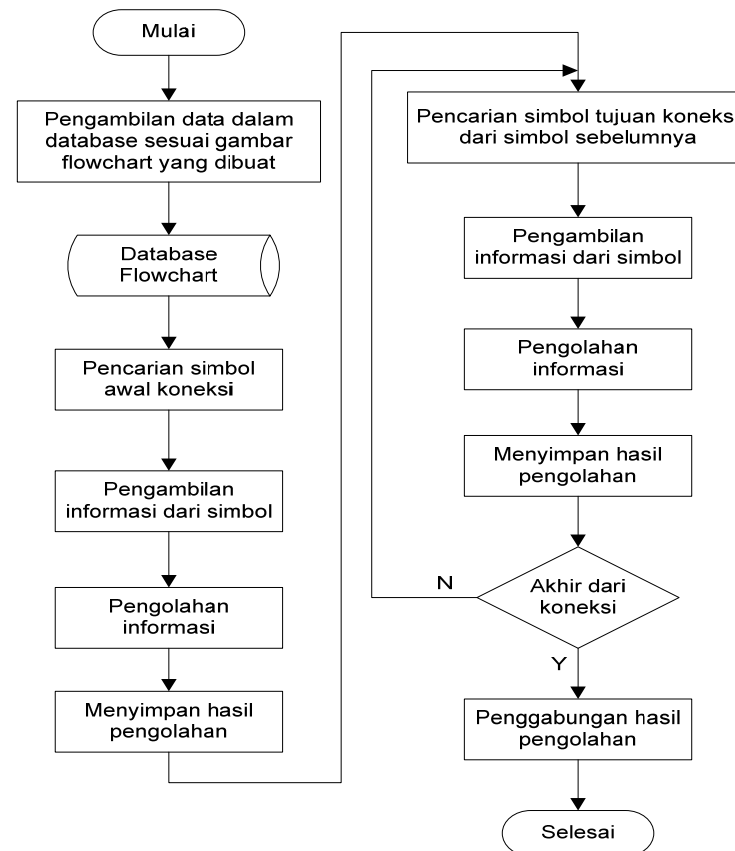


Gambar 2. Flowchart sistem keseluruhan

..

3.4.2 Flowchart Konversi

Proses konversi dimulai dengan pengambilan informasi sesuai dengan *flowchart* yang dibuat. Kemudian dicari kode simbol yang menjadi awal interkoneksi, lalu informasi dari kode simbol tersebut diambil dan diolah menjadi kode program *PL/SQL*, lalu hasil pengolahannya disimpan. Setelah itu, dicari kode simbol yang menjadi tujuan interkoneksi dari simbol yang sebelumnya, kemudian informasi dari kode simbol tersebut diambil dan diolah menjadi kode program *PL/SQL*, lalu hasil pengolahannya ..



Gambar 3 Flowchart konversi

4 PENGUJIAN DAN ANALISIS HASIL

4.1 Gambaran Umum

Aplikasi konversi *flowchart* ke bahasa pemrograman *PL/SQL MySQL* merupakan aplikasi yang digunakan untuk mengkonversi gambar *flowchart* yang dibuat oleh pengguna menjadi kode program bahasa pemrograman *PL/SQL MySQL*. Aplikasi ini terdiri dari proses koneksi ke database *MySQL*, proses pemilihan jenis *flowchart*, penggambaran simbol *flowchart*, *editing* simbol *flowchart*, interkoneksi antar simbol, proses konversi menjadi kode program bahasa pemrograman *PL/SQL MySQL*, eksekusi hasil konversi, penyimpanan gambar *flowchart* dan open gambar *flowchart* yang

disimpan. Proses pencarian kode simbol tujuan interkoneksi, pengambilan informasi dan pengolahannya serta penyimpanan hasil pengolahan terus dilakukan sampai mencapai simbol akhir dari interkoneksi. Setelah mencapai simbol akhir seluruh hasil pengolahan yang disimpan tadi digabung untuk dijadikan kode program *PL/SQL* yang utuh. *Flowchart* proses konversi dapat dilihat pada gambar dibawah ini.

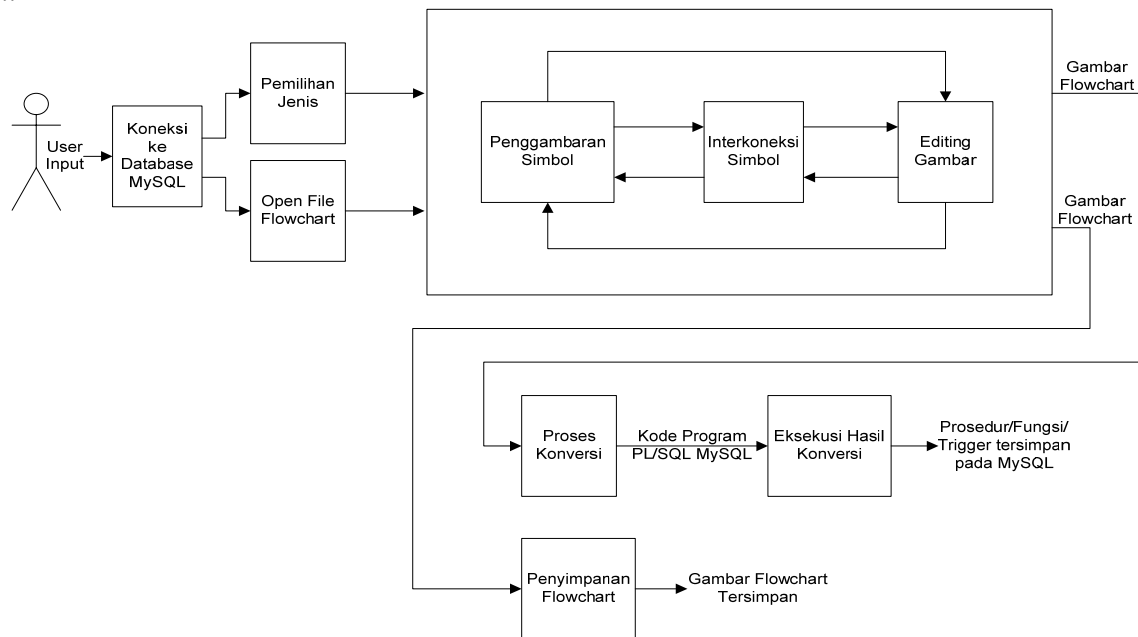
sebelumnya dibuat. Aplikasi ini diharapkan dapat membantu proses pembelajaran pembuatan *flowchart* dan pemahaman tentang bahasa pemrograman *PL/SQL MySQL*.

4.2 Skenario Penggunaan Aplikasi

Penggunaan aplikasi konversi *flowchart* ke bahasa pemrograman *PL/SQL MySQL* ini dibagi menjadi 9 bagian utama yaitu koneksi ke database *MySQL*, pemilihan jenis *flowchart*, penggambaran simbol *flowchart*, *editing* simbol *flowchart*, interkoneksi antar simbol, proses konversi menjadi kode program bahasa pemrograman *PL/SQL MySQL*, eksekusi hasil konversi, penyimpanan *flowchart* dan open gambar *flowchart* yang sebelumnya dibuat.

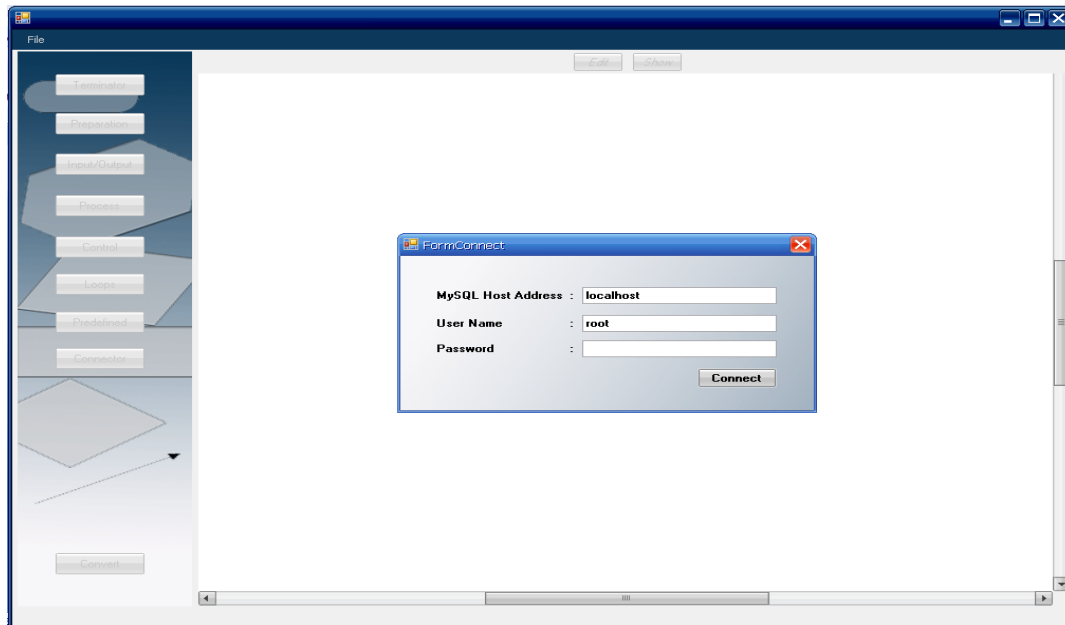
Tampilan awal dari aplikasi ini dapat dilihat pada gambar dibawah ini. Pada awal dijalankannya ..

aplikasi ini, terjadi proses *menge-load master stencil* simbol flowchart Visio



Gambar 4. Skema gambaran umum aplikasi

..



Gambar 5. Tampilan awal aplikasi

..

4.2.1 Skenario Koneksi ke Database MySQL

Proses pertama yang dilakukan dalam menggunakan aplikasi ini adalah proses koneksi ke database MySQL. Proses koneksi dapat dilakukan

pada database MySQL yang ada di komputer lokal atau yang ada pada komputer lain. Ketika tombol 'Connect' diklik, terjadi proses pengecekan *privileges user*, aplikasi ini dapat digunakan jika user

memiliki *privileges* minimal *select*, *insert*, *update* dan *delete*. Setelah itu, terjadi proses koneksi ke *database* 'db_flowchart_plsql', jika *database* tersebut belum ada pada MySQL komputer yang dikoneksikan maka akan dilakukan proses *restore database* 'db_flowchart_plsql' tersebut.

4.2.2 Skenario Pemilihan Jenis Flowchart

Proses pertama yang dilakukan dalam pembuatan *flowchart* pada aplikasi konversi *flowchart* ke bahasa pemrograman PL/SQL MySQL adalah pemilihan jenis *flowchart* yang akan dibuat. Jenis *flowchart* dapat berupa prosedur atau fungsi atau *trigger*. Inputan yang diisikan dalam proses pemilihan jenis *flowchart* ini disimpan dalam tabel 'tb_flowchart'.

4.2.3 Skenario Penggambaran Simbol

Penggambaran simbol *flowchart* pada aplikasi ini dilakukan dengan cara mengklik tombol sesuai dengan simbol *flowchart* yang akan digambarkan. Simbol yang digambarkan dapat berupa *terminator*, *preparation*, *input/output*, *process*, *predefined process*, statemen kontrol dan statemen perulangan. Simbol yang digambarkan akan diberikan nama kode otomatis oleh program dengan awalan sesuai dengan simbol yang dibuat dan diikuti dengan nomer urut dari simbol tersebut (misalkan: T1 untuk simbol *terminator* pertama atau PRE3 untuk simbol *preparation* ketiga). Untuk pembuatan simbol statemen perulangan dan statemen kontrol, dibuat terpisah dari halaman utama. Data dari simbol-simbol yang digambarkan akan disimpan pada tabel 'tb_detailflowchart' jika berada pada halaman utama, pada tabel 'tb_statecontrol' jika berada pada halaman statemen kontrol dan pada tabel 'tb_loops' jika berada pada halaman statemen perulangan.

4.2.4 Skenario Editing

Proses *editing* simbol *flowchart* yang telah dibuat pada aplikasi ini dilakukan dengan cara memilih salah satu simbol yang akan di-*edit* kemudian mengklik tombol 'edit' yang ada. Proses yang terjadi pada saat tombol 'edit' diklik adalah proses mencari nama dari simbol yang diklik (kode simbol), kemudian mencari isi dari simbol berdasarkan kode simbol tersebut. Isi simbol ini diambil dari *database* pada tabel 'tb_detailflowchart' field 'text_simbol' jika penggambaran pada halaman utama. Jika penggambaran pada halaman statemen kontrol, isi simbol ini diambil dari tabel 'tb_statecontrol' field 'text_simbol' dan jika penggambaran pada halaman statemen perulangan, isi simbol diambil dari tabel 'tb_loops' field 'text_simbol'. Setelah selesai dilakukan perubahan maka data pada tabel akan di-*update*.

4.2.5 Skenario Interkoneksi Antar Simbol

Proses interkoneksi antar simbol digunakan untuk mengkoneksikan simbol-simbol yang digambarkan. Proses ini dilakukan dengan cara mengklik tombol 'connector' yang ada, lalu menentukan simbol asal dan tujuan interkoneksi kemudian menghubungkannya. Data dari proses interkoneksi disimpan pada *database*, jika proses interkoneksi pada halaman utama maka disimpan pada tabel 'tb_connector', jika pada halaman statemen kontrol maka disimpan pada tabel 'tb_connector_statecontrol' dan jika pada halaman statemen perulangan disimpan pada tabel 'tb_connector_loops'.

4.2.6 Skenario Proses Konversi

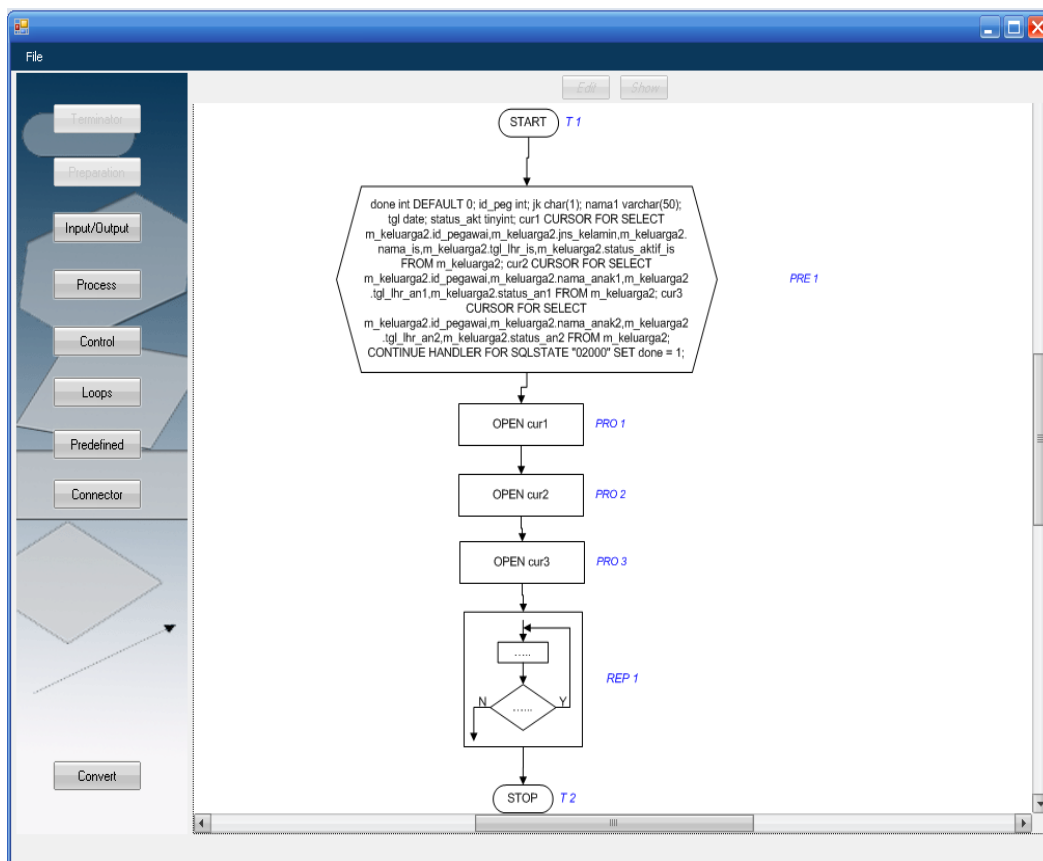
Proses konversi dilakukan jika gambar *flowchart* sudah selesai dibuat. Proses konversi ini dilakukan dengan mengklik tombol 'Convert'.

Ketika tombol 'Convert' diklik proses konversi dilakukan yaitu dengan cara sebagai berikut:

1. Pertama dicari simbol tujuan dari simbol pertama interkoneksi yaitu simbol *terminator start* pada tabel 'tb_connector'.
2. kemudian dicari id_simbol dan isi dari simbol tujuan interkoneksi tersebut pada tabel 'tb_detailflowchart'. Id simbol digunakan untuk mengetahui jenis dari simbol tujuan tersebut.
3. Jika simbol tujuan tersebut berupa simbol *preparation* maka isi simbol yang merupakan data variabel atau *cursor* dipisahkan per variabel/*cursor*. Kemudian ditambahkan kata 'Declare' pada awal data variabel/*cursor*, lalu disimpan. Jika simbol tujuan berupa simbol *input output* maka isi simbol disimpan di dalam sebuah variabel untuk input output. Jika simbol tujuan berupa simbol *process* maka isi simbol disimpan di dalam sebuah variabel untuk proses tanpa menghilangkan isi variabel sebelumnya (jika ada). Jika simbol berupa simbol *predefined process* maka ditambahkan kata 'Call' pada awal isi simbol lalu disimpan pada variabel proses tadi tanpa menghilangkan isi variabel sebelumnya (jika ada). Jika simbol berupa statemen kontrol maka hasil konversi dari statemen kontrol yang sebelumnya sudah dilakukan yang disimpan pada tabel 'tb_convert_scontrol' diambil, lalu disimpan pada variabel proses tanpa menghilangkan isi variabel sebelumnya (jika ada). Jika simbol berupa statemen perulangan maka hasil konversi dari statemen perulangan yang sebelumnya sudah dilakukan yang disimpan pada tabel 'tb_convert_loop' diambil, lalu disimpan pada variabel proses tanpa menghilangkan isi variabel sebelumnya (jika ada).

4. Pertama dicari simbol tujuan dari simbol pertama interkoneksi yaitu simbol *terminator start* pada tabel 'tb_connector'.
5. kemudian dicari id_simbol dan isi dari simbol tujuan interkoneksi tersebut pada tabel 'tb_detailflowchart'. Id simbol digunakan untuk mengetahui jenis dari simbol tujuan tersebut.
6. Jika simbol tujuan tersebut berupa simbol *preparation* maka isi simbol yang merupakan data variabel atau *cursor* dipisahkan per variabel/*cursor*. Kemudian ditambahkan kata 'Declare' pada awal data variabel/*cursor*, lalu disimpan. Jika simbol tujuan berupa simbol *input output* maka isi simbol disimpan di dalam sebuah variabel untuk input output. Jika simbol tujuan berupa simbol *process* maka isi simbol disimpan di dalam sebuah variabel untuk proses tanpa menghilangkan isi variabel sebelumnya (jika ada). Jika simbol berupa simbol *predefined process* maka ditambahkan kata 'Call' pada awal isi simbol lalu disimpan pada variabel proses tadi tanpa menghilangkan isi variabel sebelumnya (jika ada). Jika simbol berupa statemen kontrol maka hasil konversi dari
- statemen kontrol yang sebelumnya sudah dilakukan yang disimpan pada tabel 'tb_convert_scontrol' diambil, lalu disimpan pada variabel proses tanpa menghilangkan isi variabel sebelumnya (jika ada). Jika simbol berupa statemen perulangan maka hasil konversi dari statemen perulangan yang sebelumnya sudah dilakukan yang disimpan pada tabel 'tb_convert_loop' diambil, lalu disimpan pada variabel proses tanpa menghilangkan isi variabel sebelumnya (jika ada).
7. Simbol yang sebelumnya menjadi simbol tujuan dijadikan simbol awal, setelah itu kembali pada proses no.2. Proses ini terus diulang-ulang sampai simbol akhir dari interkoneksi. Jika sudah mencapai simbol akhir interkoneksi maka proses selanjutnya dilakukan (no.5).
8. Proses selanjutnya adalah dari sisi variable yang telah disimpan dengan format berikut. Dan hasil konversi ini ditampilkan pada gambar 7.

..



Gambar 6. Tampilan ketika gambar *flowchart* telah jadi

Untuk Flowchart prosedur:

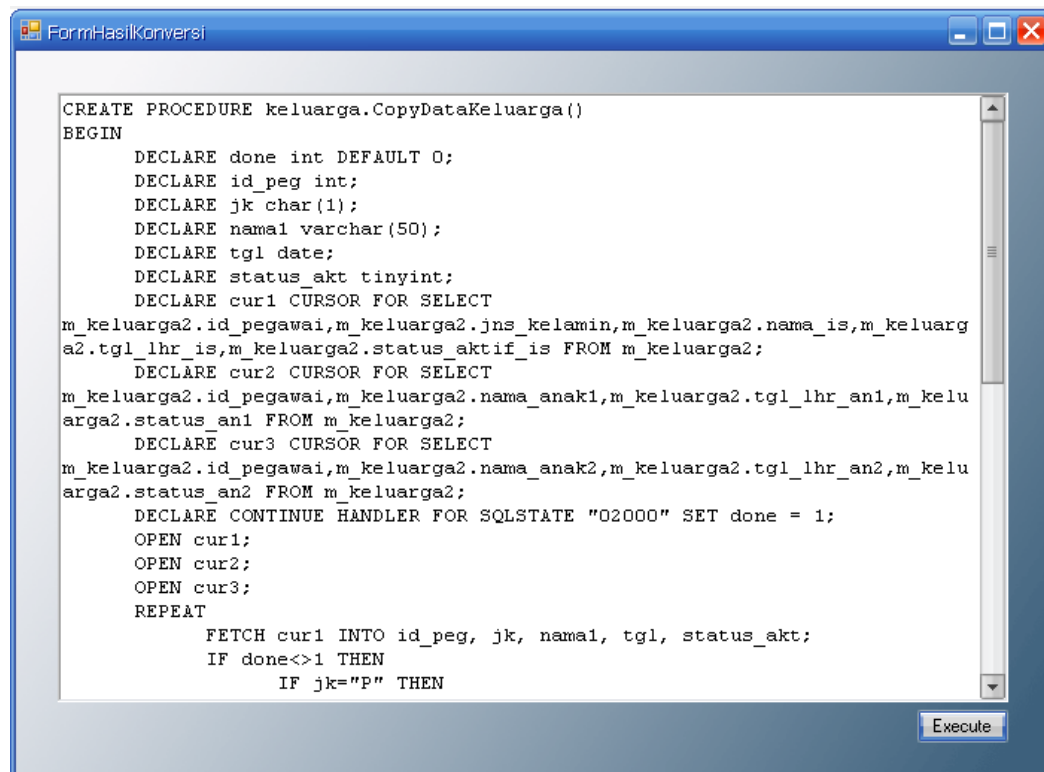
```
CREATE PROCEDURE nama_database. nama_flowchart
(isi dari variabel input output)
BEGIN
    Isi dari variabel preparation
    Isi dari variabel proses
END;
```

Untuk Flowchart fungsi:

```
CREATE FUNCTION nama_database. nama_flowchart
(isi dari variabel input output) RETURN tipe_data_return
BEGIN
    Isi dari variabel preparation
    Isi dari variabel proses
END;
```

Unntuk flowchart trigger:

```
CREATE TRIGGER nama_database. nama_flowchart
komponen_trigger yang dipilih pada awal pembuatan flowchart
FOR EACH ROW BEGIN
    BEGIN
        Isi dari variabel preparation
        Isi dari variabel proses
    END;
```



Gambar 7. Tampilan form hasil konversi

..

4.2.7 Skenario Eksekusi Hasil Konversi

Proses ini berfungsi untuk mengeksekusi *query* hasil konversi. Setelah eksekusi *query* dilakukan maka prosedur atau fungsi atau *trigger* yang dibuat akan tersimpan pada *database MySQL*.

4.2.8 Skenario Penyimpanan Flowchart

Jika dilakukan penyimpanan pada gambar *flowchart* yang dibuat maka gambar *flowchart* yang dibuat akan disimpan dalam bentuk *file Visio (*.vsd)*, yang nama *file*-nya adalah *id flowchart* dari gambar *flowchart* yang dibuat diikuti dengan nama *flowchart*-nya. Jika gambar *flowchart* tidak disimpan, maka akan terjadi proses penghapusan data *flowchart* pada *database* sesuai dengan *id_flowchart* gambar *flowchart* yang dibuat.

4.2.9 Skenario Open Gambar Flowchart yang Sebelumnya Dibuat

Proses *open* gambar *flowchart* yang sebelumnya dibuat dilakukan dengan cara: pertama dilakukan proses pencarian *id_flowchart* dari nama *flowchart* yang dipilih. Kemudian terjadi proses *open file* gambar *flowchart* dengan nama *id_flowchart* diikuti dengan nama *flowchart* yang dipilih ditambah ekstensi *.vsd*. Setelah gambar *flowchart* dibuka, terjadi proses *peng-copy-an* seluruh simbol dan halaman yang ada pada *file* gambar *flowchart* yang dibuka lalu *mem-paste* pada dokumen *kanvas* yang aktif. Setelah itu, *file* gambar *flowchart* yang dibuka tadi di-*close*.

5 PENUTUP

5.1 Simpulan

Terdapat beberapa simpulan yang diperoleh selama dilakukan proses perancangan dari Aplikasi Konversi *Flowchart* ke Kode Program Bahasa Pemrograman *PL/SQL MySQL* antara lain sebagai berikut:

1. Aplikasi ini merupakan suatu aplikasi yang digunakan untuk membantu masyarakat/orang-orang yang ingin belajar pemrograman *PL/SQL MySQL* untuk pembuatan kode program *PL/SQL MySQL* dari sebuah *flowchart* yang dibuat.
2. Aplikasi Konversi *Flowchart* ke Kode Program Bahasa Pemrograman *PL/SQL MySQL* terdiri dari proses koneksi ke *database MySQL*, pemilihan jenis *flowchart*, penggambaran simbol *flowchart*, *editing* simbol *flowchart*, interkoneksi antar simbol, proses konversi menjadi kode program bahasa pemrograman *PL/SQL MySQL*, eksekusi hasil konversi, penyimpanan gambar *flowchart* dan *open* gambar *flowchart* yang sebelumnya dibuat.
3. Pada aplikasi ini, penggambaran statemen kontrol dan statemen perulangan dibuat terpisah dari gambar utama. Pada gambar utama statemen kontrol dan statemen perulangan ini diwakili oleh satu simbol.

4. Validasi pada aplikasi mencakup proses pemilihan jenis *flowchart*, proses penggambaran simbol, proses interkoneksi dan proses *editing*. Validasi ini digunakan agar *flowchart* yang digambar dapat dikonversi dengan baik dan benar.
5. Pertumbuhan data dalam *database* pada aplikasi Konversi *Flowchart* ke Kode Program Bahasa Pemrograman *PL/SQL MySQL* merupakan pertumbuhan data rendah, karena penambahan data perharinya tidak terlalu signifikan.

5.2 Saran

Guna pengembangan sistem lebih lanjut, maka ada beberapa hal yang perlu diperhatikan antara lain sebagai berikut :

1. Dalam pengembangan Aplikasi Konversi *Flowchart* ke Kode Program Bahasa Pemrograman *PL/SQL MySQL*, hendaknya dapat mengkonversi *flowchart* untuk pemrograman yang lebih kompleks.
2. Dalam pengembangan Aplikasi Konversi *Flowchart* ke Kode Program Bahasa Pemrograman *PL/SQL MySQL*, hendaknya berbasis web agar dapat diakses dari mana saja.

6 DAFTAR PUSTAKA

- [1] Gulutzan, P. 2005. *MySQL 5.0 Stored Procedures*. MySQL AB.
- [2] Hariyanto, B. 2004. *Teori Bahasa, Otomata, dan Komputasi serta Terapannya*. Informatika: Bandung.
- [3] Hartono, J. 1999. *Analisis dan Desain Sistem Informasi: Pendekatan Terstruktur Teori dan Praktik Aplikasi Bisnis*. Andi: Yogyakarta.
- [4] Kadir, A. 1999. *Pemrograman Pascal Buku 1*. Andi: Yogyakarta.
- [5] Mackenzie, dkk. 2004. *Belajar Sendiri dalam 21 Hari Visual Basic .NET*. Andi: Yogyakarta.
- [6] Microsoft Corporation. 2003. *Microsoft Office Visio 2003 SDK Documentation*.
- [7] Rickyanto, I. 2003. *Membuat Aplikasi Windows dengan Visual Basic. NET*. PT Elek Media Komputindo: Jakarta.
- [8] Sidik, B. 2005. *MySQL Untuk Pengguna, Administrator, dan Pengembangan Aplikasi Web*. Informatika: Bandung.