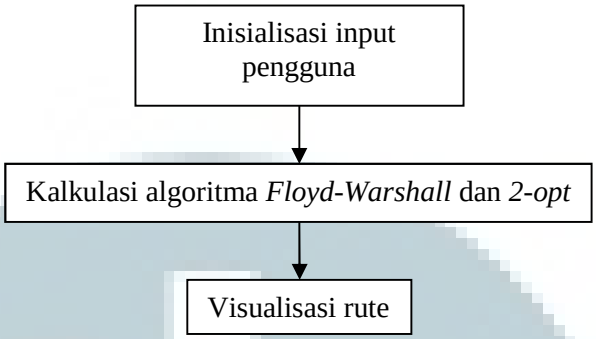
BAB III

ANALISIS DAN PERANCANGAN

Pada bab ini akan dipaparkan analisis yang dilakukan dalam pengerjaan skripsi ini. Analisis diawali dengan pengenalan tentang gudang yang dijadikan tempat prnyimpanan barang, lalu dilanjutkan dengan pengenalan terhadap konsep metode *graph*, algoritma *Flyod Warshall* dan *2-opt* yang diperlukan berdasarkan teori yang telah dijelaskan pada bab 2. Setelah itu, dilakukan analisis terhadap optimasi yang akan menjadi fokus dalam skripsi ini. Sedangkan tahap implementasi akan dilanjutkan di bab 4.

3.1 Analisis Kebutuhan Sistem

Analisis kebutuhan untuk mendapatkan hasil terakhir dari sistem yang dibuat adalah analisis mengenai berbagai tahapan proses yang dibutuhkan oleh sistem untuk mendapatkan hasil yang diinginkan. Berikut ini adalah sebuah gambaran secara umum mengenai tahapan-tahapan analisis sistem aplikasi penentuan rute optimum menggunakan algoritma *Floyd-Warshall* dan *2-opt* yang akan dibuat, yaitu:



Gambar 3.1. *Global Flowchart* Analisis kebutuhan sistem

Bebagai metode analisis dilakukan dalam semua proses tersebut untuk menghasilkan sebuah rute pengambilan barang yang optimum melalui sistem aplikasi penentuan rute pengambilan barang di gudang. Sistem secara umum akan membutuhkan sebuah unit input dari pengguna untuk mendapatkan data berupa rute yang akan digunakan dalam proses pengambilan barang digudang.

3.1.1 Metode Analis Algoritma *Floyd Warshall* dan *2-opt*

Aplikasi untuk menentukan rute optimum ini dirancang dengan menggunakan metode *graph* kemudian menggunakan algoritma *Floyd-Warshall* dan *2-opt*. Untuk melihat proses masukan (input) dan proses keluaran (output) dinyatakan dengan tampilan pada komputer yang dibuat dengan menggunakan bahasa pemrograman C#.Net (*C Sharp* dot net) yang akan diperjelas dengan diagram alur (*Flow Chart*). Pada tahapan ini digunakan notasi-notasi untuk menggambarkan arus data dimana akan sangat membantu dalam proses komunikasi dengan pengguna (*user)*.

3.1.2 Analisis Kebutuhan Proses

Kebutuhan proses dalam sistem penentuan rute optimum antara lain:

1. Proses pembuatan *node* atau titik pengambilan barang dan *vertex* atau rute pada denah gudang.
2. Proses penentuan tempat pengambilan barang selanjutnya.
3. Proses perhitungan jarak tempuh masing-masing titik pengambilan barang.
4. Proses penyeleksian jarak terpendek.

3.1.3 Analisis Kebutuhan Input

Input atau masukkan dari aplikasi penentuan rute terpendek ini berupa parameter-parameter yang diperlukan dalam metode *Graph*, Algoritma *Floyd Warshall* dan *2-opt* yaitu:

1. Data *Graph* berupa banyaknya *node* atau titik (n) termasuk koordinat (x,y).

   Simbol n merupakan banyaknya node atau titik pengambilan barang yang masuk ke dalam daftar pengambilan barang. Proses ini menggunakan peta gudang yang berbentuk dua dimensi yang menggunakan titik ordinat X dan Y yang menentukan titik pengambilan barang dari peta gudang tersebut. Pengguna (*user)* mengisikan kode-kode barang yang akan diambil yang nantinya akan muncul titik koordinat pengambilan barang sampai kembali lagi ke titik asal yang bertujuan untuk mencari jarak yang paling optimal.
2. Jarak dari tiap-tiap node atau titik.

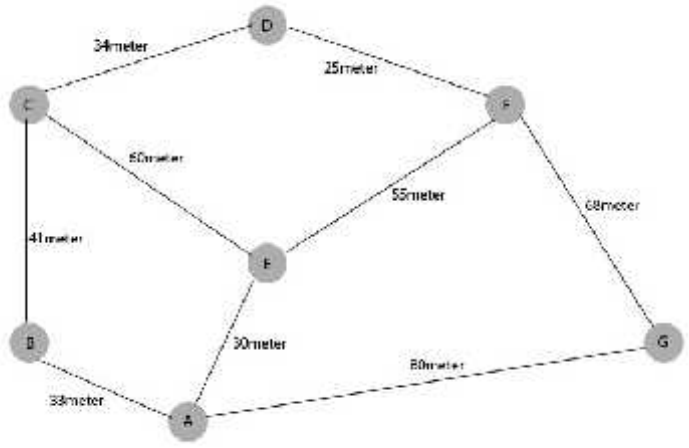
   Dalam pembuatan program aplikasi ini dibutuhkan data berupa jarak dari tiap-tiap titik pengambilan barang di dalam gudang.

3.1.4 Analisis Kebutuhan Output

Data keluaran yang diperoleh dari proses aplikasi penentuan rute optimum ini adalah rute optimum dari titik atau koordinat pengambilan barang yang telah ditentukan. Rute berupa peta dua dimensi yang disertai dengan jarak tempuh minimum dari tiap-tiap koordinat yang telah ditentukan.

3.2 Studi Kasus

Terdapat suatu *graph* berbobot yang merepresentasikan kondisi keterhubungan antar posisi pengambilan barang digudang dengan ilustrasi sebagai berikut:



Gambar 3.2. Representasi keterhubungan antar posisi pengambilan barang

Misalkan seseorang harus mengambil barang di A, B, dan C. Berikut adalah cara perhitungan dengan menggunakan metode algoritma *Floyd-Warshall*

Tahap 1:

$f_1(s) = cx_1s$

Perhitungan tahap 1 dapat kita lihat pada tabel 3.1.

Tabel 3.1. Tabel perhitungan tahap 1

S	Solusi optimum	
	f_1(s)	x_1
B	33	A
E	30	A
G	80	A

Tahap 2:

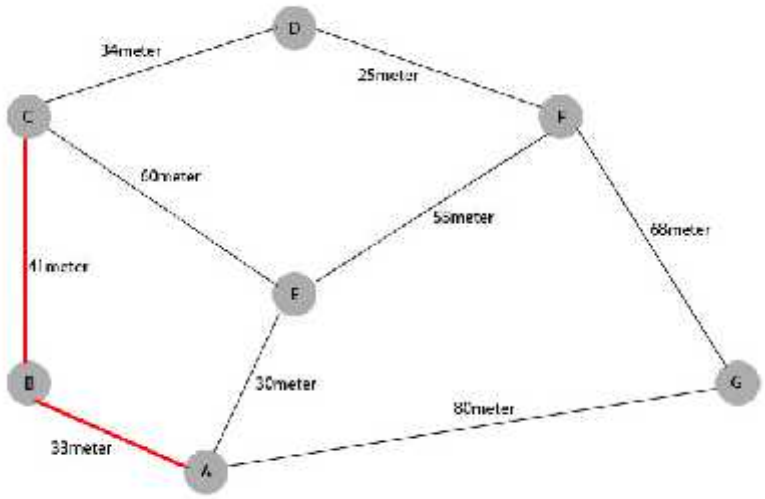
$f_2(s) = \min s_2 \{cx_2s + f_1(x_2)\}$

Pada perhitungan di tahap 2 ini ditambahkan nilai 30 meter hasil perhitungan dari tabel 3.1 ke posisi-posisi berikutnya. Hasil perhitungan tahap 2 dapat dilihat seperti pada tabel 3.2.

Tabel 3.2. Tabel perhitungan tahap 2

x_2s	$f_2(x_2,s) = cx_2s + f_1(x_2)$			Solusi optimum	
	B	E	G	f_2(s)	x_2
C	74	90	∞	74	B
F	∞	85	148	148	G

Pada tabel 3.2 kita sudah ditemukan hasil jarak ke posisi C, yaitu posisi B ke posisi C sebesar 74 meter, dan posisi E ke posisi C 90 meter. Maka dari hasil perhitungan pada tabel 3.2 tersebut dipilih rute dari posisi B ke posisi C, karena jaraknya yang paling optimal. Dari hasil pencarian jalur terpendek dari posisi A ke posisi C menggunakan algoritma *Floyd-Warshall* (pemrogramnan dinamis), ditentukan bahwa jarak terpendek dari posisi A menuju posisi C adalah 74 meter dengan jalur A – B – C. Hasil representasi *graph* dengan metode *Floyd-Warshall* dapat dilihat pada gambar 3.4.



Gambar 3.3. Repserentasi keterhubungan antar posisi dengan algoritma *Floyd-Warshall*

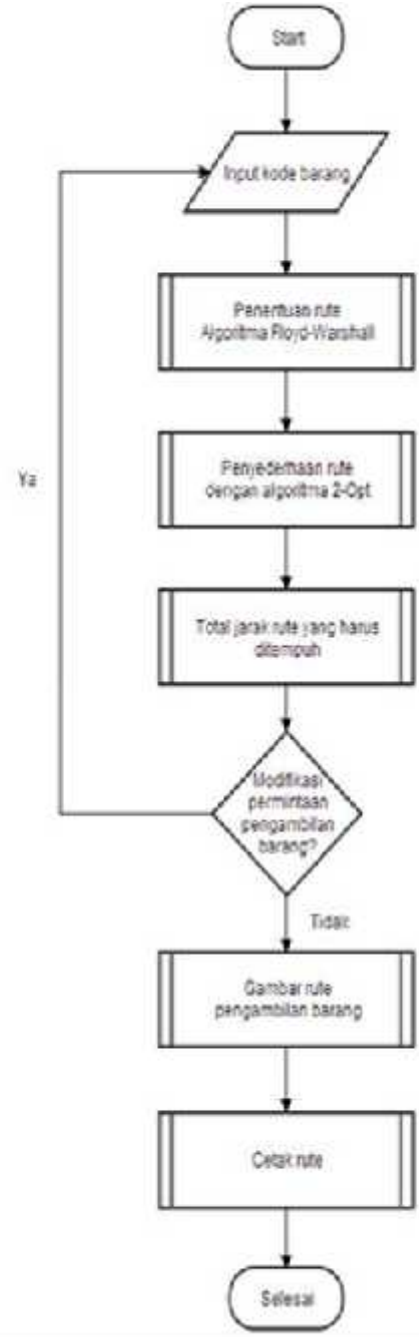
3.3 Perancangan sistem

Sistem dirancang sedemikian rupa untuk memenuhi berbagai kebutuhan dan tuntutan yang digunakan untuk menghasilkan sebuah aplikasi rute pengambilan barang di gudang. Berbagai *form* dan fungsi akan diberikan untuk membantu pengguna memenuhi keinginannya.

Berikut ini akan dijelaskan detail rancangan sistem yang dibuat pada program aplikasi ini meliputi sebuah *form* utama untuk menghasilkan rute pengambilan barang yang optimum.

3.3.1 Rancangan Program

Pada bagian rancangan program ini digunakan *flowchart* dan *sequence diagram* untuk menunjukkan alur kontrol dan jalannya program. Berikut ini adalah *flowchart* yang menunjukkan alur kontrol pada setiap modul.

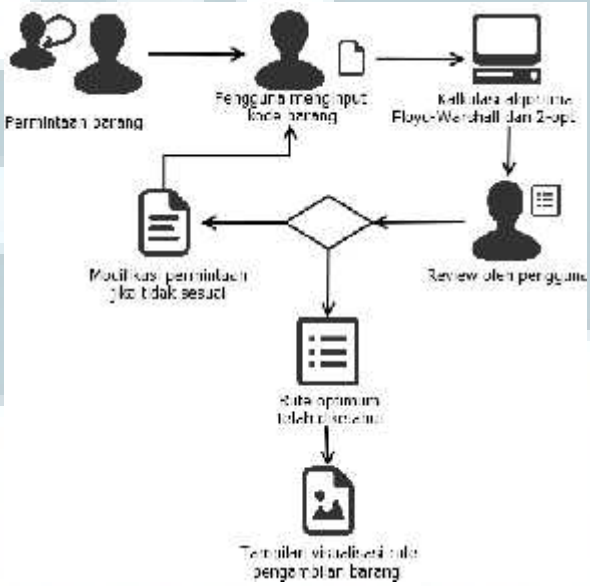


Gambar 3.4. *Flowchart* Sistem Kontrol *Modul*

Secara umum modul sistem akan berjalan sesuai dengan *flowchart* di atas. Pertama pengguna akan menginput kode-kode barang di gudang yang akan diambil. Setelah itu pengguna menjalankan kalkulasi algoritma *Floyd-Warshall*

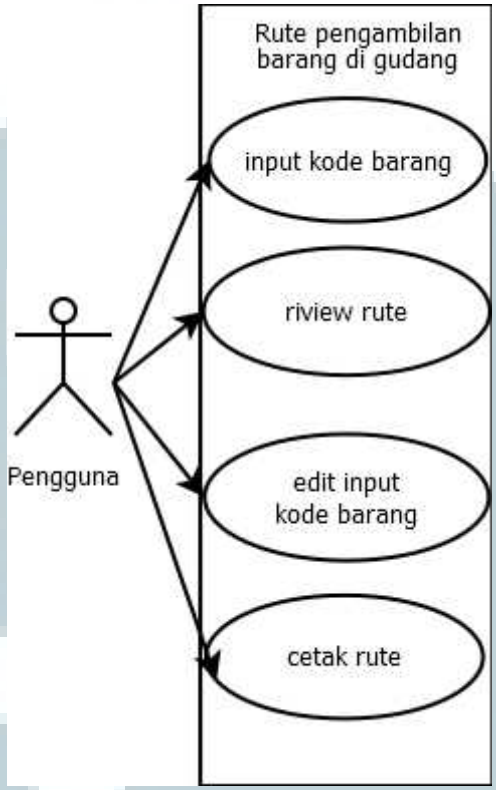
dan *2-opt*. Setelah itu pengguna menentukan apakah hasil kalkulasi tersebut telah sesuai seperti yang diinginkan. Bila telah sesuai maka pengguna akan menampilkan rute yang terbentuk dan mengeksport ke dalam gambar atau denah gudang yang sudah diberikan *node* atau titik berikut *vertex* dalam pengambilan barang yang kemudian dicetak. Bila tidak sesuai, maka pengguna bisa mengulangi proses tersebut.

a. *Rich Picture*



Gambar 3.5. *Rich picture* dari aplikasi penentuan rute pengambilan barang

b. *Use Case Diagram*

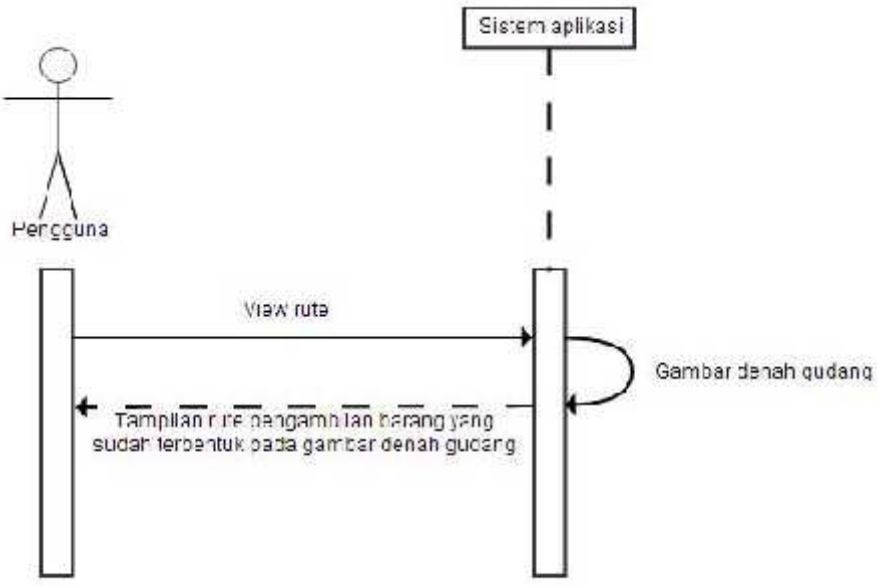


Gambar 3.6. *Use Case Diagram* aplikasi penentuan rute pengambilan barang

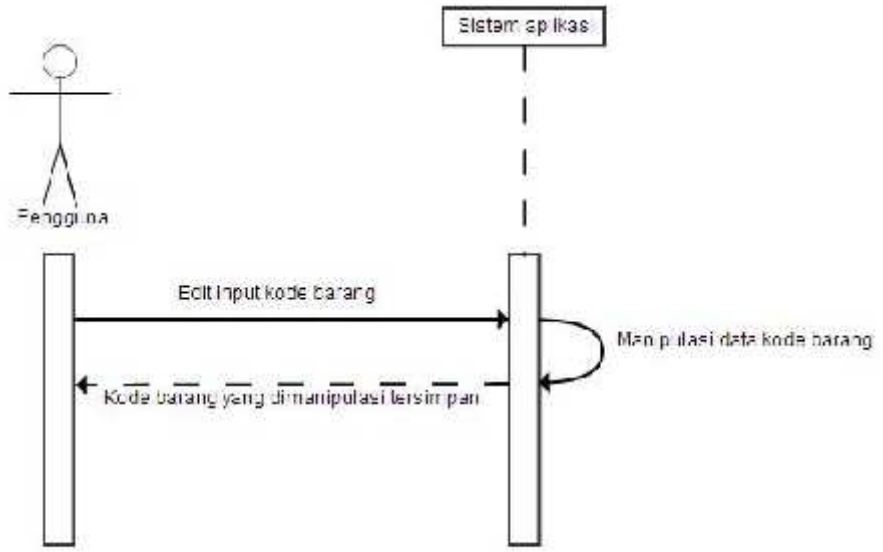
c. *Sequence Diagram*

Sequence diagram di bawah ini dijelaskan tentang aliran data atau program yang diproses oleh sistem. Gambar 3.6 menggambarkan urutan kejadian interaksi pengguna aplikasi dengan GUI.

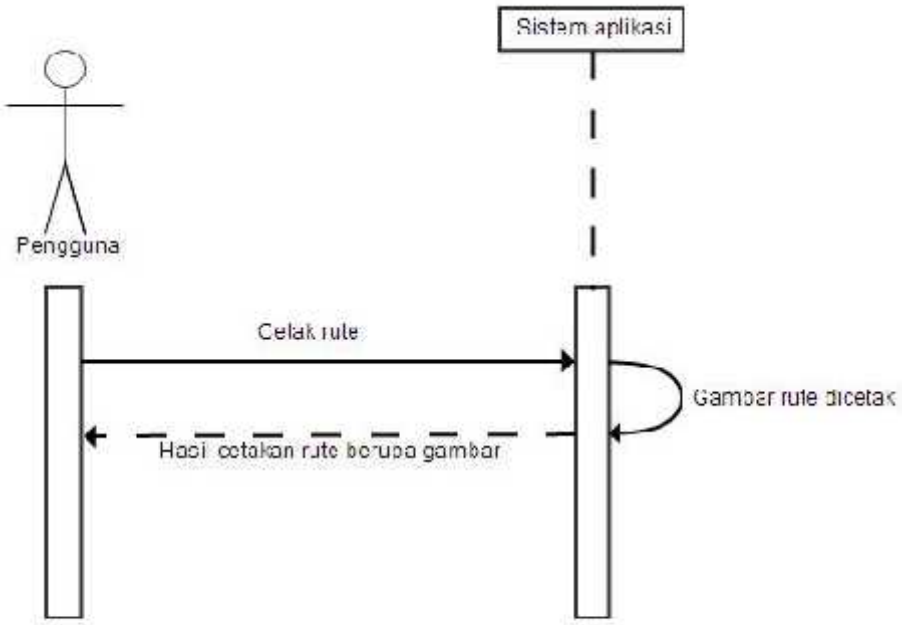
Gambar 3.7. *Sequence Diagram* input dan inisialisasi titik pengambilan barang



Gambar 3.8. *Sequence* Diagram menampilkan gambar rute yang terbentuk



Gambar 3.9. *Sequence Diagram* edit kode barang

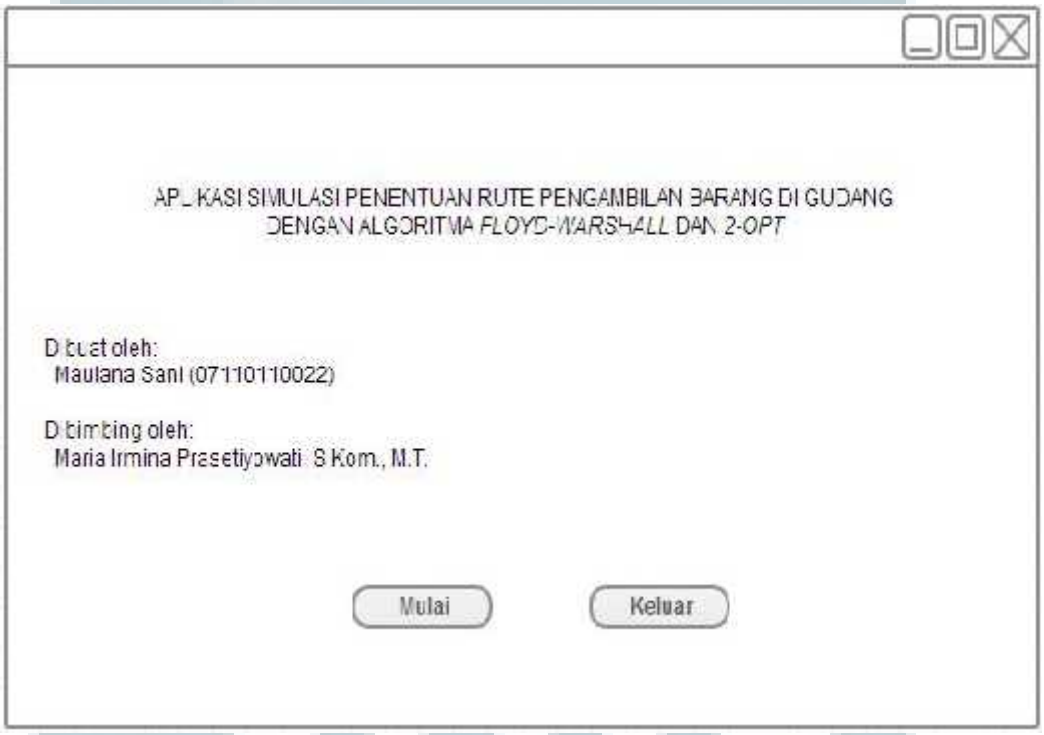


Gambar 3.10 *Sequence Diagram* cetak rute

3.3.2 Rancangan Layar

3.3.2.1 Rancangan Layar Utama

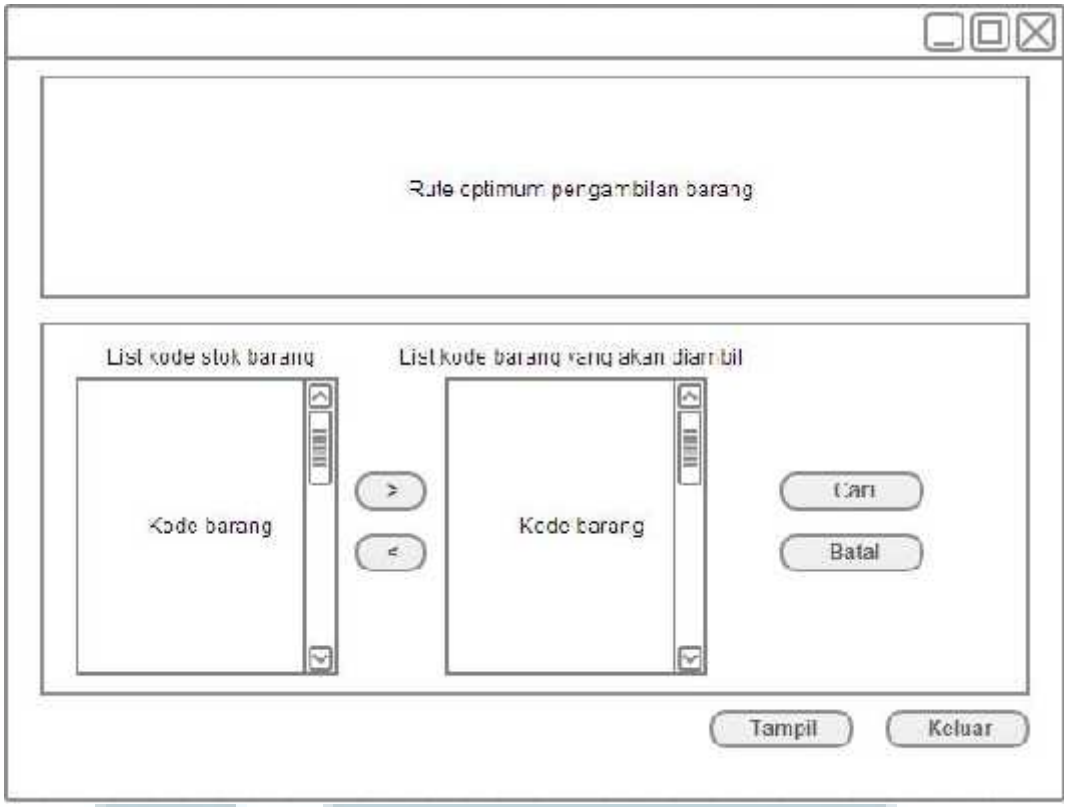
Layar utama merupakan layar utama pada aplikasi dimana layar ini merupakan layar yang pertama kali muncul sewaktu menjalankan aplikasi serta layar terakhir yang tampil pada saat aplikasi berakhir. Komponen yang terdapat pada layar menu utama hanya berupa layar yang merupakan pengenalan dan pengantar untuk menggunakan aplikasi terdiri dari komponen tombol mulai untuk memulai aplikasi penentuan rute pengambilan barang dan tombol keluar untuk keluar dari aplikasi.



Gambar 3.11. Rancangan layar Utama

3.3.2.2 Rancangan Layar Input

Rancangan layar input menampilkan kode-kode barang yang akan diambil sebagai input untuk memasukkan data tentang kode barang. Jumlah semua kode barang yang dapat dimasukkan sebagai input maksimal 100 kode barang. Pada list kode stok barang berada kotak di sebelah kiri dan list kode barang yang akan diambil berada di kotak sebelah kanan yang merupakan permintaan. Tombol cari digunakan untuk mencari hasil masukkan dari pengguna dan tombol batal digunakan untuk menghapus semua isi kotak dalam list kode barang yang akan diambil. Tampilan rute optimum pengambilan barang di gudang yang mana hanya berupa urutan-urutan kode barang yang akan diambil. Tombol tampil akan menampilkan gambar denah gudang beserta *node* atau titik-titik yang merupakan posisi tempat pengambilan barang yang sudah diinput oleh pengguna.



Gambar 3.12. Rancangan layar Menu *input*

3.2.2.3 Rancangan Layar Output

Tampilan pada layar ini menampilan denah gudang yang sudah ditandai oleh *node* atau titik-titik posisi beserta rute pengambilan barang. Terdapat dua komponen tombol yang mana tombol cetak merupakan fungsi untuk mencetak gambar yang sudah disisipi titik rute pengambilan barang dan tombol batal apabila pengguna tidak ingin mencetak gambar tersebut.

Gambar 3.13. Rancangan form layar *output*