BAB III
PERANCANGAN SISTEM

3.1 Deskripsi Sistem Pencarian Jalur Terpendek

Pencarian jalur terpendek menggunakan algoritma dijkstra menjadi permasalahan yang akan dipecahkan dan dibahas pada tugas akhir ini, aplikasi pencarian jalur terpendek berdasarkan bobot atau panjanganya jalur yang dilewati dari node asal ke node tujuan, dan tidak tergantung dari berapa jumlah node atau titik yang dilewati.

3.2 Perancangan Sistem

Pada perancangan sistem akan dibuat suatu program yang dapat dijalankan pada PC maupun laptop, yang dapat digunakan untuk mencari jalur yang paling efisien dilihat dari segi panjang jalur yang hendak ditempuh.

Kebutuhan untuk membangun sistem tersebut adalah sebagai berikut :

1. *Hardware*

   Kebutuhan *hardware* untuk sistem ini adalah :

   a. *Processor* : 400 MHz atau lebih baik
   b. *Memory* : 256 MB SDRAM atau lebih baik
   c. 475 MB ruang harddisk (setelah instalasi)
   d. 800 x 600 atau resolusi yang lebih tinggi dengan 256 warna.
   e. *Keyboard* dan *mouse*.

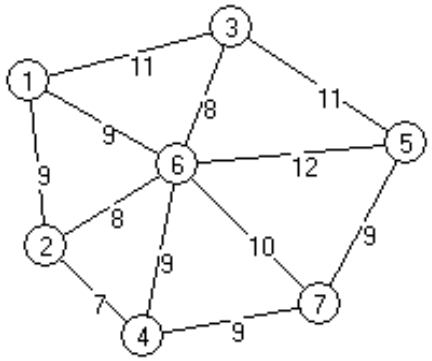
2. *Software*

   Kebutuhan *software* dari aplikasi untuk sistem ini adalah :

   a. OS : Windows 98 atau lebih baik
   b. Delphi 7

3.2.1 Data dalam Graf

Gambar 3.1 menunjukkan contoh dari penempatan node beserta bobotnya.



Gambar 3.1 Graf dengan 7 node yang tidak berarah dan berbobot

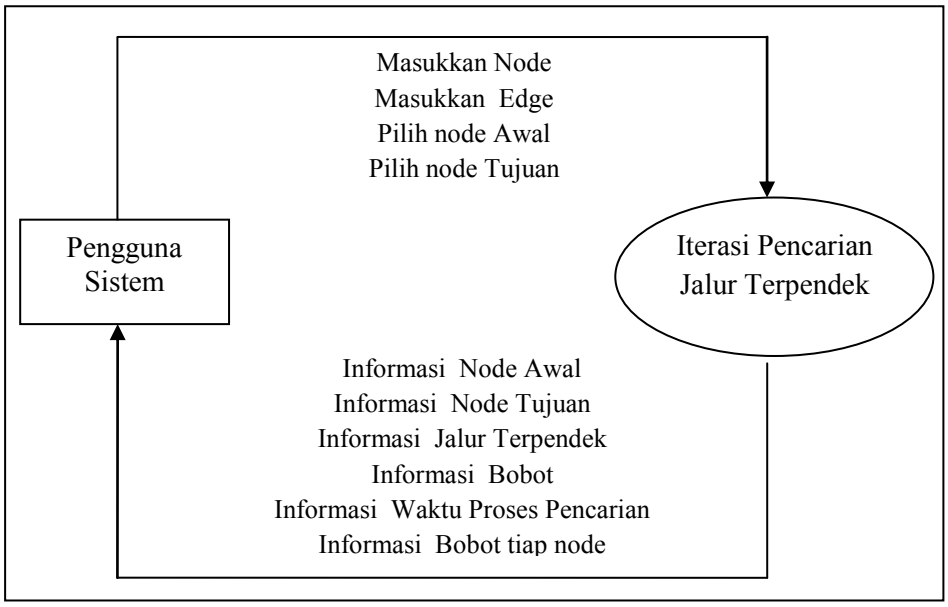
Dalam tabel 3.1 menunjukkan bobot antar node yang tersambung ke node lainnya dengan mengacu pada graf dalam gambar 3.1.

Tabel 3.1 Nilai Bobot

Node	Link ke-Node	Bobot
1	2	9
1	3	11
1	6	9
2	1	9
2	4	7
2	6	8
3	1	11
3	5	11
3	6	8
4	2	7
4	6	9
4	7	9
5	3	11
5	6	12
5	7	9
6	1	9
6	2	8
6	3	8
6	4	9
6	5	12
6	7	10
7	4	9
7	5	9
7	6	10

3.2.2 Diagram Konteks

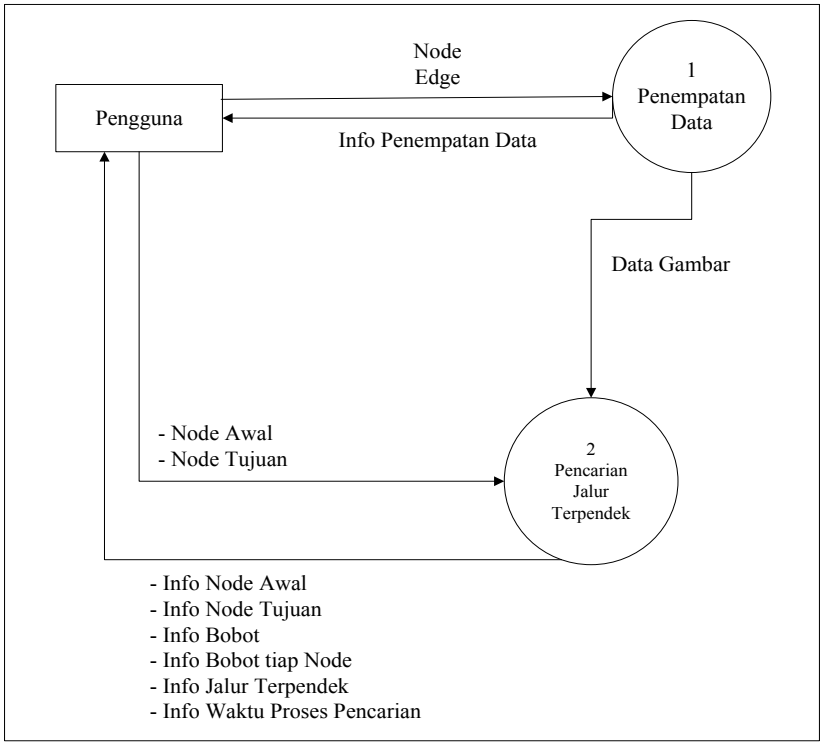
Diagram Konteks merupakan suatu model yang menjelaskan secara global bagaimana data digunakan dan diinformasikan untuk proses, atau dengan kata lain merupakan gambaran umum mengenai sistem yang menggambarkan aliran data kedalam dan keluar sistem. Berikut ini adalah diagram dari sistem yang akan dibangun.



Gambar 3.2 Diagram konteks

3.2.3 Diagram Alir Data

Setelah diagram konteks terbentuk, tahapan selanjutnya adalah membuat diagram alir data (*Data Flow Diagram*/DFD) untuk menjelaskan proses input data sistem pencarian jalur terpendek yang telah terbentuk.



Gambar 3.3 DFD Level 1

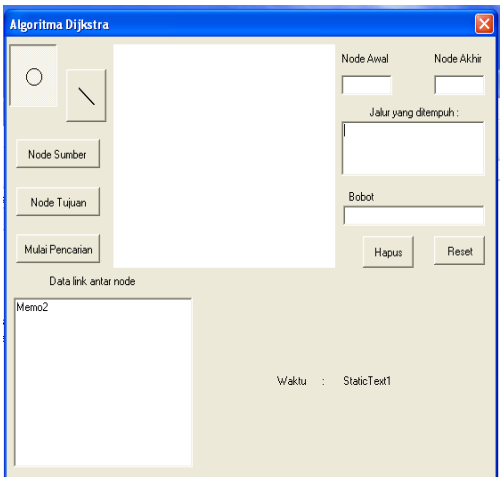
Dari DFD *Level* 1 terdapat dua tahapan, yaitu :

1. Pada proses penempatan data, pengguna diharuskan untuk memasukkan node dan menyambungkan dengan *edge* pada node lainnya dan data berupa gambar graf.

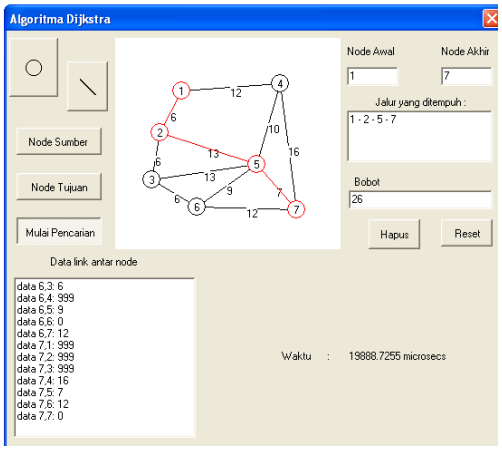
2. Pada proses pencarian jalur terpendek, diharuskan memilih node mana yang akan dijadikan node awal dan node tujuan yang bertujuan untuk menampilkan hasil simulasi, yaitu Informasi Bobot, Informasi Bobot Tiap Node, Informasi Jalur Terpendek, dan Informasi Waktu Proses Pencarian.

3.2.4 Perancangan Input/Output Sistem

Perancangan T01

	1. Jika Node di klik maka akan keluar node yang mulai dari node ke-1 2. Jika garis (\) di klik maka dapat menghubungkan antar node. 3. Jika Node Sumber di klik, lalu pilih node dan akan tampil di Node Awal (V1) 4. Jika Node Tujuan di klik, lalu pilih node dan akan tampil di Node Akhir (V2) 5. Jika Mulai Pencarian di klik, menghasilkan data untuk Jalur yang ditempuh beserta Bobot juga data antar node yang berkaitan dan waktu yang ditempuh. Dapat dilihat pada T02

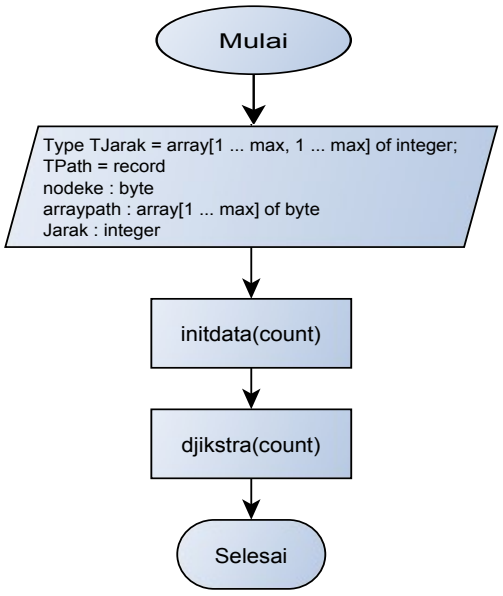
Perancangan T02

	1. Output dari aplikasi *Shortest Path* dengan menggunakan algoritma Dijkstra.

3.2.5 Perancangan Prosedural

A. Prosedur Program Utama

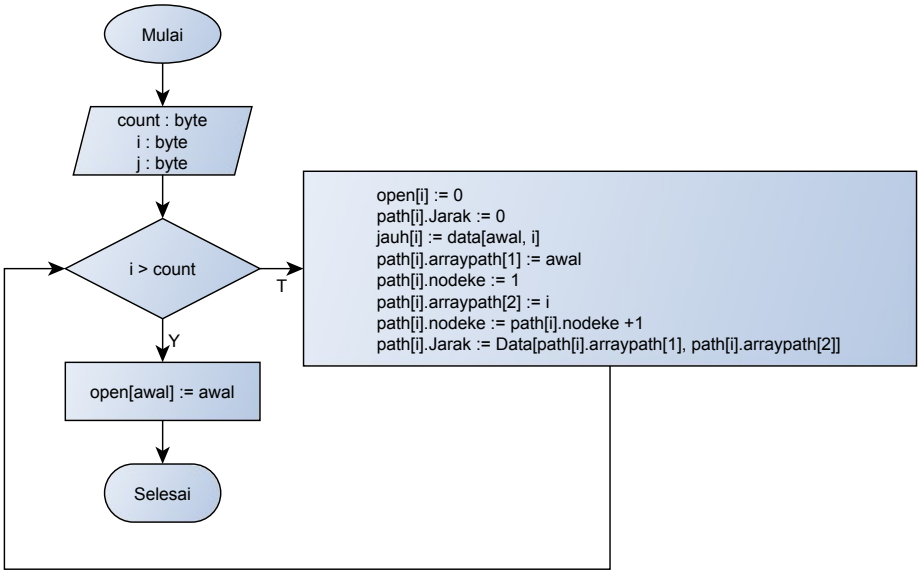
Pada bagian ini akan dibuat deskripsi rinci dari perangkat lunak, yaitu algoritma yang akan digunakan pada sistem yang akan dibuat, perancangan terlihat pada gambar 3.5.



Gambar 3.4 Prosedur Program Utama

B. Prosedur Initdata

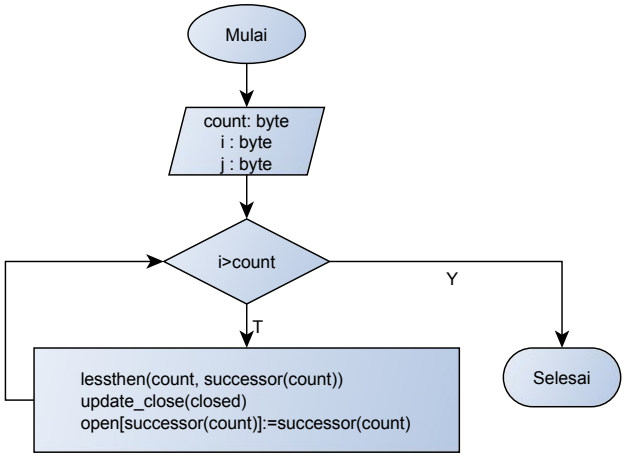
Prosedur ini berfungsi untuk menginisialisasikan data dalam pencarian jalur terpendek.



Gambar 3.5 Prosedur Initdata

C. Prosedur Dijkstra

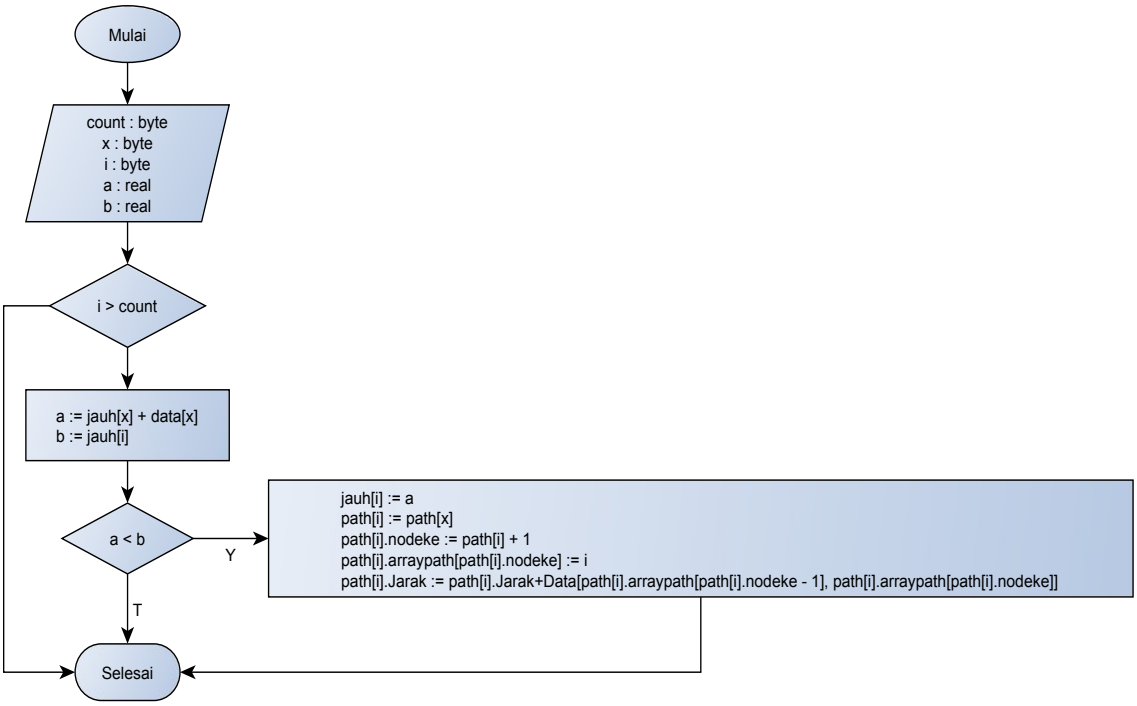
Prosedur ini berfungsi untuk memulai pencarian jalur terpendek dengan cara algoritma dijkstra.



Gambar 3.6 Prosedur Dijkstra

D. Prosedur Lessthen

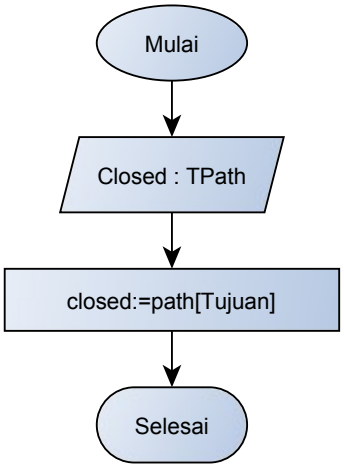
Prosedur ini berfungsi untuk mencari jalur terpendek dengan cara memasukkan node yang bertetangga ke dalam *open list*, dan membandingkan node pada jalur yang berbeda.



Gambar 3.7 Prosedur Lessthen

E. Prosedur Update

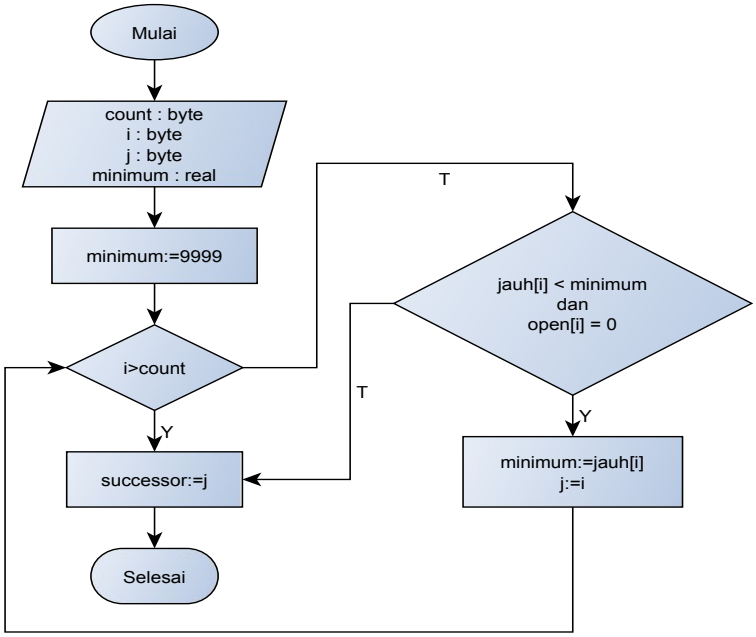
Prosedur ini berfungsi mencari jalur terpendek dengan cara memasukkan node ke dalam *closed list.*



Gambar 3.8 Prosedur Update

F. Prosedur Open

Prosedur ini berfungsi untuk mencari jalur terpendek dengan cara menentukkan node mana yang terlebih dahulu diproses dalam *open list*, proses dilakukan berdasarkan bobot yang paling minimum diantara node lainnya.



Gambar 3.9 Prosedur Open