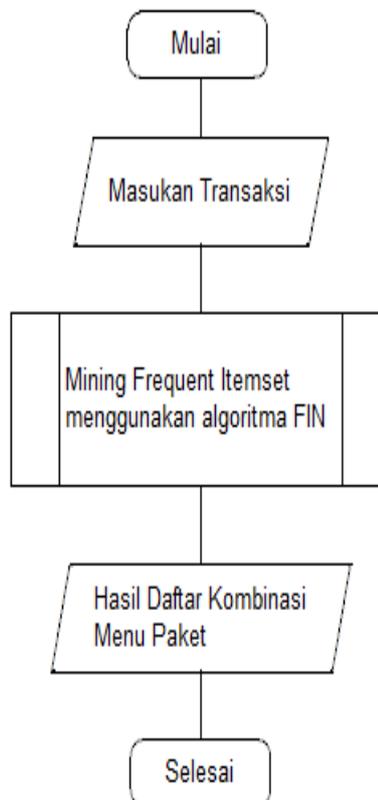


## BAB 4 PERANCANGAN

### 4.1 Deskripsi Sistem

Sistem yang dikembangkan oleh penulis adalah sistem pembuatan menu paket otomatis. Sistem yang mampu menghasilkan menu paket yang representatif dengan pilihan konsumen. Menu paket yang tersusun terdiri dari daftar kombinasi *item* makanan dan *item* minuman yang dikombinasikan menggunakan algoritma *FIN*. Tahapan algoritma *FIN* dalam menghasilkan kombinasi menu paket adalah mengkonstruksi *POC Tree*, pembentukan *frequent 1-itemset*, pembentukan *frequent 2-itemset* dan pembentukan *frequent 3-itemset*.

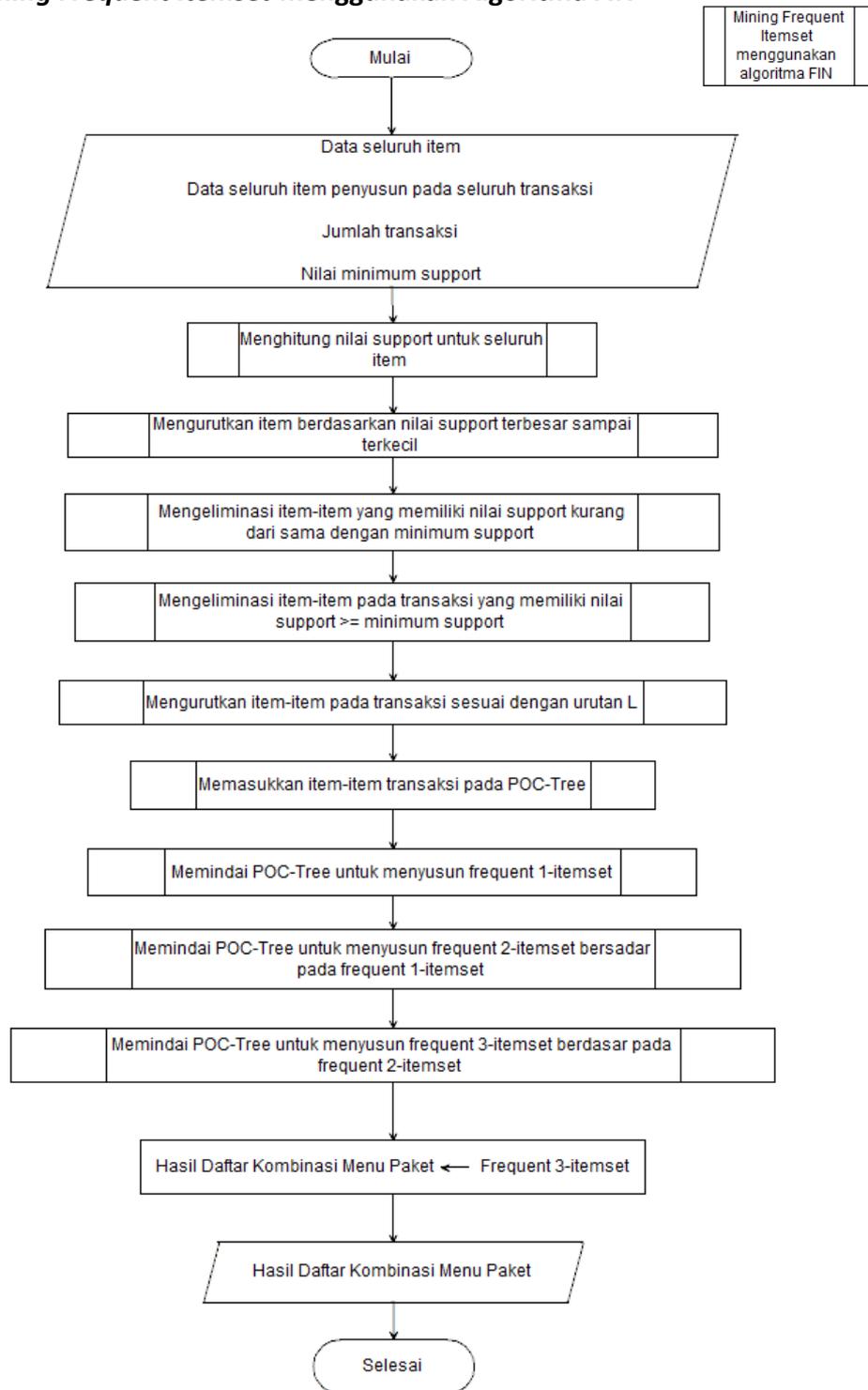
### 4.2 Perancangan Perangkat Lunak



Gambar 4.1 Diagram alir sistem Penyusun Menu Paket

Pada Gambar 4.1 perancangan sistem penyusun menu paket otomatis melalui beberapa tahapan yaitu memasukkan data transaksi, kemudian melakukan *mining frequent itemset* dengan algoritma *FIN*. Memasukkan satu per satu nota pesanan atau *struk* pada *database* sistem sebagai *dataset* transaksi. *Dataset* transaksi sebagai masukan proses *Mining Frequent Itemset* menggunakan algoritma *FIN*. Keluaran dari sistem adalah hasil daftar kombinasi menu paket.

#### 4.2.1 Mining Frequent Itemset menggunakan Algoritma FIN

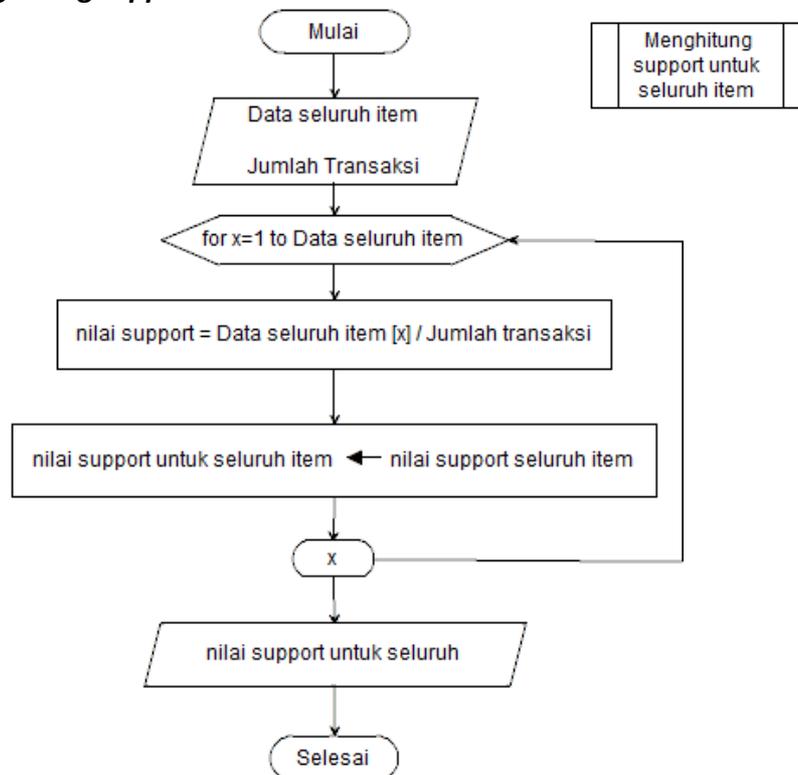


Gambar 4.2 Diagram alir *Mining Frequent Itemset* menggunakan algoritma *FIN*

Diagram alir pada Gambar 4.2 menjelaskan tahapan proses algoritma *FIN* dalam melakukan *mining frequent itemset* untuk menghasilkan menu paket. Masukan sistem adalah data seluruh item, data seluruh item penyusun pada

seluruh transaksi, jumlah transaksi, dan nilai *minimum support*. Tahapan pertama adalah menghitung nilai *support* untuk seluruh item. Tahapan kedua, mengurutkan item berdasarkan nilai *support* terbesar sampai terkecil. Tahapan ketiga, mengeliminasi item-item yang memiliki nilai *support* kurang dari sama dengan *minimum support*. Tahapan keempat, mengeliminasi item-item pada transaksi yang memiliki nilai *support*  $\geq$  *minimum support*. Tahapan kelima, mengurutkan item-item pada transaksi sesuai dengan urutan  $L$ . Tahapan keenam, memasukkan item-item transaksi pada *POC-tree*. Tahapan ketujuh, memindai *POC-tree* untuk menyusun *frequent 1-itemset*. Tahapan kedelapan, memindai *POC-tree* untuk menyusun *frequent 2-itemset* berdasar pada *frequent 1-itemset*. Tahapan kesembilan, memindai *POC-Tree* untuk menyusun *frequent 3-itemset* berdasar pada *frequent 2-itemset*. Hasil dari *frequent 3-itemset* digunakan untuk menyusun hasil daftar kombinasi menu paket. Keluaran dari sistem adalah hasil daftar kombinasi menu paket.

#### 4.2.1.1 Menghitung *support* untuk seluruh item

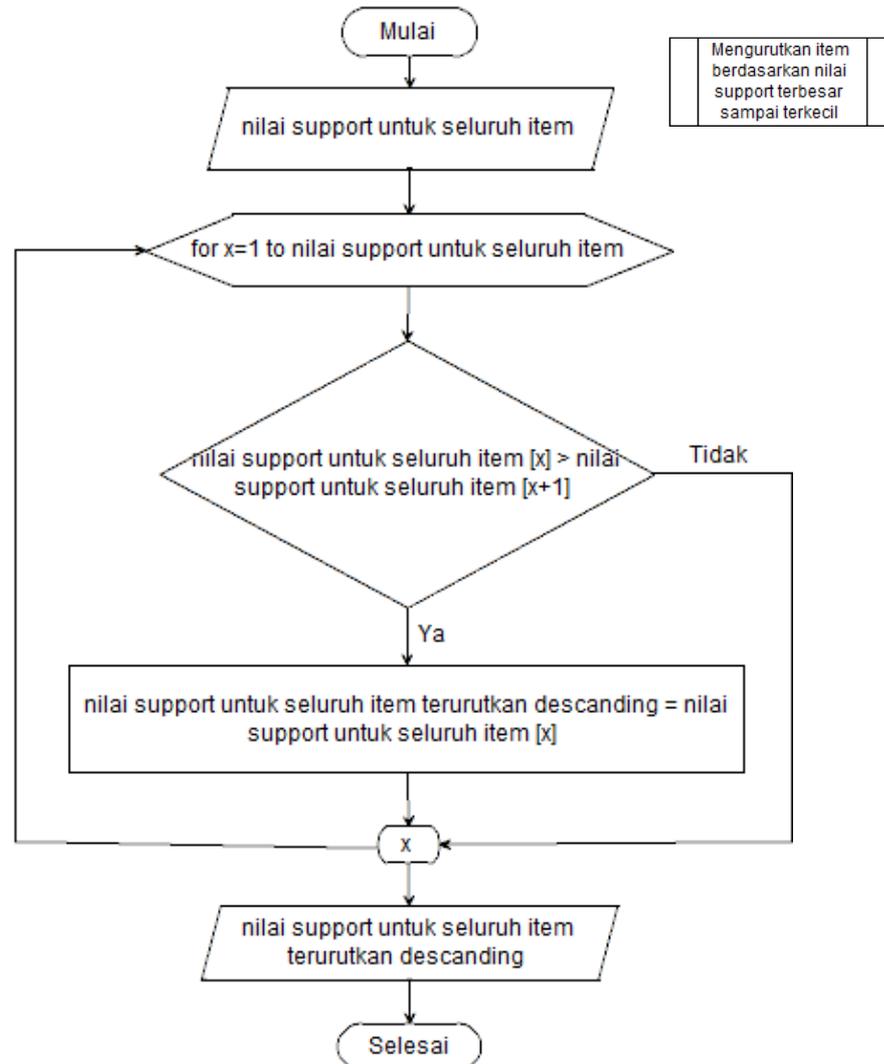


Gambar 4.3 Diagram alir Menghitung *support* untuk seluruh item

Diagram alir pada Gambar 4.3 menjelaskan proses menghitung nilai *support* untuk seluruh item daftar menu. Masukan sistem adalah data seluruh item dan jumlah transaksi. Dilakukan perulangan dari indeks  $x=1$  sampai data seluruh item habis. Dilanjutkan proses perhitungan nilai *support* yaitu nilai data seluruh item pada indeks  $x$  dibagi jumlah transaksi. Dilanjutkan menyimpan hasil perhitungan nilai *support* pada nilai *support* untuk seluruh item. Keluaran dari sistem adalah nilai *support* untuk seluruh item. Nilai *support* untuk seluruh item menjadi

masukannya untuk tahapan proses selanjutnya yaitu mengurutkan item berdasarkan nilai *support* terbesar sampai terkecil.

#### 4.2.1.2 Mengurutkan item berdasarkan nilai *support* terbesar sampai terkecil

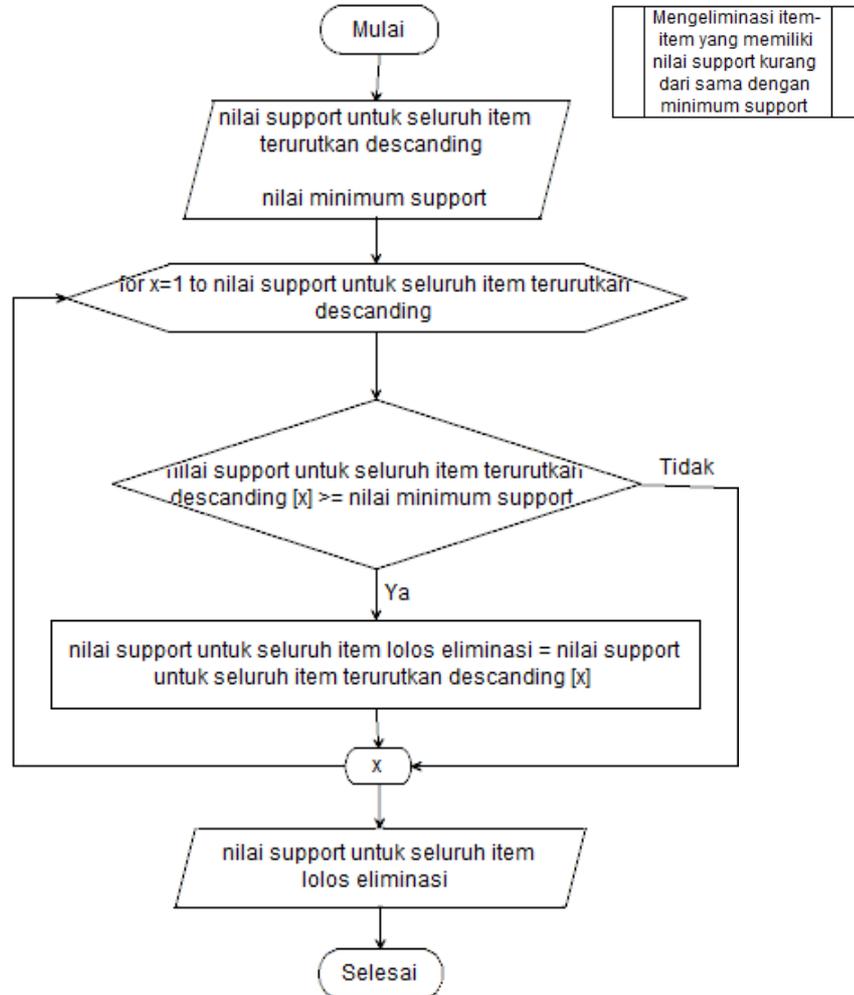


Gambar 4.4 Diagram alir Mengurutkan item berdasarkan nilai *support* terbesar sampai terkecil

Diagram alir pada Gambar 4.4 menjelaskan proses mengurutkan item berdasarkan nilai *support* terbesar sampai nilai *support* terkecil. Masukan sistem adalah nilai *support* untuk seluruh item. Proses pertama adalah melakukan perulangan dari indeks  $x=1$  sampai nilai *support* untuk seluruh item habis. Untuk setiap perulangan dilakukan proses percabangan, menyeleksi nilai *support* untuk seluruh item  $[x]$  lebih besar dari pada nilai *support* untuk seluruh item  $[x+1]$ . Apabila lolos percabangan maka dilanjutkan proses memasukkan nilai *support* untuk seluruh item  $[x]$  pada nilai *support* untuk seluruh item terurutkan *descending*. Proses terus diulang sampai perulangan selesai. Keluaran dari sistem adalah nilai *support* untuk seluruh item terurutkan *descending*. Nilai *support* untuk seluruh item terurutkan *descending* menjadi masukan untuk proses

selanjutnya yaitu mengeliminasi item-item yang memiliki nilai *support* kurang dari sama dengan minimum *support*.

#### 4.2.1.3 Mengeliminasi item-item yang memiliki nilai *support* kurang dari sama dengan minimum *support*

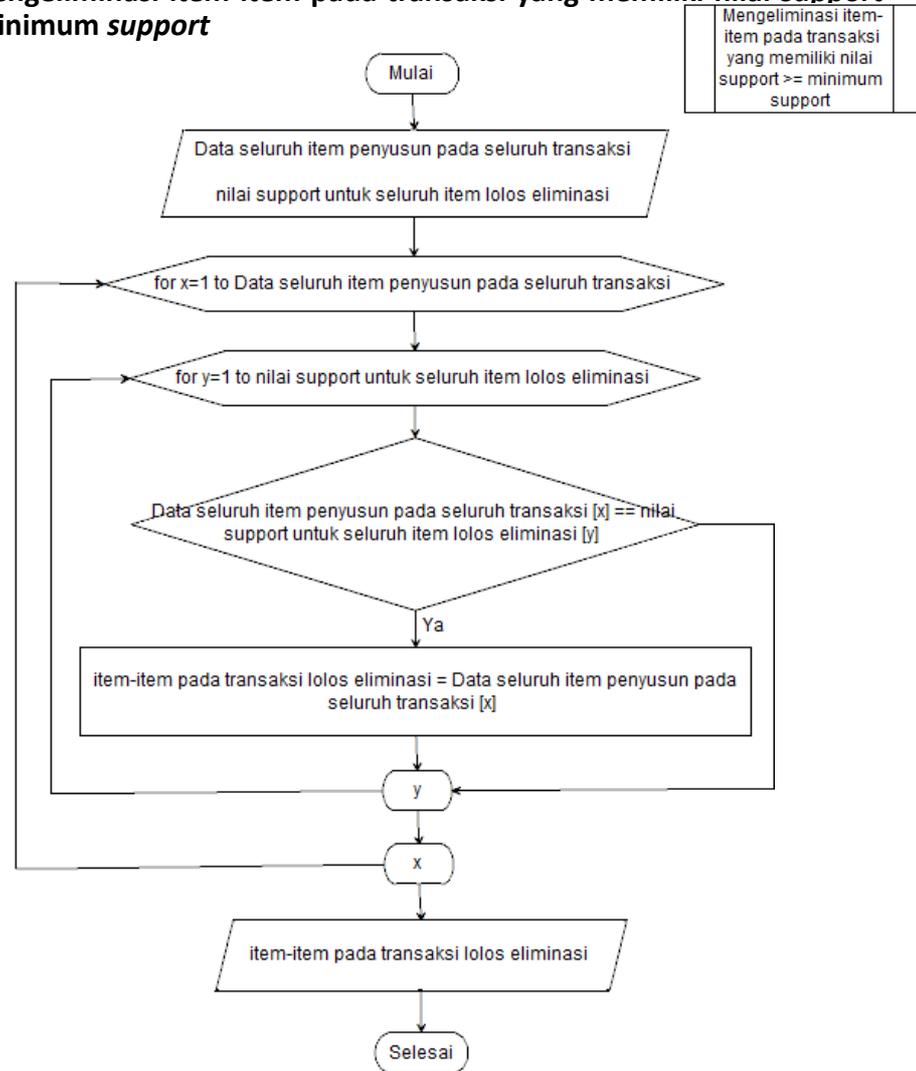


Gambar 4.5 Diagram alir Mengeliminasi item-item yang memiliki nilai *support* kurang dari sama dengan minimum *support*

Diagram alir pada Gambar 4.5 menjelaskan proses mengeliminasi item-item yang memiliki nilai *support* kurang dari sama dengan minimum *support*. Item-item yang dieliminasi adalah item *infrequent*. Masukan dari sistem adalah nilai *support* untuk seluruh item terurutkan *descending* dan nilai minimum *support*. Dilakukan proses perulangan dari indeks  $x=1$  sampai nilai *support* untuk seluruh item terurutkan *descending* habis. Untuk setiap perulangan dilakukan proses percabangan, menyeleksi nilai *support* untuk seluruh item terurutkan *descending*  $[x]$  lebih dari sama dengan minimum *support*. Apabila lolos percabangan maka dilakukan proses memasukkan nilai *support* untuk seluruh item terurutkan *descending*  $[x]$  pada nilai *support* untuk seluruh item lolos eliminasi. Proses kembali diulang sampai perulangan selesai. Keluaran dari sistem adalah nilai *support* untuk seluruh item lolos eliminasi. Nilai *support* untuk seluruh item lolos

eliminasi menjadi masukan untuk proses selanjutnya yaitu mengeliminasi item-item pada transaksi yang memiliki nilai *support*  $\geq$  minimum *support*.

#### 4.2.1.4 Mengeliminasi item-item pada transaksi yang memiliki nilai *support* $\geq$ minimum *support*

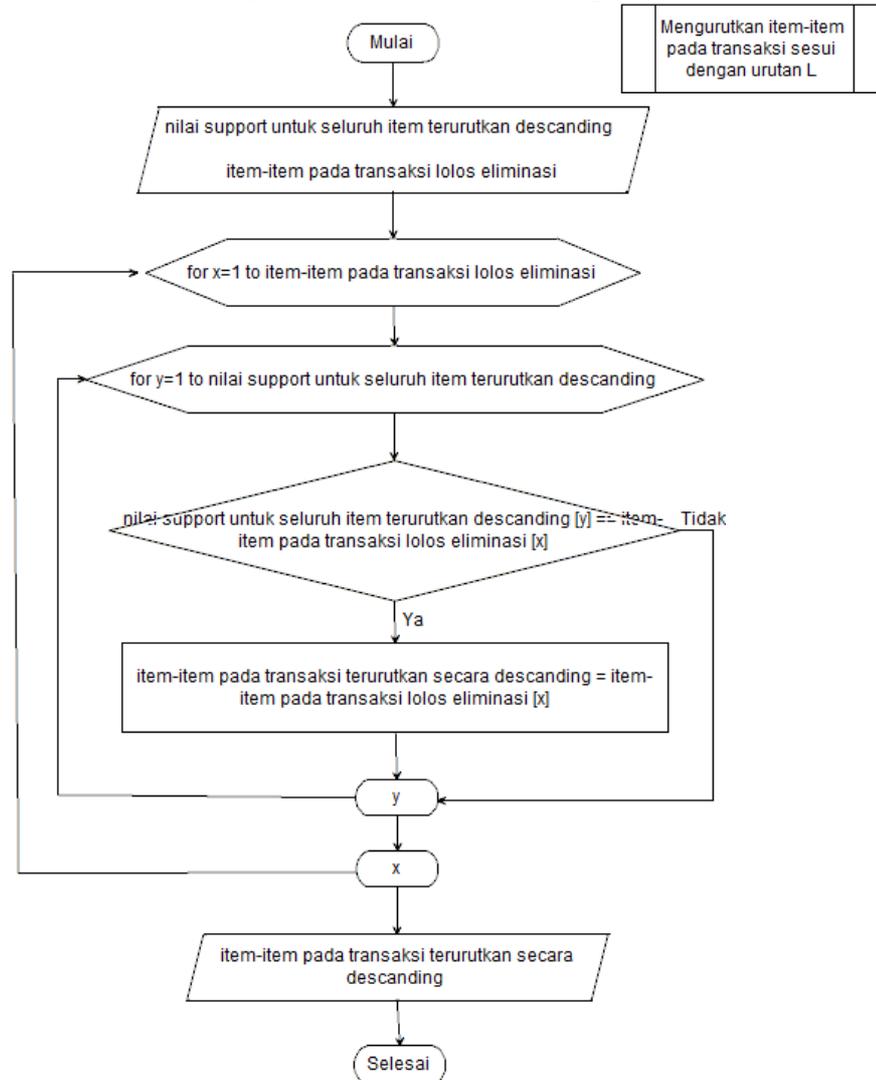


Gambar 4.6 Diagram alir Mengeliminasi item-item pada transaksi yang memiliki nilai *support*  $\geq$  minimum *support*

Diagram alir pada Gambar 4.6 menjelaskan proses mengeliminasi item-item pada transaksi yang memiliki nilai *support*  $\geq$  minimum *support*. Masukan dari sistem adalah data seluruh item penyusun pada seluruh transaksi dan nilai *support* untuk seluruh item lolos eliminasi. Terdapat dua tahapan perulangan yaitu proses perulangan pertama, untuk indeks  $x=1$  sampai data seluruh item penyusun pada seluruh transaksi habis. Dilanjutkan proses perulangan kedua, untuk indeks  $y=1$  sampai nilai *support* untuk seluruh item lolos eliminasi habis. Untuk setiap proses perulangan dilakukan proses percabangan, yang melakukan pengecekan data seluruh item penyusun pada seluruh transaksi  $[x]$ , apabila memiliki nilai yang sama dengan nilai *support* untuk seluruh item lolos eliminasi  $[x]$ . Apabila lolos pengecekan maka dilanjutkan proses memasukkan data seluruh

item penyusun pada seluruh transaksi  $[x]$  pada item-item transaksi lolos eliminasi. Proses terus diulang sampai pada kedua perulangan telah habis. Keluaran dari sistem adalah item-item pada transaksi lolos eliminasi. Item-item pada transaksi lolos eliminasi menjadi masukan untuk proses selanjutnya yaitu mengurutkan item-item pada transaksi sesuai dengan urutan  $L$ .

#### 4.2.1.5 Mengurutkan item-item pada transaksi sesuai dengan urutan $L$

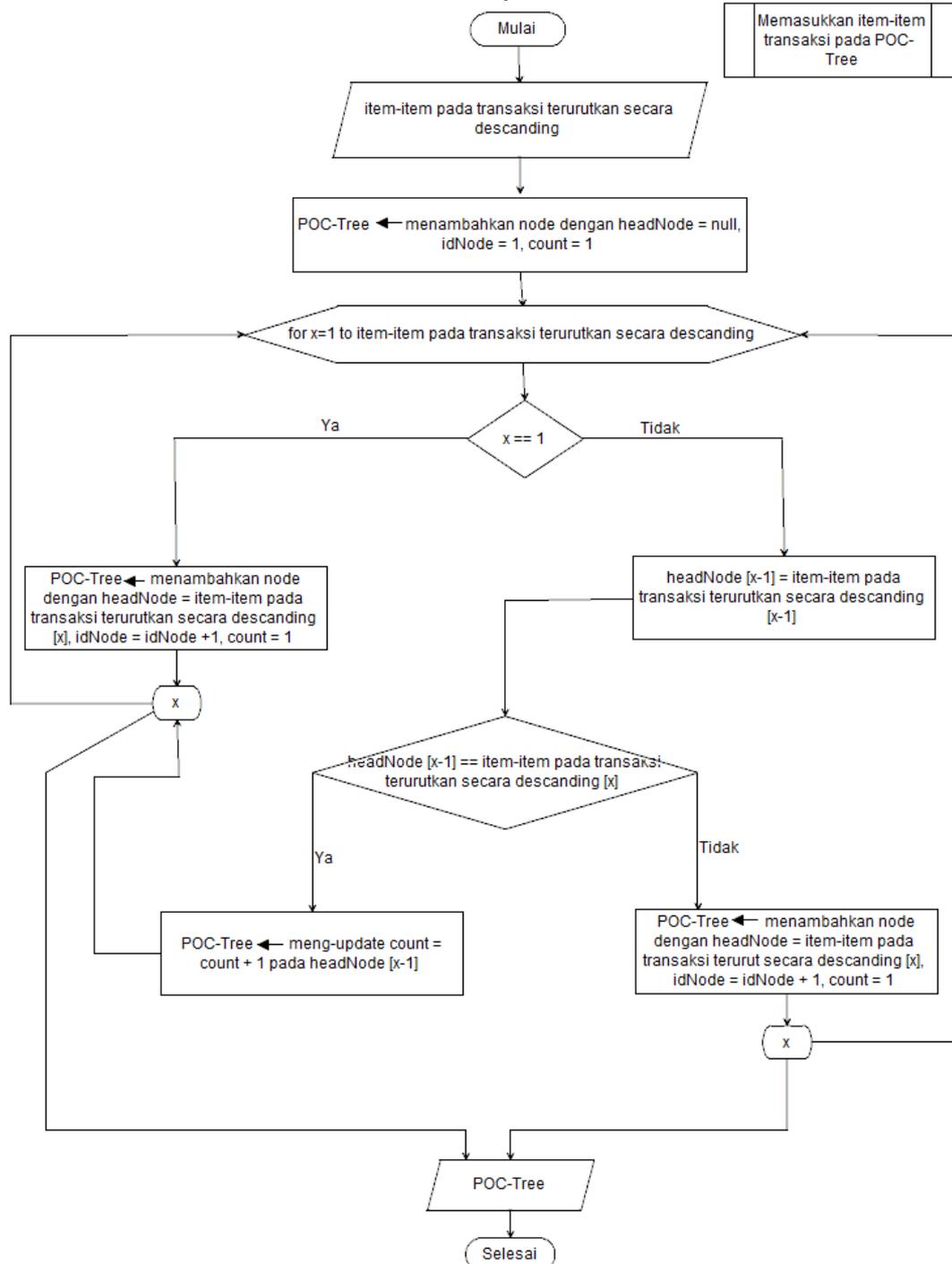


Gambar 4.7 Diagram alir Mengurutkan item-item pada transaksi sesuai dengan urutan  $L$

Diagram alir pada Gambar 4.7 menjelaskan proses mengurutkan item-item pada transaksi sesuai dengan urutan  $L$ . Masukan dari sistem adalah nilai *support* untuk seluruh item terurutkan *descending* dan item-item pada transaksi lolos eliminasi. Terdapat dua tahapan proses perulangan yaitu perulangan pertama, untuk indeks  $x=1$  sampai item-item pada transaksi lolos eliminasi habis. Perulangan kedua, untuk indeks  $y=1$  sampai nilai *support* untuk seluruh item terurutkan *descending* habis. Untuk setiap proses perulangan dilakukan proses pengecekan nilai *support* untuk seluruh item terurutkan *descending*  $[y]$  memiliki nilai sama dengan item-item pada transaksi lolos eliminasi  $[x]$ . Apabila lolos

pengecekan maka masuk pada proses memasukkan item-item pada transaksi lolos eliminasi [x] pada, item-item pada transaksi terurutkan secara *descending*. Proses terus terulang sampai kedua perulangan habis. Keluaran dari sistem adalah item-item pada transaksi terurutkan secara *descending*. Item-item pada transaksi terurutkan secara *descending* menjadi masukan untuk proses selanjutnya yaitu memasukkan item-item transaksi pada *POC-Tree*.

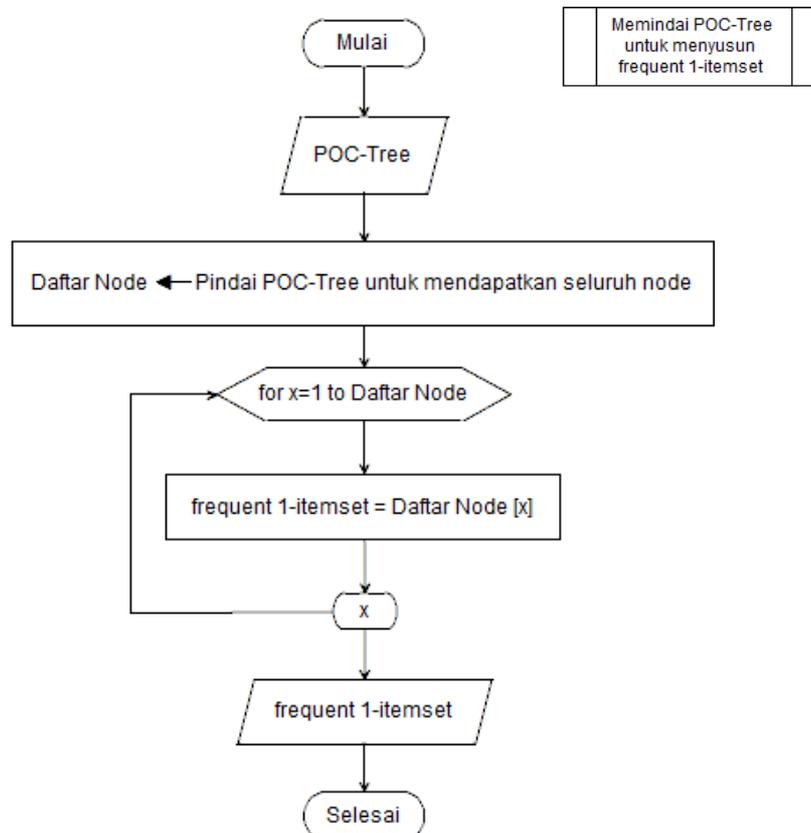
#### 4.2.1.6 Memasukkan item-item transaksi pada *POC-Tree*



Gambar 4.8 Diagram alir Memasukkan item-item transaksi pada *POC-Tree*

Pada diagram alir pada gambar 4.8 menjelaskan proses memasukkan item-item transaksi pada *POC-tree*. Masukan dari sistem adalah item-item pada transaksi terurutkan secara *descending*. Proses pertama yang dilakukan adalah menambahkan node dengan  $headNode = null, idNode = 1, count = 1$  pada *POC-Tree*. Kemudian dilanjutkan proses perulangan dari indeks  $x=1$  sampai item-item pada transaksi terurutkan secara *descending* habis. Untuk setiap perulangan dilakukan proses pengecekan untuk  $x$  sama dengan 1. Apabila lolos pengecekan masuk ke proses menambahkan *node* dengan  $headNode =$  item-item pada transaksi terurutkan secara *descending*  $[x], idNode = idNode + 1, count = 1$ . Apabila tidak lolos pengecekan maka masuk ke proses, memasukkan item-item pada transaksi terurutkan secara *descending*  $[x-1]$  pada  $headNode [x-1]$ . Untuk proses yang masuk ke proses ini, selanjutnya dilakukan pengecekan  $headNode [x-1] ==$  item-item pada transaksi terurutkan secara *descending*  $[x]$ . Apabila lolos pengecekan maka masuk ke proses meng-*update*  $count = count + 1$  pada  $headNode [x-1]$  untuk *POC-Tree*. Apabila tidak lolos pengecekan maka menambahkan *node* dengan  $headNode =$  item-item pada transaksi terurut secara *descending*  $[x], idNode = idNode + 1, count = 1$ . Seluruh proses terus diulang sampai perulangan habis. Keluaran dari sistem adalah *POC-Tree*. *POC-Tree* menjadi masukan untuk proses selanjutnya yaitu memindai *POC-Tree* untuk menyusun *frequent 1-itemset*.

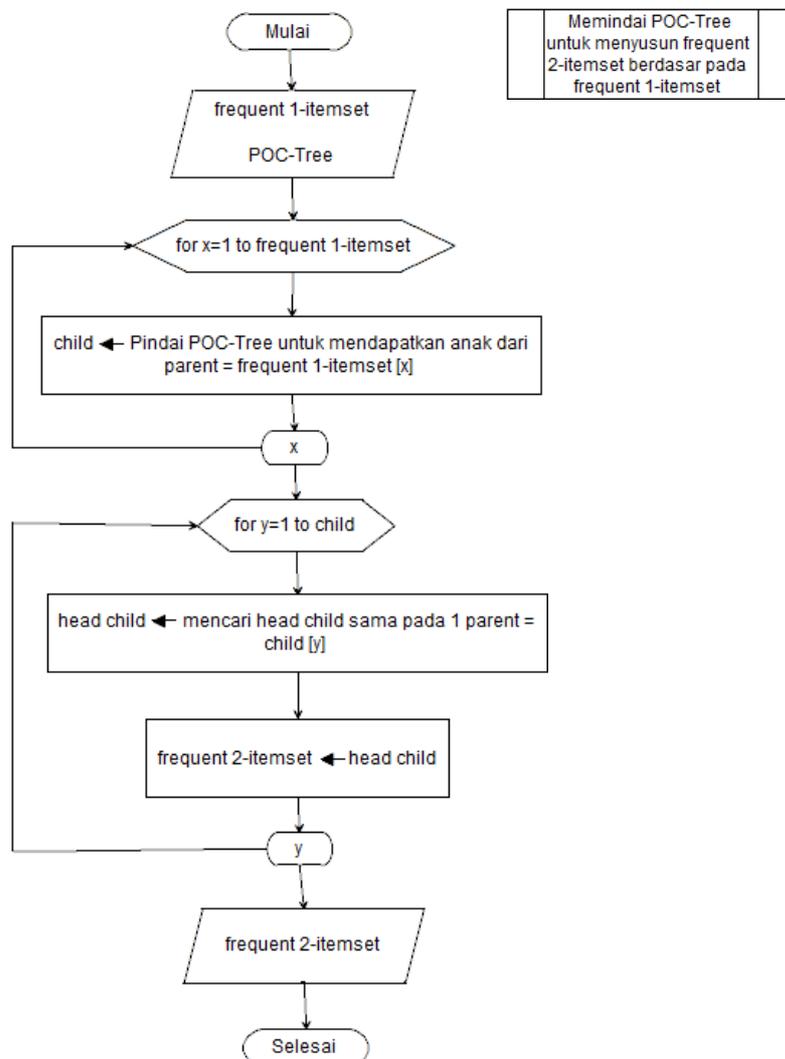
#### 4.2.1.7 Memindai *POC-Tree* untuk menyusun *frequent 1-itemset*



Gambar 4.9 Diagram alir Memindai *POC-Tree* untuk menyusun *frequent 1-itemset*

Diagram alir pada Gambar 4.9 menjelaskan proses memindai *POC-Tree* untuk menyusun *frequent 1-itemset*. Masukan dari sistem adalah *POC-Tree*. Proses pertama yang dilakukan adalah memindai *POC-Tree* untuk mendapatkan seluruh *node* kemudian disimpan pada *DaftarNode*. Kemudian dilakukan proses perulangan dari indeks  $x=1$  sampai *DaftarNode* habis. Untuk setiap perulangan dilakukan proses memasukkan *DaftarNode [x]* pada *frequent 1-itemset*. Proses terus diulang sampai perulangan selesai. Keluaran dari sistem adalah *frequent 1-itemset*. *Frequent 1-itemset* menjadi masukan untuk proses selanjutnya yaitu memindai *POC-Tree* untuk menyusun *frequent 2-itemset* berdasarkan pada *frequent 1-itemset*.

#### 4.2.1.8 Memindai *POC-Tree* untuk menyusun *frequent 2-itemset* berdasar pada *frequent 1-itemset*

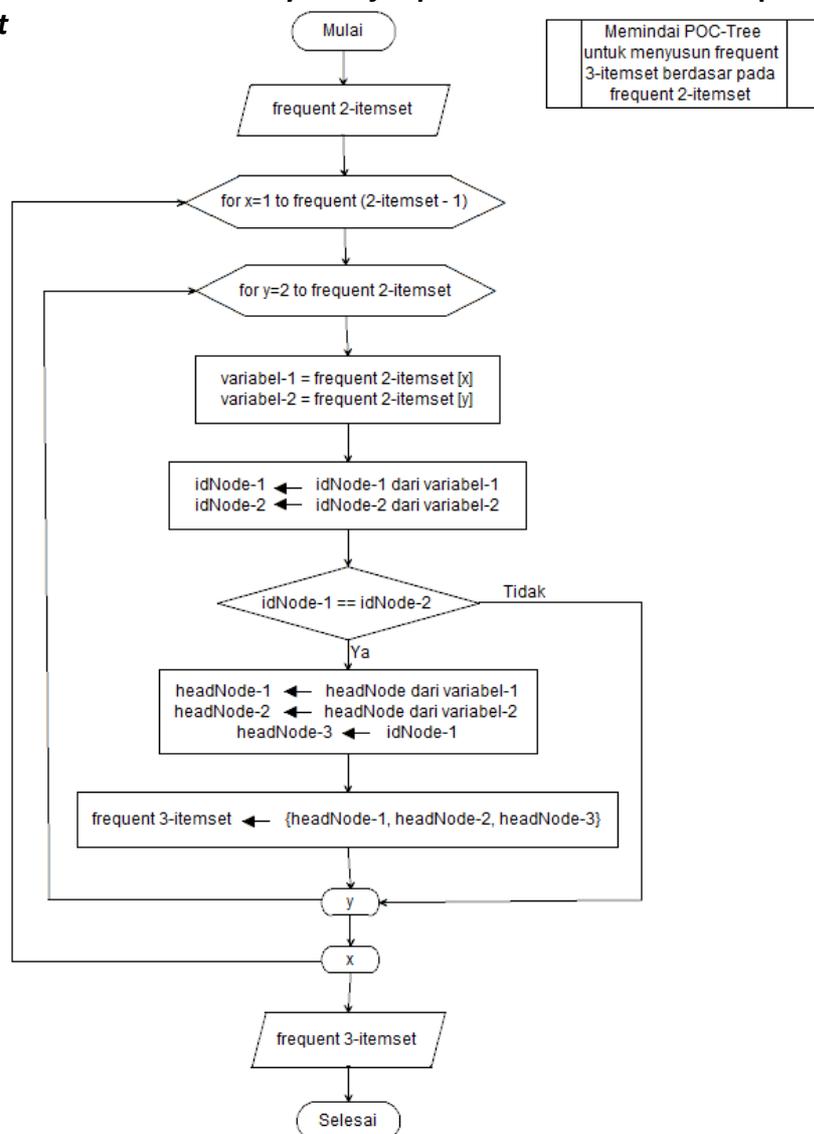


Gambar 4.10 Diagram alir Memindai *POC-Tree* untuk menyusun *frequent 2-itemset* berdasar pada *frequent 1-itemset*

Diagram alir pada Gambar 4.10 menjelaskan proses memindai *POC-Tree* untuk menyusun *frequent 2-itemset* berdasarkan pada *frequent 1-itemset*.

Pertama dilakukan proses perulangan pertama dilanjutkan proses perulangan kedua. Pada perulangan pertama dilakukan perulangan dari indeks  $x=1$  sampai *frequent 1-itemset* habis. Untuk setiap perulangan dilakukan proses memindai *POC-Tree* untuk mendapatkan anak dari *parent = frequent 1-itemset [x]*, kemudian disimpan pada *child*. Proses terus dilakukan sampai perulangan pertama habis. Kemudian dilanjutkan perulangan kedua mulai dari indeks  $y=1$  sampai *child* habis. Untuk setiap perulangan dilakukan proses mencari *headChild* sama pada 1 *parent = child [y]*, kemudian disimpan pada *headChild*. Kemudian dilanjutkan proses selanjutnya yaitu menyimpan *headChild* pada *frequent 2-itemset*. Proses terus diulang sampai perulangan kedua selesai. Keluaran dari sistem adalah *frequent 2-itemset*. *Frequent 2-itemset* menjadi masukan untuk proses selanjutnya yaitu memindai *POC-Tree* untuk menyusun *frequent 3-itemset* berdasar pada *frequent 2-itemset*.

#### 4.2.1.9 Memindai *POC-Tree* untuk menyusun *frequent 3-itemset* berdasar pada *frequent 2-itemset*



Gambar 4.11 Memindai *POC-Tree* untuk menyusun *frequent 3-itemset* berdasar pada *frequent 2-itemset*

Diagram alir pada Gambar 4.11 menjelaskan proses memindai *POC-Tree* untuk menyusun *frequent 3-itemset* berdasar pada *frequent 2-itemset*. Masukan dari sistem adalah *frequent 2-itemset*. Proses pertama yang dilakukan adalah melakukan proses perulangan dimulai dari indeks  $x=1$  sampai (*frequent 2-itemset* – 1) habis. Untuk setiap perulangan dilakukan proses perulangan kedua yaitu perulangan yang dimulai dari indeks  $y=2$  sampai *frequent 2-itemset* habis. Untuk setiap perulangan kedua dilakukan proses memasukkan nilai *frequent 2-itemset* [ $x$ ] pada *variabel-1* dan memasukkan nilai *frequent 2-itemset* [ $y$ ] pada *variabel-2*. Dilanjutkan pengecekan *idNode-1* sama dengan *idNode-2*. Apabila lolos maka masuk pada proses memasukkan *headNode* dari *variabel-1* pada *headNode-1*, memasukkan *headNode* dari *variabel-2* pada *headNode-2*, memasukkan *idNode-1* pada *headNode-3*. Dilanjutkan proses selanjutnya yaitu menyimpan *headNode-1*, *headNode-2*, *headNode-3* pada format {*headNode-1*, *headNode-2*, *headNode-3*} pada *frequent 3-itemset*. Keluaran dari sistem adalah *frequent 3-itemset*. *Frequent 3-itemset* menjadi masukan untuk proses selanjutnya yaitu membentuk hasil daftar kombinasi menu paket.

### 4.3 Perhitungan Manual

Perhitungan manual pada sistem pembuat menu paket otomatis menggunakan algoritma *FIN*. Langkah-langkah perhitungan manual seperti berikut:

1. Menghitung nilai *support* untuk setiap *item* pada daftar menu.
2. Mengurutkan *item* berdasarkan nilai *support* terbesar sampai terkecil.
3. Mengeliminasi seluruh *item* yang memiliki nilai *support* kurang dari minimum *support*.
4. Mengeliminasi *item-item* pada setiap transaksi yang memiliki nilai *support* kurang dari minimum *support*.
5. Mengurutkan *item-item* yang tersisa untuk setiap transaksi sesuai dengan urutan  $L$ .
6. Memasukkan *item-item* pada setiap transaksi pada *POC Tree*.
7. Memindai *POC tree* untuk menyusun *frequent 1-itemset*.
8. Memindai *POC tree* untuk menyusun *frequent 2-itemset* yang berdasar pada *frequent 1-itemset*.
9. Memindai *POC tree* untuk menyusun *frequent 3-itemset* yang berdasar pada *frequent 2-itemset*.
10. Menyusun kombinasi menu paket makanan dan minuman berdasarkan *frequent 3-itemset*.

Untuk melakukan penghitungan manual pada penelitian "Implementasi *Association Rule Mining* untuk menemukan Menu Paket Makanan dengan Algoritma *FIN* menggunakan *Nodesets*", digunakan dataset transaksi yang

berjumlah 31 transaksi. Transaksi tercatat pada *struk* pembayaran dan nota pesanan.

Tabel 4.1 *Dataset* Transaksi

ID_TRANSAKSI	ID_ITEM
1	140, 131, 114, 93
2	140, 58
3	9, 1, 137, 118
4	118, 148, 68, 1, 90
5	1, 88, 48, 26, 5, 145, 138, 29
6	29, 1, 150, 93
7	47, 2, 118, 138
8	84, 10, 131, 117, 1
9	1, 56, 46, 117, 118
10	123
11	51, 47, 3, 150, 117, 131
12	59, 95
13	13, 58, 117, 154, 118, 3
14	72, 1, 69, 90, 117, 131, 91
15	74, 1, 127, 151
16	117, 150, 154, 88, 49, 40, 1
17	40, 1
18	117, 154, 9, 69, 1
19	76, 125, 138
20	93, 117, 113
21	89, 1, 123, 131, 44
22	40, 1
23	68, 135, 137
24	41, 117, 138, 1, 89
25	25, 27, 34, 30, 118, 1
26	7, 42, 25, 96, 123, 138, 118, 150, 3
27	1, 68, 10, 90, 40, 117, 138
28	1, 40, 9, 117, 150, 69, 154, 90, 76, 110
29	68, 145, 118
30	52
31	9, 1, 123, 95, 42

Keterangan: Nama *item* diwakili dengan *id\_Item* yang berbentuk angka. Nama setiap *item* dapat dilihat pada Lampiran B.

**Langkah 1. Menghitung nilai *support* untuk setiap *item* pada daftar menu.**

Berdasarkan rumus *support* pada persamaan 2.1 dilakukan penghitungan nilai *support* untuk seluruh transaksi. Hasil penghitungan nilai *support* ditampilkan pada tabel 4.2

Tabel 4.2 Penghitungan Nilai *Support* untuk Seluruh *Item*

<b>Id_Item</b>	<b>Jumlah Kemunculan <i>Item</i> Untuk 31 Transaksi</b>	<b>Jumlah Transaksi</b>	<b>Jumlah Kemunculan <i>Item</i> untuk 31 Transaksi / Jumlah Transaksi</b>
1	18	31	0,580645161290323
2	1	31	0,032258064516129
3	3	31	0,096774193548387
4	0	31	0
5	1	31	0,032258064516129
6	0	31	0
7	1	31	0,032258064516129
8	1	31	0,032258064516129
9	4	31	0,129032258064516
10	2	31	0,064516129032258
11	0	31	0
12	0	31	0
13	1	31	0,032258064516129
14	0	31	0
15	0	31	0
16	0	31	0
17	0	31	0
18	0	31	0
19	0	31	0
20	0	31	0
21	0	31	0
22	0	31	0
23	0	31	0
24	2	31	0
25	0	31	0,064516129032258
26	1	31	0,032258064516129
27	1	31	0,032258064516129
28	2	31	0
29	0	31	0,064516129032258

Tabel 4.2 Penghitungan Nilai *Support* untuk Seluruh *Item* (lanjutan)

<b>Id_Item</b>	<b>Jumlah Kemunculan <i>Item</i> Untuk 31 Transaksi</b>	<b>Jumlah Transaksi</b>	<b>Jumlah Kemunculan <i>Item</i> untuk 31 Transaksi / Jumlah Transaksi</b>
30	0	31	0,032258064516129
31	1	31	0
32	1	31	0
33	0	31	0
34	0	31	0,032258064516129
35	0	31	0
36	0	31	0
37	0	31	0
38	0	31	0
39	0	31	0
40	2	31	0,161290322580645
41	0	31	0,032258064516129
42	5	31	0,064516129032258
43	1	31	0
44	1	31	0,032258064516129
45	2	31	0
46	1	31	0,032258064516129
47	0	31	0,064516129032258
48	1	31	0,032258064516129
49	0	31	0,032258064516129
50	1	31	0
51	1	31	0,032258064516129
52	0	31	0,032258064516129
53	0	31	0
54	0	31	0
55	1	31	0
56	1	31	0,032258064516129
57	2	31	0
58	0	31	0,064516129032258
59	1	31	0,032258064516129
60	0	31	0
61	0	31	0
62	0	31	0
63	0	31	0
64	4	31	0

Tabel 4.2 Penghitungan Nilai *Support* untuk Seluruh *Item* (lanjutan)

<b>Id_Item</b>	<b>Jumlah Kemunculan <i>Item</i> Untuk 31 Transaksi</b>	<b>Jumlah Transaksi</b>	<b>Jumlah Kemunculan <i>Item</i> untuk 31 Transaksi / Jumlah Transaksi</b>
65	3	31	0
66	0	31	0
67	0	31	0
68	0	31	0,129032258064516
69	0	31	0,096774193548387
70	0	31	0
71	0	31	0
72	2	31	0,032258064516129
73	0	31	0
74	0	31	0,032258064516129
75	0	31	0
76	1	31	0,064516129032258
77	0	31	0
78	1	31	0
79	0	31	0
80	0	31	0
81	1	31	0
82	0	31	0
83	0	31	0
84	0	31	0,032258064516129
85	0	31	0
86	0	31	0
87	0	31	0
88	3	31	0,064516129032258
89	0	31	0,064516129032258
90	2	31	0,129032258064516
91	0	31	0,032258064516129
92	2	31	0
93	2	31	0,096774193548387
94	1	31	0
95	4	31	0,064516129032258
96	0	31	0,032258064516129
97	0	31	0
98	0	31	0
99	0	31	0

Tabel 4.2 Penghitungan Nilai *Support* untuk Seluruh *Item* (lanjutan)

<b>Id_Item</b>	<b>Jumlah Kemunculan <i>Item</i> Untuk 31 Transaksi</b>	<b>Jumlah Transaksi</b>	<b>Jumlah Kemunculan <i>Item</i> untuk 31 Transaksi / Jumlah Transaksi</b>
100	0	31	0
101	0	31	0
102	1	31	0
103	0	31	0
104	1	31	0
105	0	31	0
106	0	31	0
107	0	31	0
108	0	31	0
109	0	31	0
110	0	31	0,032258064516129
111	0	31	0
112	0	31	0
113	8	31	0,032258064516129
114	11	31	0,032258064516129
115	0	31	0
116	0	31	0
117	1	31	0,354838709677419
118	1	31	0,258064516129032
119	0	31	0
120	1	31	0
121	0	31	0
122	1	31	0
123	0	31	0,129032258064516
124	4	31	0
125	0	31	0,032258064516129
126	0	31	0
127	0	31	0,032258064516129
128	2	31	0
129	0	31	0
130	0	31	0
131	6	31	0,161290322580645
132	0	31	0
133	2	31	0
134	0	31	0

Tabel 4.2 Penghitungan Nilai *Support* untuk Seluruh *Item* (lanjutan)

<b>Id_Item</b>	<b>Jumlah Kemunculan <i>Item</i> Untuk 31 Transaksi</b>	<b>Jumlah Transaksi</b>	<b>Jumlah Kemunculan <i>Item</i> untuk 31 Transaksi / Jumlah Transaksi</b>
135	0	31	0,032258064516129
136	0	31	0
137	0	31	0,064516129032258
138	5	31	0,193548387096774
139	0	31	0
140	0	31	0,064516129032258
141	0	31	0
142	1	31	0
143	0	31	0
144	0	31	0
145	0	31	0,064516129032258
146	4	31	0
147	0	31	0
148	0	31	0,032258064516129
149	0	31	0
150	0	31	0,161290322580645
151	0	31	0,032258064516129
152	0	31	0
153	2	31	0
154	0	31	0,129032258064516
155	0	31	0
156	1	31	0
157	0	31	0
158	5	31	0
159	1	31	0
160	0	31	0
161	0	31	0
162	0	31	0
163	0	31	0
164	0	31	0
165	0	31	0
166	0	31	0
167	0	31	0
168	0	31	0

**Langkah 2. Mengurutkan *item* berdasarkan nilai *support* terbesar sampai terkecil.**

Berdasarkan Tabel 4.2 Penghitungan Nilai *Support* untuk Seluruh *Item*, telah didapat nilai *support* untuk masing-masing *item*. *Item-item* pada Tabel 4.2 diurutkan secara *descending* (dari nilai terbesar sampai nilai terkecil) berdasarkan nilai *support*-nya. Hasil pengurutan ditampilkan pada Tabel 4.3

Tabel 4.3 Hasil Pengurutan *Item* secara *descending* berdasarkan nilai *support*

<b>Id_Item</b>	<b>Nilai Support</b>
1	0,580645161290323
117	0,354838709677419
118	0,258064516129032
138	0,193548387096774
40	0,161290322580645
131	0,161290322580645
150	0,161290322580645
9	0,129032258064516
68	0,129032258064516
90	0,129032258064516
123	0,129032258064516
154	0,129032258064516
3	0,096774193548387
69	0,096774193548387
93	0,096774193548387
10	0,064516129032258
25	0,064516129032258
29	0,064516129032258
42	0,064516129032258
47	0,064516129032258
58	0,064516129032258
76	0,064516129032258
88	0,064516129032258
89	0,064516129032258
95	0,064516129032258
137	0,064516129032258
140	0,064516129032258
145	0,064516129032258
2	0,032258064516129

Tabel 4.3 Hasil Pengurutan *Item* secara *descending* berdasarkan nilai *support* (lanjutan)

<b>Id_Item</b>	<b>Nilai Support</b>
5	0,032258064516129
7	0,032258064516129
8	0,032258064516129
13	0,032258064516129
26	0,032258064516129
27	0,032258064516129
30	0,032258064516129
34	0,032258064516129
41	0,032258064516129
44	0,032258064516129
46	0,032258064516129
48	0,032258064516129
49	0,032258064516129
51	0,032258064516129
52	0,032258064516129
56	0,032258064516129
59	0,032258064516129
72	0,032258064516129
74	0,032258064516129
84	0,032258064516129
91	0,032258064516129
96	0,032258064516129
110	0,032258064516129
113	0,032258064516129
114	0,032258064516129
125	0,032258064516129
127	0,032258064516129
135	0,032258064516129
148	0,032258064516129
151	0,032258064516129
4	0
6	0
11	0
12	0
14	0

Tabel 4.3 Hasil Pengurutan *Item* secara *descending* berdasarkan nilai *support* (lanjutan)

<b>Id_Item</b>	<b>Nilai Support</b>
15	0
16	0
17	0
18	0
19	0
20	0
21	0
22	0
23	0
24	0
28	0
31	0
32	0
33	0
35	0
36	0
37	0
38	0
39	0
43	0
45	0
50	0
53	0
54	0
55	0
57	0
60	0
61	0
62	0
63	0
64	0
65	0
66	0
67	0
70	0

Tabel 4.3 Hasil Pengurutan *Item* secara *descending* berdasarkan nilai *support* (lanjutan)

<b>Id_Item</b>	<b>Nilai Support</b>
71	0
73	0
75	0
77	0
78	0
79	0
80	0
81	0
82	0
83	0
85	0
86	0
87	0
92	0
94	0
97	0
98	0
99	0
100	0
101	0
102	0
103	0
104	0
105	0
106	0
107	0
108	0
109	0
111	0
112	0
115	0
116	0
119	0
120	0
121	0

Tabel 4.3 Hasil Pengurutan *Item* secara *descending* berdasarkan nilai *support* (lanjutan)

<b>Id_Item</b>	<b>Nilai Support</b>
122	0
124	0
126	0
128	0
129	0
130	0
132	0
133	0
134	0
136	0
139	0
141	0
142	0
143	0
144	0
146	0
147	0
149	0
152	0
153	0
155	0
156	0
157	0
158	0
159	0
160	0
161	0
162	0
163	0
164	0
165	0
166	0
167	0
168	0

**Langkah 3. Mengeliminasi *item* yang memiliki nilai *support* kurang dari minimum *support***

Untuk perhitungan manual digunakan nilai minimum *support* = 10 %. Nilai desimal minimum *support* = 0,10. Apabila terdapat *item* yang memiliki nilai *support* lebih kecil dari pada minimum *support*, maka *item* tersebut dihapus.

Tabel 4.4 Hasil Eliminasi *Item* yang memiliki nilai *support* lebih dari minimum *support*

ID_ITEM	Nilai Support
1	0.580645161290323
117	0.354838709677419
118	0.258064516129032
138	0.193548387096774
40	0.161290322580645
131	0.161290322580645
150	0.161290322580645
9	0.129032258064516
68	0.129032258064516
90	0.129032258064516
123	0.129032258064516
154	0.129032258064516

**Langkah 4. Mengeliminasi *item-item* pada setiap transaksi yang memiliki nilai *support* kurang dari minimum *support*.**

Dilakukan eliminasi terhadap *item infrequent* pada daftar *item-item* untuk setiap transaksi pada Tabel 4.1. *Item infrequent* adalah *item* yang memiliki nilai *support* kurang dari nilai *minimum support*.

Tabel 4.5 Hasil Eliminasi *item-item* untuk setiap transaksi

ID_TRANSAKSI	ID_ITEM
1	131
3	9, 1, 118
4	118, 68, 1, 90
5	1, 138
6	1, 150
7	118, 138
8	131, 117, 1
9	1, 117, 118

Tabel 4.5 Hasil Eliminasi *item-item* untuk setiap transaksi (lanjutan)

ID_TRANSAKSI	ID_ITEM
10	123
11	150, 117, 131
13	117, 154, 118
14	1, 90, 117, 131
15	1
16	117, 150, 154, 40, 1
17	40, 1
18	117, 154, 9, 1
19	138
20	117
21	1, 123, 131
22	40 1
23	68
24	117, 138, 1
25	118, 1
26	123, 138, 118, 150
27	1, 68, 90, 40, 117, 138
28	1, 40, 9, 117, 150, 154, 90
29	68, 118
31	9, 1, 123

**Langkah 5. Mengurutkan *item-item* yang tersisa untuk setiap transaksi sesuai dengan urutan *L*.**

Pada Tabel 4.5, *item-item* penyusun setiap transaksi masih dalam urutan awal berdasarkan user *input*. *Item-item* setiap transaksi untuk dapat dimasukkan ke dalam *POC-tree* harus dalam urutan sesuai dengan urutan *L*. Urutan *L* adalah urutan sesuai dengan nilai *support* seluruh *item* dari nilai terbesar sampai terkecil. Pada Tabel 4.6, *item-item* pada Tabel 4.5 diurutkan sesuai dengan urutan *L*.

Tabel 4.6 Hasil mengurutkan *Item-item* tersisa untuk setiap transaksi sesuai urutan *L*.

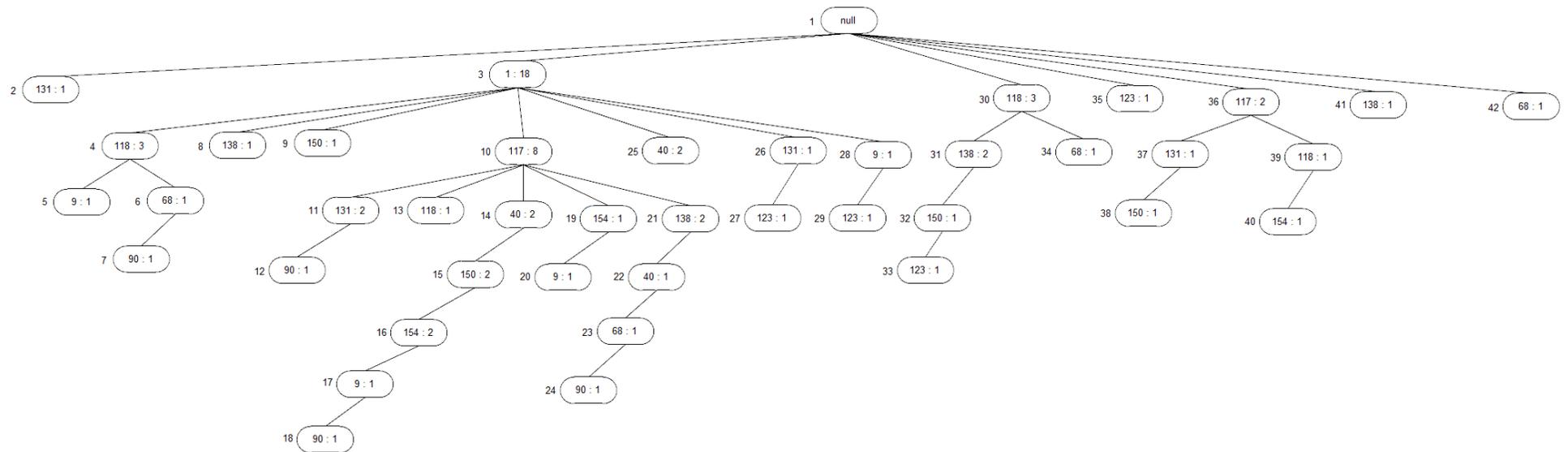
ID_TRANSAKSI	ID_ITEM
1	131
3	1, 118, 9
4	1, 118, 68, 90
5	1, 138
6	1, 150

Tabel 4.6 Hasil mengurutkan *Item-item* tersisa untuk setiap transaksi sesuai urutan *L* (lanjutan).

<b>ID_TRANSAKSI</b>	<b>ID_ITEM</b>
7	118, 138
8	1, 117, 131
9	1, 117, 118
10	123
11	117, 131, 150
13	117, 118, 154
14	1, 117, 131, 90
15	1
16	1, 117, 40, 150, 154
17	1, 40
18	1, 117, 154, 9
19	138
20	117
21	1, 131, 123
22	1, 40
23	68
24	1, 117, 138
25	1, 118
26	118, 138, 150, 123
27	1, 117, 138, 40, 68, 90
28	1, 117, 40, 150, 154, 9, 90
29	118, 68
31	1, 9, 123

### Langkah 6. Memasukkan *item-item* pada setiap transaksi pada POC Tree.

Seluruh *id\_item* untuk tiap transaksi pada Tabel 4.6 dimasukkan pada *POC-tree* satu per satu untuk setiap transaksi. *Item-item* pada setiap transaksi dimasukkan mulai dari *node null*. Contoh, untuk *id\_transaksi* = 1, *id\_item* 131 dimasukkan mulai dari *null*, kemudian menambahkan satu *node* baru sebagai *child* dari *node null*. Format *child node* adalah {*item-name* : *count*}. Untuk *id\_item* = 1 format *child node* menjadi {131 : 1}. Proses diulang sampai seluruh *item-item* untuk setiap transaksi selesai dimasukkan ke dalam *POC tree*. *POC-tree* sebagai struktur data untuk menyimpan seluruh transaksi, sehingga tidak diperlukan lagi untuk memindai *database* untuk mendapatkan *item-item* setiap transaksi.



Gambar 4.12 Hasil *POC-tree* setelah dimasukkan oleh 31 transaksi

**Langkah 7. Memindai POC tree untuk menyusun frequent 1-itemset**

Setelah seluruh *item-item* pada setiap transaksi tersimpan pada *POC-tree*, maka dilakukan pemindaian *tree* untuk menyusun *frequent 1-itemset*. *Frequent 1-itemset* terdiri dari kombinasi 1 *item* menu makanan atau minuman. Menggunakan format *nodeset frequeunt 1-itemset* yaitu  $\{(preorder_i, count_i)\}$ . *Preorder* adalah nilai *preorder* dari *node*, *count* adalah nilai jumlah *node* dilalui oleh setiap transaksi.

Format Penulisan *Nodeset frequent 1-itemset*:  $item \rightarrow \{(preorder_i, count_i)\}$

- 1  $\rightarrow \{(3, 18)\}$
- 117  $\rightarrow \{(10, 8)\}$
- 118  $\rightarrow \{(4, 3), (13, 1), (30, 3), (38, 1)\}$
- 138  $\rightarrow \{(8, 1), (21, 2), (31, 2), (40, 1)\}$
- 40  $\rightarrow \{(14, 2), (22, 1), (25, 2)\}$
- 131  $\rightarrow \{(2, 1), (11, 2), (26, 1), (36, 1)\}$
- 150  $\rightarrow \{(9, 1), (15, 2), (32, 1), (37, 1)\}$
- 9  $\rightarrow \{(5, 1), (17, 1), (20, 1), (28, 1)\}$
- 68  $\rightarrow \{(6, 1), (23, 1), (41, 1)\}$
- 90  $\rightarrow \{(7, 1), (12, 1), (18, 1), (24, 1)\}$
- 123  $\rightarrow \{(27, 1), (29, 1), (33, 1), (34, 1)\}$
- 154  $\rightarrow \{(16, 2), (19, 1), (39, 1)\}$

**Langkah 8. Memindai POC tree untuk menyusun frequent 2-itemset yang berdasar pada frequent 1-itemset.**

Setelah *frequent 1-itemset* telah terbentuk maka dilanjutkan untuk membuat *frequent 2-itemset*. *Frequent 2-itemset* adalah kombinasi dari 2 *item* makanan atau minuman. *Frequent 2-itemset* dibentuk dengan cara memindai *POC-tree* menggabungkan 2 *node* dengan persyaratan masih memiliki *ancestor* yang sama. Berdasarkan *POC-tree* yang terbentuk pada Gambar 4.8, maka didapatkan daftar *parent* beserta *child* yang dimiliki *parent* pada Tabel 4.7. Format Penulisan pada *Child* adalah  $\{idNode : HeadNode\}$ , *idNode* adalah nilai *id node* pada *POC-tree*, dan *headNode* adalah nilai *head* atau *id item* pada *POC-tree*.

Tabel 4.7 Daftar *Parent* beserta *Child*

<i>Parent</i>	<i>Child</i>
117	{11:131, 12:90, 13:118, 14:40, 15:150, 16:154, 17:9, 18:90, 19:154, 20:9, 21:138, 22:40, 23:68, 24:90, 37:131, 38:150, 39:18, 40:154}

Tabel 4.7 Daftar *Parent* beserta *Child* (lanjutan)

<b>Parent</b>	<b>Child</b>
1	{4:118, 5:9, 6:68, 7:90, 8:138, 9:150, 10:117, 11:131, 12:90, 13:118, 14:40, 15:150, 16:154, 17:9, 18:90, 19:154, 20:9, 21:138, 22:40, 23:68, 24:90, 37:131, 38:150, 39:118, 40:154, 25:40, 26:131, 27:123, 28:9, 29:123}
118	{5:9, 6:68, 7:90, 31:138, 32:150, 33:123, 34:1, 40:154}
138	{22:40, 23:68, 24:90, 32:150, 33:123}
40	{15:150, 16:154, 17:9, 18:90, 23:68, 24:90}
131	{12:90, 27:123, 38:150}
150	{16:154, 17:9, 18:90, 33:123}
9	{18:90, 29:123}
66	{7:90, 24:90}
90	{}
123	{}
154	{17:9, 18:90, 20:9}

Untuk mendapatkan dua *child* yang sama masih dalam 1 *ancestor* maka dilakukan, dengan cara mencari *child* yang memiliki 2 *head node* yang sama untuk suatu *parent*. Contoh pada *parent* 117 maka ditemukan *child* dengan 2 *head node* yang sama pada 11:131 dan 37:131. Berdasarkan cara ini maka ditemukan seluruh *head node* yang memiliki *ancestor* yang sama pada Tabel 4.8. Hasil dari pasangan *head* → 2 *head Node* yang sama untuk 1 *Parent* menjadi hasil dari *frequent 2-itemset*.

Tabel 4.8 Hasil 2 *Head Node* sama pada 1 *Parent*

<b>Head</b>	<b>2 Head Node yang sama untuk 1 Parent</b>
117	{11:131, 37:131}
117	{12:90, 18:90}
117	{13:118, 39:118}
117	{12:90, 24:90}
117	{18:90, 24:90}
117	{14:40, 22:40}
117	{15:150, 38:150}
117	{16:154, 19:154}
117	{16:154, 40:154}
117	{19:154, 40:154}
1	{4:118, 39:118}
1	{5:9, 17:9}
1	{5:9, 20:9}

Tabel 4.8 Hasil 2 *Head Node* sama pada 1 *Parent* (lanjutan)

<i>Head</i>	<i>2 Head Node yang sama untuk 1 Parent</i>
1	{17:9, 20:9}
1	{5:9, 28:9}
1	{17:9, 28:9}
1	{20:9, 28:9}
1	{6:68, 23:68}
1	{7:90, 18:90}
1	{7:90, 24:90}
1	{12:90, 18:90}
1	{12:90, 24:90}
1	{18:90, 24:90}
1	{8:138, 21:138}
1	{9:150, 15:150}
1	{9:150, 38:150}
1	{9:150, 38:150}
1	{11:131, 37:131}
1	{11:131, 26:131}
1	{37:131, 26:131}
1	{14:40, 22:40}
1	{14:40, 25:40}
1	{22:40, 25:40}
1	{16:154, 19:154}
1	{16:154, 40:154}
1	{19:154, 40:154}
1	{27:123, 29:123}
40	{8:90, 24:90}
68	{7:90, 24:90}
154	{17:9, 20:9}

### Langkah 9. Memindai *POC tree* untuk menyusun *frequent 3-itemset* yang berdasar pada *frequent 2-itemset*

Seluruh *frequent 2-itemset* terbentuk maka dilanjutkan untuk membentuk *frequent 3-itemset*. *Frequent 3-itemset* dibentuk dari memindai *POC-tree* untuk menghasilkan kombinasi 3 *item* menu makanan atau minuman. Syarat untuk mengkombinasikan dua *frequent 2-itemset* memiliki nilai *preorder* serta *count* yang saling beririsan antara dua *frequent 2-itemset*. Contoh cara untuk menemukan *mining frequent 3-itemset* yaitu mencari 1 hasil *frequent 2-itemset*

yang memiliki *idNode* serta *headNode* yang sama dengan 1 hasil *frequent 2-itemset* yang lain pada Tabel 4.8. Seperti  $117 \rightarrow \{11:131, 37:131\}$  dengan  $1 \rightarrow \{11:131, 26:131\}$ . Antara  $117 \rightarrow \{11 : 131, 37 : 131\}$  memiliki *idNode* dan *idHead* yang sama pada 11:131 pada  $1 \rightarrow \{11:131, 26:131\}$ . Pada 11:131 menunjukkan 11 adalah *idNode* serta 131 adalah *headNode*. Sehingga 11:131 adalah irisan dari  $117 \rightarrow \{11:131, 37:131\}$  dengan  $1 \rightarrow \{11:131, 26:131\}$ . Untuk membentuk kombinasi 3 *item* pada *frequent 3-itemset* maka *parent* 117 pada  $117 \rightarrow \{11:131, 37:131\}$ , *parent* 1 pada  $1 \rightarrow \{11:131, 26:131\}$ , dan *head node* 131 pada irisan  $\{11:131\}$  menjadi 1 117 131. Berdasarkan pada Tabel 4.8 didapatkan hasil *frequent 3-itemset* sebagai berikut :

Simbol  $\cap$  adalah irisan antara 2 hasil *frequent 2-itemset*

Kombinasi ke-1

$117 \rightarrow \{11:131, 37:131\}$

$1 \rightarrow \{11:131, 26:131\}$   
 $\cap$   
 $1 \cap 117 \rightarrow \{11:131\}$

Hasil *Frequent 3-itemset*  $\rightarrow 1 \ 117 \ 131$

Kombinasi ke-2

$117 \rightarrow \{12:90, 18:90\}$

$1 \rightarrow \{7:90, 18:90\}$   
 $\cap$   
 $1 \cap 117 \rightarrow \{18:90\}$

Hasil *Frequent 3-itemset*  $\rightarrow 1 \ 117 \ 90$

Kombinasi ke-3

$117 \rightarrow \{13:118, 39:118\}$

$1 \rightarrow \{4:118, 39:118\}$   
 $\cap$   
 $1 \cap 117 \rightarrow \{39:118\}$

Hasil *Frequent 3-itemset*  $\rightarrow 1 \ 117 \ 118$

Kombinasi ke-4

$117 \rightarrow \{14:40, 22:40\}$

$1 \rightarrow \{14:40, 25:40\}$   
 $\cap$   
 $1 \cap 117 \rightarrow \{14:40\}$

Hasil *Frequent 3-itemset*  $\rightarrow 1 \ 117 \ 40$

Kombinasi ke-5

$117 \rightarrow \{15:150, 38:150\}$

$1 \rightarrow \{9:150, 38:150\}$   
 $\cap$   
 $1 \cap 117 \rightarrow \{38:150\}$

Hasil *Frequent 3-itemset*  $\rightarrow 1 \ 117 \ 150$

Kombinasi ke-6

117 → {16:154, 19:154}

1 → {16:154, 40:154}    ∩

1 117 → {16:154}

Hasil *Frequent 3-itemset* → 1 117 154

Kombinasi ke-7

1 → {7:90, 24:90}

40 → {8:90, 24:90}    ∩

1 40 → {24:90}

Hasil *Frequent 3-itemset* → 1 40 90

Kombinasi ke-8

1 → {7:90, 18:90}

68 → {7:90, 24:90}    ∩

1 68 → {7:90}

Hasil *Frequent 3-itemset* → 1 68 90

Kombinasi ke-9

1 → {17:9, 28:9}

154 → {17:9, 20:9}    ∩

1 154 → {17:9}

Hasil *Frequent 3-itemset* → 1 154 9

Kombinasi ke-10

117 → {18:90, 24:90}

40 → {18:90, 24:90}    ∩

117 40 → {18:90}

Hasil *Frequent 3-itemset* → 117 40 90

Kombinasi ke-11

117 → {12:90, 24:90}

68 → {7:90, 24:90}    ∩

117 68 → {24:90}

Hasil *Frequent 3-itemset* → 117 68 90

**Langkah 10. Menyusun kombinasi menu paket makanan dan minuman berdasarkan *frequent 3-itemset*.**

Berdasarkan hasil kombinasi *item* makanan atau minuman pada *frequent 3-itemset* dapat dibuat kombinasi Menu Paket Makanan. Kombinasi *id\_item* pada *frequent 3-itemset* dikonversi menjadi nama *item* menu dengan *query* pada *database* menu.

Tabel 4.9 Hasil Kombinasi Menu Paket Makanan

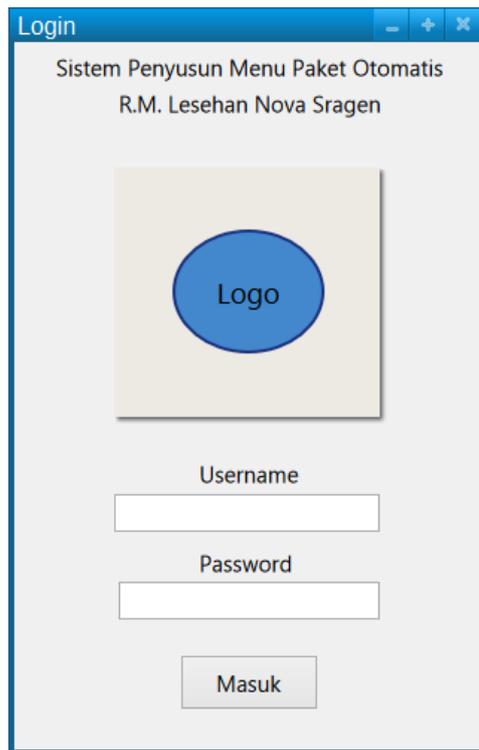
No	Hasil Kombinasi <i>Frequent 3-itemset</i>	Hasil Kombinasi Menu Paket Makanan
1	1 117 131	Nasi Putih, Ice Tea, Ice Lemon Tea
2	1 117 90	Nasi Putih, Ice Tea, Nila Bakar / biji
3	1 117 118	Nasi Putih, Ice Tea, Ice Jeruk
4	1 117 40	Nasi Putih, Ice Tea, Ayam Goreng / biji
5	1 117 150	Nasi Putih, Ice Tea, Hot Tea
6	1 117 154	Nasi Putih, Ice Tea, Ades Tanggung Biasa
7	1 40 90	Nasi Putih, Ayam Goreng / biji, Kentang Goreng
8	1 68 90	Nasi Putih, Steak Ayam, Kentang Goreng
9	1 154 9	Nasi Putih, Ades Tanggung Biasa, Nila Bakar / biji
10	117 40 90	Ice Tea, Ayam Goreng / biji, Kentang Goreng
11	117 68 90	Ice Tea, Steak Ayam, Kentang Goreng

#### 4.4 Perancangan Desain Antarmuka

Perancangan desain antarmuka digunakan untuk mendesain antarmuka yang akan diimplementasikan pada sistem. Antarmuka didesain menyesuaikan dengan pengembangan aplikasi yang berbasis *desktop*. Desain antarmuka terdiri dari berbagai *frame* seperti *frame login*, *frame masukan*, *frame transaksi*, dan *frame menu paket*.

##### 4.4.1 Desain Antarmuka *Frame Login*

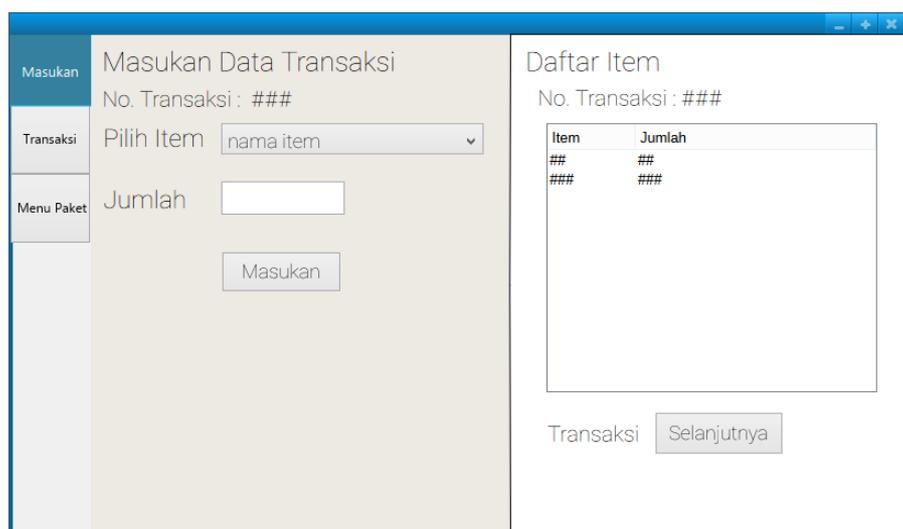
Perancangan desain antarmuka untuk *frame Login*. *Frame login* terdiri dari komponen antarmuka nama sistem, logo rumah makan, masukan *username*, masukan *password*, dan tombol untuk melakukan *login*. Antarmuka *login* digunakan untuk pengecekan pengguna yang dapat menggunakan sistem.



Gambar 4.13 Antarmuka *Frame Login*

#### 4.4.2 Desain Antarmuka *Frame Masukan*

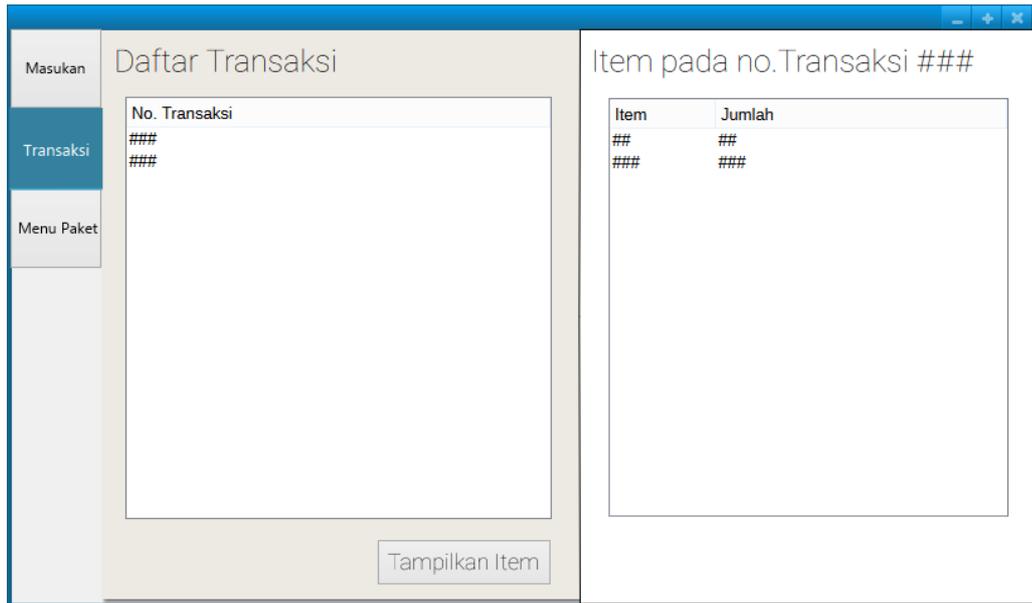
Antarmuka *Frame Masukan* digunakan oleh pengguna untuk memasukan data ke dalam *database* sistem. Data yang digunakan oleh pengguna adalah *struk* pembayaran atau nota pesanan. Pada *frame Masukan* terdapat dua *workspace* yaitu *workspace* pertama untuk memasukkan data pada bagian kiri, *workspace* kedua untuk menampilkan data *item* untuk satu transaksi yang telah tersimpan pada *database* pada bagian kanan.



Gambar 4.14 Antarmuka *Frame Masukan*

#### 4.4.3 Desain Antarmuka *Frame* Transaksi

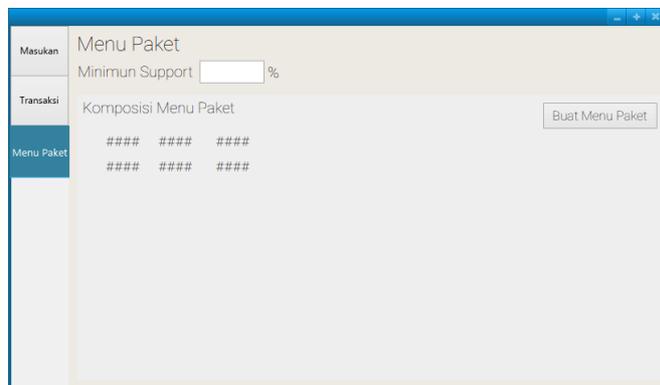
Antarmuka *Frame* Transaksi digunakan oleh pengguna untuk melihat daftar *item* untuk suatu transaksi, yang telah dimasukkan sebelumnya pada menu Masukan. Pengguna dapat memilih No. Transaksi yang diinginkan untuk melihat secara spesifik daftar *item-item* penyusun transaksi pada *workspace Item* pada bagian kanan. Pada *workspace* bagian kiri terdapat Tabel yang berisi seluruh daftar No. Transaksi yang dapat ditekan oleh pengguna untuk melihat daftar *item* untuk No. Transaksi tersebut.



Gambar 4.15 Antarmuka *Frame* Transaksi

#### 4.4.4. Desain Antarmuka *Frame* Menu Paket

Antarmuka *Frame* Menu Paket digunakan oleh pengguna untuk menghasilkan menu paket. Pengguna memasukkan nilai minimum *Support* yang diinginkan sebagai *threshold* menu paket yang akan dihasilkan oleh sistem. Setelah penggunaan mengisi nilai *minimum support* kemudian menekan tombol “Buat Menu Paket”, kombinasi menu paket ditampilkan sistem pada *workspace* Komposisi Menu Paket.



Gambar 4.16 Antarmuka *Frame* Menu Paket