BAB 3
ANALISIS DAN PERANCANGAN SISTEM

3.1 Analisis Sistem

Analisis sistem adalah penguraian dari suatu informasi yang utuh kedalam bagian-bagian komponennya dengan maksud untuk mengidentifikasi dan mengevaluasi permasalahan, kesempatan, hambatan yang terjadi dan kebutuhan yang diharapkan sehingga dapat diusulkan perbaikan. Analisis Sistem ini bertujuan untuk menjelaskan permasalahan yang muncul pada saat pembangunan sistem, hal ini bertujuan untuk membantu ketika proses perancangan sistem berlangsung. Tugas utama analisis sistem dalam tahap ini adalah menemukan kelemahan-kelemahan dari sistem yang berjalan sehingga dapat diusulkan perbaikannya. Dalam analisis sistem ini meliputi beberapa bagian, yaitu :

1. Analisis Masalah
2. Analisis Penerepan Metode CRISP-DM
3. Analisis Spesifikasi Kebutuhan Perangkat Lunak
4. Analisis Penerapan Kebutuhan *Fungsional*
5. Analisis Penerapan Kebutuhan *Non Fungsional*

3.1.1 Analisis Masalah

Analisis masalah yang dilakukan adalah melihat dan mengidentifikasi permasalahan atau kendala di dalam penelitian yang dilakukan. Masalah dapat diartikan sebagai suatu pernyataan yang menyatakan tentang situasi yang memerlukan pemecahan. Perlu ada analisis masalah melalui rumusan masalah yang sudah ditentukan sebelumnya. Berdasarkan hasil pengamatan, diketahui bahwa analisis masalahnya adalah sebagai berikut :

1. Dalam menentukan barang-barang apa saja yang akan dijual secara paket, CV.Sultan Jaya Mandiri mengandalkan data barang yang banyak terjual, sehingga belumlah cukup dijadikan sebagai dasar pengambilan keputusan untuk membuat sebuah paket. Karena barang yang terjual belum tentu efektif bila dijual secara bersamaan sebagai paket barang.

2. Pemaketan produk dilakukan pada hari besar tertentu seperti idul fitri, natal dan hari-hari besar lainnya.

3.1.2 Analisis Penerapan Metode CRISP-DM

Metode pembangunan perangkat *data mining* yang digunakan dalam penelitian ini adalah *Cross-Industry Standard Process for Data Mining* (CRISP-DM).

3.1.2.1 Pemahaman Bisnis

Tahapan pemahaman bisnis merupakan tahapan pertama dilakukan dalam kerangka kerja CRISP-DM. Dalam tahapan bisnis ini terdapat beberapa tahapan lainnya, yaitu :

1. Tujuan Bisnis

   Dalam proses bisnisnya CV. Sultan Jaya Mandiri mempunyai tujuan bisnis yaitu untuk memenuhi permintaan *customer* untuk meningkatkan penjualan.

2. Penentuan Sasaran *Data Mining*

   Tujuan dari penerapan *data mining* pada pola pembelian *customer* ini adalah untuk mengetahui bagaimana pola penentuan paket berdasarkan dari data yang sudah ada untuk meningkatkan penjualan.

3.1.2.2 Pemahaman Data

Dalam penelitian ini adalah data transaksi yang terjadi di CV. Sultan Jaya Mandiri yang berasal dari *file microsoft excel* (.xlsx). Namum pada tahap ini, data yang digunakan hanyalah data transaksi penjualan pada tanggal 01/04/2015. Berikut merupakan penjelasan dari masing-masing *atribut* yang terdapat pada tabel data transaksi yang dapat dilihat pada tabel 3.1 dibawah ini :

Tabel 3. 1 Struktur Data Transaksi

Keterangan		
Fungsi	Untuk mengetahui jenis produk –produk yang paling sering terjual oleh customer	
Format	Microsoft Excel (.xlsx)	
Atribut	No	No urut

Keterangan		
Atribut	Tanggal Transaksi	Tanggal proses pembelian
	No Transaksi	Nomor pembelian
	Kode Barang	Kode dari setiap barang
	Nama Barang	Nama barang yang dibeli
	Kategori	Kategori dari setiap nama barang
	Jumlah	Jumlah barang yang dibeli
	Harga	Harga brang yang dibeli
	Total	Total harga yang dibeli

3.1.2.3 Persiapan Data

Persiapan data adalah hal yang harus dilakukan dalam proses *data mining*, karena tidak semua data atau atribut dalam data digunakan pada data *mining*. Proses ini dilakukan agar data yang akan digunakan sesuai dengan kebutuhan. Adapun tahapan-tahapan *preprocessing* data dalam penelitian ini adalah sebagai berikut :

1. Ekstraksi Data

   Ekstraksi data dapat diartikan sebagai proses pengambilan data dari sumber data dalam rangka untuk melanjutkan proses pengolahan data ketingkat selanjutnya ataupun untuk menyimpan data hasil *ekstrak* tersebut. Dalam penelitian ini, data yang berasal dari *flat file* berformat *microsoft excel* di-*ekstrak*, kemudian disimpan kedalam sebuah *database* agar memudahkan dalam proses pengolahan data. Data yang digunakan sebanyak 171 *record* dari 37 transaksi. Data transaksi penjualan terdapat dalam table D-1 dalam lampiran D.

2. Pembersihan Data

   Pembersihan Data merupakan proses menghilangkan *noise* dan menghilangkan data yang tidak relevan atau inkonsisten. Data yang memiliki kategori *non-food* akan diberisihkan dikarenakan hanya kategori *food* dan sembako saja yang akan dijadikan sebagai paket barang. Setelah proses

cleaning jumlah data menjadi 163 *record* dari 38 transaksi. Data yang telah dibersihkan dari hasil ekstraksi data terdapat dalam table D-2 dalam lampiran.

3. Pemilihan Atribut

   Pemilihan atribut adalah proses pemilihan atribut data yang akan digunakan, sehingga data tersebut dapat diproses sesuai dengan kebutuhan *data mining*. Dalam penelitian ini, atribut yang akan digunakan yakni no transaksi dan kode barang yang memiliki tujuan masing-masing, seperti pada no transaksi berguna untuk membedakan pembelian dari yang lain, kode barang berguna untuk mengetahui barang apa saja yang dibeli dalam satu transaksi. Jumlah data setelah proses pemilihan atribut sebanyak 163 dari 37 transaksi. Hasil pemilihan atribut dapat dilihat pada table D-3 dalam lampiran D.

4. Setelah semua proses telah berhasil dilalui dan data transaksi telah sesuai dengan kebutuhan yang diperlukan dalam proses *data mining*, maka data transaksi penjualan tersebut sudah dapat digunakan untuk diproses dalam sistem *data mining* dengan 163 *record* data dari 37 transaksi.

3.1.2.4 Pemodelan

Pemodelan merupakan tahap untuk membuat model atau desain sistem yang akan dibangun. Didalam tahap ini akan dipilih metode yang cocok untuk menyelesaikan masalah di CV. Sultan Jaya Mandiri dalam pembuatan paket barang. Metode yang akan digunakan adalah metode *associaiton rule* dengan algoritma *fp-growth*.

Metode dasar *association rule* dibagi menjadi dua tahap yaitu tahap pencarian pola frekuensi tinggi dan pembentukan atau perhitungan asosiasi dari frekuensi yang sudah didapatkan.

1. Pencarian pola frekuensi tinggi

Tahap ini mencari pola *item* yang memenuhi syarat *minimum* dari nilai *support* dalam *database*. Menurut Lorase, kita bebas menentukan nilai *minimum support* dan *minimum confidence* sesuai kebutuhan. Untuk melihat banyaknya variasi data yang diperoleh, maka pada kasus ini nilai *minimum support* yang akan

digunakan yaitu 7% dari 38 sampel data transaksi yang digunakan yaitu sebesar 2,66 ≈ *3*.

Berikut adalah tahap pencarian frekuensi tinggi dari setiap item pada data transaksi penjualan.

1. Langkah pertama yang dilakukan yaitu menuliskan setiap kode barang dan memberikan nilai *frequent*, dimana nilai *frequent* tersebut didapat dengan cara menghitung kemunculan setiap barang pada data transaksi penjualan. Dari hasil perhitungan *frequent* dapat diketahui jumlah *item* yang terdapat pada data transaksi sebanyak 65 *item*.

   Adapun hasil pengolahan dapat dilihat pada table 3.2 dibawah ini.

Tabel 3. 2 *Frequent Item*

Kode Barang	*Frekuensi*
ASD1	2
ASK1	7
ASC1	2
AKL1	1
AA1	3
AG1	1
BGP1	1
BB1	3
BPK1	4
BB1	1
BR1	3
BTR1	2
BG7	1
CI500	1
ET1	3
EJ1	4

Kode Barang	*Frekuensi*
FO1	2
FF1	4
FYK1	3
GDC3	4
GM1	3
GP1	5
HXA1	1
ICM1	1
IRAB1	3
IKRC1	1
KDK1	2
KAFM31	3
KPB1	1
KRK1	2
KB1	3
KKA1	7

Kode Barang	*Frekuensi*
KLWC1	3
KH4	2
KBA1	4
LDU1	2
MS3	1
MS1	4
MGS1	2
MS2	5
MZE1	2
NB2	3
OJD1	2
PF1	3
PP1	1
PKC1	2
PK1	3
PIC1	1

Kode Barang	*Frekuensi*
PO1	3
RBM1	1
RT1	3
SS5	1
SIAB1	4
SA5	1
SPAB1	4
SPG1	2
SOY1	3
SS1	1
SE1	1
TAN1	1
TU1	1
TE1	3
TA1	6
TULM1	3
TS28	1

2. Langkah kedua adalah menghilangkan *item* yang memiliki *frequent* kurang dari nilai *minimum support* yang telah ditentukan sebelumnya, dalam kasus ini nilai *minimum support* yang telah ditentukan adalah sebesar 3. Dimana *item* yang kurang dari 3 akan dihilangkan dan *item* yang memiliki *frequent* lebih atau sama dengan 3 akan akan digunakan untuk ke proses berikutnya. Setelah dilakukan proses eliminasi terhadap data yang tidak memenuhi *minimum support*, maka jumlah *item* yang memenuhi *minimum support* yaitu sebanyak 31 *item*. Berikut adalah data yang sudah memenuhi nilai *minimum support* terdapat pada tabel 3.3.

Tabel 3. 3 *Item* Yang Memenuhi *Minimum Support*

Kode Kode Barang	*Support Count*
ASK1	7
AA1	3
BB1	3
BPK1	4
BR1	3
ET1	3
EJ1	4
FF1	4
FYK1	3
GDC3	4
GM1	3
GP1	5
IRAB1	3
KAFM31	3
KB1	3

Kode Barang	*Support Count*
KKA1	7
KLWC1	3
KBA1	4
MS1	4
MS2	5
NB2	3
PF1	3
PK1	3
PO1	3
RT1	3
SIAB1	4
SPAB1	4
SOY1	3
TE1	3
TA1	6
TULM1	3

3. Langkah ketiga yaitu memberikan *priority* untuk setiap *item*. Setelah mendapatkan data yang memenuhi nilai *minimum support* seperti pada tabel 3.3 diatas, selanjutnya melakukan proses pengurutan secara *descending* dari kemunculan setiap frekuensi (*frequent*) yang muncul dan memberikan *priority* sebagai tanda dari setiap *item* yang memiliki *frequent* yang paling besar sampai dengan terkecil. Jika terdapat kode barang dengan *frequent* yang sama maka yang didahulukan adalah kode barang yang memiliki susunan huruf paling awal berdasarkan susunan huruf dalam alphabet. Jumlah *item* yang dicari *priority* nya yaitu sebanyak 31 *item.*

Berikut ini adalah hasil pengurutan *frequent item* berdasarkan proritas terdapat pada tabel 3.4.

Tabel 3. 4 Pemberian nilai *Priority*

Kode Barang	*Support Count*	*Priority*
ASK1	7	1
KKA1	7	2
TA1	6	3
GP1	5	4
MS2	5	5
BPK1	4	6
EJ1	4	7
FF1	4	8
GDC3	4	9
KBA1	4	10
MS1	4	11
SIAB1	4	12
SPAB1	4	13
AA1	3	14
BB1	3	15
BR1	3	16
ET1	3	17
FYK1	3	18
GM1	3	19
IRAB1	3	20
KAFM31	3	21
KB1	3	22
KLWC1	3	23
NB2	3	24
PF1	3	25

Kode Barang	*Support Count*	*Priority*
PK1	3	26
PO1	3	27
RT1	3	28
SOY1	3	29
TE1	3	30
TULM1	3	31

4. Setelah semua *item* diketahui nilai *priority*-nya maka data transaksi yang mengandung *item* yang memenuhi *minimum support* akan diurutkan berdasarkan nilai *priority*. Pada tahap ini *item* yang memiliki *frequent* tertinggi atau nilai *priority* tertinggi akan didahulukan penulisannya. Pada tahap ini, jumlah transaksi yang didaptkan yaitu sebesar 37 transaksi.
   Hasil dari pengurutan data berdasarkan *priority* terdapat pada tabel 3.5.

Tabel 3. 5 Hasil Pengurutan data Berdasarkan *Priority*

No Transaksi	Kode Barang
137505	BPK1
137505	EJ1
137505	KBA1
137505	PO1
137506	FF1
137506	SPAB1
137506	IRAB1
137507	BPK1
137507	SPAB1
137507	BB1
137507	IRAB1
137507	PK1
137507	PO1

No Transaksi	Kode Barang
137507	RT1
137508	SIAB1
137508	BB1
137508	KB1
137508	NB2
137509	TA1
137509	EJ1
137509	KBA1
137509	BR1
137509	KB1
137510	ASK1
137510	SIAB1
137510	SPAB1

No Transaksi	Kode Barang
137510	KB1
137510	NB2
137511	TA1
137511	GP1
137511	BR1
137511	KAFM31
137512	ASK1
137512	KKA1
137512	SPAB1
137512	KAFM31
137513	TA1
137513	BR1
137513	KAFM31
137514	ASK1
137514	KKA1
137514	ET1
137514	PF1
137515	TA1
137515	GP1
137515	GM1
137515	TULM1
137516	GP1
137516	KBA1
137516	MS2
137516	EJ1
137516	GM1
137517	MS2
137517	MS1

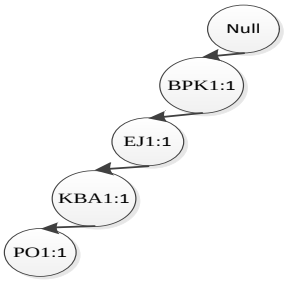
No Transaksi	Kode Barang
137517	RT1
137518	TULM1
137519	KKA1
137519	FF1
137520	GP1
137520	MS2
137520	AA1
137520	PF1
137520	TE1
137521	KKA1
137521	KLWC1
137522	FF1
137522	SIAB1
137522	AA1
137522	IRAB1
137523	ASK1
137523	PK1
137524	GDC3
137524	AA1
137524	KLWC1
137524	SOY1
137525	RT1
137525	TE1
137526	ET1
137527	FF1
137527	SOY1
137528	ET1
137528	TE1

No Transaksi	Kode Barang
137529	GP1
137531	MS1
137531	GM1
137532	BPK1
137532	FYK1
137532	PK1
137533	ASK1
137533	KKA1
137533	EJ1
137533	GDC3
137533	KBA1
137534	GDC3
137535	TA1
137535	SIAB1
137536	ASK1
137536	KKA1
137536	KLWC1

No Transaksi	Kode Barang
137537	ASK1
137537	MS2
137537	MS1
137538	KKA1
137538	FYK1
137538	PF1
137539	GDC3
137539	FYK1
137539	PO1
137540	BB1
137540	SOY1
137541	BPK1
137541	NB2
137541	TULM1
137542	TA1
137542	MS2
137542	MS1

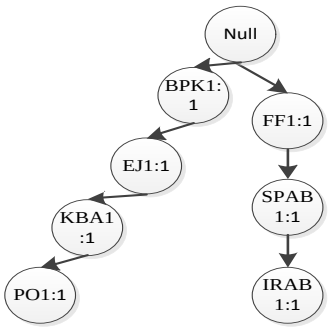
5. Langkah selanjutnya membuat *fp- tree*. Berikut adalah tahap-tahap dalam pembuatan *fp-tree* yaitu :

1. Transaksi 137505 :BPK1, EJI1, KBA1, PO1
   a. Lakukan pembacaan pada data transaksi pertama yang akan diproses.
   b. Lakukan pengecekan pada *root* yang diinisialisasikan dengan *Null*, karena *root* belum mempunyai anak (*child*), maka data yang dibaca menjadi anak pertama dari *root*. Dan *support count* dari masing-masing *node* bertambah 1, kecuali *root*.



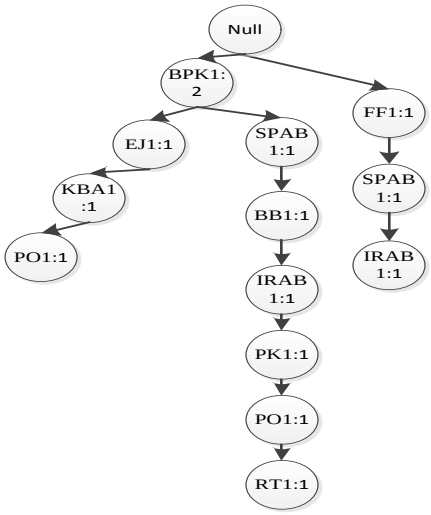
Gambar 3. 1 Hasil pembentukan *FP-Tree* Setelah Transaksi 137505

2. Transaksi 137506 :FF1, SPAB1, IRAB1
   a. Lakukan pembacaan pada data transaksi kedua yang akan diproses.
   b. Lakukan pengencekan pada *root*, karena *root* sudah mempunyai anak dan data yang dibaca berbeda dengan data yang dibaca sebelumnya maka itu menjadi anak terakhir dari *root* dan menjadi *sibling* dari *node* data pertama. Dan *support count* dari *node* tersebut bertambah 1.



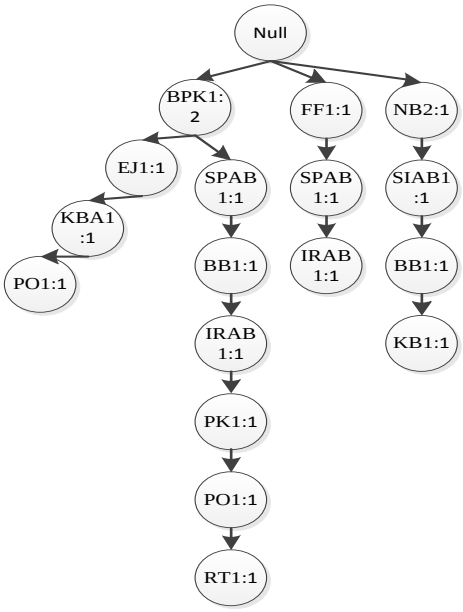
Gambar 3. 2 Hasil pembentukan *FP-Tree* setelah Transaksi 137506

3. Transaksi 137507 :BPK1, SPAB1, BB1, IRAB1, PK1, PO1, RT1
   a. Lakukan pembacaan pada data transaksi ketiga yang akan diproses.
   b. Lakukan pengencekan pada *root*, karena *root* sudah mempunyai anak dan data yang dibaca sama dengan data pertama yang dibaca namun berbeda lintasan maka *node* pada lintasan yang lain tidak bertambah *support count*-nya, hanya data yang dibaca saja yang bertambah *support count*-nya.



Gambar 3. 3 Hasil pembentukan *FP-Tree* setelah Transaksi 137507

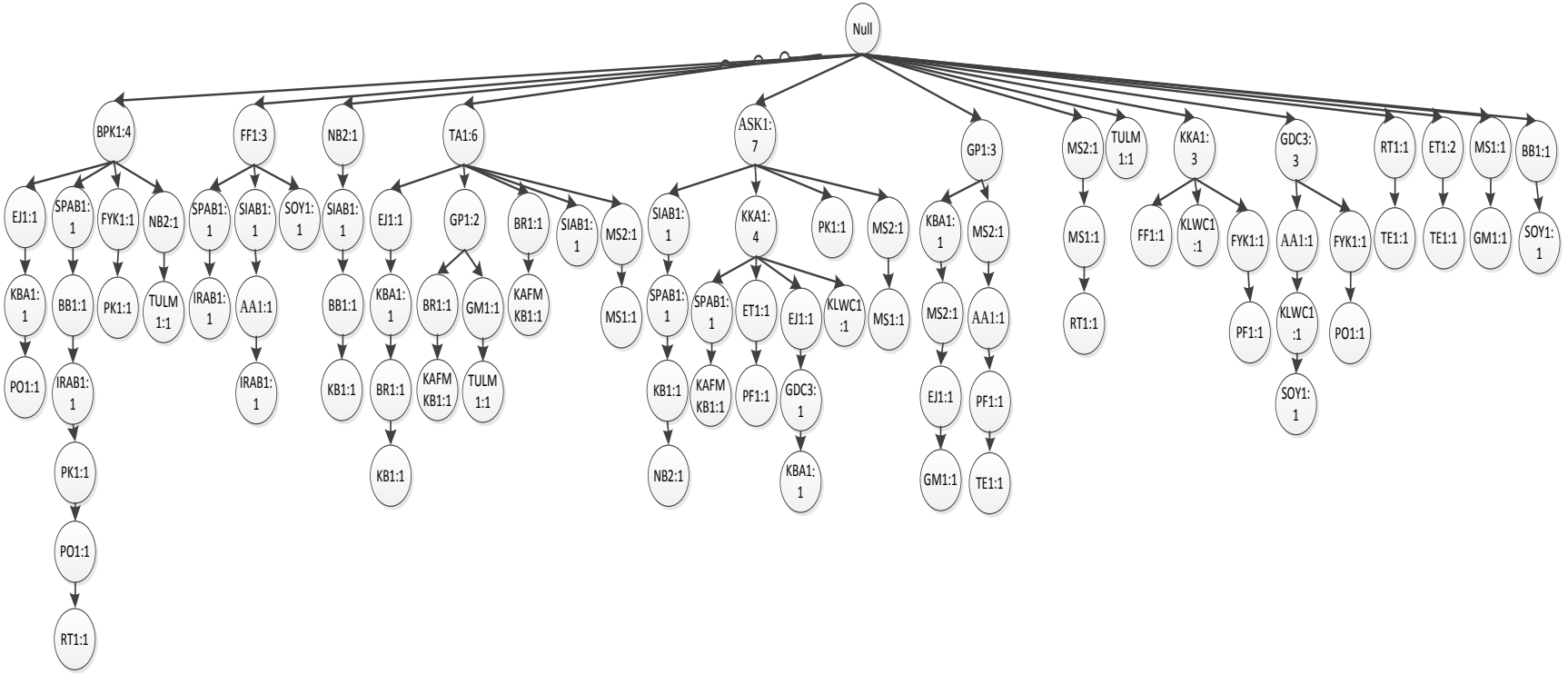
4. Transaksi 137508 : NB2, SIAB1, BB1, KB1
   a. Lakukan pembacaan pada data transaksi keempat yang akan diproses.
   b. Lakukan pengencekan pada *root*, karena *root* sudah mempunyai anak dan data yang dibaca berbeda dengan data yang dibaca sebelumnya maka itu menjadi anak terakhir dari *root*. Dan *support count* dari *node* tersebut bertambah 1.



Gambar 3. 4 Hasil pembentukan *FP-Tree* setelah Transaksi 137508

5. Transaksi 137542 :TA1, MS2, MS1

Setelah proses pembentukan *tree* trakhir maka terbentuklah tree dari semua transaksi yang akan digunakan seperti yang terdapat pada gambar 3.5 berikut ini.



Gambar 3. 5hasil pembentukan *FP-Tree* setelah pembacaan Transaksi 137542

Pembentukan lengkap *fp-tree* dapat dilihat pada table D-4 dalam lampiran D.

Setelah tahap pembangunan *fp-tree* dari sekumpulan data transaksi dilakukan, maka akan dilakukan pembangkitan *conditional pattern base, conditional fp-tree, frequent pattern.*

1. Tahapan Pembangkitan *Conditional Pattern Base*

   Dalam tahap ini mencari *support count* terkecil dari setiap *item* sampai dengan terbesar berdasarkan hasil *priority* sebelumnya. Selanjutnya adalah mencari lintasan awal dan pola akhir untuk setiap *item.* Pembangkitan *conditional pattern base* didapatkan melalui *FP-Tree* yang telah dibangun sebelumnya. Jumlah *item* yang didapatkan *conditional pattern base*-Nya adalah sebanyak 27 *item.* Berikut ini adalah hasil perhitungan *conditional pattern base* dapat dilihat pada table 3.6.

   Tabel 3. 6 Hasil perhitungan *Condiitonal Pattern Base*

Items	***Conditional Pattren Base***
TULM1	• BPK1, NB2:1 • TA1, GP1, GM1:1
TE1	• GP1, MS2, AA1, PF1:1 • RT1:1 • ET1:1
SOY1	• FF1:1 • GDC3, AA1, KLWC1:1 • BB1:1
RT1	• BPK1, SPAB1, BB1, IRAB1, PK1, PO1:1 • MS2, MS1:1
PO1	• BPK1, EJ1, KBA1:1 • BPK1, SPAB1, BB1, IRAB1, PK1:1 • GDC3, FYK1:1

Items	*Conditional Pattren Base*
PK1	• BPK1, SPAB1, BB1, IRAB1:1 • BPK1, FYK1:1 • ASK1:1
PF1	• ASK1, KKA1, ET1 • GP1, MS2, AA1:1 • KKA1, FYK1:1
NB2	• BPK1:1 • SIAB1, BB1, KB1:1 • ASK1, SIAB1, SPAB1, KB1:1
KLWC1	• ASK1, KKA1:1 • KKA1:1 • GDC3, AA1:1
KB1	• SIAB1, BB1:1 • TA1, EJ1, KBA1, BR1:1 • ASK1, SIAB1, SPAB1:1
KAFM31	• TA1, GP1, BR1:1 • TA1, BR1:1 • ASK1, KKA1, SPAB1:1
IRAB1	• BPK1, SPAB1, BB1:1 • , SPAB1:1 • FF1, SIAB1, AA1:1
GM1	• TA1, GP1:1 • GP1, KBA1, MS2, EJ1:1 • MS1:1
ET1	• ASK1, KKA1:1
BR1	• TA1, EJ1, KBA1:1 • TA1, GP1:1 • TA1:1

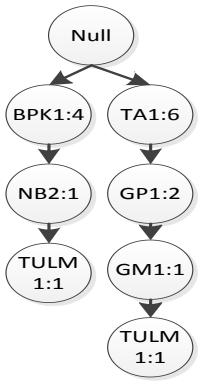
Items	*Conditional Pattren Base*
BB1	• BPK1, SPAB1:1 • SIAB1:1
AA1	• FF1, SIAB1:1 • GP1, MS2:1 • GDC3:1
SPAB1	• BPK1:1 • FF1:1 • ASK1, SIAB1:1 • ASK1, KKA1:1
SIAB1	• FF1:1 • TA1:1 • ASK1:1
MS1	• KBA1, MS2:1 • ASK1, MS2:1 • MS2:1
KBA1	• BPK1, EJ1:1 • TA1, EJ1:1 • ASK1, KKA1, EJ1, GDC3:1 • GP1:1
GDC3	• ASK1, KKA1, EJ1:1
FF1	• KKA1:1
EJ1	• BPK1:1 • TA1:1 • ASK1, KKA1:1 • GP1, KBA1, MS2:1
MS2	• TA1:1 • ASK1:1 • GP1, KBA1:1

Items	*Conditional Pattren Base*
	• GP1:1
GP1	• TA1:2
KKA1	• ASK1:4

2. Tahap Pembangkitan *Conditional FP-Tree*

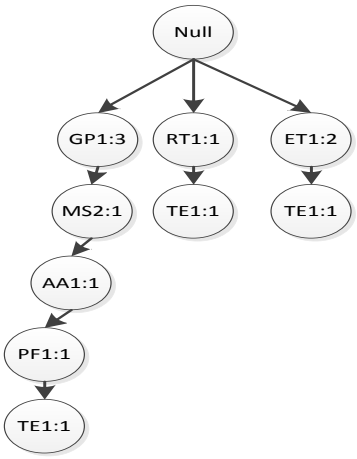
   Pada tahap ini setiap *item* yang ada pada table 3.6 diatas akan dibuat *conditional fp-tree,* dan apabila ada *item* yang *support count*-nya memenuhi *minimum support* maka akan dibangkitkan dengan *conditional FP-Tree.*

   1. Tahap pembentukan *conditional fp-tree* untuk *item* TULM1 dengan *conditional pattern base* { BPK1, NB2:1 }, { TA1, GP1, GM1:1} yaitu *null* karena tidak memenuhi *minimum support count* ≥ 3 dan tidak termasuk kedalam *item* yang *frequent*.



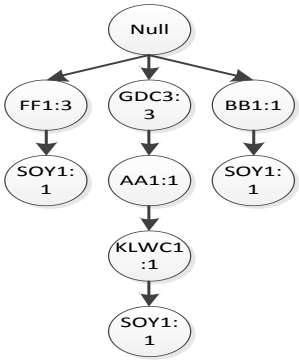
Gambar 3. 6 Lintasan Mengandung Simpul TULM1

   2. *Conditional Fp-tree* untuk *item* TE1 dengan *conditional pattern base* { GP1, MS2, AA1, PF1:1} { RT1:1} {ET1:1} yaitu *null* karena walaupun *support count*-nya memenuhi nilai *minimum support*, tetapi pada proses pembangkitan tidak didapatkan *support count* yang memenuhi nilai *minimum support*.



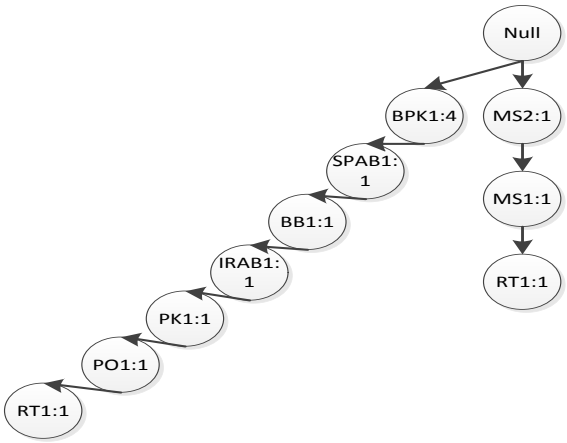
Gambar 3. 7 Lintasan Mengandung SImpul TE1

3. *Conditional Fp-tree* untuk *item* SOY1 dengan *Conditional Pattern Base* { FF1:1} {GDC3, AA1, KLWC1:1} {BB1:1} yaitu *null* karena walaupun *support count*-nya memenuhi nilai *minimum support*, tetapi pada proses pembangkitan tidak didapatkan *support count* yang memenuhi nilai *minimum support*.



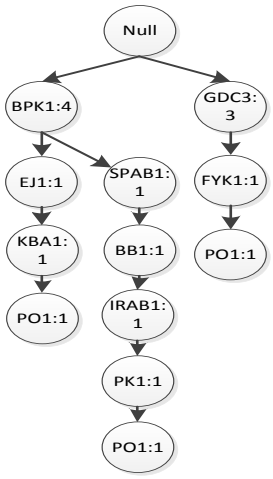
Gambar 3. 8 Lintasan Mengandung Simpul SOY1

4. *Conditional Fp-tree* untuk *item* RT1 dengan *Conditional Pattern Base* {BPK1, SPAB1, BB1, IRAB1, PKI1, PO1:1} {MS2, MS1:1} yaitu *null* karena tidak memenuhi *support count* ≥ 3 dan tidak termasuk kedalam *item* yang *frequent*.



Gambar 3. 9 Lintasan Mengandung Simpul RT1

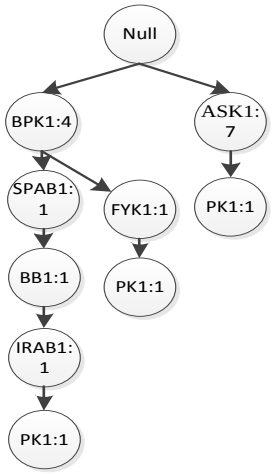
5. *Conditional Fp-tree* untuk *item* PO1 dengan *Conditional Pattern Base* {BPK1, EJ1, KBA1:1} {BPK1, SPAB1, BB1, IRAB1, PK1:1} {GDC3, FYK1:1} yaitu *null* karena walaupun *support count*-nya memenuhi nilai *minimum support*, tetapi pada proses pembangkitan tidak didapatkan *support count* yang memenuhi nilai *minimum support*.



Gambar 3. 10 Lintasan Mengandung SImpul PO1

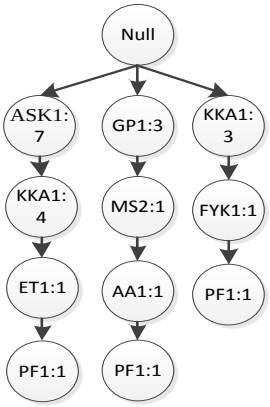
6. *Conditional Fp-tree* untuk *item* PK1 dengan *Conditional Pattern Base* {BPK1, SPAB1, BB1, IRAB1:1} {BPK1, FYK1:1} {ASK1:1} yaitu *null* karena walaupun *support count*-nya memenuhi nilai *minimum*

support, tetapi pada proses pembangkitan tidak didapatkan *support count* yang memenuhi nilai *minimum support.*



Gambar 3. 11 Lintasan Mengandung Simpul PK1

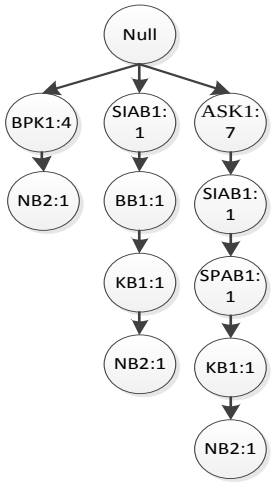
7. *Conditional Fp-tree* untuk *item* PF1 demngan *Conditional Pattern Base* { ASK1, KKA1, ET1:1} {GP1, MS2, AA1:1} {KKA1, FYK1:1} yaitu *null* karena walaupun *support count*-nya memenuhi nilai *minimum support,* tetapi pada proses pembangkitan tidak didapatkan *support count* yang memenuhi nilai *minimum support.*



Gambar 3. 12 Lintasan Mengandung Simpul PF1

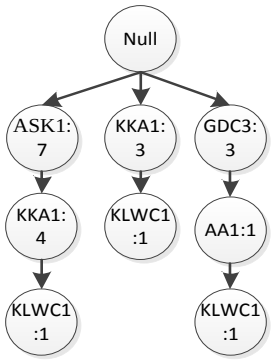
8. *Conditional Fp-tree* untuk *item* NB2 dengan *Conditional Pattern Base* {BPK1:1} {SIAB1, BB1, KB1:1} {ASK1, SIAB1, SPAB1, KB1:1} yaitu *null* karena walaupun *support count*-nya memenuhi nilai

minimum support, tetapi pada proses pembangkitan tidak didapatkan *support count* yang memenuhi nilai *minimum support*.



Gambar 3. 13 Lintasan Mengandung Simpul NB2

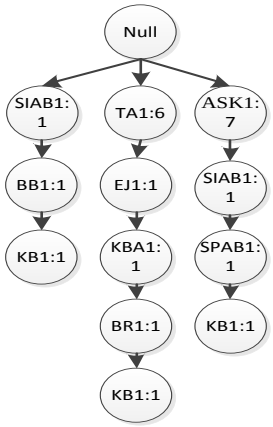
9. *Conditional Fp-tree* untuk *item* KLWC1 dengan *Conditional Pattern Base* {ask1, kka1:1} {KKA1:1} {GDC3, AA1:1} yaitu *null* karena walaupun *support count*-nya memenuhi nilai *minimum support*, tetapi pada proses pembangkitan tidak didapatkan *support count* yang memenuhi nilai *minimum support*.



Gambar 3. 14 Lintasan Mengandung Simpul KLWC1

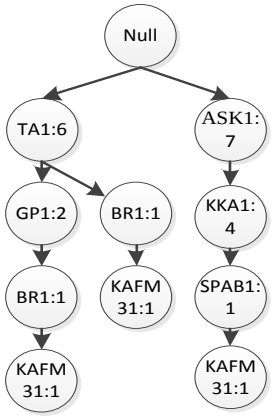
10. *Conditional Fp-tree* untuk *item* KB1 dengan *Conditional Pattern Base* {SIAB1, BB1:1} {TA1, EJ1, KBA1, BR1:1} {ASK1, SIAB1, SPAB1:1} yaitu *null* karena walaupun *support count*-nya memenuhi

nilai *minimum support*, tetapi pada proses pembangkitan tidak didapatkan *support count* yang memenuhi nilai *minimum support*.



Gambar 3. 15 Lintasan Mengandung Simpul KB1

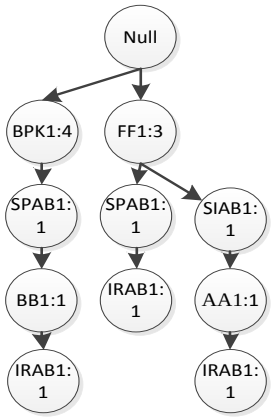
11. *Conditional Fp-tree* untuk *item* KAFM31 dengan *Conditional Pattern Base* {TA1, GP1, BR1:1} {TA1, BR1:1} {ASK1, KKA1, SPAB1:1} yaitu *null* karena walaupun *support count*-nya memenuhi nilai *minimum support*, tetapi pada proses pembangkitan tidak didapatkan *support count* yang memenuhi nilai *minimum support.*



Gambar 3. 16 Lintasan Mengandung Simpul KAFM31

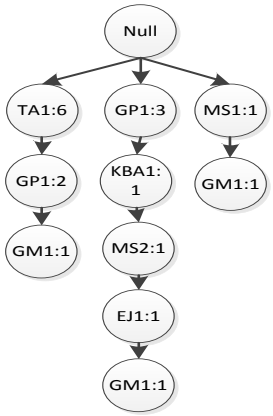
12. *Conditional Fp-tree* untuk *item* IRAB1 dengan *Conditional Pattern Base* {BPK1, SPAB1, BB1:1} {SPAB1:1} {FF1, SIAB1, AA1:1} yaitu *null* karena walaupun *support count*-nya memenuhi nilai

minimum support, tetapi pada proses pembangkitan tidak didapatkan *support count* yang memenuhi nilai *minimum support*.



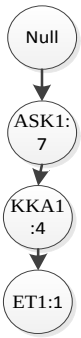
Gambar 3. 17 Lintasan Mengandung Simpul IRAB1

13. *Conditional Fp-tree* untuk *item* GM1 dengan *Conditional Pattern Base* {TA1, GP1:1} {GP1, KBA1, MS2, EJ1:1} {MS1:1} yaitu *null* karena walaupun *support count*-nya memenuhi nilai *minimum support*, tetapi pada proses pembangkitan tidak didapatkan *support count* yang memenuhi nilai *minimum support*.



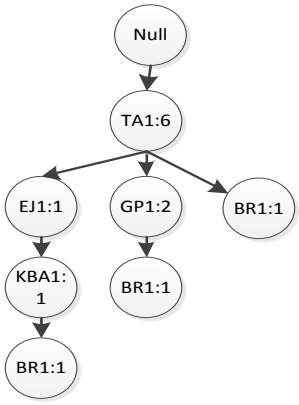
Gambar 3. 18 Lintasan Mengandung Simpul GM1

14. *Conditional Fp-tree* untuk *item* ET1 dengan *Conditional Pattern Base* {ASK1, KKA1:1} yaitu *null* karena tidak memenuhi *minimum support count* ≥ 3 dan tidak termasuk kedalam *item* yang *frequent*.



Gambar 3. 19 Lintasan Mengandung Simpul ET1

15. *Conditional Fp-tree* untuk *item* BR1 dengan *Conditional Pattern Base* {TA1, EJ1, KBA1:1} {TA1, GP1:1} {TA1:1} yang setelah dijumlahkan berdasarkan *support count* = 3 telah memenuhi nilai *minimum support* ≥ 3.



Gambar 3. 20 Lintasan Mengandung Simpul BR1

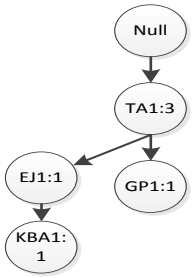
Beriku adalah tahapan-tahapannya:

1. Jumlahkan *support count* yang memenuhi *minimum support* yang akan dibangkitkan dengan *conditional FP-Tree*.

   Dalam *conditional FP-Tree* untuk *item* BR1, terdapat 3 lintasan.

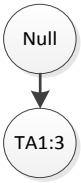
   Pada lintasan yang berakhiran BR1 akan dihilangkan dan *support count* yang melewati lintasan pertama yang berakhiran BR1, akan disamakan dengan *support count* BR1.

Proses selanjutnya adalah apabila dalama satu lintasan melintasi dua lintasan atau lebih, maka *support count*-nya akan selalu bertambah 1.



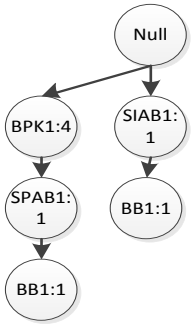
Gambar 3. 21 *Conditional FP-Tree BR1* Setelah Dibangkitkan

2. Setelah proses pembangkitan *FP-Tree*, maka proses terakhir adalah mencari *frequent pattern* yang memenuhi nilai *minimum support*. TA1:3 adalah satu-satunya *item* yang didapatkan yang memenuhi nilai *minimum support.* Dan *frequent pattern* yang didapatkan adalah {BR1, TA1:3}



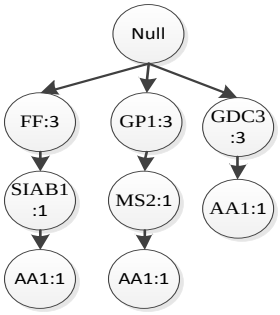
Gambar 3. 22 *Conditional FP-Tree* BR1, TA1 :3Hasil *Frequent Pattern*

16. *Conditional Fp-tree* untuk *item* BB1 dengan *Conditional Pattern Base* {BPK1, SPAB1:1} {SIAB1:1} yaitu *null* karena tidak memenuhi *minimum support count* ≥ 3 dan tidak termasuk kedalam *item* yang *frequent.*



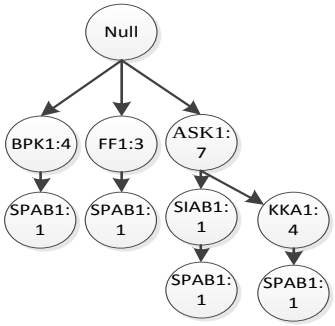
Gambar 3. 23 Lintasan Mengandung Simpul BB1

17. *Conditional Fp-tree* untuk *item* AA1 dengan *Conditional Pattern Base* {FF1, SIAB1:1} {GP1, MS2:1}{GDC3:1} yaitu *null* karena walaupun *support count*-nya memenuhi nilai *minimum support*, tetapi pada proses pembangkitan tidak didapatkan *support count* yang memenuhi nilai *minimum support*.



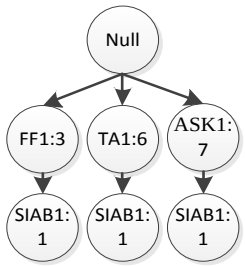
Gambar 3. 24 Lintasan Mengandung Simpul AA1

18. *Conditional Fp-tree* untuk *item* SPAB1 dengan *Conditional Pattern* Base {BPK1:1} {FF1:1} {ASK1, SIAB1:1} {ASK1, KKA1:1} yaitu *null* karena walaupun *support count*-nya memenuhi nilai *minimum support*, tetapi pada proses pembangkitan tidak didapatkan *support count* yang memenuhi nilai *minimum support*.



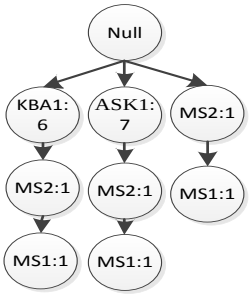
Gambar 3. 25 Lintasan Mengandung Simpul SPAB1

19. *Conditional Fp-tree* untuk *item* SIAB1 dengan *Conditional Pattern Base* {FF1:1} {TA1:1} {ASK1:1} yaitu *null* karena walaupun *support count*-nya memenuhi nilai *minimum support*, tetapi pada proses pembangkitan tidak didapatkan *support count* yang memenuhi nilai *minimum support.*



Gambar 3. 26 Lintasan Mengandung Simpul SIAB1

20. *Conditional Fp-tree* untuk *item* MS1 dengan *Conditional Pattern Base* {KBA1, MS2:1} {ASK1, MS2:1} {MS2:1} yang setelah dijumlahkan berdasarkan *support count* = 3 telah memenuhi nilai *minimum support* ≥ 3.



Gambar 3. 27 Lintasan Mengandung Simpul MS1

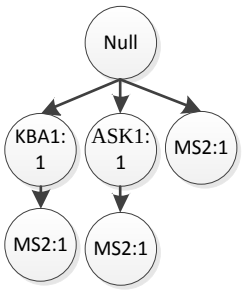
Beriku adalah tahapan-tahapannya:

1. Jumlahkan *support count* yang memenuhi *minimum support* yang akan dibangkitkan dengan *conditional FP-Tree.*

   Dalam *conditional FP-Tree* untuk *item* MS1, terdapat 3 lintasan.

   Pada lintasan yang berakhiran MS1 akan dihilangkan dan *support count* yang melewati lintasan pertama yang berakhiran MS1, akan disamakan dengan *support count* MS1.

   Proses selanjutnya adalah apabila dalama satu lintasan melintasi dua lintasan atau lebih, maka *support count*-nya akan selalu bertambah 1.



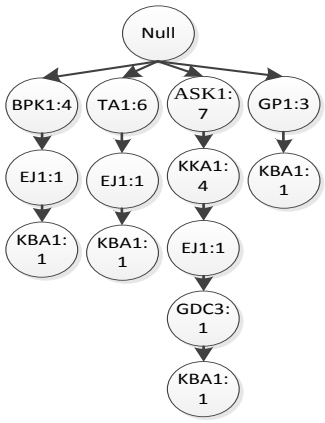
Gambar 3. 28 *Conditional FP-Tree* MS1 Setelah Dibangkitkan

2. Setelah proses pembangkitan *FP-Tree,* maka proses terakhir adalah mencari *frequent pattern* yang memenuhi nilai *minimum support.* MS2:3 adalah satu-satunya *item* yang didapatkan yang memenuhi nilai *minimum support.* Dan *frequent pattern* yang didapatkan adalah {MS1, MS2:3}



Gambar 3. 29 *Conditional FP-Tree* MS1, MS2:3 Hasil *Frequent Pattern*

21. *Conditional Fp-tree* untuk *item* KBA1 dengan *Conditional Pattern Base* {BPK1, EJ1:1} {TA1, EJ1:1} {ASK1, KKA1, EJ1, GDC3:1} {GP1:1} yang setelah dijumlahkan berdasarkan *support count* = 4 telah memenuhi nilai *minimum support* ≥ 3.



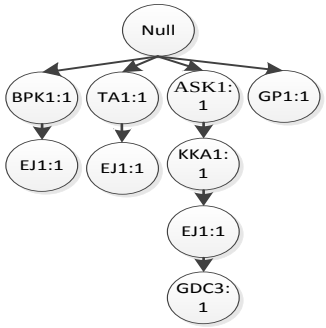
Gambar 3. 30 Lintasan Mengandung Simpul KBA1

Beriku adalah tahapan-tahapannya:

1. Jumlahkan *support count* yang memenuhi *minimum support* yang akan dibangkitkan dengan *conditional FP-Tree.*

   Dalam *conditional FP-Tree* untuk *item* KBA1, terdapat 4 lintasan. Pada lintasan yang berakhiran KBA1 akan dihilangkan dan *support count* yang melewati lintasan pertama yang berakhiran KBA1, akan disamakan dengan *support count* KBA1.

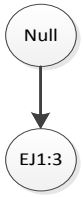
   Proses selanjutnya adalah apabila dalama satu lintasan melintasi dua lintasan atau lebih, maka *support count*-nya akan selalu bertambah 1.



Gambar 3. 31 *Conditional FP-Tee* KBA1 Setelah Dibangkitkan

2. Setelah proses pembangkitan *FP-Tree*, maka proses terakhir adalah mencari *frequent pattern* yang memenuhi nilai *minimum support*.

   EJ1:3 adalah satu-satunya *item* yang didapatkan yang memenuhi nilai *minimum support*. Dan *frequent pattern* yang didapatkan adalah {KBA1, EJ1:3}



Gambar 3. 32 *Conditional FP-Tree* KBA1, EJ1 Hasil *Frequent Pattern*

22. *Conditional Fp-tree* untuk *item* GDC3 dengan *Conditional Pattern Base* {ASK1, KKA1, EJI:1} yaitu *null* karena tidak memenuhi

minimum support count ≥ 3 dan tidak termasuk kedalam *item* yang *frequent.*



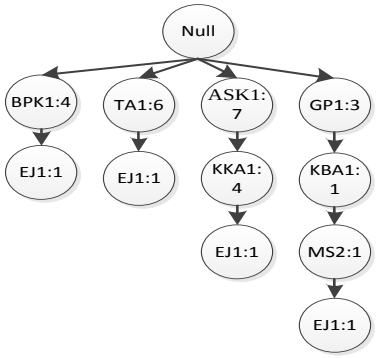
Gambar 3. 33 Lintasan Mengandung Simpul GDC3

23. *Conditional Fp-tree* untuk *item* FF1 dengan *Conditional Pattern Base* {KKA1:1} yaitu *null* karena tidak memenuhi minimum *support count* ≥ 3 dan tidak termasuk kedalam *item* yang *frequent.*



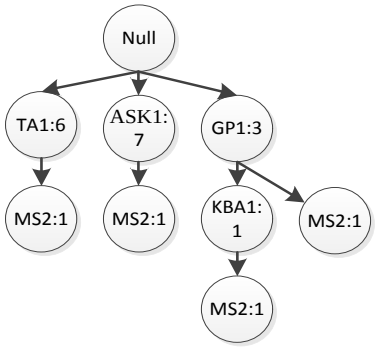
Gambar 3. 34 Lintasan Mengandung Simpul FF1

24. *Conditional Fp-tree* untuk *item* EJ1 dengan *Conditional Pattern Base* {BPK1:1} {TA1:1} {ASK1, KKA1:1} {GP1, KBA1, MS2:1} yaitu *null* karena walaupun *support count*-nya memenuhi nilai *minimum support*, tetapi pada proses pembangkitan tidak didapatkan *support count* yang memenuhi nilai *minimum support.*



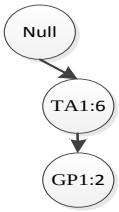
Gambar 3. 35 Lintasan Mengandung Simpul EJ1

25. *Conditional Fp-tree* untuk *item* MS2 dengan *Conditional Pattern Base* {TA1:1} {ASK1:1} {GP1, KBA1:1} {GP1:1} yaitu *null* karena walaupun *support count*-nya memenuhi nilai *minimum support*, tetapi pada proses pembangkitan tidak didapatkan *support count* yang memenuhi nilai *minimum support.*



Gambar 3. 36 Lintasan Mengandung MS2

26. *Conditional Fp-tree* untuk *item* GP1 dengan *Conditional Pattern Base* {TA1:2} yaitu *null* karena tidak memenuhi *minimum support coun t* ≥ 3 dan tidak termasuk kedalam *item* yang *frequent.*



Gambar 3. 37 Lintasan Mengandung GP1

27. *Conditional Fp-tree* untuk *item* KKA1 dengan *Conditional Pattern Base* {ASK1:4} yang setelah dijumlahkan berdasarkan *support count* = 4 telah memenuhi nilai *minimum support* ≥ 3.



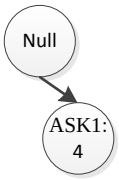
Gambar 3. 38 Lintasan Mengandung KKA1

Beriku adalah tahapan-tahapannya:

1. Jumlahkan *support count* yang memenuhi *minimum support* yang akan dibangkitkan dengan *conditional FP-Tree.*

   Dalam *conditional FP-Tree* untuk *item* KKA1, terdapat 3 lintasan. Pada lintasan yang berakhiran KKA1 akan dihilangkan dan *support count* yang melewati lintasan pertama yang berakhiran KKA1, akan disamakan dengan *support count* KKA1.

   Proses selanjutnya adalah apabila dalama satu lintasan melintasi dua lintasan atau lebih, maka *support count*-nya akan selalu bertambah 1.



Gambar 3. 39 *Conditional FP-Tree* KKA1 Setelah Dibangkitkan

2. Setelah proses pembangkitan *FP-Tree,* maka proses terakhir adalah mencari *frequent pattern* yang memenuhi nilai *minimum support*.

   ASK1:4 adalah satu-satunya *item* yang didapatkan yang memenuhi nilai *minimum support*. Dan *frequent pattern* yang didapatkan adalah {KKA1, ASK1:3}



Gambar 3. 40 *Conditional FP-Tree* KKA1, ASK1:4 Hasil *Frequent Pattern*

3. Tahap Pencarian *Frequent Pattern*

   Setelah mendapatkan hasil dari *conditional FP-Tree* dari tabel *conditional pattern base,* ditemukan beberapa *frequent pattern* yang didapatkan.

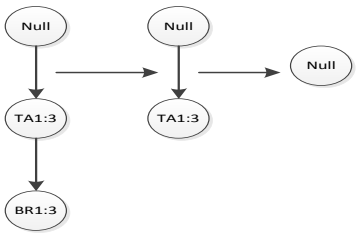
Jumlah *item* yang didapatkan pada tahap pembangkitan *fp-tree* yaitu sebanyak 4 *item*. Hasil perhitungan dari *Conditional FP Tree* terdapat pada tabel 3.7.

Tabel 3. 7 Hasil Perhitungan *Conditional FP Tree*

Item	**Hasil *Conditional FP Tree***
BR1	{TA1:3}
MS1	{MS2:3}
KBA1	{EJ1:3}
KKA1	{ASK1:4}

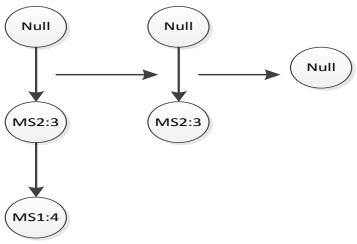
Selanjutnya membuat *conditional FP-Tree* untuk setiap *pattern base*, berikut langkah-langkah yang akan dilakukan:

a. Akumulasikan stiap *item* untuk stiap *frequent pattern* yang terdapat pada table diatas.
b. Buat *conditional fp-tree* dari setiap *frequent pattern* yang didapatkan.
   1. Untuk *item* BR1, *frequent pattern* yang didapatkan adalah {TA1:3}



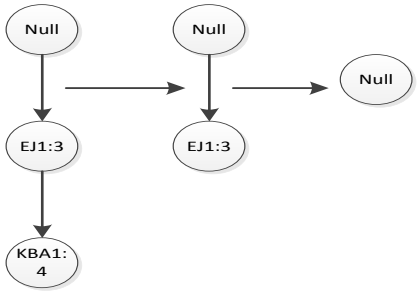
Gambar 3. 41 *Conditional FP-Tree* untuk *item* BR1

   2. Untuk *item* MS1, *frequent pattern* yang didapatkan adalah {MS2:3}



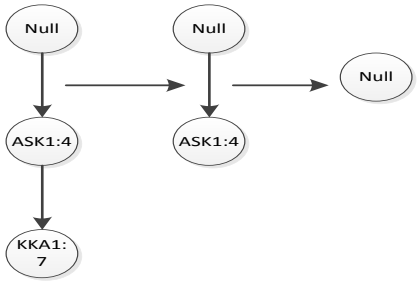
Gambar 3. 42 *Conditional FP-Tree* untuk *item* MS1

3. Untuk *item* KBA1, *frequent pattern* yang didapatkan adalah {EJ1:3}



Gambar 3. 43 *Conditional FP-Tree* untuk *item* KBA1

4. Untuk *item* KKA1, *frequent pattern* yang didapatkan adalah {ASK1:4}



Gambar 3. 44 *Conditional FP-Tree* untuk *item* KKA1

c. Proses selanjutnya setelah mendapatkan *frequent pattern* dari setiap *conditional fp-tree* adalah menentukan *suffix* dari setiap *frequent pattern*, seperti pada table 3.8 di bawah ini. *Frequent pattern* yang didapatkan yaitu sebanyak 4 item.

Tabel 3. 8 Hasil *Frequent Pattern*

Suffix	*Frequent pattern*
BR1	BR1-TA1
MS1	MS1-MS2
KBA1	KBA1-EJ1
KKA1	KKA1-ASK1

2. Perhitungan Asosiasi

Dari hasil *frequent pattern* yang didapatkan pada table 3.8 di atas akan di *generate* untuk mendapatkan *rule* dengan *minimum confidence* 50% dengan masing-masing *pattern* dikombinasikan dengan *pattern* lain. Dengan menggunakan rumus yang ada pada bagian bab 2 untuk memperoleh nilai *confidence*. *Rules* yang didapatkan sebanyak 8 *rules*. Hasil perhitungan *minimum support* dan *minimum confidence* terdapat pada tabel 3.9.

Tabel 3. 9 Hasil Perhitungan *Confidence* dan *Generate Rule*

Rule	***Minimum Confidence***
BR1 → TA1	(3/3) *100 =100%
TA1 → BR1	(3/6) * 100 =50%
MS1 → MS2	(3/4) *100 =75%
MS2 → MS1	(3/5) * 100 = 60%
KBA1→EJ1	(3/4) *100 = 75%
EJ1→KBA1	(3/4) *100 = 75%
KKA1→ASK1	(4/7)*100 =57,14%
ASK1→KKA1	(4/7)*100 =57,14%

Rule yang telah memenuhi nilai *minimum confidence* kemudian disesuaikan dengan tabel barang. Sehingga kode barang yang terdapat pada *rule* sesuai dengan nama brangnya dan dapat dijadikan informasi. Tabel barang dapat dilihat pada Tabel D-6 di Lapmiran D.

Informasi yang dihasilkan berupa kombinasi *item* dan nama barang berdasarkan kode barang yang sesuai pada tabel D-6 di Lampiran. Hasilnya dapat dilihat pada tabel 3.10.

Tabel 3. 10 Paket Barang dalam Bentuk Awal

Rule	**Paket Barang**
BR1 → TA1	BERAS - TELUR AYAM
MS1 → MS2	MASAKO SAPI - MINYAK SAYUR

Rule	Paket Barang
KBA1→EJ1	KUKU BIMA ANGUR - EXTRA JOS
KKA1→ASK1	KOPI KAPAL API MIX - ABC SUSU KOPI & GULA

Dari hasil pengolahan data ini kita dapat mengambil sebuah keputusan mana barang yang cocok untuk dijadikan sebuah paket penjualan barang. Sebagai contoh, bila kita ingin membuat sebuah paket penjualan barang yang berisikan dua buah barang, maka kita bisa menggunakan data telur dan beras dengan nilai *confidence* 100% sebagi barang-barang yang akan dijadikan paket penjualan. Namun jika ingin membuat paket penjualan barang dengan jumlah barang lebih dari dua maka data penjualan yang digunakan harus ditambahkan dan nilai *support* harus diturunkan agar *rules* yang didapatkan menghasilkan lebih dari dua buah kombinasi barang sehingga paket dapat dibentuk dengan jumlah item lebih banyak.

3.1.3 Analisis Spesifikasi Kebutuhan Perangkat Lunak

Spesifikasi kebutuhan perangkat lunak adalah kebutuhan kebutuhan apa saja yang diperlukan untuk membangun sebuah perangkat lunak sesuai dengan kebutuhan penggunanya. Spesifikasi kebutuhan perangkat lunak dibagi menjadi dua yaitu kebuthan *non-fungsional* dan kebutuhan *fungsional*.

3.2.1 Analisis Kebutuhan *Non-Fungsional*

Analisis kebutuhan *non fungsional* dilakukan untuk mengetahui spesifikasi kebutuhan sistem. Spesifikasi kebutuhan melibatkan analisa perangkat keras (*hardware*), analisa perangkat lunak (*software*), analisa pengguna (*user*).

3.2.1.1 Analisis Kebutuhan Perangkat Keras

Perangkat keras pendukung dalam pembangunan perangkat lunak ini adalah sebagai berikut:

1. Analisa spesifikasi kebutuhan perangkat keras pada sistem yang sedang berjalan yang digunakan oleh direktur CV. Sultan Jaya Mandiri adalah:
   a. *Processor* :Intel Core Duo
   b. *Hardisk* :160 GB

c. *Memory* :2 GB

2. Analisa spesifikasi *minimum* kebutuhan perangkat keras pada sistem yang akan dibangun membutuhkan:
   a. *Processor* berkecepatan minimal 2 Ghz
   b. *Harddisk* 1 GB untuk menyimpan data
   c. *Memory* 512 MB
3. Evaluasi kebutuhan perangkat keras

   Dari data spesifikasi perangkat keras yang digunakan oleh direktur CV. Sultan Jaya Mandiri seperti diatas, perangkat keras yang telah dimiliki sudah cukup untuk dapat menjalankan aplikasi pengolahan data penjualan yang akan dibuat.

3.2.1.2 Analisis Kebutuhan Perangkat Lunak

Kebutuhan perangkat lunak pendukung merupakan hal penting bagi pembangunan perangkat lunak yang sedang dirancang. Perangkat lunak pendukung dalam pembangunan perangkat lunak ini adalah sebagai berikut:

1. Analisis kebutuhan perangkat lunak pada sistem yang sedang berjalan yang digunakan oleh direktur CV. Sultan Jaya Mandiri adalah:
   a. Sistem operasi Windows 7
   b. Microsoft Office
2. Analisis spesifikasi kebutuhan perangkat lunak untuk menjalankan sistem yang akan dibangun adalah :
   a. Sistem operasi Windows 7
   b. MySQL sebagai *database server*
   c. Visual studio 2013
3. Evaluasi kebutuhan perangkat lunak

   Berdasarkan analisis spesifikasi perangkat lunak yang digunakan oleh direktur CV.Sultan Jaya Mandiri, spesifikasi perangkat lunak yang akan digunakan kurang memenuhi spesifikasi perangkat lunak yang dibutuhkan. Kekurangan tersebut dapat diatasi dengan meng-*install* MySQL dan Visual Studio pada computer yang digunakan.

3.2.1.3 Alnalisis pengguna

Analisis pengguna dibuat untuk memenuhi siapa saja dan seperti apa karakteristik yang dimiliki oleh orang yang dapat menggunakan aplikasi yang akan dibuat. Untuk lebih jelas mengenai karakteristik pengguna, dapat dilihat pada tabel 3.11.

Tabel 3. 11 Karakteristik Pengguna

Pengguna	Tanggung Jawab	Hak Akses	Tingkat Pendidikan	Tingkat Keterampilan	Pengalaman
Direktur sebagai pemilik perusahaan	Mengoperasikan aplikasi sesuai dengan tugas yang diberikan	Melakukan kegiatan yang berhubungan dengan isi dari tampilan yang ada pada aplikasi.	Minimal Sekolah Menengah Atas (SMA)	Memahami konten yang tersedia pada aplikasi	Minimal mampu mengoperasikan komputer

Dengan kondisi yang terdapat pada tabel diatas, dapat diambil kesimpulan bahwa aplikasi yang dibuat dapat dijalankan oleh *staff* di perusahaan tersebut. Pegawai *Staff* dalam hal ini adalah orang yang bekerja di bagian produksi yang akan menggunakan aplikasi yang akan dibangun.

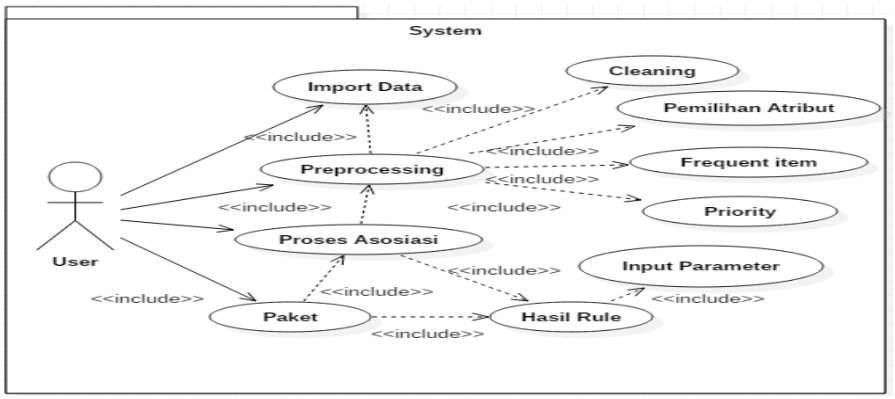
3.2.2 Analisis Kebutuhan Fungsional

Analisis kebutuhan fungsional bertujuan untuk perancangan terhadap aplikasi akan dibangun. Apikasi yang akan dibangun menggunakan pendekatan berorientasi objek dengan menggunakan pemodelan UML. Pemodelan yang akan digunakan untuk memodelkan terdiri dari diagram *use case*, *sqence diagram, diagram aktivitas,* dan *class diagram*.

3.2.2.1 Diagram *Use Case*

Use case atau diagram *use case* merupakan pemodelan untuk kelakuan (*behavior)* sistem yang akan dibuat. Diagram *use case* yang terdapat pada sistem yang akan dibangun terdiri dari satu *user* dan sembilan *use case*. Diagram *use*

case dapat dilihat pada gambar 3.45.Penjelasan *user* dan *use case* dapat dilihat pada Tabel 3.12 dan Tabel 3.13.



Gambar 3. 45 Diagram *Use Case*

Tabel 3. 12 Definisi *User*

User	Deskripsi
User	Bertugas untuk melakukan *import* data, menginputkan nilai *minimum support* dan *minimum confidence*.

Tabel 3. 13 Deskripsi *Use Case*

No	*Use Case*	Deskripsi
1	*Import* Data	Sistem mengharuskan *user* yang belum melakukan *import* data untuk melakukan *import* data terlebih dahulu lalu menyimpan data tersebut kedalam *database*.
2	*Prepocessing*	Data yang sudah disimpan dalam *database* kemudian sistem melakukan proses *cleaning* dan *pemilihan atribut*.
3	*Cleaning*	Sistem melakukan pembersihan data pada data yang memiliki kategori *Non Food* jika user sudah melakukan *import* data.
4	Pemilihan Atribut	Sistem melakukan pemilihan atribut No Transaksi dan Kode barang jika *user* sudah melakukan *import* data dan *cleaning*.

No	*Use Case*	Deskripsi
5	*Frequent Item*	Sistem melakukan perhitungan *Frequent Item* jika *user* sudah melakukan *import* data, *cleaning* dan *pemilihan atribut*.
6	*Priority*	Sistem akan melakukan *Priority* berdasarkan *friority* jika *user* sudah melakukan *import* data, *cleaning*, *pemilihan atribut*, dan *Frequent Item*.
7	Proses Assosiasi	Sistem melakukan proses *frequent pattern*, *rule* dan melihat hasil asosiasi. Sedangkan *user* menginputkan nilai *minimum support* dan *minimum confidence* terlebih dahulu.
8	*Input* Parameter	*User* menginputkan nilai *minimum support* dan *minimum confidence* sebagai parameter .
9	Hasil *Rule*	Sistem menampilkan informasi dari hasil seleksi *rule.* Informasi yang ditampilkan berupa kombinasi nama barang dank ode barang.
10	Paket	Sistem menampilkan informasi rekomendasi paket. Informasi yang ditampilkan berupa rekomendasi paket barang.

Skenario *Use Case*

1. **Skenario *Use Case Import* Data**

Skenario *use case import* data menggambarkan langkah – langkah aksi *user* terhadap sistem untuk melakukan *import* data transaksi penjualan yang akan disimpan ke dalam *database.* Skenario *use case import data* dapat dilihat pada Tabel 3.14.

Tabel 3. 14 Skenario *Use Case Import* Data

Use case Name	***Import* Data**	
Goal In Context	*Import* data transaksi penjualan dan menyimpan ke dalam *database*.	
Precondition	*User* menyiapkan data transaksi.	
Successful End Condition	Data transaksi berhasil disimpan ke dalam *database.*	
Failed End Condition	Data transaksi tidak berhasil disimpan ke dalam *database*.	
Actors	*User*	
Triger	*User* memasukan data transaksi penjualan.	
Main Flow	Step	Action
	1	*User* memasukan data transaksi penjualan.
	2	Sistem melakukan proses validasi terhadap masukan.
	3	Sistem menampilkan data transaksi jika format data yang dimasukan sesuai.
	4	Sistem menampilkan notifikasi kesalahan jika format data yang dimasukan salah.
	5	*User* melakukan *request* terhadap sistem untuk menyimpan data transaksi ke dalam *database.*
	6	Sistem menyimpan data transaksi kedalam *database.*
Extension	Step	Branching Action
	1.1	*User* memilih membatalkan proses *import.*
	1.2	Sistem menampilkan notifikasi kesalahan .

2. Skenario *Use Case Preprocesing*

Skenario *use case preprocessing* menjelaskan langkah-langkah untuk melakukan proses cleaning dan pemilihan atribut. Skenario *use case preprocesing* dapat dilihat pada Tabel 3.15.

Tabel 3. 15 Skenario *Use Case Prepocessing*

Use case Name	***Preprocesing***	
Goal In Context	Sistem melakukan proses *cleaning* dan *pemilihan atribut*, *Frequent Item* dan *Priority*.	
Precondition	Data transaksi telah disimpan didalam *database* .	
Successful End Condition	Sistem berhasil melakukan proses data *selection* dan data *cleaning*, *Frequent Item* dan *Priority*.	
Failed End Condition	Sistem tidak berhasil melakukan proses *cleaning* dan pemilihan atribut, *Frequent Item* dan *Priority*.	
Actors	*User*.	
Triger	*User* melakukan *request* proses *cleaning* dan pemilihan atribut , *Frequent Item* dan *Priority* kepada sistem.	
Included Cases	*Cleaning* dan *pemilihan atribut*, *Frequent Item* dan *Priority*.	
Main Flow	Step	*Action*
	1	*User* melakakukan *request cleaning.*
	2	Sistem melakukan proses *cleaning.*
	3	Sistem menyimpan hasil proses *cleaning.*
	4	Sistem menampilkan data hasil *cleaning.*
	5	*User* melakukan *request pemilihan atribut.*
	6	Sistem melakukan proses *pemilihan atribut.*
	7	Sistem menyimpan hasil proses *pemilihan atribut* kedalam *database.*
	8	Sistem menampilkan data *pemilihan atribut.*
	9	*User* melakukan *request Frequent Item*
	10	Sistem melakukan proses *Frequent Item*

Use case name	*Preprocessing*	
Main Flow	Step	*Action*
	11	Sistem menyimpan hasil *Frequent Item* ke dalam *database*
	12	Sistem menampilkan data hasil *Frequent Item*
	13	*User* melakukan *request Priority*
	14	Sistem melakukan proses *Priority*
	15	Sistem menyimpan hasil *Priority* ke dalam *database*
	16	Sistem menampilkan hasil *Priority*
Extension	Step	*Branching Action*
	-	-

3. Skenario *Use Case Cleaning*

Skenario *use case cleaning* menggambarkan langkah-langkah melakukan pembersihan terhadap data transaksi penjualan yang memiliki kategori *Non Food* terhadap data transaksi penjualan yang telah melewati proses *import data*. Skenario *use case cleaning* dapat dilihat pada table 3.16.

Tabel *3. 16* Skenario *Use Case Cleaning*

Use case Name	***Cleaning***
Goal In Context	Sistem melakukan pemberishan pada data yang memiliki kategori *Non Food.*
Precondition	Sudah melalu proses *import data*.
Successful End Condition	Sistem berhasil melakukan pemberishan pada data yang mempunyai kategori *Non Food.*
Failed End Condition	Sistem tidak berhasil melakukan pemberishan pada data yang mempunyai kategori *Non Food.*
Actors	*User*.
Triger	*User* melakukan *request cleaning* kepada sistem.
Included Cases	-

Use case Name	***Cleaning***	
Main Flow	Step	Action
	1	User melakukan *request cleaning* kepada sistem .
	2	Sistem melakukan *cleaning* pada data yang memiliki kategori *Non Food.*
	3	Sistem menyimpan hasil *cleaning* kedalam *database.*
	4	Sistem menampilkan hasil *cleaning.*
Extension	Step	Branching Action
	-	-

4. Skenario *Use Case* pemilihan atribut

Skenario *use case* pemilhan *atribut* menggambarkan langkah-langkah melakukan pemilihan atribut terhadap data transaksi yang telah melewati proses *cleaning*. Skenario *use case pemilihan atribut* dapat dilihat pada table 3.17.

Tabel 3. 17 Skenario *Use Case Pemilihan atribut*

Use case Name	***Pemilihan atribut***
Goal In Context	Sistem melakukan pemilihan atribut No transaksi dan kodeBarang.
Precondition	Data transaksi penjualan telah melalui proses *cleaning.*
Successful End Condition	Sistem berhasil memilih atribut No transaksi dan kodeBarang.
Failed End Condition	Sistem tidak berhasil memilih atribut No transaksi dan kodeBarang.
Actors	*User*.
Triger	*User* melakukan *request pemilihan atribut* kepada sistem.

Use case Name	***Pemilihan atribut***	
Included Cases	-	
Main Flow	Step	Action
	1	*User* melakukan *request* pemilihan *atribut* kepada sistem .
	2	Sistem melakukan proses pemilihan *atribut* no transaksi dan kode barang.
	3	Sistem menyimpan hasil pemilihan *atribut* ke dalam *database.*
	4	Sistem menampilkan data hasil pemilihan *atribut.*
Extension	Step	Branching Action
	-	-

5. Skenario *Use Case Frequent Item*

Skenario *use case Frequent Item* menggambarkan langkah-langkah melakukan perhitungan *frequency* terhadap *item* yang terdapat pada data transaksi dan telan melewati proses pemilihan atribut. Skenario *use case* data *Frequent Item* dapat dilihat pada tabel 3.18.

Tabel 3. 18 Skenario *Use Case Frequent Item*

Use case Name	***Frequent Item***
Goal In Context	Menghitung jumlah kemunculan tiap item dalam data transaksi penjualan.
Precondition	Posisi data sudah dilakukan *cleaning.*
Successful End Condition	Sistem berhasil menampilkan frekuensi tiap *item.*
Failed End Condition	Sistem tidak berhasil melakukan perhitungan.
	Frekuensi *Item*.
Actors	*User*
Triger	*User* melakukan *request Frequent Item* kepada

Use case Name	***Frequent Item***	
	sistem .	
Main Flow	Step	Action.
	1	User melakukan *request Frequent Item.*
	2	Sistem melakukan proses penghitungan frekuensi tiap *item* dan diurutkan secara *Descending* berdasarkan frekuensinya.
	3	Sistem menyimpan data hasil *Frequent Item* ke dalam *database*
	4	Sistem menampilkan hasil *Frequent Item*

6. Skenario *Use Case Priority*

Skenario *use case Priority* menggambarkan langkah-langkah melakukan pengurutan data berdasarkan *priority* setelah melewati proses *frecuency item*. Skenario *use case Priority* dapat dilihat pada tabel 3.20.

Tabel 3. 19 Skenario Use Case Priority

Use case Name	***Priority***	
Goal In Context	Data terurut berdasarkan *priority*	
Precondition	Data sudah melewati proses *Frequent Item*	
Successful End Condition	Sistem berhasil melakukan *priority*	
Failed End Condition	Sistem tidak berhasil melakukan *priority*	
Actors	*Manager*	
Triger	*User* melakukan *request Priority* kepada sistem	
Main Flow	*Step*	*Action*
	1	*User* melakukan *request Priority*
	2	Sistem melakukan proses pengurutan data beradasrkan *priority*
	3	Sistem menyimpan hasil *Priority* kedalam *database*

Use case Name	***Priority***	
Main Flow	*Step*	*Action*
	4	Sistem menampilkan hasil *Priority*

7. Skenario *Use case* Proses asosiasi

Skenario *use case* proses *assosiasi* menggambarkan langkah-langkah aksi *user* untuk melakukan proses *frequent pattern, rule*, dan melihat hasil asosiasi. Skenario *use case* proses asosiasi dapat dilihat pada table 3.20.

Tabel 3. 20 Skenario *Use Case* Proses Asosiasi

Use case name	**Proses Asosiasi**	
Goal In Context	Sistem mendapatkan *rule* lalu menampilkan *rule* tersebut menjadi sebuah informasi	
Precondition	Telah melakukan proses *preprocessing*	
Successful End Condition	Sistem berhasil mendapatkan *rule* lalu menampilkan *rule* tersebut menjadi sebuah informasi	
Failed End Condition	Sistem tidak berhasil mendapatkan *rule*	
Actors	*User*	
Triger	*User* memasukan nilai *minimum support* dan *minimum confidence* yang valid	
Included Cases	*Frequent itemset*, *minimum support*, *rule*, *minimum confidence* dan lihat hasil analisa	
Main Flow	Step	Action
	1	*User* memasukan nilai minimum *support* dan *confidence*
	2	Sistem menghitung *support* dan *confidence*
	3	Sistem melakukan *validasi* terhadap masukan yang dilakukan oleh *user*
	4	Sistem menghapus data yang memiliki

Use case name	**Proses Asosiasi**	
Main Flow	Step	Action
		nilai *support* dan *confidence* yang ≤ nilai *minimum support* dan *confidence*
	5	Sistem menampilkan data yang memiliki nilai *support* dan *confidence* ≥ nilai minimum *support* dan *confidence*
	6	Sistem menampilkan hasil seleksi *rule*
	7	Sistem pelakukan proses pembentukan paket barang
	8	Sistem menampilkan informasi bentuk paket
Extension	Step	Branching Action
	1.1	Sistem menampilkan notifikasi nilai minimum *support* yang diinputkan tidak sesuai
	2.1	Sistem menampilkan notifikasi nilai minimum *confidence* yang diinputkan
	3.1	Sistem tidak mendapatkan *rule* karena nilai minimum *confidence* yang diinputkan oleh user terlalu tinggi

8. Skenario *Use Case Imput* Parameter

Skenario *use case input* parameter menggambarkan langkah-langkah melakukan *input* nilai *minimum support* dan *minimum confidence* yang dilakukan oleh *user* setelah proses *prepocesing* dilakukan. Skenario *use case input* parameter dapat dilihat pada tabel3.21.

Tabel 3. 21 *Use Case* Skenario *Input* Parameter

Use case Name	*Input* Parameter	
Goal In Context	Sistem mendapatkan nilai *minimum support* dan *confidence* untuk proses asosiasi	
Precondition	*User* menginputkan nilai *minimum support dan confidence*	
Successful End Condition	Sistem berhasil mendapatkan nilai *support* dan *confidence* sesuai dari *input*an *user*	
Failed End Condition	Sistem tidak berhasil mendapatkan nilai *support* dan *confidence*	
Actors	*User*	
Triger	*User* menginputkan nilai *minimum support dan confidence* yang *valid*	
Main Flow	Step	Action
	1	*User* menginputkan nilai *minimum support dan confidence*
	2	Sistem melakukan *validasi* terhadap inputan parameter yang dilakukan oleh *user*
	3	Sistem meminta *user* kembali melakukan *input* nilai parameter yang sesuai
	4	Sistem menampilkan nilai parameter yang diinputkan oleh *user*
Extension	*Step*	*Branching Action*
	2.1	Sistem menampilkan *notifikasi* nilai *minimum support* dan *confidence* yang diinputkan tidak sesuai
	2.2	Sistem tidak mendapatkan nilai *minimum support dan confidence* karena nilai yang inputkan oleh *user* tidak sesuai

9. Skenario *Use case* Hasil *Rule*

Skenario *use case* hasil *rule* menggambarkan langkah-langkah aksi *user* untuk melakukan proses pembuatan *rule*. Skenario *use case rule* dapat dilihat pada table 3.22.

Tabel 3. 22 Skenario *Use Case* Hasil *Rules*

Use case Name	*Rule*	
Goal In Context	Sistem melakukan pembentukan *rule*	
Precondition	Sudah melewati proses *input* parameter	
Successful End Condition	Sistem berhasil melakukan pembentukan *rule*	
Failed End Condition	Sistem tidak berhasil melakukan pembentukan *rule*	
Actors	*User*	
Triger	*User* menginputkan nilai parameter	
Included Cases	*Minimum confidence*	
Main Flow	Step	Action
	1	*User* menginputkan nilai *minimum support* &*confidence*
	2	Sistem mendapatkan nilai *minimum support* dan *confidence*
	3	Sistem menghitung nilai *support* dan *confidence*
	4	Sistem menyeleksi data yang memenuhi dan tidak memenuhi nilai *minimum support* dan *confidence*
	5	Sistem menghapus data yang kurang dari *minimum support* dan *confidence*
	6	Sistem menampilkan data yang memenuhi nilai *minimum support* dan *confidence*

Use case Name	*Rule*	
Main Flow	Step	Action
	7	Sistem menampilkan informasi berupa *rule* yang dihasilkan
Extension	Step	Branching Action
	1.1	Sistem menampilkan pesan nilai *minimum confidence* yang diinputkan tidak sesuai
	1.2	Sistem gagal mendapatkan *rule* karena nilai *confidence* yang diinputkan oleh *user* terlalu tinggi

10. Skenario *Use Case* Paket

Skenario *use case* paket menggambarkan langkah-langkah aksi *user* untuk melihat hasil paket. Skenario *use case rules* dapat dilihat pada table 3.23.

Tabel 3. 23 Skenario *Use Case* Paket

Use case Name	Paket	
Goal In Context	Sistem melakukan pembentukan paket	
Precondition	Sudah melewati proses *rule*	
Successful End Condition	Sistem berhasil melakukan pembentukan *paket*	
Failed End Condition	Sistem tidak berhasil melakukan pembentukan paket	
Actors	*User*	
Triger	*User* menginputkan nilai parameter	
Included Cases	*minimum confidence*	
Main Flow	*Step*	*Action*
	1	*User* melakukan *request* bentuk paket yang akan digunakan sebagai paket penjualan

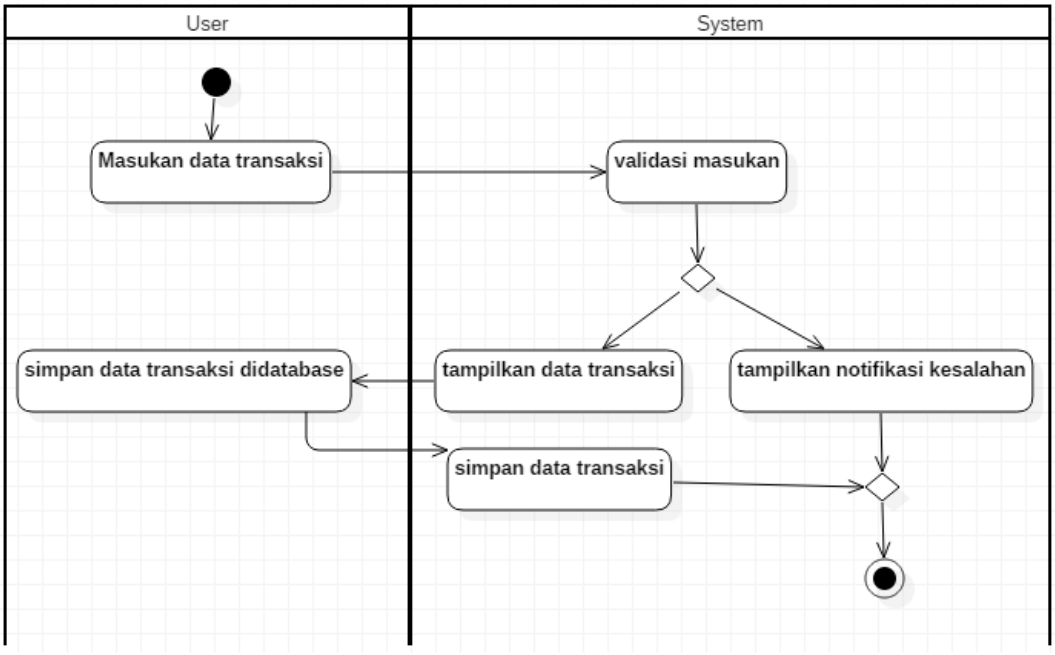
Use case Name	Paket	
Main Flow	*Step*	*Action*
	2	Sistem melakukan proses pemilihan rule untuk dijadikan paket
	3	Sistem menyimpan hasil proses pembentukan paket
	4	Sistem menampilkan hasil proses pembentukan paket penjualan barang

3.2.2.2 Diagram Aktivitas

Diagram Aktivitas menggambarkan rangkaian aliran dari aktivitas dari setiap proses. Berikut adalah diagram aktivitas dari perangkat lunak yang dibangun:

a. Diagram Aktivitas *Import* Data

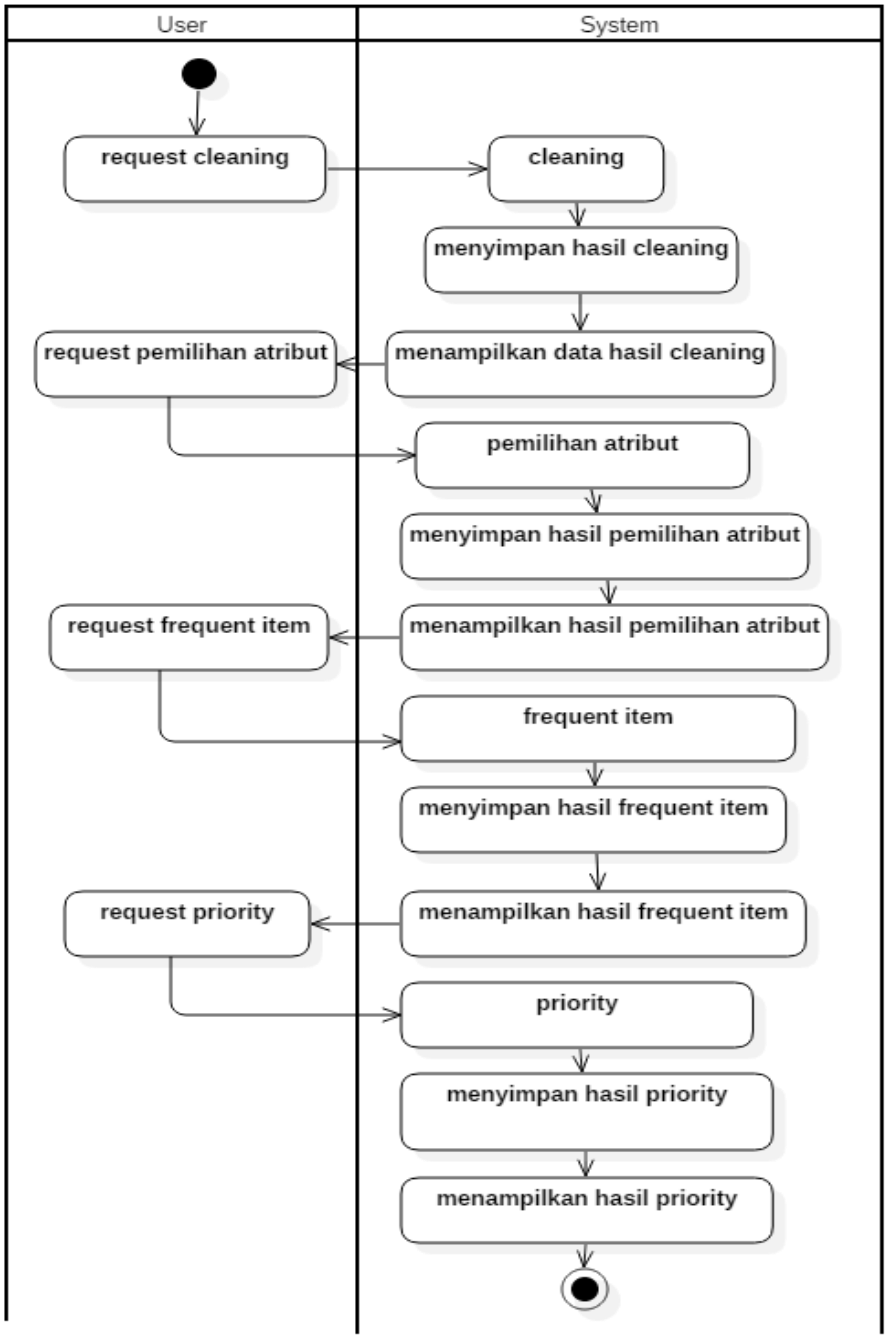
Diagram Aktivitas import data menjelaskan aktivitas *actor* dalam melakukan proses *import* data. *Diagram Aktivitas import* data dapat dilihat pada Gambar 3.46.



Gambar 3. 46 Diagram *Activity Import* data

b. Diagram Aktivitas *Preprocessing*

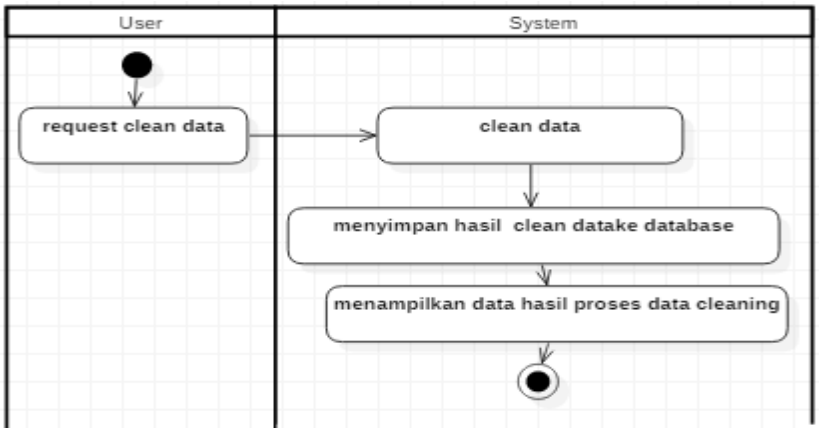
Diagram Aktivitas preprocessing menjelaskan aktivitas sistem dalam melakukan proses pengolahan data. *Diagram Aktivitas preprocessing* dapat dilihat pada Gambar 3.47.



Gambar 3. 47 Diagram *Activity Prepocesing*

c. Diagram Aktivitas *Cleaning*

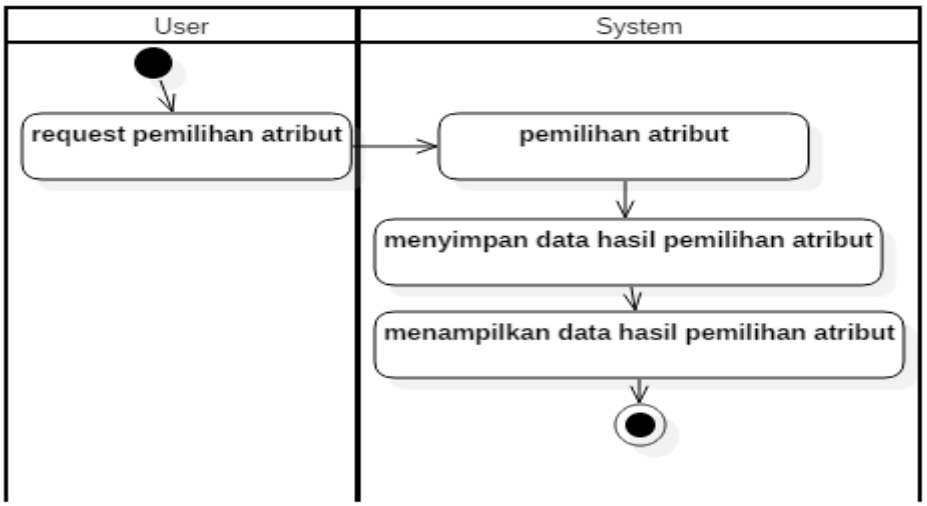
Diagram Aktivitas pembersihan data (*Cleaning*) menjelaskan aktivitas sistem dalam proses pembersihan data. *Diagram Aktivitas cleaning* dapat dilihat pada Gambar 3.48.



Gambar 3. 48 Diagram *Activity Cleaning*

d. Diagram Aktivitas Pemilihan atribut

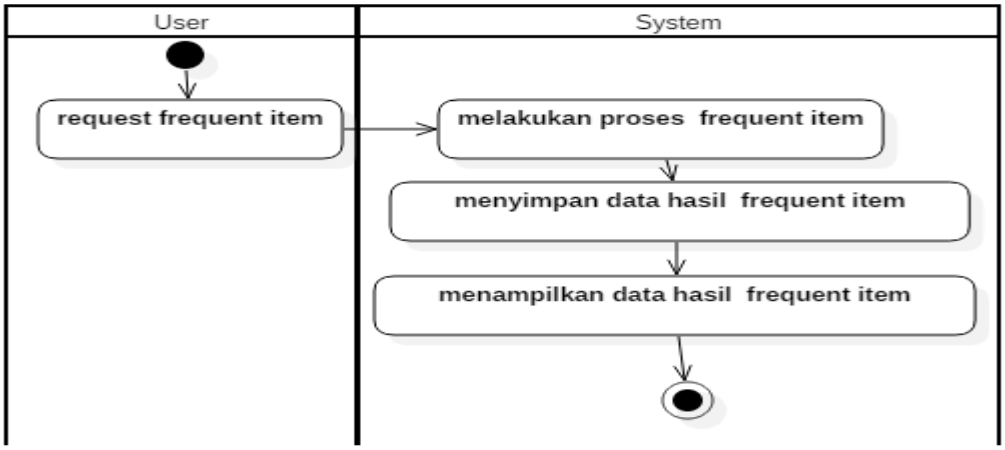
Diagram Aktivitas pemilihan atribut menjelaskan aktivitas sistem dalam melakukan pengaturan proses pemilihan atribut data. *Diagram Aktivitas* pemilihan atribut dapat dilihat pada Gambar 3.49.



Gambar 3. 49 Diagram *Activity* Pemilihan atribut

e. Diagram Aktivitas *Frquency item*

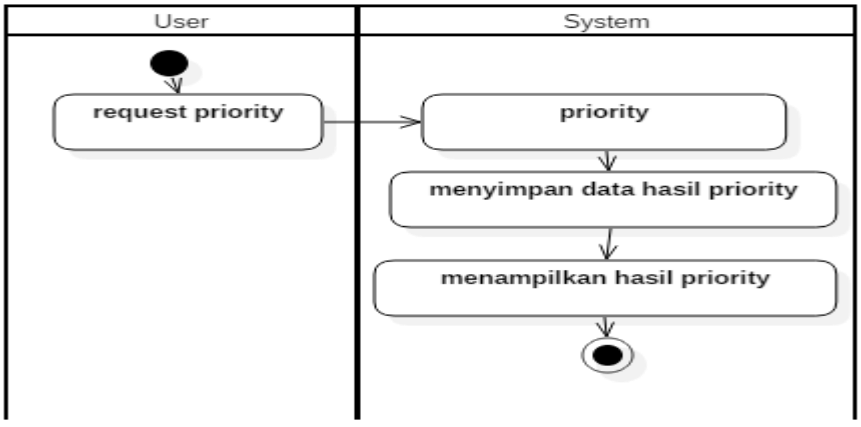
Diagram Aktivitas Hitung *Frequent* menjelaskan aktivitas sistem dalam melakukan proses Penghitungan jumlah kemunculan tiap *item*. *Diagram Aktivitas* Hitung *Frequent* dapat dilihat pada gambar 3.50.



Gambar 3. 50 Diagram Activity Frequent Item

f. Diagram Aktivitas *Priority*

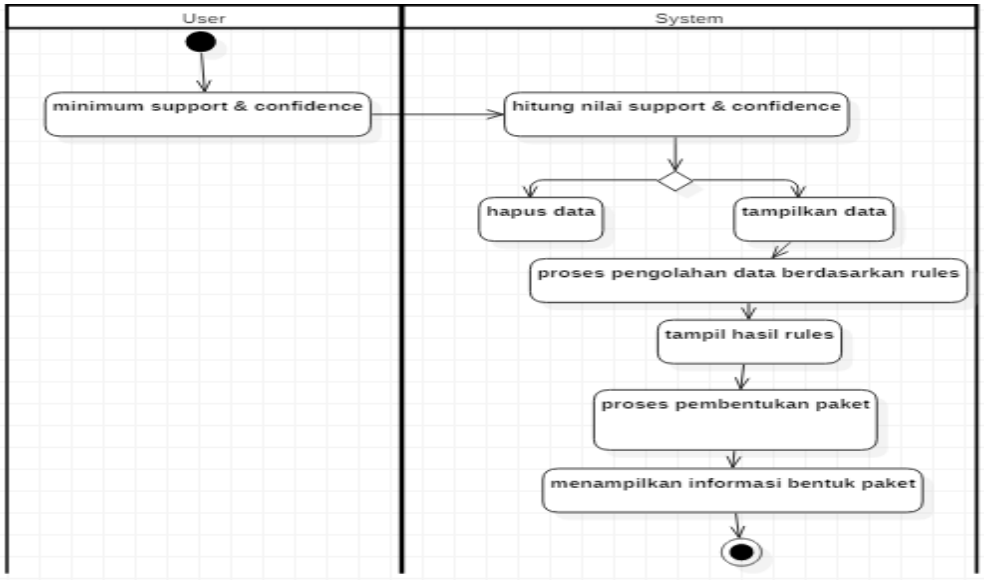
Diagram Aktivitas Priority menjelaskan aktivitas sistem dalam melakukan proses pengurutan data berdasarkan *priority*. *Diagram Aktivitas Priority* dapat dilihat pada gambar 3.51.



Gambar 3. 51 Diagram *Activity Priority*

g. Diagram Aktivitas Asosiasi

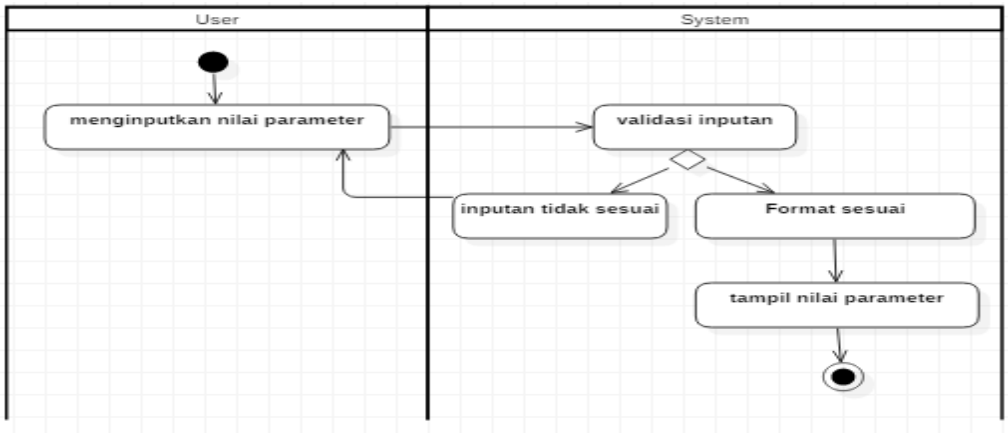
Diagram Aktivitas proses asosiasi menjelaskan aktivitas *user* dan sistem dalam melakukan proses pencarian *frequent pattern , rule* dan melihat hasil asosiasi. *Diagram Aktivitas proses asosiasi* dapat dilihat pada gambar 3.52.



Gambar 3. 52 Diagram *Activity* Proses Asosiasi

h. Diagram Aktivitas *Input* Parameter

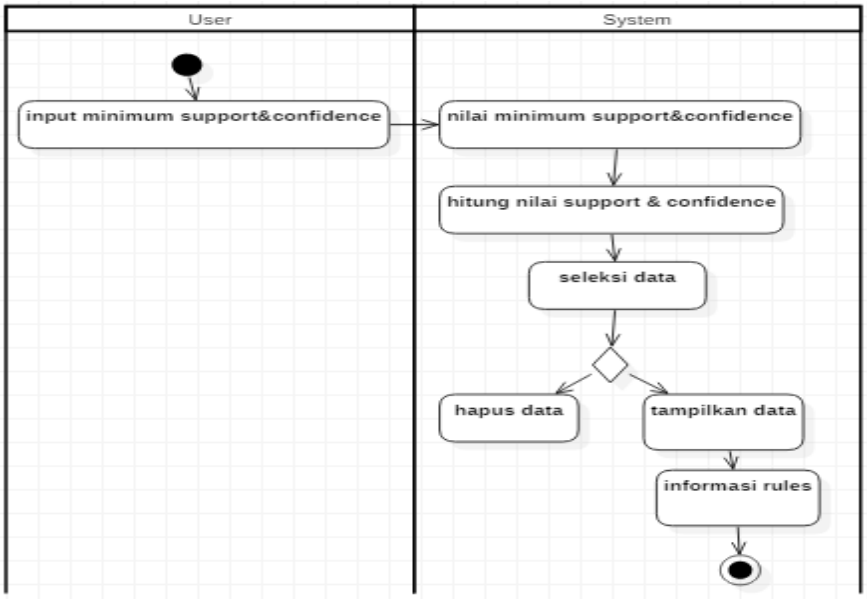
Diagram Aktivitas *Input parameter* menjelaskan aktivitas *user* dan sistem dalam melakukan proses *input* nilai *minimum support dan confidence*. Diagram Aktivitas *input parameter* dapat dilihat pada gambar 3.53.



Gambar 3. 53 Diagram *Activity Input* Parameter

i. Diagram Aktivitas Hasil *Rule*

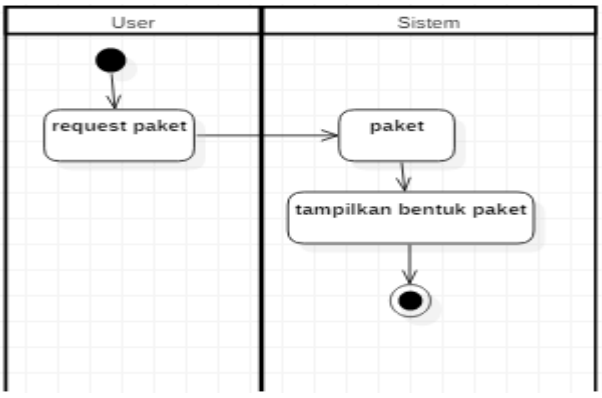
Diagram Aktivitas hasil *rules* menjelaskan aktivitas sistem dalam melakukan proses menemukan *rule. Diagram Aktivitas rules* dapat dilihat pada gambar 3.54.



Gambar 3. 54 Diagram *Activity* hasil *Rules*

j. Diagram Aktivitas Paket

Diagram Aktivitas paket menjelaskan aktivitas sistem dalam melakukan proses menampilkan bentuk paket. *Diagram Aktivitas* paket dapat dilihat pada gambar 3.55.

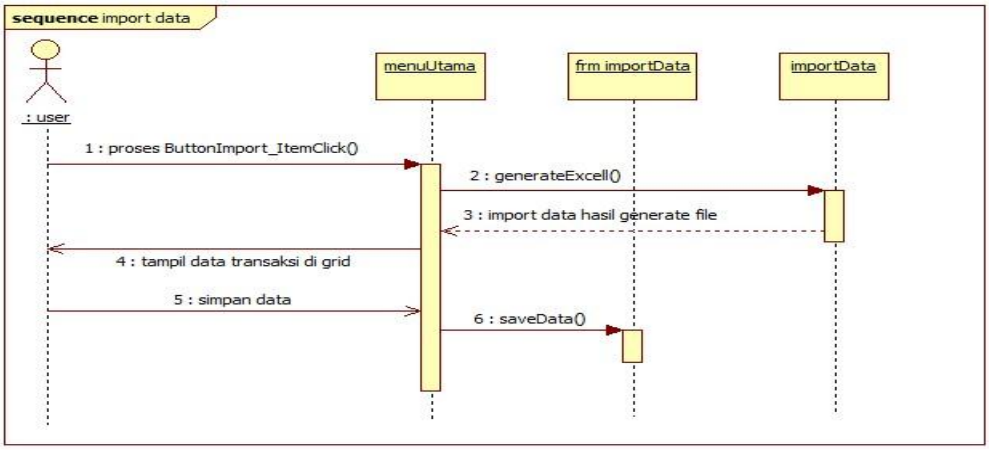


Gambar 3. 55 Diagram *Activity* Paket

3.2.2.3 Diagram Sekuen

a. Diagram Sekuen *Impor*t Data

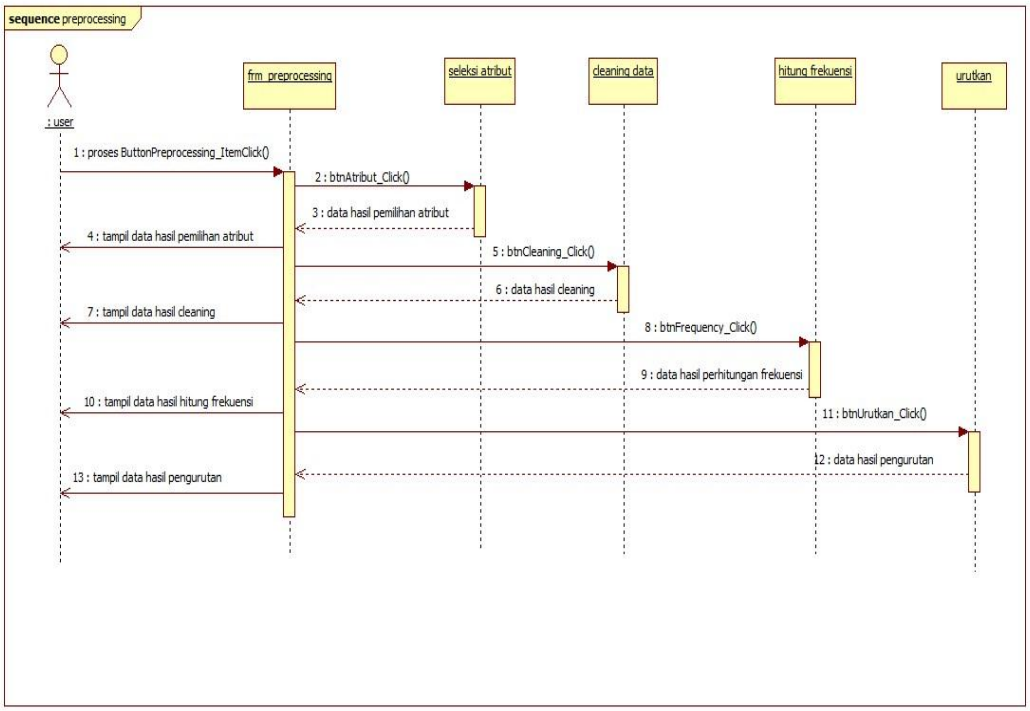
Diagram sekuen *import* data dapat dilihat pada Gambar 3.56.



Gambar 3. 56 Diagram Sekuen *Import* Data

b. Diagram Sekuen *Prepossesing*

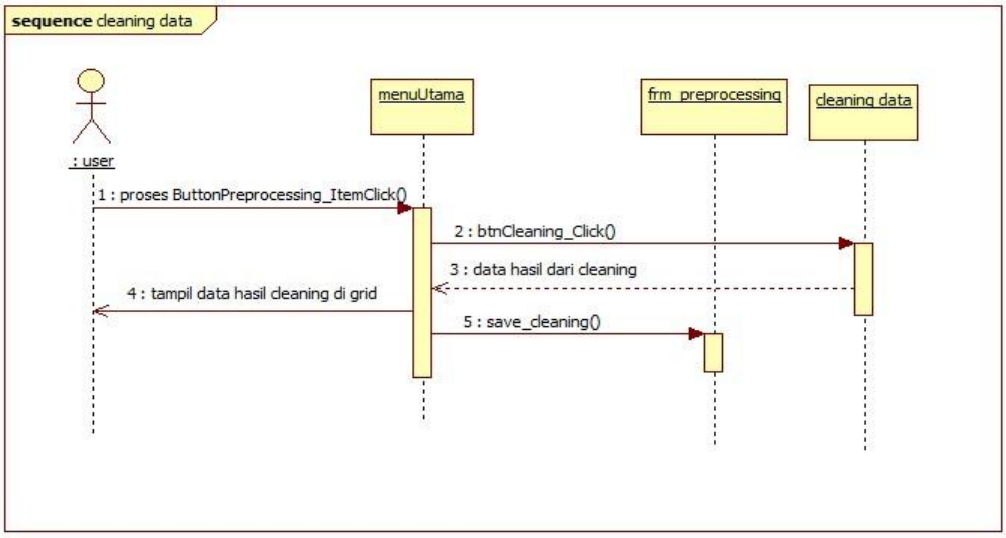
Diagram sekuen prepocessing data dapat dilihat pada Gambar 3.57.



Gambar 3. 57 Diagram Sekuen *Prepocesing*

c. Diagram Sekuen *Cleaning*

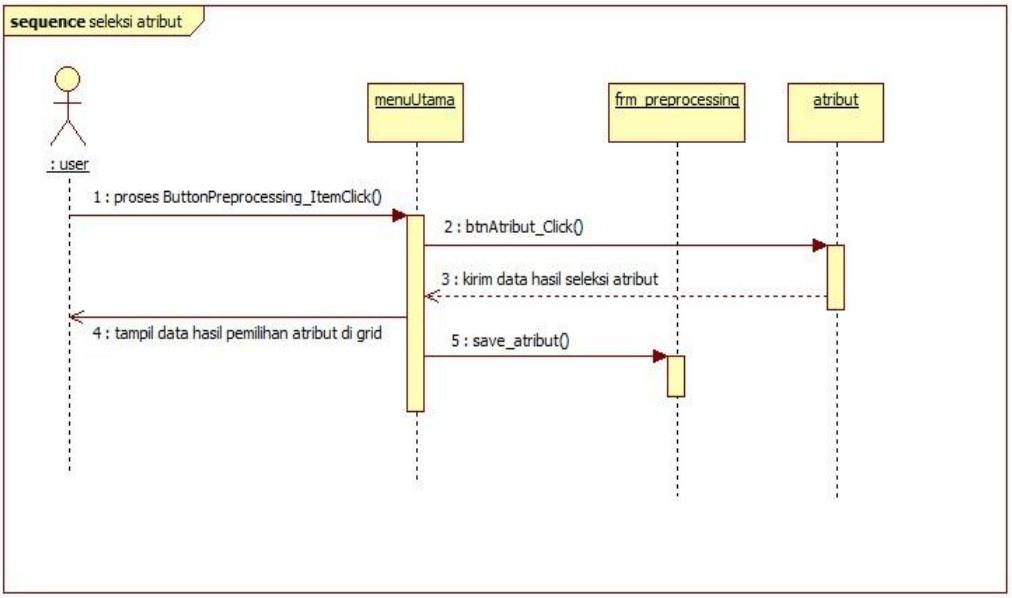
Diagram sekuen *cleaning* data dapat dilihat pada Gambar 3.58.



Gambar 3. 58 Diagram Sekuen *Cleaning*

d. Diagram Sekuen Pemilihan Atribut

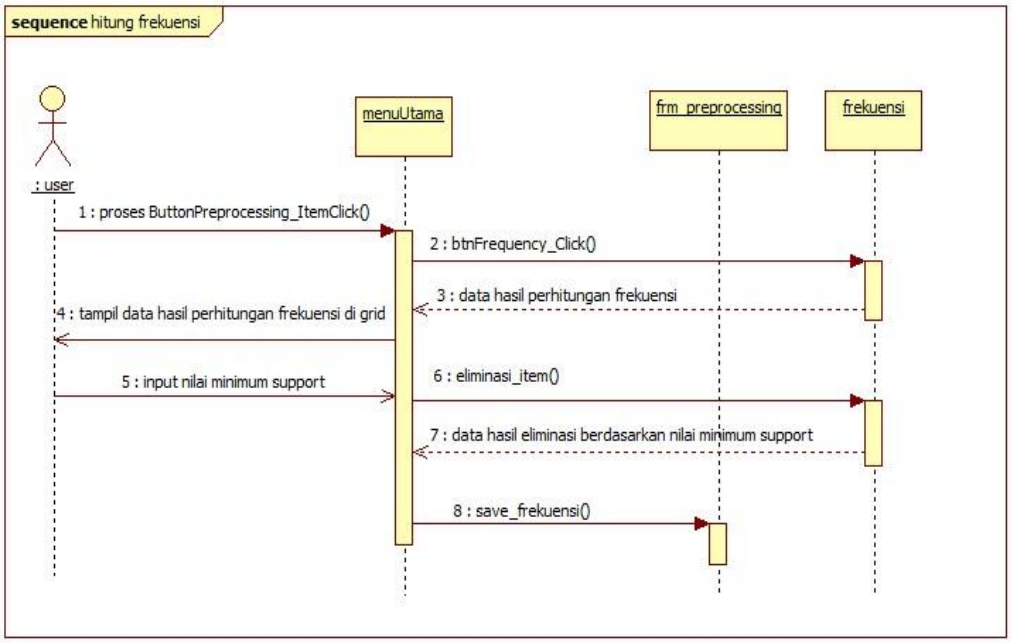
Diagram sekuen pemilihan atribut dapat dilihat pada Gambar 3.59.



Gambar 3. 59 Diagram Sekuen Pemilihan atribut

e. Diagram Sekuen Hitung *Frequency*

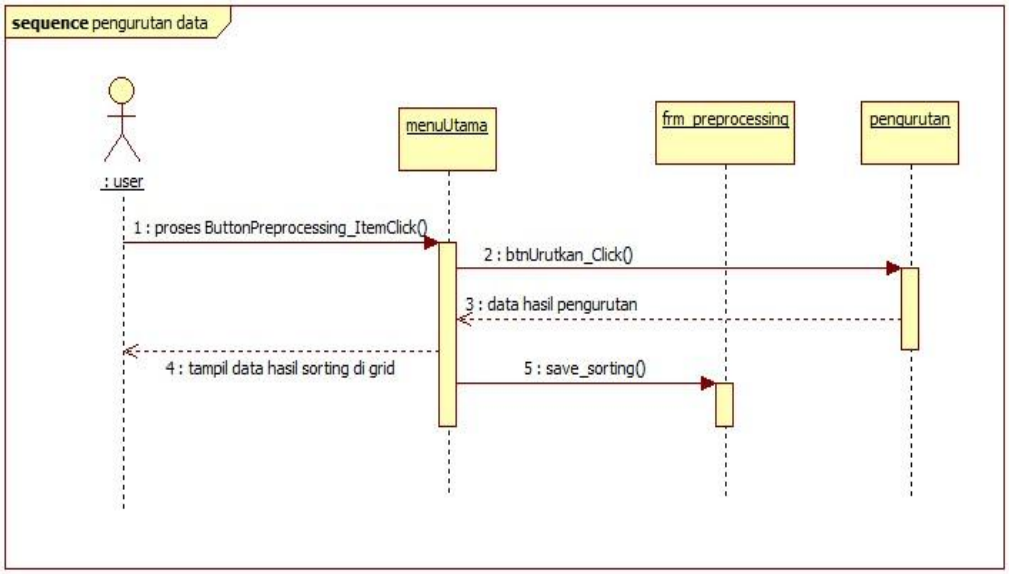
Diagram sekuen hitung *frequency* dapat dilihat pada Gambar 3.60.



Gambar 3. 60 Diagram Sekuen *Frequent Item*

f. Diagram Sekuen *Priority*

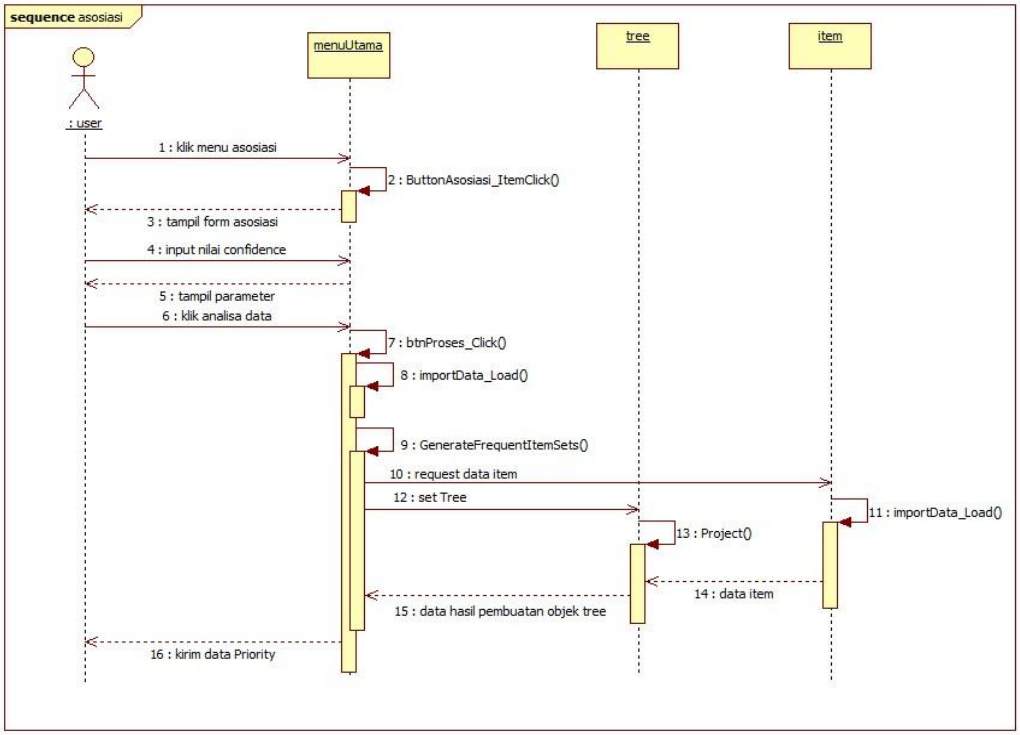
Diagram sekuen *priority* dapat dilihat pada Gambar 3.61.



Gambar 3. 61 Diagram Sekuen *Priority*

g. Diagram Sekuen Asossiasi

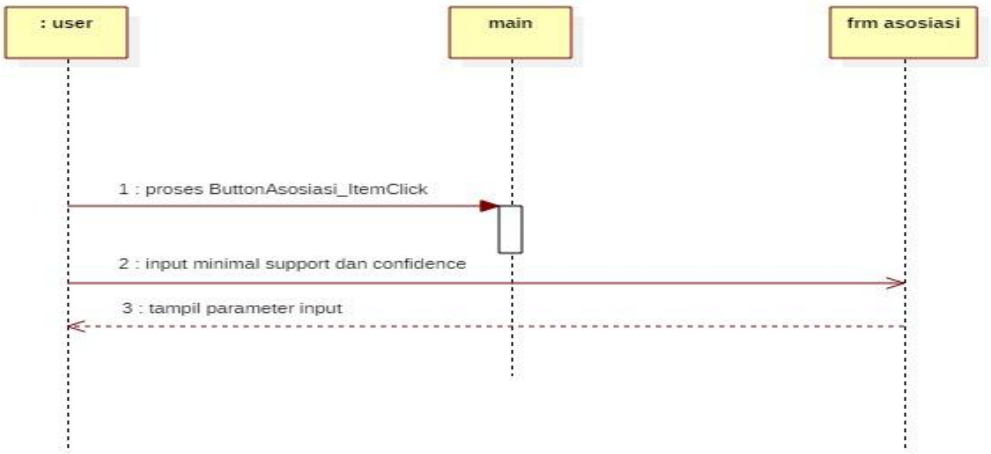
Diagram sekuen Asossiasi dapat dilihat pada Gambar 3.62.



Gambar 3. 62 Diagram Sekuen Asosiasi

h. Diagram sekuen *input* parameter

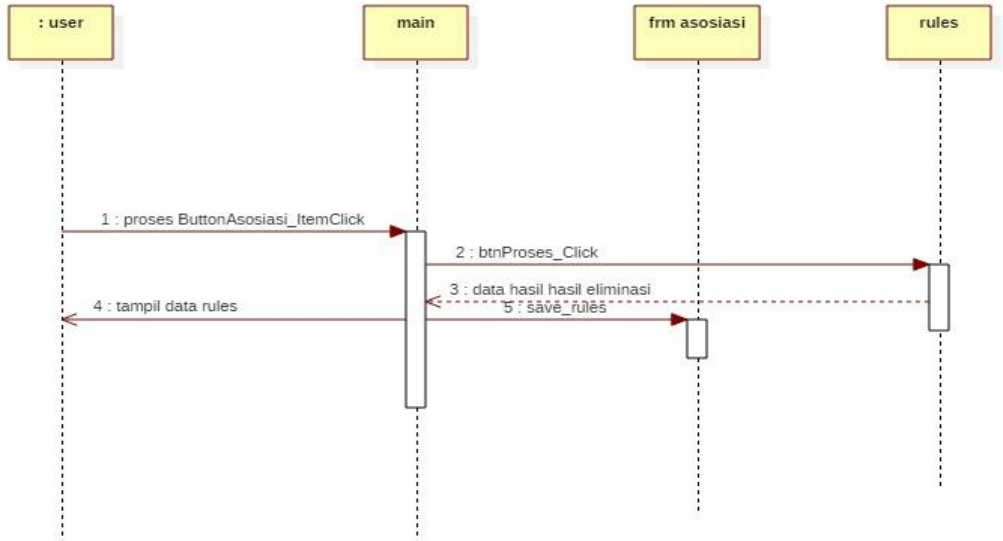
Diagram sekuen *input* parameter dapat dilihat pada gambar 3.63.



Gambar 3. 63 Diagram Sekuen I*nput* Parameter

i. *Sekuence* diagram lihat *rules*

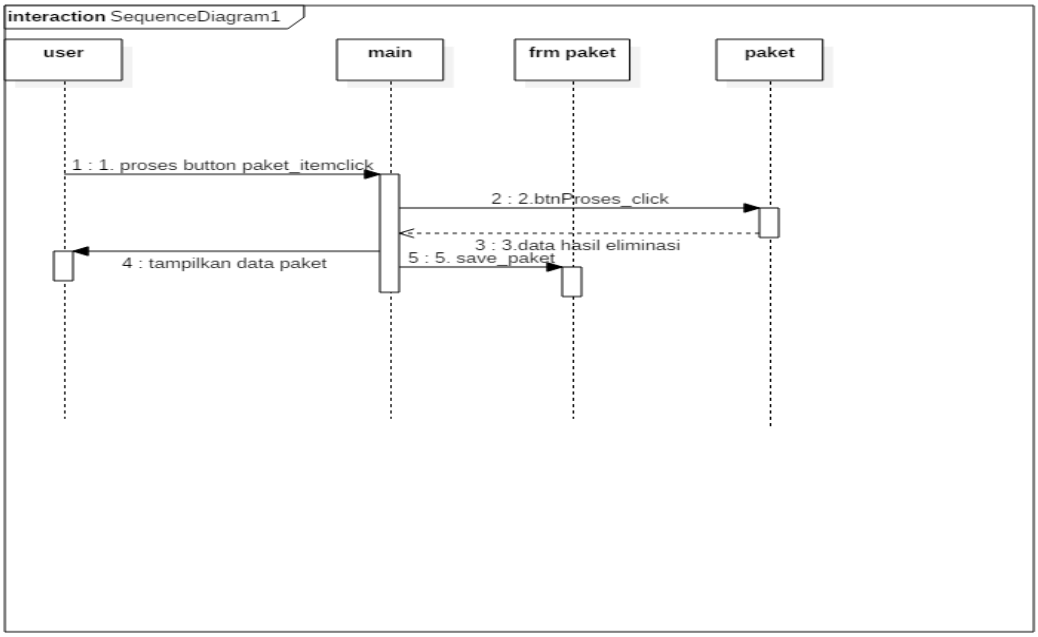
Diagram sekuen lihat *rules* dapat dilihat pada gambar 3.64.



Gambar 3. 64 Diagram Sekuen Lihat *Rules*

a. Sekuen Diagram Paket

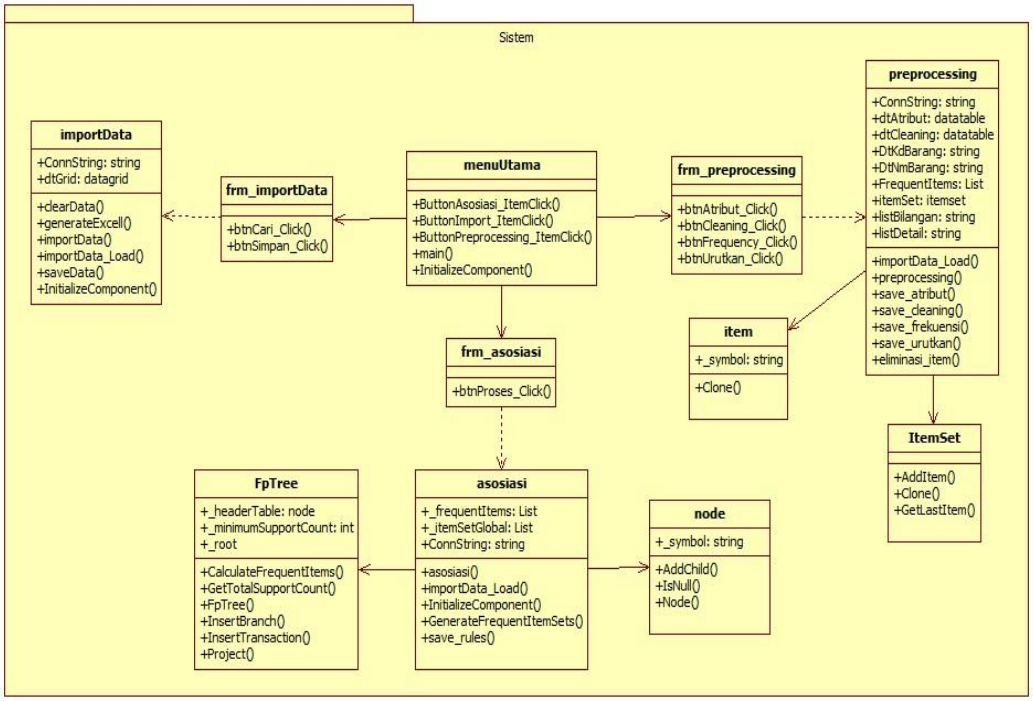
Diagram sekuen lihat *rules* dapat dilihat pada gambar 3.65.



Gambar 3. 65 Diagram Sekuen Paket

3.2.2.4 Diagram Kelas

Diagram Kelas dapat dilihat pada Gambar 3.66.



Gambar 3. 66 Digram Class

3.2.2.5 Perancangan Sistem

Proses perancangan yang akan dilakukan pada tahap ini adalah perancangan sistem basis data, perancangan struktur menu, perancangan antar muka, perancangan pesan, jaringan semantik, dan perancangan prosedural.

3.2.2.6 Perancangan *class*

Brikut ini adalah perancangan yang digunakan pada sistem yang akan dibangun.

Tabel 3. 24 Struktur *Class* Menu Utama

Nama *Class*: MenuUtama		
Nama	Jenis	*Type*
buttonAsosiasi_ItemClick()	Method	Public
ButtonPreprocessing_ItemClick()	Method	Public
Main()	Method	Public
InitializeComponent()	Method	Public

Tabel 3. 25 Struktur *Class Importdata*

Nam *Class:importData*		
Nama	Jenis	*Type*
ClearData()	Method	Public
generateExcel()	Method	Public
importData ()	Method	Public
importData_Load()	Method	Public
saveData()	Method	Public
InitializeComponent()	Method	Public
ConString:string	Atribut	Public
dtGrid: datagrid	Atribut	Public

Tabel 3. 26 Struktur *Class* Frm_*Importdata*

Nama *Class*: frm_importData		
Nama	Jenis	*Type*
btnCari_Click()	Method	Public
btnSimpan_Click()	Method	Public

Tabel 3. 27 Struktur *Class Prepocessing*

Nam *Class:preprocessing*		
Nama	Jenis	*Type*
importData_Load()	Method	Public
Preprocessing()	Method	Public
Save_atribut()	Method	Public
Save_cleaning()	Method	Public
Save_frekuensi()	Method	Public
Save_urutkan()	Method	Public
Eliminasi_item()	Method	Public
ConnString: string	Atribut	Public
dtAtribut:database	Atribut	Public

Nam *Class:preprocessing*		
Nama	Jenis	*Type*
dtCleaning:database	Atribut	Public
DtKdBarang:string	Atribut	Public
DtNamaBarang:string	Atribut	Public
FrequentItems:List	Atribut	Public
itemSet:itemset	Atribut	Public
LstBilangan: string	Atribut	Public
LstDetail:string	Atribut	Public

Tabel 3. 28 Struktur *Class* Frm_*Prepocessing*

Nama *Class*: frm_*prepocessing*		
Nama	Jenis	*Type*
btnAtribut_Click()	Method	Public
btnCleaning_Click()	Method	Public
btnFrequency_Click()	Method	Public
btnUrutkan_Click()	Method	Public

Tabel 3. 29 Struktur *Class Item*

Nama *Class: item*		
Nama	Jenis	*Type*
Clone()	Method	Public
_symbol:string	Atribut	Public

Tabel 3. 30 struktur *class itemset*

Nama Class: ItemSet		
Nama	Jenis	Type
AddItem()	Method	Public
Clone()	Method	Public
GetLastItem()	Method	Public

Tabel 3. 31 Struktur *Class* Asosiasi

Nam *Class*: Asosiasi		
Nama	Jenis	*Type*
Asosiasi()	Method	Public
importData_Load()	Method	Public
InitializerComponen()	Method	Public
GenerateFrequentItemSets()	Method	Public
Save_rules()	Method	Public
_frequentItems: List	Atribut	Public
_itemSetGlobal: List	Atribut	Public
ConnString: string	Atribut	Public

Tabel 3. 32 Struktur *Class* Frm_Asosiasi

Nam *Class*: Frm_Asosiasi		
Nama	Jenis	*Type*
btnProses_Click()	Method	Public

Tabel 3. 33 Struktur *Class Fp_Tree*

Nam *Class*: *FpTree*		
Nama	Jenis	*Type*
CalculateFrequentItems()	Method	Public
GetTotalSupportCount()	Method	Public
FpTree()	Method	Public
InsertBranch()	Method	Public
InsertTransaction()	Method	Public
Project()	Method	Public
_headerTable: node	Atribut	Public
_minimumSupportCount: int	Atribut	Public
_root	Atribut	Public

Tabel 3. 34 Struktur *Class Node*

Nama *Class: Node*		
Nama	Jenis	*Type*
AddChid()	Method	Public
IsNull()	Method	Public
Node()	Method	Pulic
_symbol: string	Atribut	Public

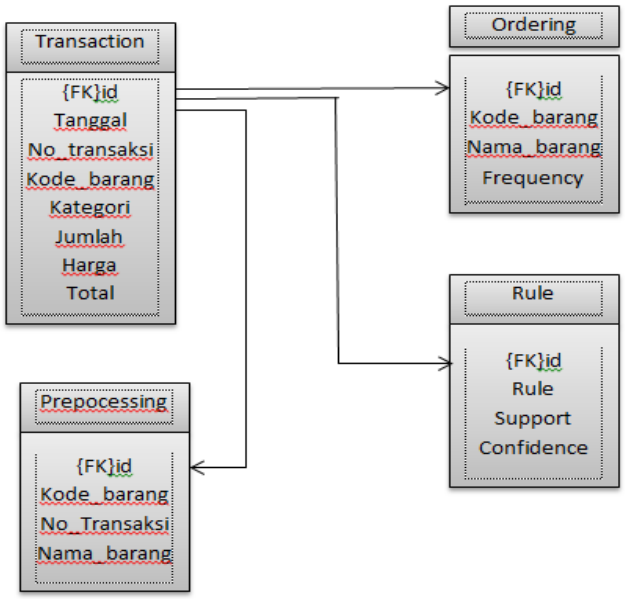
3.2.2.7 Perancangan Basis Data

Perancangan Basis Data adalah menciptakan atau merancang kumpulan data yang terhubung dan tersimpan secara bersama-sama. Tabel yang akan digunakan dalam program ini antara lain:

1. Tabel *Ordering*
2. Tabel *Transaction*
3. Tabel *Prepocessing*
4. Tabel *Rules*

3.2.2.7.1 Skema Relasi

Skema relasi dari aplikasi yang akan dibangun dapat dilihat pada gambar 3.67.



Gambar 3. 67 Skema Relasi

3.2.2.7.2 Struktur Tabel

Struktur tabel pada aplikasi yang akan dibangun terdiri dari 4 tabel dengan rincian sebagai berikut:

1. Tabel *Ordering*

   Tabel *ordering* untuk menyimpan data hasil *import* file excel. Tabel *ordering* dapat dilihat pada tabel 3.35.

Tabel 3. 35 Tabel *Ordering*

Nama Atribut	Tipe Data	Panjang	Keterangan
id	Int	10	PK
Kode_barang	varchar	10	
Nama_barang	Varchar	100	
Frequency	Int	11	

2. Tabel *Transactions*

Tabel data *transaction* untuk menyimpan data hasil *import* file excel. Tabel *transactions* dapat dilihat pada tabel 3.36.

Tabel 3. 36 Tabel *Transaction*

Nama Atribut	Tipe Data	Panjang	Keterangan
Id	Int	10	FK
Tanggal	Text	-	
No_Transaksi	Varchar	50	
Kode_barang	Text	-	
Nama_barang	Text	-	
Kategori	Text	-	
Jumlah	Int	10	
Harga	Int	10	
Total	Int	10	

3. Tabel *Prepocesising*

   Tabel *preprocessing* berfungsi untuk menyimpan data hasil *preprocessing*. Tabel *preprocessing* dapat dilihat pada tabel 3.37.

Tabel 3. 37 Tabel *Prepocessing*

Nama Atribut	Tipe Data	Panjang	Keterangan
Id	Int	10	FK
Kode_barang	Varchar	10	
No_Transaksi	Varchar	50	
Nama_barang	Varchar	100	
Kategori	Varchar	20	

4. Tabel *rules*

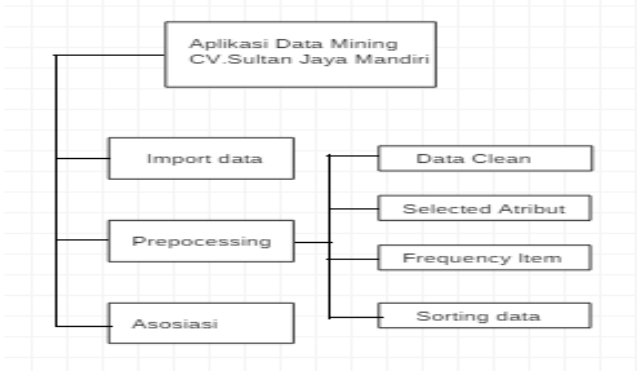
   Tabel *rules* berfngsi untuk menyimpan data hasil *rules*. Tabel *rules* dapat dilihat pada tabel 3.38.

Tabel 3. 38 Tabel *Rules*

Nama Atribut	Tipe Data	Panjang	Keterangan
Id	Int	10	FK
Rule	Text	-	
Support	Int	11	
Confidence	Int	11	

3.2.2.7.3Perancangan Struktur Menu

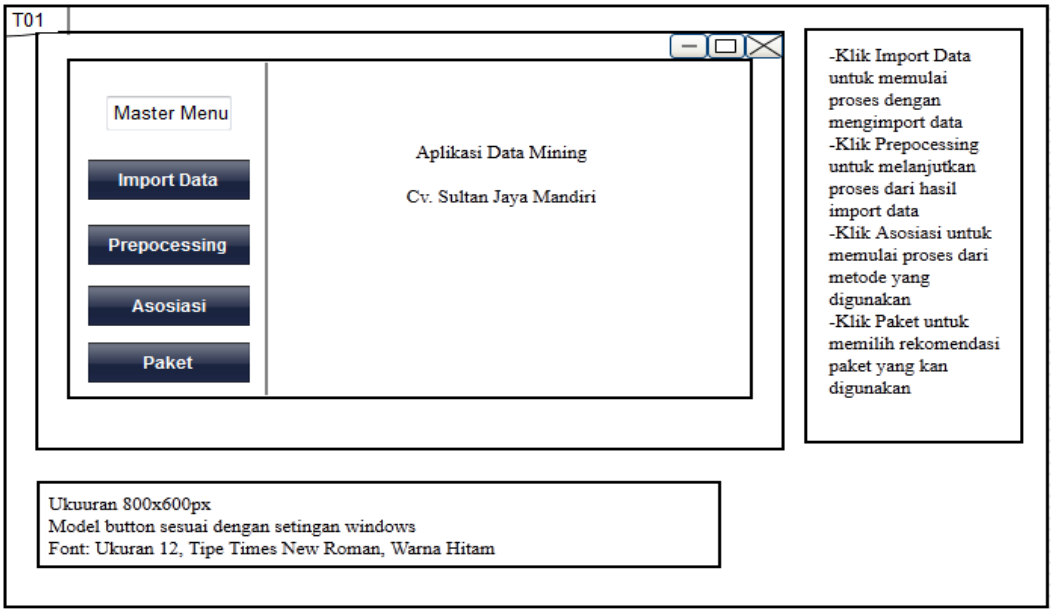
Dalam pembangunan perangkat lunak diperlukan struktur menu untuk memudahkan pengguna dalam menavigasikan fungsional dari sebuah sistem. Adapun perancangan struktur menu untuk aplikasi yang akan dibangun dapat dilihat pada gambar 3.67.



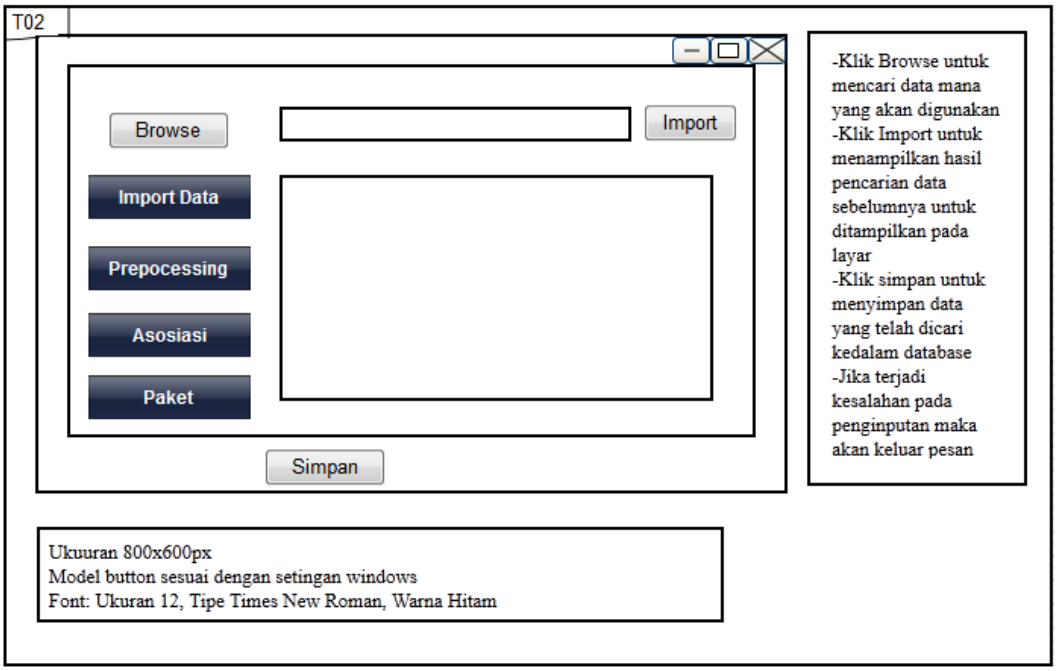
Gambar 3. 68 Struktur Menu

3.2.2.7.4Perancangan Antar Muka Jendela Aplikasi

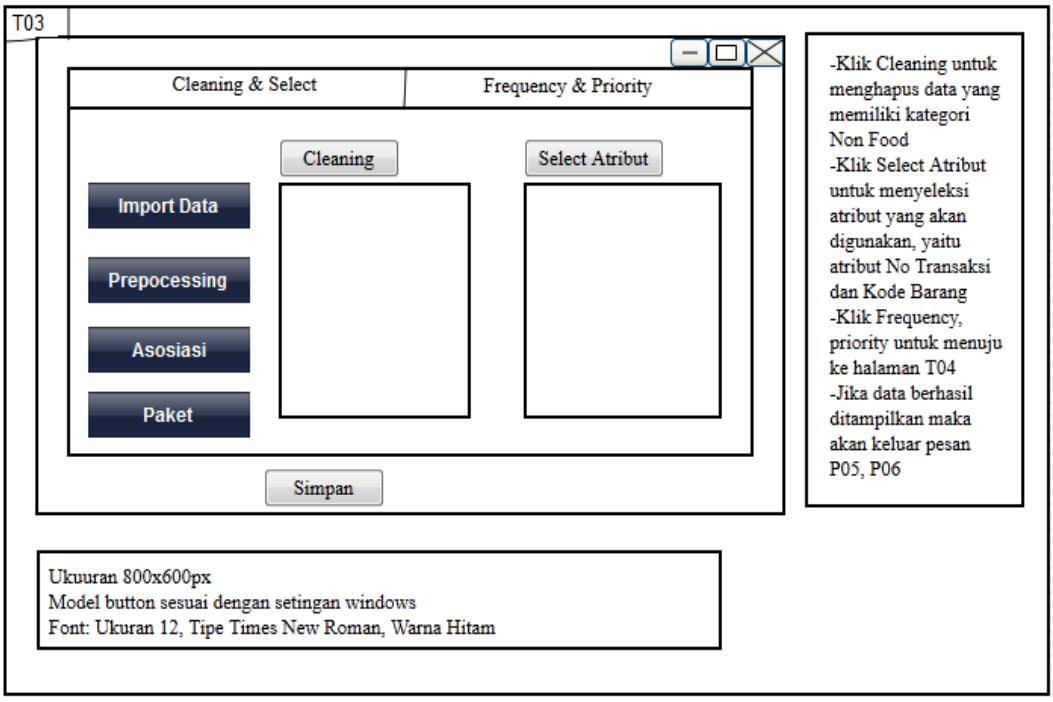
Perancangan antar muka berfungsi untuk memberikan gambaran tentang perangkat lunak yang akan dibangun, supaya lebih memudahkan dalam pembuatan perangkat lunak nantinya. Adapun perancangan antar muka jendela aplikasi yang akan dibangun akan dijelaskan sebagai berikut:



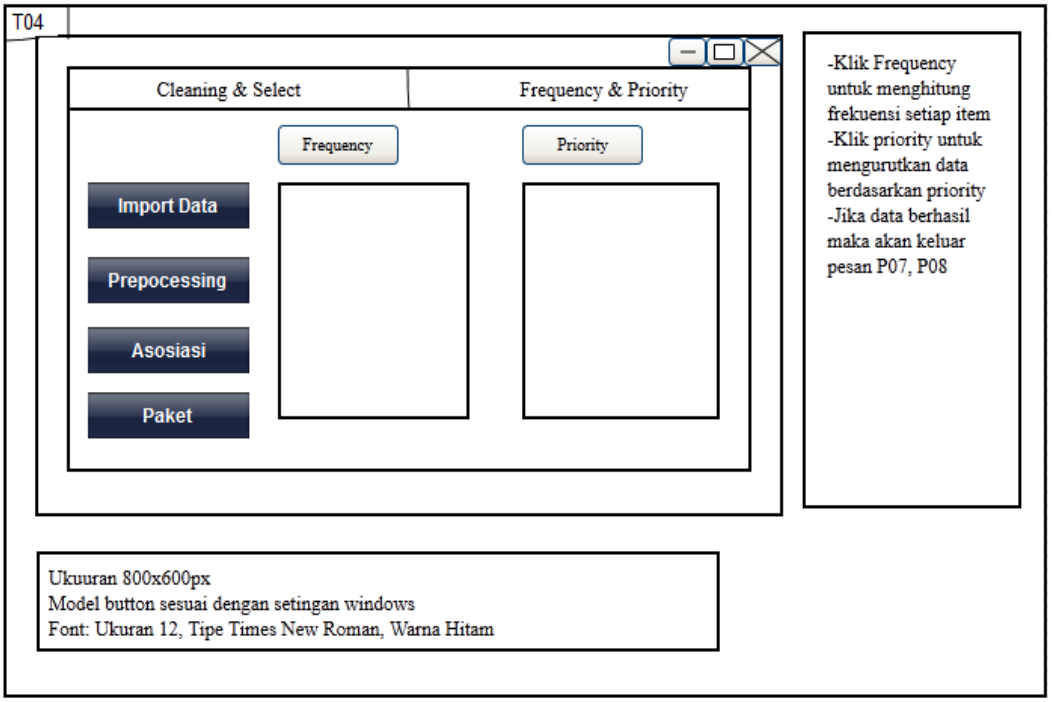
Gambar 3. 69 Tampilan Antar Muka Menu Utama



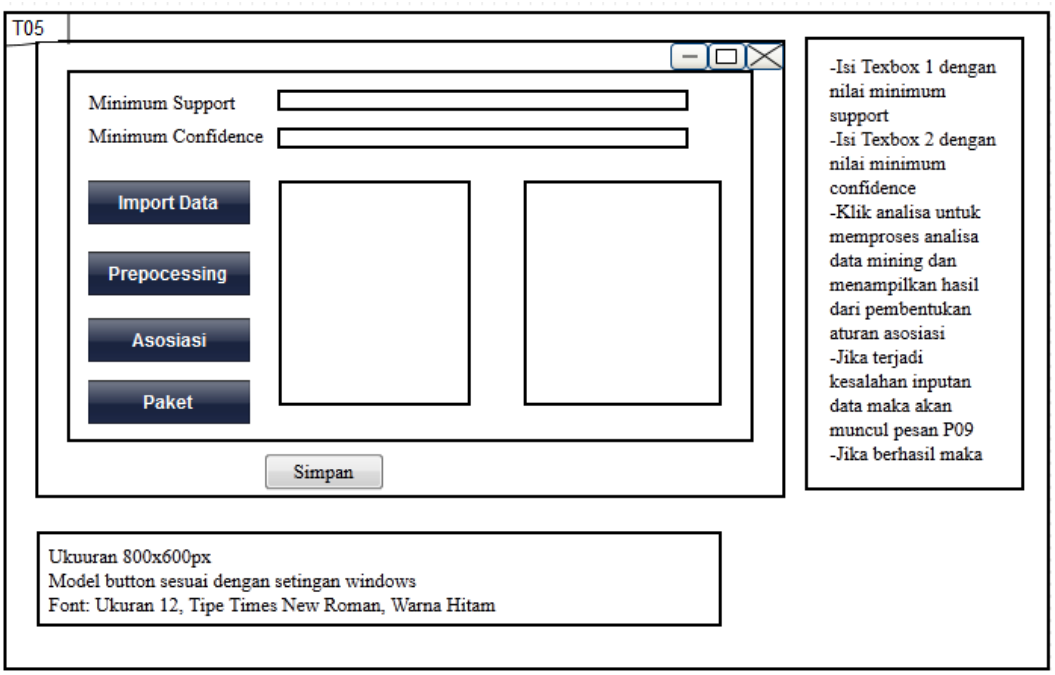
Gambar 3. 70 Tampilan Antar Muka *Import* Data



Gambar 3. 71 Tampilan Antar Muka *Prepocesing1*



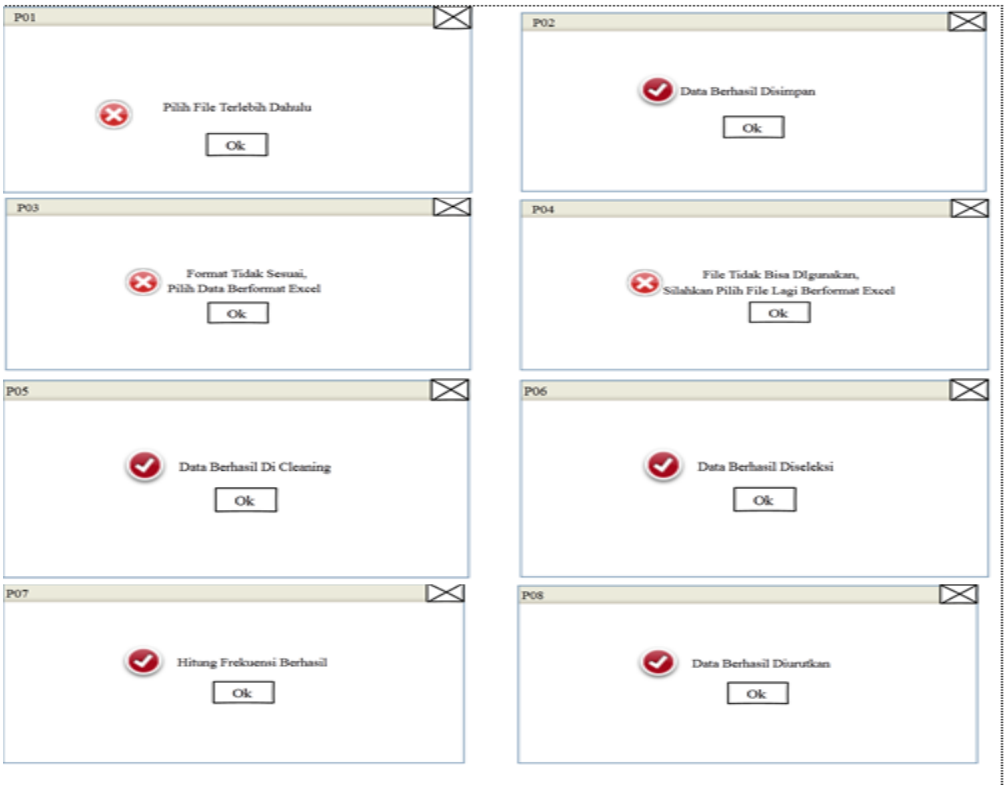
Gambar 3. 72 Tampilan Antar Muka *Prepocesing2*



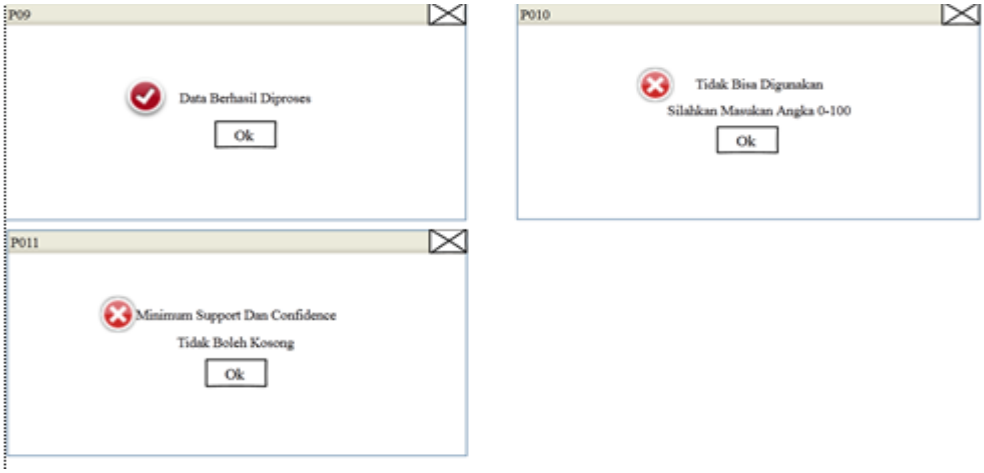
Gambar 3. 73 Tampilan Antar Muka Halaman Proses Asosisasi

3.2.2.7.5 Perancangan Antar Muka Pesan

Adapun perancangan pesan dapat dilihat pada gambar 3.73:



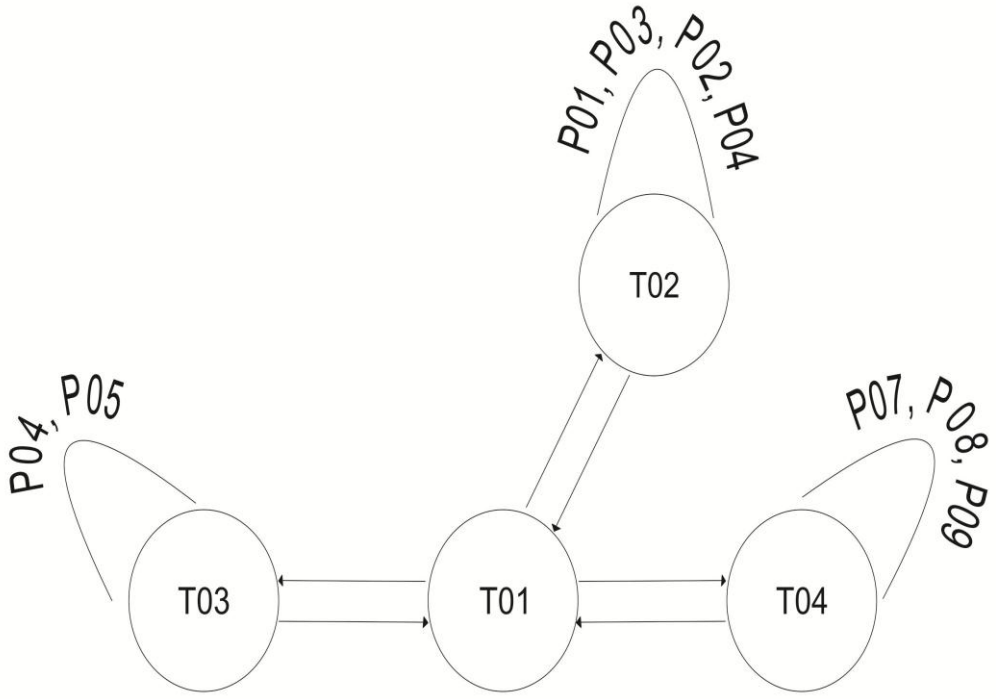
Gambar 3. 74 Tampilan Antar Muka Pesan1



Gambar 3. 75 Tampilan Antar Muka Pesan2

3.2.2.7.6 Perancangan Jaringan Semantik

Perancangan semantik dapat dilihat pada gambar 3.75.



Gambar 3. 76 Jaringan Semantik