BAB 3
ANALISIS DAN PERANCANGAN SISTEM

3.1 Analisis Sistem

Analisis sistem (*System Analysis*) dapat didefinisikan sebagai penguraian dari suatu sistem informasi yang utuh ke dalam bagian-bagian komponennya dengan maksud untuk mengidentifikasikan dan mengevaluasi permasalahan-permasalahan, kesempatan-kesempatan, hambatan-hambatan yang terjadi dan kebutuhan-kebutuhan yang diharapkan sehingga dapat diusulkan perbaikan-perbaikannya. Dalam analisa sistem ini meluputi beberapa bagian, yaitu:

1. Analisis Masalah
2. Analisis *Crisp-DM*
3. Analisis Kebutuhan Non Fungsional
4. Analisis Kebutuhan Fungsional

3.1.1 Analisis Masalah

Analisis masalah yang dilakukan adalah mengidentifikasikan permasalahan atau kendala di dalam penelitian yang dilakukan. Masalah dapat didefinisikan sebagai suatu pernyataan yang diinginkan untuk dipecahkan. Perlu ada analisis masalah melalui rumusan masalah yang sudah ditentukan sebelumnya. Berdasarkan hasil pengamatan, diketahui bahwa masalahnya adalah sebagai berikut :

Cv. Bukit Manikam mengalami kendala dalam penentuan paket penjualan yang akan ditawarkan ke pelanggan. Kendala yang terjadi dikarenakan paket yang ditawarkan kurang laku terjual, maka dari permasalahan tersebut Cv. Bukit Manikam membutuhkan informasi penentuan paket barang yang tepat. Berdasarkan penjelasan di atas untuk penyelesaian masalah yang terjadi, perlu adanya penerapan *data mining*.

3.1.2 Analisis 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 Business Understanding

Business Understanding merupakan tahapan pertama yang dilakukan dalam kerangka kerja *CRISP-DM*. Dalam tahapan bisnis ini terbagi menjadi dua bagian, yaitu:

1. Identifikasi Tujuan Bisnis

   Tujuan bisnis Cv. Bukit Manikam yaitu menjual paket barang untuk mencapai target penjualan paket yang sudah ditentukan.

2. Menilai Situasi Perusahaan

   Menilai situasi merupakan tahapan untuk menjelaskan sumber daya, asumsi dan batasan yang terdapat pada Cv. Bukit Manikam.

   a. Sumber daya

      1. Sumber daya data

         Sumber daya data yang terdapat pada Cv. Bukit Manikam meliputi data nota transaksi penjualan dan katalog.

      2. Sumber daya perangkat keras

         Sumber daya perangkat keras yang terdapat pada Cv. Bukit Manikam meliputi perangkat komputer yang digunakan oleh kepala cabang, supervisor, dan bagian administrasi.

      3. Sumber daya personil

         Sumber daya personil yang terdapat pada Cv. Bukit Manikam meliputi kepala cabang, supervisor, *marketing*, *sales*, kurir dan bagian gudang.

   b. Asumsi

      1. Data yang digunakan dalam penelitian ini adalah data transaksi berupa *file* excel periode januari 2016.
      2. Hasil dari penelitian ini berupa *rules* yang difilter sebagai acuan untuk pembentukan paket barang.

   c. Batasan

      Batasan paket dalam penelitian ini adalah dalam satu paket hanya terdapat satu *brand* saja, sebagai contoh yamaha dengan yamaha.

3. Penentuan Sasaran *Data mining*

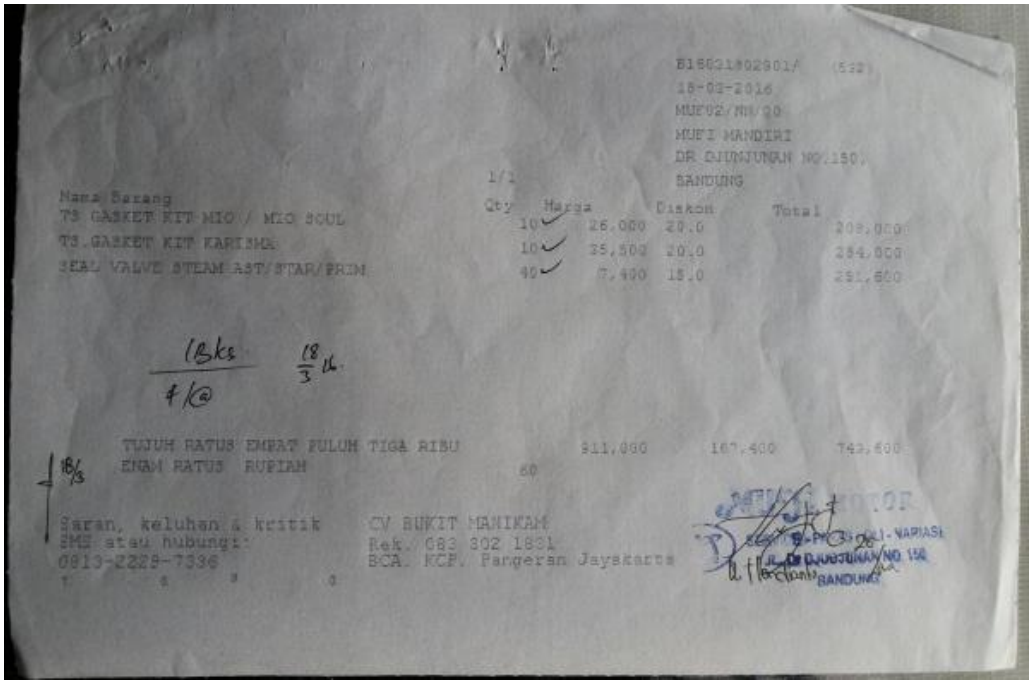
Penentuan sasaran penerapan *data mining* adalah mengetahui pasangan barang untuk dijadikan dasar keputusan Cv. Bukit Manikam dalam menentukan paket penjualan.

3.1.2.2 Data Understanding

Tahapan pemahaman data merupakan tahapan kedua yang dilakukan dalam kerangka kerja *CRISP-DM*. Dalam tahapan pemahaman data ini terbagi menjadi beberapa bagian, yaitu:

1. Mengumpulkan data awal

Tahap pertama dalam pemahaman data dimulai dengan pengumpulan data, sumber data yang didapat dalam penelitian ini adalah berupa data nota transaksi penjualan di Cv. Bukit Manikam, yang dapat dilihat pada gambar 3.1



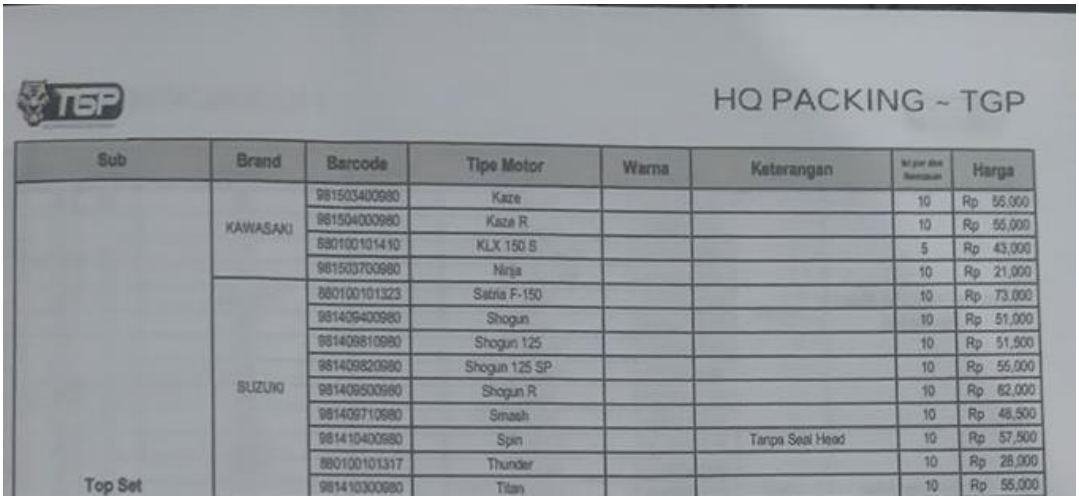
Gambar 3. 1 Sumber Data Nota

Lalu data dari nota penjualan dipindahkan ke dalam *file Microsoft Excel* (.xlsx) seperti contoh pada tabel 3.1

Tabel 3. 1 Contoh Data Transaksi Yang Dipakai

Nomor Faktur	Tanggal	Nama Barang	Qty	Harga	Diskon	Total Harga
B16010100001/	01/01/2016	Seal Valve Steam Beat Hi-Q	40	Rp 7.400	20	Rp 296.000
B16010100001/	01/01/2016	Ts Gasket Kit Vixion Hi-Q	20	Rp 30.000	20	Rp 600.000
B16010100001/	01/01/2016	Pac.R.Crank Cover Vega Z R	20	Rp 6.000	20	Rp 120.000
B16010100001/	01/01/2016	Pac.Muffler Mio	10	Rp 2.100	20	Rp 21.000
B16010100001/	01/01/2016	Seal Valve Steam Jupiter Mx Hi-Q	30	Rp 6.200	20	Rp 186.000
B16010100002/	01/01/2016	Ts Gasket Kit Mio / Mio Soul	15	Rp 26.000	20	Rp 390.000
B16010100002/	01/01/2016	Pac.R.Crank Cover Vega Z R	5	Rp 6.000	20	Rp 30.000
B16010100002/	01/01/2016	Ts Gasket Kit Vixion Hi-Q	5	Rp 30.000	20	Rp 150.000
B16010100002/	01/01/2016	Seal Valve Steam Jupiter Mx Hi-Q	60	Rp 6.200	20	Rp 372.000
B16010100003/	01/01/2016	Seal Valve Steam Jupiter Mx Hi-Q	5	Rp 6.200	20	Rp 31.000
B16010100003/	01/01/2016	Visor Beat Injection (Abu-Abu)	5	Rp 28.000	20	Rp 140.000
B16010100003/	01/01/2016	Pac.Muffler Mio	5	Rp 2.100	20	Rp 10.500
B16010100003/	01/01/2016	Ts Gasket Kit Mio / Mio Soul	10	Rp 26.000	20	Rp 260.000
B16010100003/	01/01/2016	Pac.R.Crank Cover Vega Z R	40	Rp 6.000	20	Rp 240.000

Untuk pemaketan sendiri, dibutuhkan kode barang dan *brand* yang akan dibutuhkan dalam tahap selanjutnya. Untuk kode barang dan *brand* didapat dari katalog, contoh dari katalog barang dapat dilihat pada gambar 3.2



Gambar 3. 2 Katalog Barang

Selanjutnya, *barcode* dan *brand* dimasukan dan disatukan ke dalam *file* data transaksi yang telah tersedia, berikut hasil setelah *barcode* dan *brand* yang telah dimasukkan dapat dilihat pada tabel 3.2

Tabel 3. 2 Data Transaksi

Nomor Faktur	Tanggal	Barcode	Brand	Nama Barang	Qty	Harga	Diskon	Total Harga
B16010100001/	01/01/2016	991204500461	Honda	Seal Valve Steam Beat Hi-Q	40	Rp 7.400	20	Rp 296.000
B16010100001/	01/01/2016	981313000980	Yamaha	Ts Gasket Kit Vixion Hi-Q	20	Rp 30.000	20	Rp 600.000
B16010100001/	01/01/2016	989337900010	Yamaha	Pac.R.Crank Cover Vega Z R	20	Rp 6.000	20	Rp 120.000
B16010100001/	01/01/2016	989307500160	Yamaha	Pac.Muffler Mio	10	Rp 2.100	20	Rp 21.000
B16010100001/	01/01/2016	991307700461	Yamaha	Seal Valve Steam Jupiter Mx Hi-Q	30	Rp 6.200	20	Rp 186.000
B16010100002/	01/01/2016	880100102212	Yamaha	Ts Gasket Kit Mio / Mio Soul	15	Rp 26.000	20	Rp 390.000
B16010100002/	01/01/2016	989337900010	Yamaha	Pac.R.Crank Cover Vega Z R	5	Rp 6.000	20	Rp 30.000
B16010100002/	01/01/2016	981313000980	Yamaha	Ts Gasket Kit Vixion Hi-Q	5	Rp 30.000	20	Rp 150.000
B16010100002/	01/01/2016	991307700461	Yamaha	Seal Valve Steam Jupiter Mx Hi-Q	60	Rp 6.200	20	Rp 372.000
B16010100003/	01/01/2016	991307700461	Yamaha	Seal Valve Steam Jupiter Mx Hi-Q	5	Rp 6.200	20	Rp 31.000
B16010100003/	01/01/2016	880203101146	Honda	Visor Beat Injection (Abu-Abu)	5	Rp 28.000	20	Rp 140.000
B16010100003/	01/01/2016	989307500160	Yamaha	Pac.Muffler Mio	5	Rp 2.100	20	Rp 10.500
B16010100003/	01/01/2016	880100102212	Yamaha	Ts Gasket Kit Mio / Mio Soul	10	Rp 26.000	20	Rp 260.000
B16010100003/	01/01/2016	989337900010	Yamaha	Pac.R.Crank Cover Vega Z R	40	Rp 6.000	20	Rp 240.000

2. Mendeskripsikan data

Tahap kedua dalam pemahaman data adalah mendeskripsikan data, yaitu data transaksi. Adapun detail informasi mengenai data yang digunakan dapat dilihat pada tabel 3.3.

Tabel 3. 3 Informasi Struktur Data

Dokumen	Keterangan		
Detail Data Transaksi	Deskripsi	Data ini berisi mengenai data transaksi yang terbentuk	
	Format	Microsoft Excel (.xlsx)	
	Atribut	Nomor Faktur	Nomor pembelian
		Tanggal	Tanggal pembelian
		Brand	Nama *brand* motor
		Barcode	Kode dari setiap barang
		Nama Barang	Nama barang yang dibeli
		Qty	Jumlah barang yang dibeli
		Harga	Harga barang yang dibeli
		Diskon	Pemotongan harga yang diberikan
		Total Harga	Total harga barang yang dibeli

3. Mengidentifikasi masalah kualitas data

Dari data transaksi yang didapat terdapat *noise, noise* yaitu data transaksi yang hanya memiliki 1 barang dalam sekali transaksi sehingga data tersebut perlu dihilangkan karena akan mempengaruhi proses *mining*.

3.1.2.3 Data Preparation

Persiapan data merupakan tahap di mana akan dilakukan pemilihan tabel dan *field* yang akan digunakan dalam proses *mining.* Persiapan data dilakukan dengan sebutan *Preprocessing Data. Preprocessing* merupakan hal yang harus dilakukan dalam proses *data mining*, karena tidak semua data atau atribut data dalam data digunakan dalam proses *data mining*. Proses ini dilakukan agar data yang digunakan sesuai dengan kebutuhan. Adapun tahapan-tahapan *preprocessing* data dalam penelitian ini adalah sebagai berikut :

1. Pemilihan Atribut

   Proses pemilihan atribut atau *selection data* adalah proses di mana atribut data akan dipilih dan diproses sesuai dengan kebutuhan *data mining*. Sebelum melakukan proses pembersihan data akan dilakukan proses *selection* atau pemilihan atribut terlebih dahulu. Karena dari data transaksi di Cv. Bukit Manikam memiliki 9 atribut, sedangkan yang dibutuhkan untuk *data mining* hanya 2 atribut, maka akan dilakukan proses seleksi dengan menyeleksi atribut yang tidak dibutuhkan, proses ini akan memudahkan pada proses pembersihan data nanti. Dalam penelitian ini, 2 atribut yang akan digunakan adalah atribut Nomor Faktur dan Nama Barang. Kedua atribut ini digunakan untuk memenuhi tujuan awal di mana akan dicari pola pembelian pelanggan berdasarkan barang yang dibeli. Atribut Nomor Faktur digunakan untuk membedakan satu transaksi dengan transaksi lainnya, dan atribut Nama Barang digunakan untuk mengetahui barang apa saja yang dibeli dalam satu transaksi. Contoh pemilihan atribut dapat dilihat pada tabel 3.4

Tabel 3. 4 Contoh Pemilihan Atribut

Nomor Faktur	Nama Barang
B16010100001/	Seal Valve Steam Beat Hi-Q
B16010100001/	Ts Gasket Kit Vixion Hi-Q
B16010100001/	Pac.R.Crank Cover Vega Z R
B16010100001/	Pac.Muffler Mio
B16010100001/	Seal Valve Steam Jupiter Mx Hi-Q
B16010100002/	Ts Gasket Kit Mio / Mio Soul
B16010100002/	Pac.R.Crank Cover Vega Z R

3. Pembersihan Data

Proses pembersihan data atau *cleaning data* adalah proses menghilangkan *noise*. *Noise* di sini yaitu data transaksi yang hanya memiliki 1 barang dalam sekali pembelian. Dalam data transaksi ini akan dilakukan pengeleminasian terhadap transaksi yang memiliki jumlah barang kurang dari 2 dalam satu kali transaksinya, karena syarat ini diperlukan dalam *Association Rules* untuk melihat keterhubungan antar 2 barang atau lebih. Setelah melakukan proses pembersihan data, dari data transaksi yang awalnya sebanyak 19 transaksi dengan jumlah *record* 85 di*cleaning* menjadi 17 transaksi dengan jumlah *record* 17. Contoh sebelum dan sesudah pembersihan data dapat dilihat pada tabel 3.5 dan 3.6

Tabel 3. 5 Contoh data yang akan dibersihkan

Nomor Faktur	Nama Barang
B16010100003/	Seal Valve Steam Jupiter Mx Hi-Q
B16010100003/	Visor Beat Injection (Abu-Abu)
B16010100003/	Pac.Muffler Mio
B16010100003/	Ts Gasket Kit Mio / Mio Soul
B16010100003/	Pac.R.Crank Cover Vega Z R
B16010100004/	Pac.Muffler Mio
B16010100005/	Seal Valve Steam Beat Hi-Q
B16010100005/	Pac.Muffler Mio
B16010100005/	Pac.Muffler Beat / Scoopy / Spacy / Vario

Tabel 3. 6 Contoh data yang telah dibersihkan

Nomor Faktur	Nama Barang
B16010100003/	Seal Valve Steam Jupiter Mx Hi-Q
B16010100003/	Visor Beat Injection (Abu-Abu)
B16010100003/	Pac.Muffler Mio
B16010100003/	Ts Gasket Kit Mio / Mio Soul
B16010100003/	Pac.R.Crank Cover Vega Z R
B16010100005/	Seal Valve Steam Beat Hi-Q
B16010100005/	Pac.Muffler Mio
B16010100005/	Pac.Muffler Beat / Scoopy / Spacy / Vario

3.1.2.4 Modelling

Pemodelan merupakan tahap untuk membangun model. Adapun tahapannya sebagai berikut :

1. Memilih Teknik *Modelling*

   Penelitian ini bertujuan untuk mendapatkan informasi mengenai pola pembelian pelanggan yang nantinya akan digunakan oleh pihak Cv. Bukit Manikam sebagai dasar pengambilan keputusan untuk menentukan barang apa saja yang bisa dijadikan sebuah paket penjualan barang. Maka dari itu teknik pemodelan yang digunakan adalah dengan menerapkan *Data mining*, data transaksi akan diolah dengan Metode *Association Rule* dengan menggunakan algoritma *CT-Pro* untuk menemukan pola-pola pembelian barang yang sering dibeli oleh pelanggan.

2. Membangun Model

   Setelah memilih model yang digunakan tahap selanjutnya adalah membangun model sesuai dengan yang ditentukan berikut merupakan tahapan pembangunan model menggunakan metode *association rules*. Metodologi dasar Association *Rule* terbagi menjadi dua tahap, yaitu mencari *frequent itemset* dan mendefinisikan *condition* dan *result*. Algoritma yang digunakan dalam penelitian ini adalah algoritma *CT-Pro*. Berikut ini adalah langkah-langkah proses pengerjaan algoritma *CT-Pro* :

1. Mencari *Frequent Itemset*

Langkah pertama adalah menghitung jumlah kemunculan tiap barang, sebelumnya akan ditentukan nilai minimum *support* untuk melihat batasan terendah munculnya barang. Dalam menentukan nilai *minimum support* dapat berubah sesuai kebutuhan *user*. Hal ini juga diperkuat dengan pernyataan pakar *data mining* Daniel T. Larose, yang menerangkan bahwa *user* bebas menentukan nilai minimum *support* (*minsup*) dan minimum *confidence* (*mincof*) sesuai kebutuhan[13]. *Minimum supporrt* yang diambil pada penelitian ini adalah 4, maka batasan barang yang muncul harus sebanyak >= 4 kali. Maka pada data transaksi, masing-masing barang dihitung jumlah kemunculan berdasarkan barangnya. Hasil dari penghitungan kemunculan barang dapat dilihat pada tabel 3.7.

Tabel 3. 7 Hasi Jumlah Kemunculan *Item*

Nama Barang	*Support Count*
Pac.Muffler Mio	11
Pac.R.Crank Cover Vega Z R	10
Seal Valve Steam Beat Hi-Q	10
Ts Gasket Kit Mio / Mio Soul	8
Ts Gasket Kit Vixion Hi-Q	8
Seal Valve Steam Jupiter Mx Hi-Q	7
Pac.Cyl Jupiter Mx	6
Pac.Muffler Beat / Scoopy / Spacy / Vario	5
Pac.Cyl Mio / Mio Soul	4
Mirror Assy Aero Yamaha (Hitam)	4
Spakbor Dpn Vario Techno 125 (Hitam)	3
Mirror Assy Elipso Hnd (White)	3
Spakbor Dpn Mio Soul (Hitam)	2
Leg Shield Container Vario (Hitam)	1
Visor Beat Injection (Abu-Abu)	1

Karena *minimum support*-nya 4, maka barang yang tidak memenuhi batas *minimum support* akan dieliminasi. Berikut adalah tabel hasil eliminasi kemunculan barang yang tidak memenuhi *minimun suport* yang diacu pada tabel 3.8.

Tabel 3. 8 Hasil Eliminasi Jumlah Kemunculan *Item*

Nama Barang	*Support Count*
Pac.Muffler Mio	11
Seal Valve Steam Beat Hi-Q	10
Pac.R.Crank Cover Vega Z R	10
Ts Gasket Kit Vixion Hi-Q	8
Ts Gasket Kit Mio / Mio Soul	8
Seal Valve Steam Jupiter Mx Hi-Q	7
Pac.Cyl Jupiter Mx	6
Pac.Muffler Beat / Scoopy / Spacy / Vario	5
Pac.Cyl Mio / Mio Soul	4
Mirror Assy Aero Yamaha (Hitam)	4

Begitu juga pada data transaksi untuk barang yang *frequent*-nya tidak memenuhi batas minimum *support* dieliminasi dan hasilnya dapat dilihat pada tabel 3.9

Tabel 3. 9 Data Transaksi yang Frequent

Nomor Faktur	Nama Barang
B16010100001/	Seal Valve Steam Beat Hi-Q
B16010100001/	Ts Gasket Kit Vixion Hi-Q
B16010100001/	Pac.R.Crank Cover Vega Z R
B16010100001/	Pac.Muffler Mio
B16010100001/	Seal Valve Steam Jupiter Mx Hi-Q
B16010100002/	Ts Gasket Kit Mio / Mio Soul

Nomor Faktur	Nama Barang
B16010200011/	Seal Valve Steam Jupiter Mx Hi-Q
B16010100011/	Pac.Cyl Jupiter Mx
B16010200011/	Seal Valve Steam Beat Hi-Q
B16010200011/	Pac.R.Crank Cover Vega Z R
B16010200013/	Ts Gasket Kit Mio / Mio Soul
B16010200013/	Ts Gasket Kit Vixion Hi-Q

Nomor Faktur	Nama Barang
B16010100002/	Pac.R.Crank Cover Vega Z R
B16010100002/	Ts Gasket Kit Vixion Hi-Q
B16010100002/	Seal Valve Steam Jupiter Mx Hi-Q
B16010100003/	Seal Valve Steam Jupiter Mx Hi-Q
B16010100003/	Pac.Muffler Mio
B16010100003/	Ts Gasket Kit Mio / Mio Soul
B16010100003/	Pac.R.Crank Cover Vega Z R
B16010100005/	Seal Valve Steam Beat Hi-Q
B16010100005/	Pac.Muffler Mio
B16010100005/	Pac.Muffler Beat / Scoopy / Spacy / Vario
B16010100006/	Seal Valve Steam Beat Hi-Q
B16010100006/	Pac.Cyl Mio / Mio Soul
B16010100006/	Pac.R.Crank Cover Vega Z R
B16010100006/	Mirror Assy Aero Yamaha (Hitam)
B16010100007/	Pac.R.Crank Cover Vega Z R
B16010100007/	Pac.Muffler Mio
B16010100007/	Ts Gasket Kit Vixion Hi-Q
B16010200008/	Pac.Muffler Beat / Scoopy / Spacy / Vario
B16010200008/	Seal Valve Steam Beat Hi-Q
B16010200008/	Pac.R.Crank Cover Vega Z R
B16010200008/	Pac.Cyl Jupiter Mx
B16010200008/	Pac.Muffler Mio
B16010200009/	Seal Valve Steam Beat Hi-Q
B16010200009/	Pac.R.Crank Cover Vega Z R
B16010200010/	Pac.Cyl Jupiter Mx
B16010200010/	Seal Valve Steam Beat Hi-Q
B16010200010/	Ts Gasket Kit Mio / Mio Soul
B16010200010/	Ts Gasket Kit Vixion Hi-Q
B16010200010/	Pac.Muffler Mio

Nomor Faktur	Nama Barang
B16010200013/	Seal Valve Steam Jupiter Mx Hi-Q
B16010300014/	Ts Gasket Kit Vixion Hi-Q
B16010300014/	Pac.Cyl Jupiter Mx
B16010300014/	Pac.Muffler Beat / Scoopy / Spacy / Vario
B16010300014/	Pac.Muffler Mio
B16010300015/	Ts Gasket Kit Vixion Hi-Q
B16010300015/	Pac.Muffler Mio
B16010300015/	Seal Valve Steam Jupiter Mx Hi-Q
B16010300015/	Ts Gasket Kit Mio / Mio Soul
B16010300015/	Mirror Assy Aero Yamaha (Hitam)
B16010300016/	Pac.Muffler Mio
B16010300016/	Seal Valve Steam Beat Hi-Q
B16010300016/	Pac.Cyl Mio / Mio Soul
B16010300016/	Pac.Cyl Jupiter Mx
B16010300016/	Mirror Assy Aero Yamaha (Hitam)
B16010300016/	Pac.R.Crank Cover Vega Z R
B16010300016/	Pac.Muffler Beat / Scoopy / Spacy / Vario
B16010300017/	Pac.Cyl Mio / Mio Soul
B16010300017/	Pac.Cyl Jupiter Mx
B16010300017/	Seal Valve Steam Beat Hi-Q
B16010300017/	Pac.Muffler Mio
B16010300018/	Seal Valve Steam Jupiter Mx Hi-Q
B16010300018/	Ts Gasket Kit Mio / Mio Soul
B16010300018/	Ts Gasket Kit Vixion Hi-Q
B16010300019/	Pac.Cyl Mio / Mio Soul
B16010300019/	Seal Valve Steam Beat Hi-Q
B16010300019/	Pac.Muffler Mio
B16010300019/	Ts Gasket Kit Mio / Mio Soul
B16010300019/	Pac.R.Crank Cover Vega Z R

Nomor Faktur	Nama Barang
B16010200010/	Pac.Muffler Beat / Scoopy / Spacy / Vario
B16010100011/	Ts Gasket Kit Mio / Mio Soul

Nomor Faktur	Nama Barang
B16010300019/	Mirror Assy Aero Yamaha (Hitam)

Selanjutnya untuk setiap barang diberikan *id global item* yaitu penomoran secara *ascending* dari jumlah *support count* terbesar ke terkecil. Berikut adalah tabel *global item table* yang dapat dilihat pada tabel 3.10

Tabel 3. 10 *Global Item Table*

Nama Barang	*Support Count*	*Id global item*
Pac.Muffler Mio	11	1
Pac.R.Crank Cover Vega Z R	10	2
Seal Valve Steam Beat Hi-Q	10	3
Ts Gasket Kit Mio / Mio Soul	8	4
Ts Gasket Kit Vixion Hi-Q	8	5
Seal Valve Steam Jupiter Mx Hi-Q	7	6
Pac.Cyl Jupiter Mx	6	7
Pac.Muffler Beat / Scoopy / Spacy / Vario	5	8
Pac.Cyl Mio / Mio Soul	4	9
Mirror Assy Aero Yamaha (Hitam)	4	10

Selanjutnya dilakukan *mapping* data yaitu memetakan data transaksi dari tabel 3.9 terhadap *global item table* pada tabel 3.10, di mana nama barang pada tabel 3.9 digantikan oleh *id global item* sesuai dengan *id global item* yang ada pada tabel 3.10. Data transaksi kemudian diurutkan dari *id global item* terkecil ke terbesar dari setiap nomor fakturnya. Berikut adalah hasil *mapping* data transaksi tabel 3.9 yang dapat dilihat pada tabel 3.11.

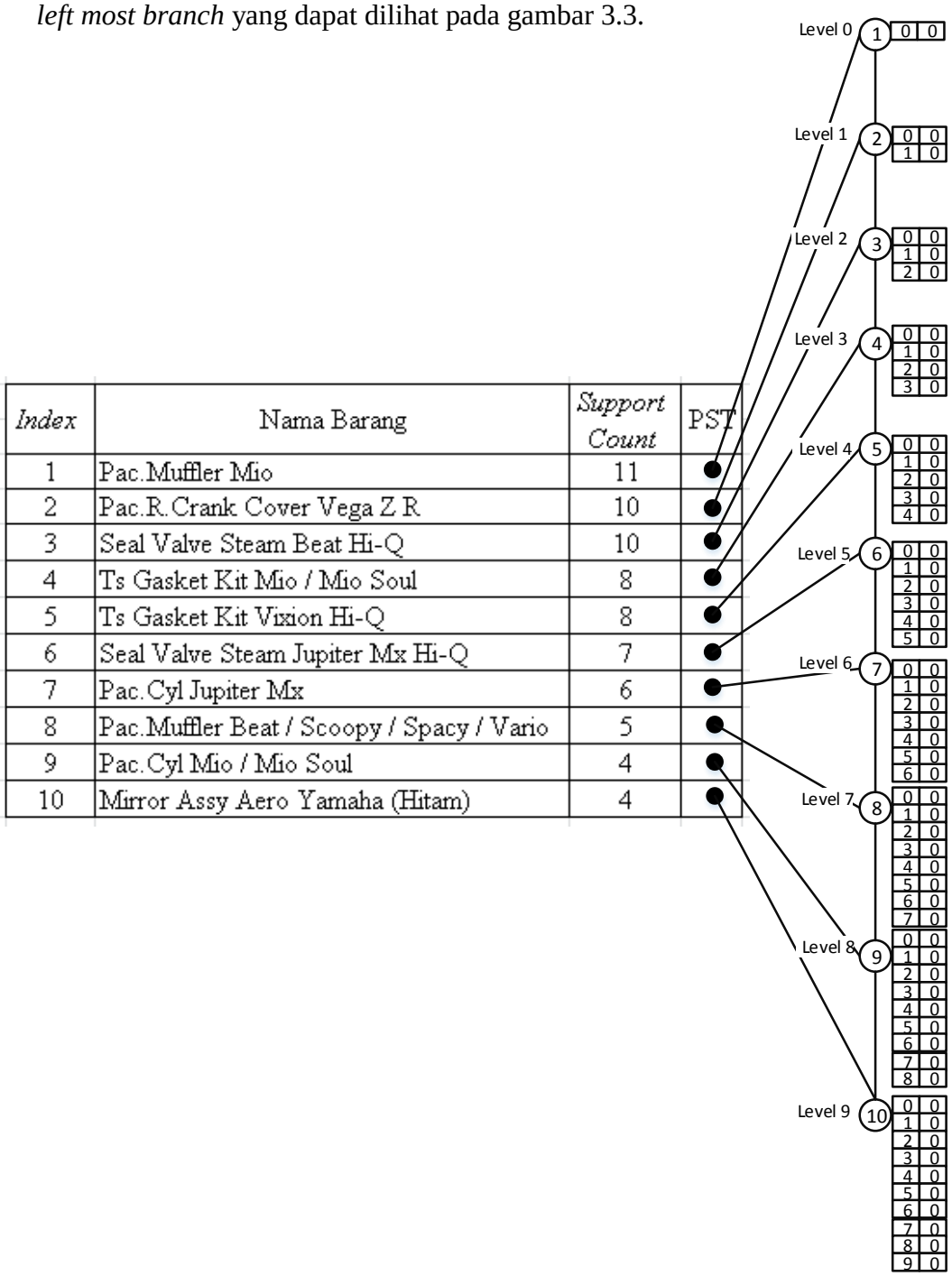
Tabel 3. 11 Tabel *Mapping*

Nomor Faktur	*Id Global Item*
B16010100001/	1, 2, 3, 5, 6
B16010100002/	2, 4, 5, 6
B16010100003/	1, 2, 4, 6
B16010100005/	1, 3, 8
B16010100006/	2, 3, 9, 10
B16010100007/	1, 2, 5
B16010200008/	1, 2, 3, 7, 8
B16010200009/	2, 3
B16010200010/	1, 3, 4, 5, 7, 8
B16010200011/	2, 3, 4, 6, 7
B16010200013/	4, 5, 6
B16010300014/	1, 5, 7, 8
B16010300015/	1, 4, 5, 6, 10
B16010300016/	1, 2, 3, 7, 8, 9, 10
B16010300017/	1, 3, 7, 9
B16010300018/	4, 5, 6
B16010300019/	1, 2, 3, 4, 9, 10

2. Membuat *CFP-Tree*

Setelah d*item*ukan *frequent itemset* kemudian dilakukan pembentukan *CFP-Tree* . Pada pembentukan *Global CFP-Tree* dimulai dengan membuat *left most branch* yang dapat dilihat pada gambar 3.3.

Index	Nama Barang	*Support Count*	PST
1	Pac.Muffler Mio	11	
2	Pac.R.Crank Cover Vega Z R	10	
3	Seal Valve Steam Beat Hi-Q	10	
4	Ts Gasket Kit Mio / Mio Soul	8	
5	Ts Gasket Kit Vixion Hi-Q	8	
6	Seal Valve Steam Jupiter Mx Hi-Q	7	
7	Pac.Cyl Jupiter Mx	6	
8	Pac.Muffler Beat / Scoopy / Spacy / Vario	5	
9	Pac.Cyl Mio / Mio Soul	4	
10	Mirror Assy Aero Yamaha (Hitam)	4	



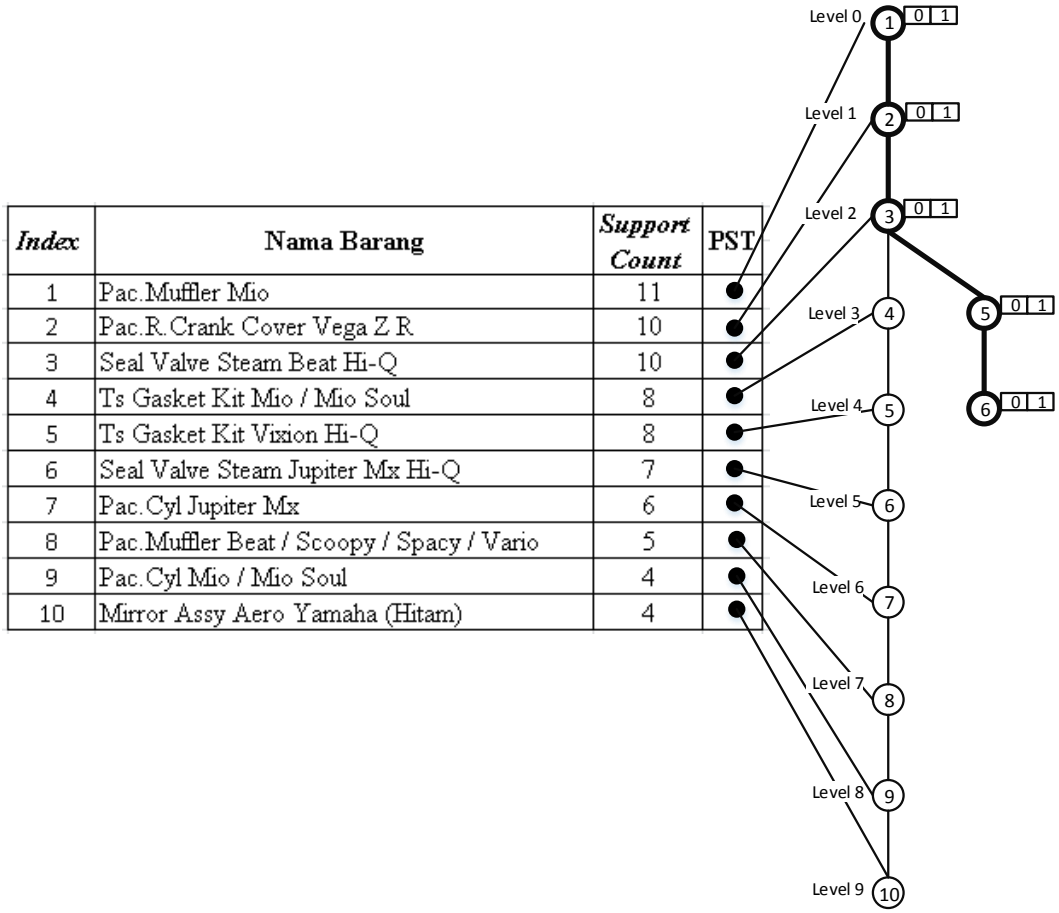
Gambar 3. 3 *Left Most Branch*

Tahap selanjutnya yaitu memasukkan setiap data transaksi ke dalam *tree*. Berikut langkah langkah membuat *CFP-Tree* :

1. Dimulai dari *root* yaitu barang yang pertama kali muncul dari nomor faktur pertama yang telah di-*mapping*, barang tersebut akan menjadi *currnode*.
2. Tambahkan *count* = 1 pada barang yang pertama muncul sesuai dengan levelnya.
3. Jika *currnode* sudah memiliki anak, dan barang yang muncul sama dengan anak tersebut maka tambahkan *count* = 1 pada level yang sama *currnode*.
4. Jika *currnode* sudah memiliki anak, tetapi data yang muncul berbeda dengan anak dari *currnode*, buat *node* baru dan tambahkan *count* = 1 pada level yang sama dengan *currnode*, hubungkan *node* baru tersebut pada *currnode* melalui *node link*. Berikut pembentukan *CFP-Tree* :

a. Nomor Faktur B16010100001/ dengan *index* : 1, 2, 3, 5 dan 6 merupakan data awal yang akan diproses. Pembentukan *CFP-Tree* yang dapat dilihat pada gambar 3.4

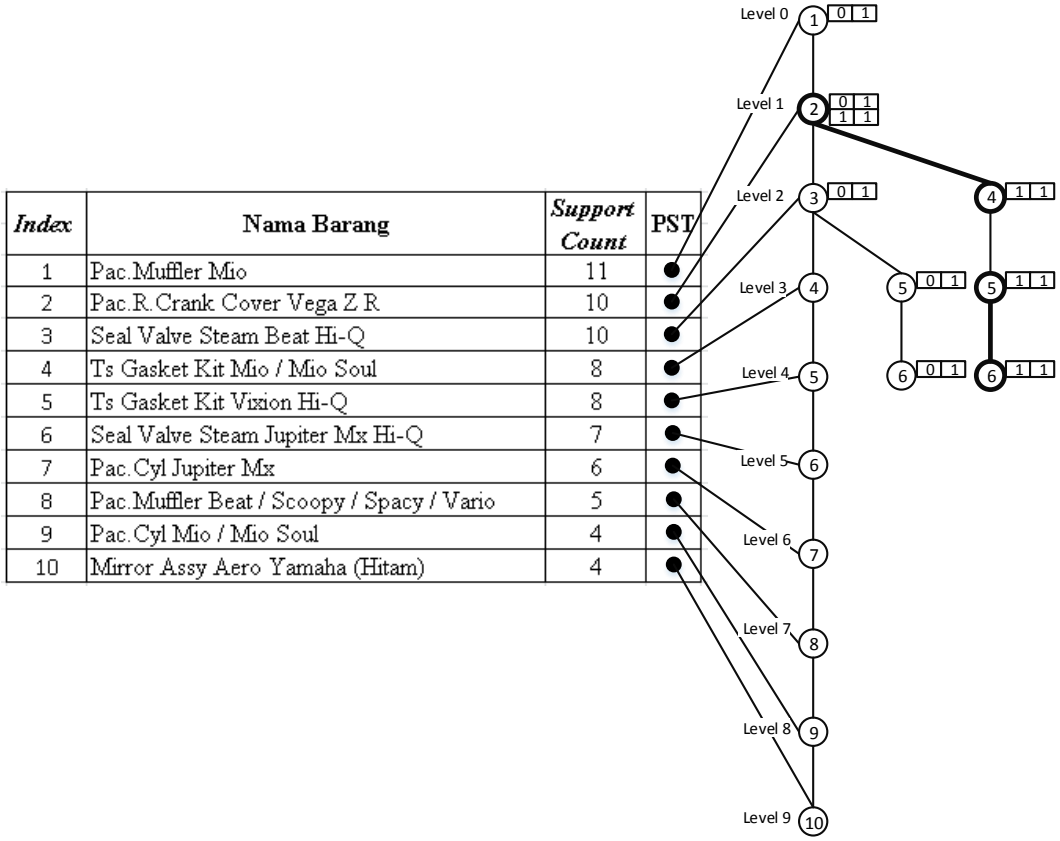
Index	Nama Barang	*Support Count*	PST
1	Pac.Muffler Mio	11	
2	Pac.R.Crank Cover Vega Z R	10	
3	Seal Valve Steam Beat Hi-Q	10	
4	Ts Gasket Kit Mio / Mio Soul	8	
5	Ts Gasket Kit Vixion Hi-Q	8	
6	Seal Valve Steam Jupiter Mx Hi-Q	7	
7	Pac.Cyl Jupiter Mx	6	
8	Pac.Muffler Beat / Scoopy / Spacy / Vario	5	
9	Pac.Cyl Mio / Mio Soul	4	
10	Mirror Assy Aero Yamaha (Hitam)	4	



Gambar 3. 4 *CFP-Tree* Nomor Faktur B16010100001/

b. Nomor Faktur B16010100002/ dengan *index* : 2, 4, 5 dan 6 akan membentuk *CFP-Tree* yang dapat dilihat pada gambar 3.5

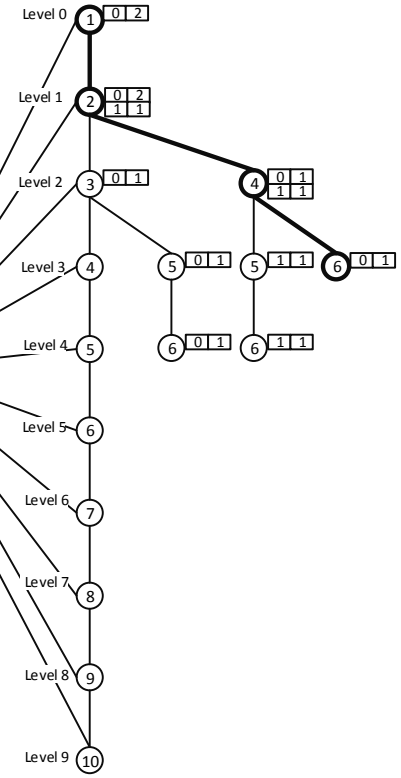
Index	Nama Barang	*Support Count*	PST
1	Pac.Muffler Mio	11	
2	Pac.R.Crank Cover Vega Z R	10	
3	Seal Valve Steam Beat Hi-Q	10	
4	Ts Gasket Kit Mio / Mio Soul	8	
5	Ts Gasket Kit Vixion Hi-Q	8	
6	Seal Valve Steam Jupiter Mx Hi-Q	7	
7	Pac.Cyl Jupiter Mx	6	
8	Pac.Muffler Beat / Scoopy / Spacy / Vario	5	
9	Pac.Cyl Mio / Mio Soul	4	
10	Mirror Assy Aero Yamaha (Hitam)	4	



Gambar 3. 5 *CFP-Tree* Nomor Faktur B16010100002/

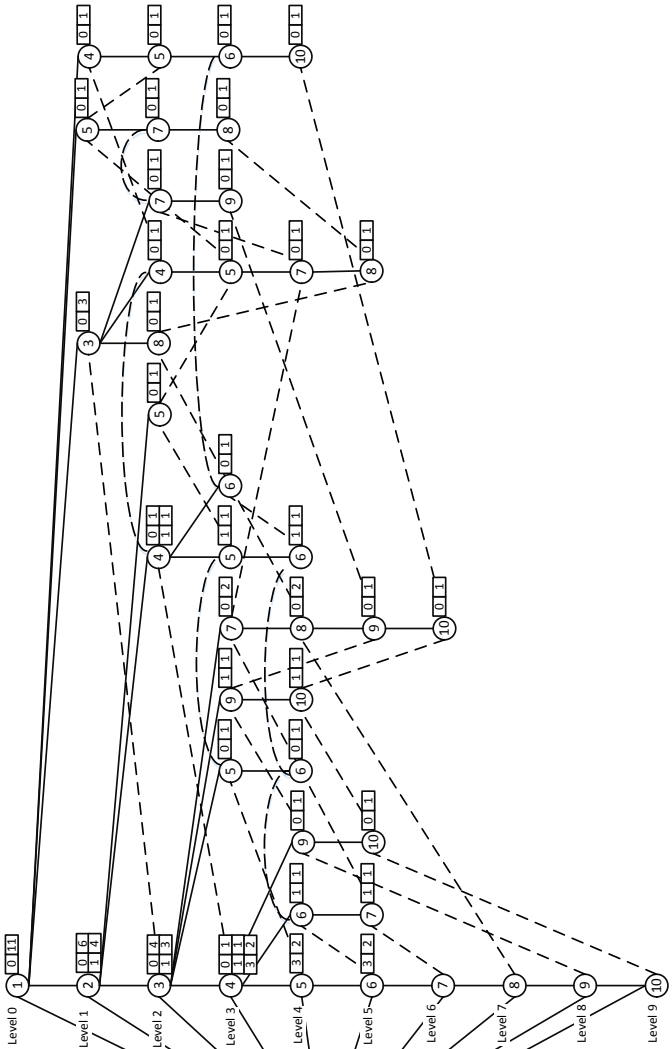
c. Nomor Faktur B16010100003/ dengan *index* : 1, 2, 4 dan 6 akan membentuk *CFP-Tree* yang dapat dilihat pada gambar 3.6

Index	Nama Barang	*Support Count*	PST
1	Pac.Muffler Mio	11	
2	Pac.R.Crank Cover Vega Z R	10	
3	Seal Valve Steam Beat Hi-Q	10	
4	Ts Gasket Kit Mio / Mio Soul	8	
5	Ts Gasket Kit Vixion Hi-Q	8	
6	Seal Valve Steam Jupiter Mx Hi-Q	7	
7	Pac.Cyl Jupiter Mx	6	
8	Pac.Muffler Beat / Scoopy / Spacy / Vario	5	
9	Pac.Cyl Mio / Mio Soul	4	
10	Mirror Assy Aero Yamaha (Hitam)	4	



Gambar 3. 6 *CFP-Tree* Nomor Faktur B16010100003/

Berikut adalah *Global CFP-Tree* yang terbentuk dari seluruh transaksi, yang diacu pada gambar 3.7.



Index	Nama Barang	*Support Count*	PST
1	Pac.Muffler Mio	11	
2	Pac.R.Crank Cover Vega Z R	10	
3	Seal Valve Steam Beat Hi-Q	10	
4	Ts Gasket Kit Mio / Mio Soul	8	
5	Ts Gasket Kit Vizion Hi-Q	8	
6	Seal Valve Steam Jupiter Mx Hi-Q	7	
7	Pac.Cyl Jupiter Mx	6	
8	Pac.Muffler Beat / Scoopy / Spacy / Vario	5	
9	Pac.Cyl Mio / Mio Soul	4	
10	Mirror Assy Aero Yamaha (Hitam)	4	

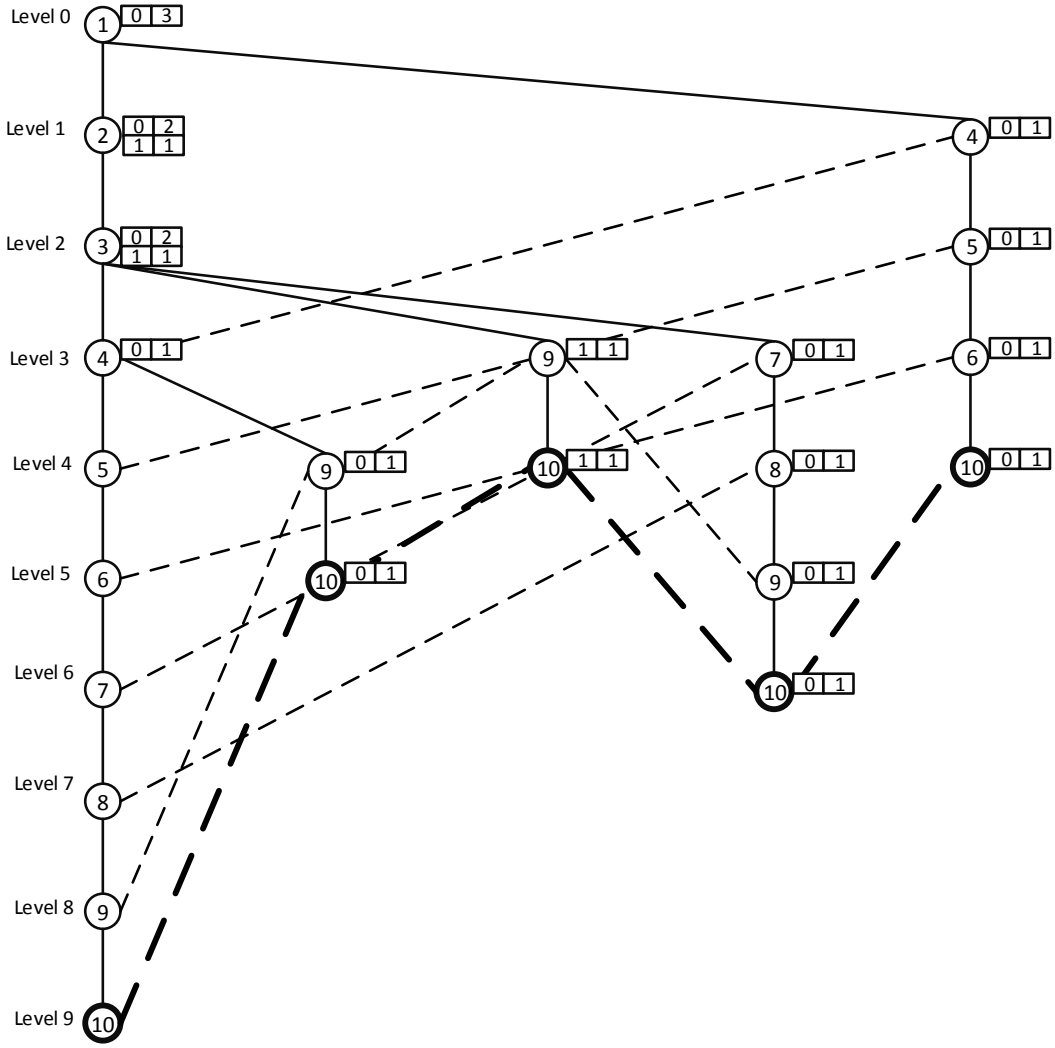
Gambar 3. 7 Global CFP-Tree

3. Melakukan penggalian *frequent patterns*

Setelah tahap pembangunan *global CFP-Tree* dari sekumpulan data transaksi, tahap selanjutnya adalah membangun *local* CFP-*Tree* untuk mencari *frequent itemset*. Berikut adalah langkah-langkah dari algoritma CT-Pro :

1. Lakukan penelusuran *node* yang berkaitan untuk mencari *node* yang memenuhi minimum *support count*, di mana pencarian dimulai dari *support count* terkecil sampai dengan *support count* terbesar karena *CT-Pro* bekerja *Bottom-Up* pada *Global CFP-Tree*. Dalam pencarian *Local Frequent Pattern Tree* berisikan lintasan *prefix* dan *suffic pattern* (pola akhiran). Untuk setiap pencarian *suffix* dilakukan dengan mencari lintasan *prefix* dari *suffix* yang dicari.
2. Selanjutnya masukan setiap *item* yang memenuhi *minimum support* ke *local item table.*
3. Berikan *index* baru pada *node* yang ada di *local item table* secara *ascending* dari *support count* terbesar ke terkecil.
4. Setelah *local item table* terbentuk lakukan pemetaan *index* baru terhadap *node* di *local item table*.
5. Setelah dilakukan pemetaan lakukan pembentukan *Local CFP-Tree* dengan aturan yang sama saat pembentukan *Global CFP-Tree*.
6. Lakukan pembentukan *Local CFP-Tree projection* yaitu di mana *index* (x) akan menjadi *root* dan *index* yang ada di *local CFP-Tree* menjadi anaknya, lakukan penelusuran di *local CFP-Tree* secara *bottom-up* dari *node* ke-i dengan menghitung *count* dari setiap *node*, jika *node* k memenuhi minimum *support* jadikan k menjadi anak dari *node* ke-i. Berikut penggalian *frequent pattern*nya :

a. Pencarian *local Frequent Pattern Tree* untuk *index* 10. Lakukan penelusuran secara *bottom-up* pada *Global CFP-Tree* yang hanya berakhiran *index* 10, untuk jelasnya dapat dilihat pada gambar 3.8



Gambar 3. 8 CFP-*Tree* Berakhiran *index* 10

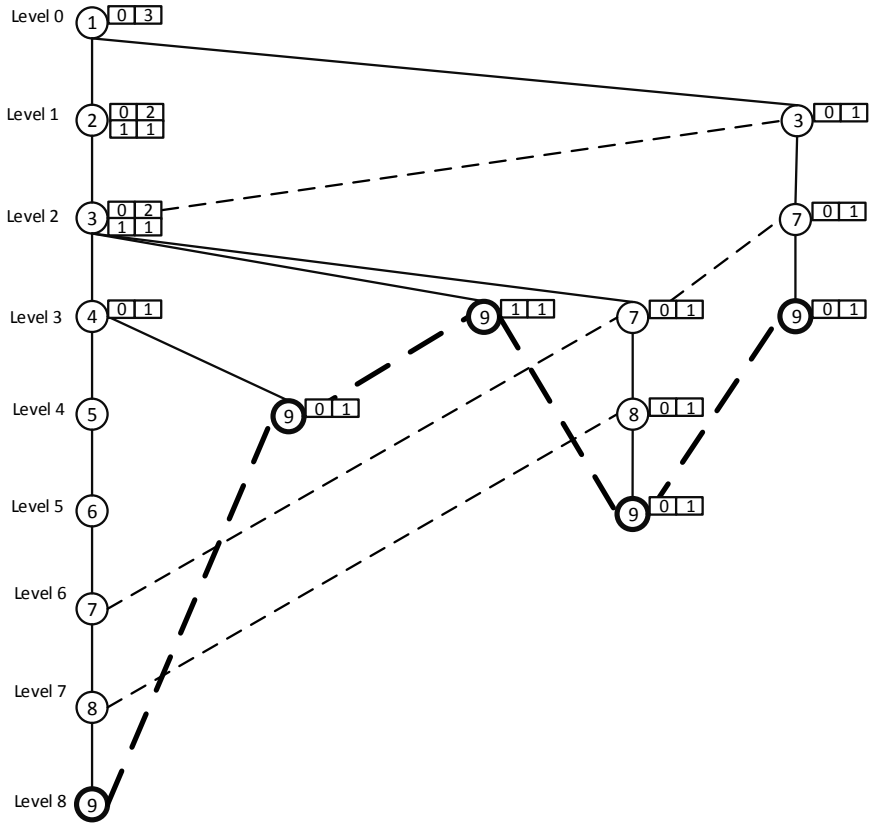
Berikut hasil penelusuran untuk *index* 10 pada *Global CFP-Tree* yang dapat dilihat pada tabel 3.12

Tabel 3. 12 Penelusuran CFP-*Tree* *index* 10

No	Lintasan
1	10,9,4,3,2,1
2	10,9,3,2
3	10,9,8,7,3,2,1
4	10,6,5,4,1

Dari tabel 3.12 didapat jumlah *support count* dari setiap *node* yaitu 1(3), 2(3), 3(3), 4(2), 5(1), 6(1), 7(1), 8(1), 9(3) Pada *node* tersebut tidak ada yang memenuhi *minimum support* 4, maka tidak terbentuk *local item table* sehingga untuk *local CFP-Tree* dan *frequent pattern*nya juga tidak akan terbentuk.

b. Pencarian *local Frequent Pattern Tree* untuk *index* 9. Lakukan penelusuran secara *bottom-up* pada *Global CFP-Tree* yang hanya berakhiran *index* 9, untuk jelasnya dapat dilihat pada gambar 3.9



Gambar 3. 9 CFP-*Tree* Berakhiran *index* 9

Berikut hasil penelusuran untuk *index* 9 pada *Global CFP-Tree* yang dapat dilihat pada tabel 3.13

Tabel 3. 13 Penelusuran CFP-*Tree* *index* 9

No	Lintasan
1	9,4,3,2,1
2	9,3,2
3	9,8,7,3,2,1
4	9,7,3,1

Dari tabel 3.13 didapat jumlah *support count* dari setiap *node* yaitu. 1(3), 2(3), 3(4), 4(1), 7(2), 8(1). Karena *minimum supportnya* 4, maka *node* yang memenuhi *minimum support* adalah 3(4). Selanjutnya *node* 3(4) di masukan ke *local item table* dengan diberikan *index* yaitu penomoran secara *ascending* dari jumlah *support count* terbesar ke terkecil. Berikut merupakan *local item table* yang terbentuk, dapat dilihat pada tabel 3.14

Tabel 3. 14 *Local Item Table*

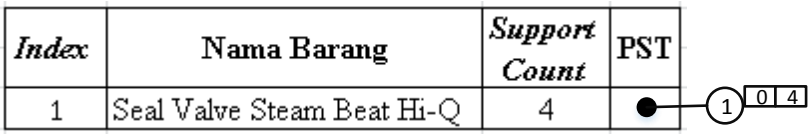
Index Local	***Index Global***	**Nama Barang**	***Support Count***
1	3	Seal Valve Steam Beat Hi-Q	4

Setelah *local item* tabel terbentuk, maka tahapan selanjutnya adalah memetakan *node* yang memenuhi nilai minimum *support* dengan nomor *index* yang sudah diberikan. Berikut hasil *mapping* yang dapat dilihat pada tabel 3.15

Tabel 3. 15 Hasil Mapping Local CFP-*Tree*

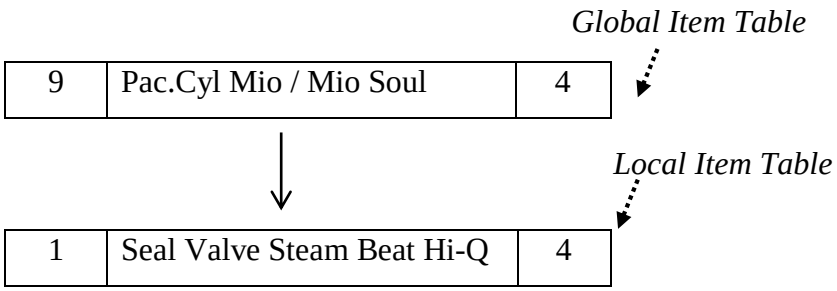
No	Lintasan
1	1
2	1
3	1
4	1

Setelah hasil *mapping* didapat, selanjutnya akan dimasukan ke *Local CFP-Tree*. Aturan pembentukan *Local CFP-Tree* sama dengan pembentukan *Global CFP-Tree*, yang membedakan adalah pada *Global CFP-Tree* yang digunakan dalam pembentukan *tree*-nya adalah *Global Item* tabel yang terbentuk dari *Global Item table*, sedangkan pada *Local CFP-Tree* yang digunakan dalam pembentukan *tree*-nya adalah *Local item* tabel yang terbentuk dari *Local Frequent item*. Berikut adalah *Local CFP-Tree* dari *index* 9 yang dapat dilihat pada gambar 3.10



Gambar 3. 10 *Local CFP-Tree* Untuk *index* 9

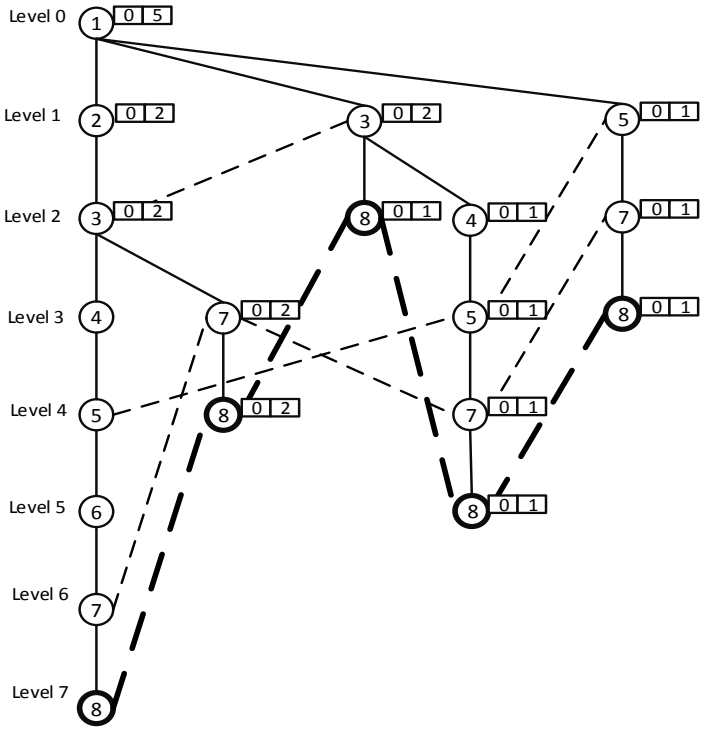
Setelah *Local CFP-Tree index* 9 terbentuk, langkah selanjutnya adalah membuat *Local CFP-Tree projection* di mana *index* 9 akan menjadi *root* dan setiap *index* di *Local CFP-Tree* akan menjadi anak dari *index* 9. Hasilnya dapat dilihat pada gambar 3.11



Gambar 3. 11 *Local CFP-Tree Projection* 9

Dari *local CFP-Tree projection* maka didapatkan *item frequent* untuk *index* 9 dengan nama barang Pac.Cyl Mio / Mio Soul adalah (9,1:4).

c. Pencarian *local Frequent Pattern Tree* untuk *index* 8. Lakukan penelusuran secara *bottom-up* pada *Global CFP-Tree* yang hanya berakhiran *index* 8, untuk jelasnya dapat dilihat pada gambar 3.12



Gambar 3. 12 CFP-*Tree* Berakhiran *index* 8

Berikut hasil penelusuran untuk *index* 8 pada *Global CFP-Tree* yang dapat dilihat pada tabel 3.16

Tabel 3. 16 Penelusuran CFP-*Tree* *index* 8

No	Lintasan
1	8,7,3,2,1
2	8,7,3,2,1
3	8,3,1
4	8,7,5,4,3,1
5	8,7,5,1

Dari tabel 3.16 didapat jumlah *support count* dari setiap *node* yaitu. 1(5), 2(2), 3(4), 5(2),7(4). Karena *minimum supportnya* 4, maka *node* yang memenuhi *minimum support* adalah 1(5), 3(4), 7(4) Selanjutnya *node* 1(5), 3(4), 7(4) dimasukan ke *Local item Table*, dimulai dari *support count* terbesar ke terkecil dengan *index* baru yang dapat dilihat pada tabel 3.17

Tabel 3. 17 *Local Item Table*

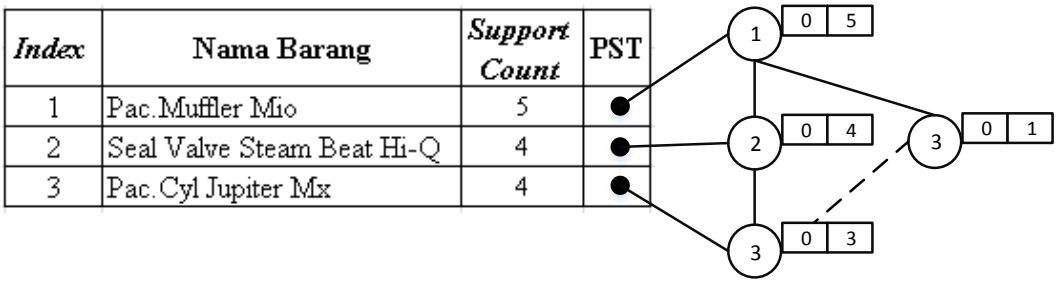
Index Local	*Index Global*	Nama Barang	*Support Count*
1	1	Pac.Muffler Mio	5
2	3	Seal Valve Steam Beat Hi-Q	4
3	7	Pac.Cyl Jupiter Mx	4

Setelah *local item* tabel terbentuk, maka tahapan selanjutnya adalah memetakan *node* yang memenuhi nilai minimum *support* dengan nomor *index* yang sudah diberikan. Berikut hasil *mapping* yang dapat dilihat pada tabel 3.18

Tabel 3. 18 Hasil Mapping Local CFP-*Tree*

No	Lintasan
1	1,2,3
2	1,2,3
3	1,2
4	1,2,3
5	1,3

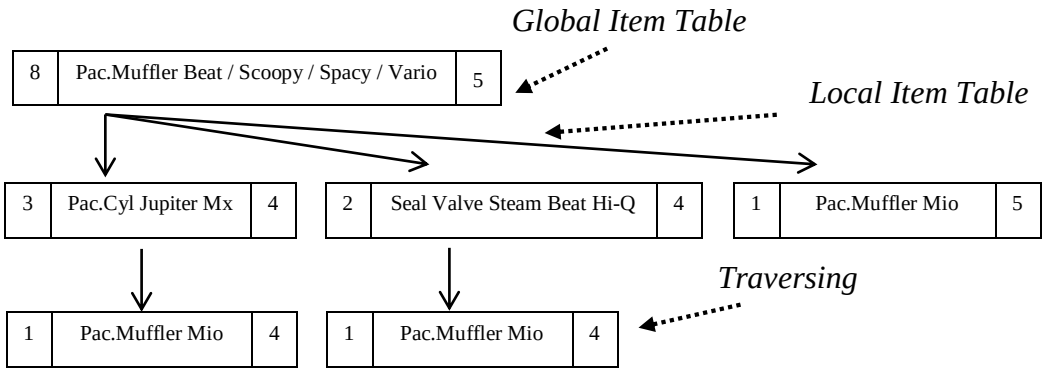
Setelah hasil *mapping* didapat, maka selanjutnya akan dimasukan ke *Local CFPTree*. Aturan pembentukan *Local CFP-Tree* sama dengan pembentukan *Global CFP-Tree*, yang membedakan adalah pada *Global CFP-Tree* yang digunakan dalam pembentukan *tree*-nya adalah *Global Item* tabel yang terbentuk dari *Global Item table*, sedangkan pada *Local CFP-Tree* yang digunakan dalam pembentukan *tree*-nya adalah *Local item* tabel yang terbentuk dari *Local Frequent item*. Berikut adalah *Local CFP-Tree* dari *index* 8 yang dapat dilihat pada gambar 3.13



Index	Nama Barang	*Support Count*	PST
1	Pac.Muffler Mio	5	
2	Seal Valve Steam Beat Hi-Q	4	
3	Pac.Cyl Jupiter Mx	4	

Gambar 3. 13 *Local CFP-Tree* Untuk *index* 8

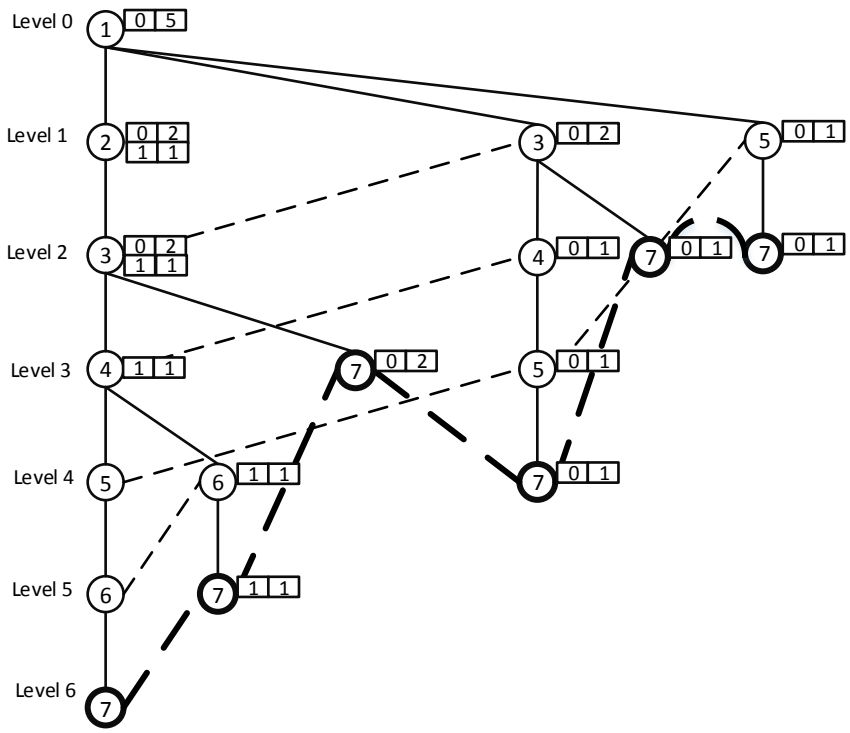
Setelah *Local CFP-Tree index* 8 terbentuk, langkah selanjutnya adalah membuat *Local CFP-Tree projection* di mana *index* 8 akan menjadi *root* dan setiap *index* di *Local CFP-Tree* akan menjadi anak dari *index* 8. Untuk menemukan *traversing*, lakukan penelusuran di *Local CFP-Tree index* 8 secara *bottom-up*, hasilnya dapat dilihat pada gambar 3.14



Gambar 3. 14 *Local CFP-Tree Projection* 8

Dari *local CFP-Tree projection* maka didapatkan *item frequent* untuk *index* 8 dengan nama barang Pac.Muffler Beat / Scoopy / Spacy / Vario adalah (8,3;4), (8,3,1;4), (8,2;4), (8,2,1;4) dan (8,1;5).

d. Pencarian *local Frequent Pattern Tree* untuk *index* 7. Lakukan penelusuran secara *bottom-up* pada *Global CFP-Tree* yang hanya berakhiran *index* 7, untuk jelasnya dapat dilihat pada gambar 3.15



Gambar 3. 15 CFP-*Tree* Berakhiran *index* 7

Berikut hasil penelusuran untuk *index* 7 pada *Global CFP-Tree* yang dapat dilihat pada tabel 3.19

Tabel 3. 19 Penelusuran CFP-*Tree* *index* 7

No	Lintasan
1	7,6,4,3,2
2	7,3,2,1
3	7,3,2,1
4	7,5,4,3,1
5	7,3,1
6	7,5,1

Dari tabel 3.19 didapat jumlah *support count* dari setiap *node* yaitu. 1(5), 2(3), 3(5), 4(2), 5(2), 6(1). Karena *minimum support-nya* 4, maka *node* yang memenuhi *minimum support* adalah 1(5) dan 3(5) Selanjutnya *node* 1(5) dan 3(5) dimasukan ke *Local item Table*, dimulai dari *support count* terbesar ke terkecil dengan *index* baru yang dapat dilihat pada tabel 3.20

Tabel 3. 20 *Local Item Table*

Index Local	*Index Global*	Nama Barang	*Support Count*
1	1	Pac.Muffler Mio	5
2	3	Seal Valve Steam Beat Hi-Q	5

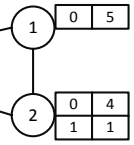
Setelah *local item* tabel terbentuk, maka tahapan selanjutnya adalah memetakan *node* yang memenuhi nilai minimum *support* dengan nomor *index* yang sudah diberikan. Berikut hasil *mapping* yang dapat dilihat pada tabel 3.21

Tabel 3. 21 Hasil Mapping Local CFP-*Tree*

No	Lintasan
1	2
2	1,2
3	1,2
4	1,2
5	1,2
6	1

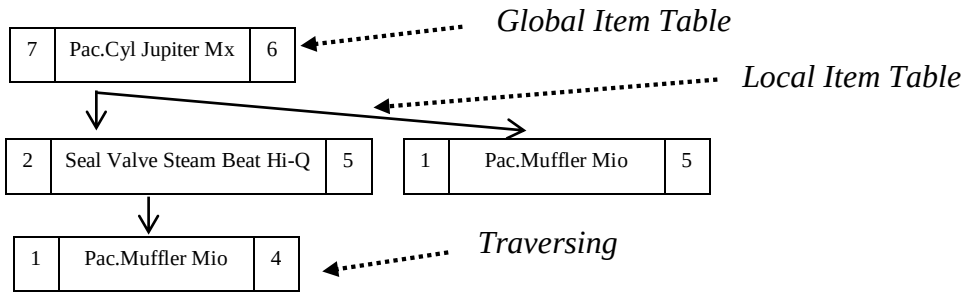
Setelah hasil *mapping* didapat, maka selanjutnya akan dimasukan ke *Local CFPTree*. Aturan pembentukan *Local CFP-Tree* sama dengan pembentukan *Global CFP-Tree*, yang membedakan adalah pada *Global CFP-Tree* yang digunakan dalam pembentukan *tree*-nya adalah *Global Item* tabel yang terbentuk dari *Global Item table*, sedangkan pada *Local CFP-Tree* yang digunakan dalam pembentukan *tree*-nya adalah *Local item* tabel yang terbentuk dari *Local Frequent item*. Berikut adalah *Local CFP-Tree* dari *index* 7 yang dapat dilihat pada gambar 3.16

Index	Nama Barang	*Support Count*	PST
1	Pac.Muffler Mio	5	
2	Seal Valve Steam Beat Hi-Q	5	



Gambar 3. 16 *Local CFP-Tree* Untuk *index* 7

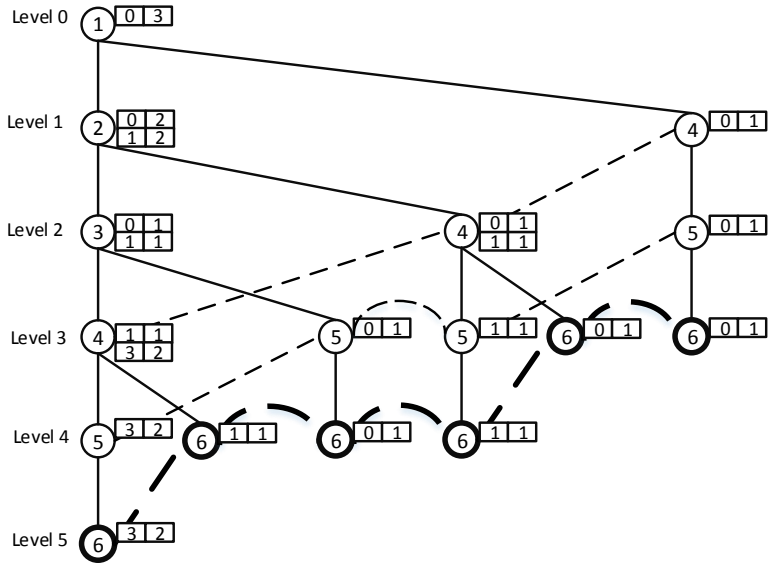
Setelah *Local CFP-Tree index* 7 terbentuk, langkah selanjutnya adalah membuat *Local CFP-Tree projection* di mana *index* 7 akan menjadi *root* dan setiap *index* di *Local CFP-Tree* akan menjadi anak dari *index* 7. Untuk menemukan *traversing*, lakukan penelusuran di *Local CFP-Tree index* 7 secara *bottom-up*, hasilnya dapat dilihat pada gambar 3.17



Gambar 3. 17 *Local CFP-Tree Projection* 7

Dari *local CFP-Tree projection* maka didapatkan *item frequent* untuk *index* 7 dengan nama barang Pac.Cyl Jupiter Mx adalah (7,2;5), (7,2,1;4) dan (7,1;5).

e. Pencarian *local Frequent Pattern Tree* untuk *index* 6. Lakukan penelusuran secara *bottom-up* pada *Global CFP-Tree* yang hanya berakhiran *index* 6, untuk jelasnya dapat dilihat pada gambar 3.18



Gambar 3. 18 CFP-*Tree* Berakhiran *index* 6

Berikut hasil penelusuran untuk *index* 6 pada *Global CFP-Tree* yang dapat dilihat pada tabel 3.22

Tabel 3. 22 Penelusuran CFP-*Tree index* 6

No	Lintasan
1	6,5,4
2	6,5,4
3	6,4,3,2
4	6,5,3,2,1
5	6,5,4,2
6	6,4,2,1
7	6,5,4,1

Dari tabel 3.22 didapat jumlah *support count* dari setiap *node* yaitu. 1(3), 2(4), 3(2), 4(6), 5(5). Karena *minimum supportnya* 4, maka *node* yang memenuhi *minimum support* adalah 2(4), 4(6) dan 5(5). Selanjutnya *node* 2(4), 4(6) dan 5(5) di masukan ke *Local item Table*, dimulai dari *support count* terbesar ke terkecil dengan *index* baru yang dapat dilihat pada tabel 3.23

Tabel 3. 23 *Local Item Table*

Index Local	***Index Global***	**Nama Barang**	***Support Count***
1	4	Ts Gasket Kit Mio / Mio Soul	6
2	5	Ts Gasket Kit Vixion Hi-Q	5
3	2	Pac.R.Crank Cover Vega Z R	4

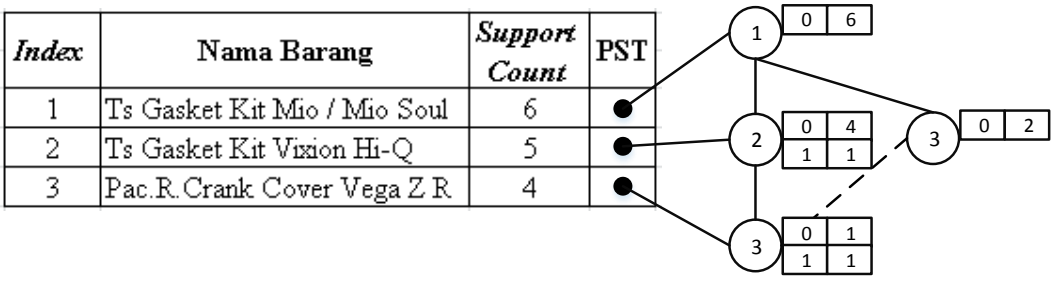
Setelah *local item* tabel terbentuk, maka tahapan selanjutnya adalah memetakan *node* yang memenuhi nilai minimum *support* dengan nomor *index* yang sudah diberikan. Berikut hasil *mapping* yang dapat dilihat pada tabel 3.24

Tabel 3. 24 Hasil Mapping Local CFP-*Tree*

No	Lintasan
1	1,2
2	1,2
3	1,3
4	2,3
5	1,2,3
6	1,3
7	1,2

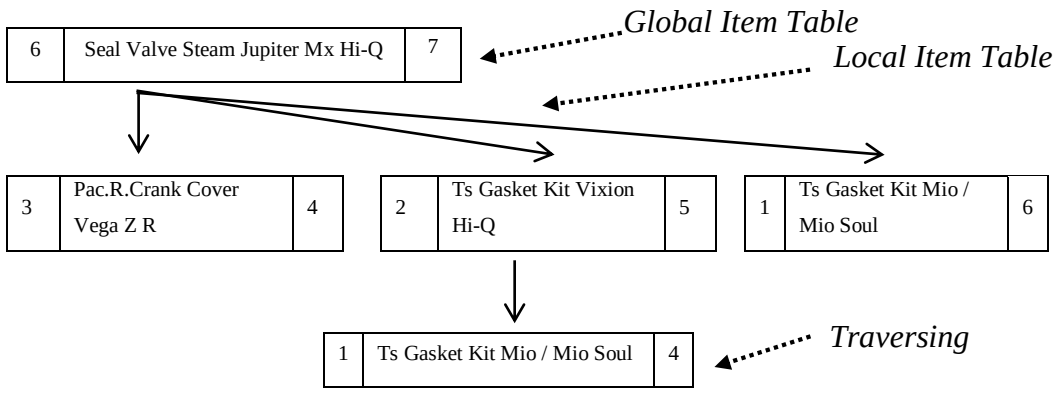
Setelah hasil *mapping* didapat, maka selanjutnya akan dimasukan ke *Local CFPTree*. Aturan pembentukan *Local CFP-Tree* sama dengan pembentukan *Global CFP -Tree*, yang membedakan adalah pada *Global CFP-Tree* yang digunakan dalam pembentukan *tree*-nya adalah *Global Item* tabel yang terbentuk dari *Global Item table*, sedangkan pada *Local CFP-Tree* yang digunakan dalam pembentukan *tree*-nya adalah *Local item* tabel yang terbentuk dari *Local Frequent item*. Berikut adalah *Local CFP-Tree* dari *index* 6 yang dapat dilihat pada gambar 3.19

Index	Nama Barang	*Support Count*	PST
1	Ts Gasket Kit Mio / Mio Soul	6	●
2	Ts Gasket Kit Vixion Hi-Q	5	●
3	Pac.R.Crank Cover Vega Z R	4	●



Gambar 3. 19 *Local CFP-Tree* Untuk *index* 6

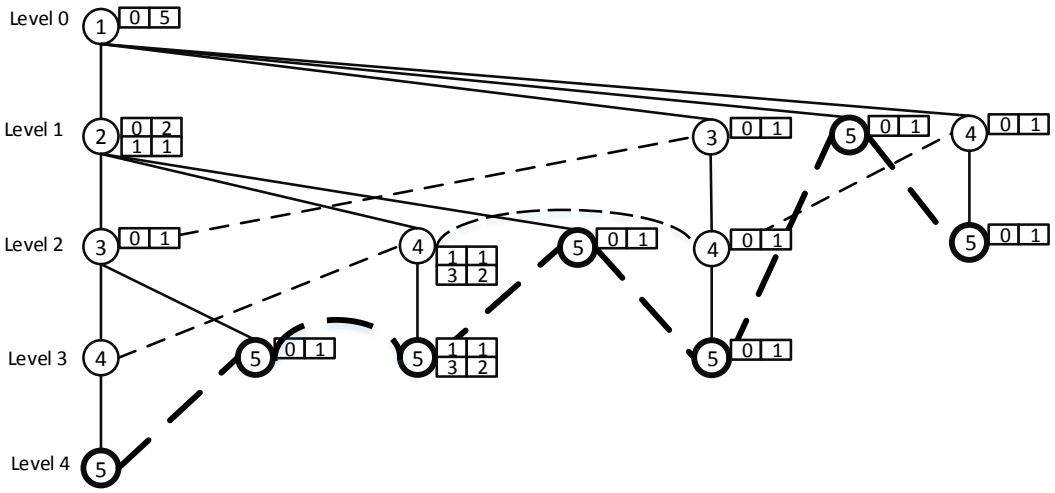
Setelah *Local CFP-Tree index* 6 terbentuk, langkah selanjutnya adalah membuat *Local CFP-Tree projection* di mana *index* 6 akan menjadi *root* dan setiap *index* di *Local CFP-Tree* akan menjadi anak dari *index* 6. Untuk menemukan *traversing*, lakukan penelusuran di *Local CFP-Tree index* 6 secara *bottom-up*, hasilnya dapat dilihat pada gambar 3.20



Gambar 3. 20 *Local CFP-Tree Projection* 6

Dari *local CFP-Tree projection* maka didapatkan *item frequent* untuk *index* 6 dengan nama barang Seal Valve Steam Jupiter Mx Hi-Q adalah (6,3;4), (6,2;5), (6,2,1;4) dan (6,1;6)

f. Pencarian *local Frequent Pattern Tree* untuk *index* 5. Lakukan penelusuran secara *bottom-up* pada *Global CFP-Tree* yang hanya berakhiran *index* 5, untuk jelasnya dapat dilihat pada gambar 3.21



Gambar 3. 21 CFP-*Tree* Berakhiran *index* 5

Berikut hasil penelusuran untuk *index* 5 pada *Global CFP-Tree* yang dapat dilihat pada tabel 3.25

Tabel 3. 25 Penelusuran CFP-*Tree index* 5

No	Lintasan
1	5,3,2,1
2	5,4
3	5,4
4	5,4,2
5	5,2,1
6	5,4,3,1
7	5,1
8	5,4,1

Dari tabel 3.25 didapat jumlah *support count* dari setiap *node* yaitu. 1(5), 2(3), 3(2), 4(5). Karena *minimum supportnya* 4, maka *node* yang memenuhi *minimum support* adalah 1(5) dan 4(5). Selanjutnya *node* 1(5) dan 4(5) dimasukan ke *Local item Table*, dimulai dari *support count* terbesar ke terkecil dengan *index* baru yang dapat dilihat pada tabel 3.26

Tabel 3. 26 *Local Item Table*

Index Local	***Index Global***	**Nama Barang**	***Support Count***
1	1	Pac.Muffler Mio	5
2	4	Ts Gasket Kit Mio / Mio Soul	5

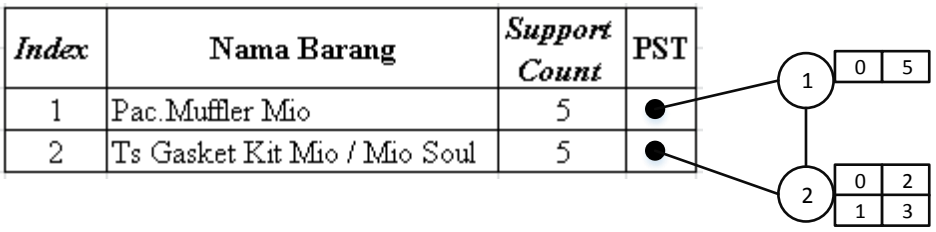
Setelah *local item* tabel terbentuk, maka tahapan selanjutnya adalah memetakan *node* yang memenuhi nilai minimum *support* dengan nomor *index* yang sudah diberikan. Berikut hasil *mapping* yang dapat dilihat pada tabel 3.27

Tabel 3. 27 Hasil Mapping Local CFP-*Tree*

No	Lintasan
1	1
2	2
3	2
4	2

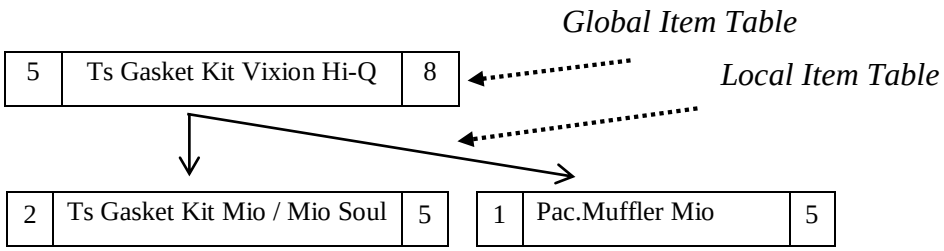
No	Lintasan
5	1
6	1,2
7	1
8	1,2

Setelah hasil *mapping* didapat, maka selanjutnya akan dimasukan ke *Local CFPTree*. Aturan pembentukan *Local CFP-Tree* sama dengan pembentukan *Global CFP-Tree*, yang membedakan adalah pada *Global CFP-Tree* yang digunakan dalam pembentukan *tree*-nya adalah *Global Item* tabel yang terbentuk dari *Global Item table*, sedangkan pada *Local CFP-Tree* yang digunakan dalam pembentukan *tree*-nya adalah *Local item* tabel yang terbentuk dari *Local Frequent item*. Berikut adalah *Local CFP-Tree* dari *index* 5 yang dapat dilihat pada gambar 3.22



Gambar 3. 22 *Local CFP-Tree* Untuk *index* 5

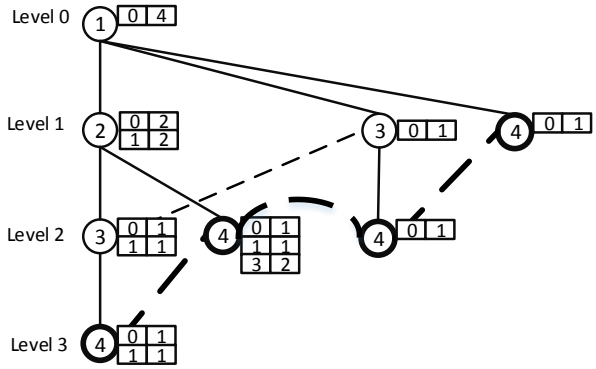
Setelah *Local CFP-Tree* *index* 5 terbentuk, langkah selanjutnya adalah membuat *Local CFP-Tree* *projection* 5 di mana *index* akan menjadi *root* dan setiap *index* di *Local CFP-Tree* akan menjadi anak dari *index* 5. Untuk menemukan *traversing*, lakukan penelusuran di *Local CFP-Tree* *index* 5 secara *bottom-up*, hasilnya dapat dilihat pada gambar 3.23



Gambar 3. 23 *Local CFP-Tree* *Projection* 5

Dari *local CFP-Tree projection* maka didapatkan *item frequent* untuk *index* 5 dengan nama barang Ts Gasket Kit Vixion Hi-Q adalah (5,2;5) dan (5,1;5).

g. Pencarian *local Frequent Pattern Tree* untuk *index* 4. Lakukan penelusuran secara *bottom-up* pada *Global CFP-Tree* yang hanya berakhiran *index* 4, untuk jelasnya dapat dilihat pada gambar 3.24



Gambar 3. 24 CFP-*Tree* Berakhiran *index* 4

Berikut hasil penelusuran untuk *index* 4 pada *Global CFP-Tree* yang dapat dilihat pada tabel 3.28

Tabel 3. 28 Penelusuran CFP-*Tree* *index* 4

No	Lintasan
1	4,3,2
2	4,3,2,1
3	4,2
4	4,2,1
5	4,3,1
6	4,1

Dari tabel 3.28 didapat jumlah *support count* dari setiap *node* yaitu. 1(4), 2(4), 3(3). Karena *minimum supportnya* 4, maka *node* yang memenuhi *minimum support* adalah 1(4) dan 2(4). Selanjutnya *node* 1(4) dan 2(4) dimasukan ke *Local item Table*, dimulai dari *support count* terbesar ke terkecil dengan *index* baru dapat dilihat pada tabel 3.29

Tabel 3. 29 *Local Item Table*

Index Local	*Index Global*	Nama Barang	*Support Count*
1	1	Pac.Muffler Mio	4
2	2	Pac.R.Crank Cover Vega Z R	4

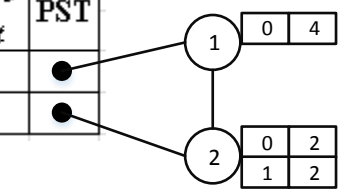
Setelah *local item* tabel terbentuk, maka tahapan selanjutnya adalah memetakan *node* yang memenuhi nilai minimum *support* dengan nomor *index* yang sudah diberikan. Berikut hasil *mapping* yang dapat dilihat pada tabel 3.30

Tabel 3. 30 Hasil Mapping Local CFP-*Tree*

No	Lintasan
1	2
2	1,2
3	2
4	1,2
5	1
6	1

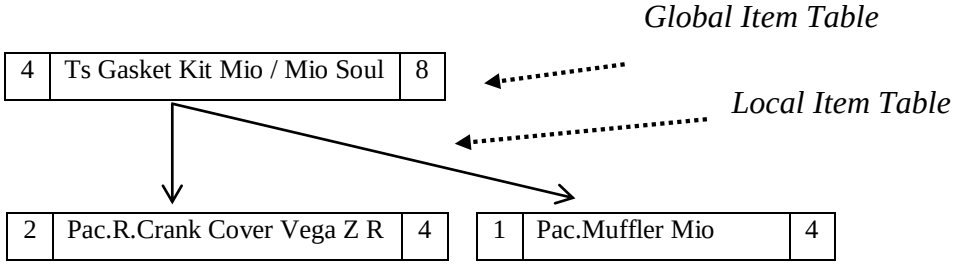
Setelah hasil *mapping* didapat, maka selanjutnya akan dimasukan ke *Local CFPTree*. Aturan pembentukan *Local CFP-Tree* sama dengan pembentukan *Global CFP-Tree*, yang membedakan adalah pada *Global CFP-Tree* yang digunakan dalam pembentukan *tree*-nya adalah *Global Item* tabel yang terbentuk dari *Global Item table*, sedangkan pada *Local CFP-Tree* yang digunakan dalam pembentukan *tree*-nya adalah *Local item* tabel yang terbentuk dari *Local Frequent item*. Berikut adalah *Local CFP-Tree* dari *index* 4 yang dapat dilihat pada gambar 3.25

Index	Nama Barang	*Support Count*	PST
1	Pac.Muffler Mio	4	●
2	Pac.R.Crank Cover Vega Z R	4	●



Gambar 3. 25 *Local CFP-Tree* Untuk *index* 4

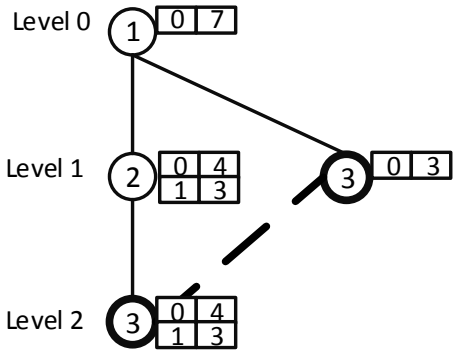
Setelah *Local CFP-Tree index* 4 terbentuk, langkah selanjutnya adalah membuat *Local CFP-Tree projection* di mana *index* 4 akan menjadi *root* dan setiap *index* di *Local CFP-Tree* akan menjadi anak dari *index* 4. Untuk menemukan *traversing*, lakukan penelusuran di *Local CFP-Tree index* 4 secara *bottom-up*, hasilnya dapat dilihat pada gambar 3.26.



Gambar 3. 26 *Local CFP-Tree Projection* 4

Dari *local CFP-Tree projection* maka didapatkan *item frequent* untuk *index* 4 dengan nama barang Ts Gasket Kit Mio / Mio Soul adalah (4,2;4) dan (4,1;4).

h. Pencarian *local Frequent Pattern Tree* untuk *index* 3. Lakukan penelusuran secara *bottom-up* pada *Global CFP-Tree* yang hanya berakhiran *index* 3, untuk jelasnya dapat dilihat pada gambar 3.27



Gambar 3. 27 CFP-*Tree* Berakhiran *index* 3

Berikut hasil penelusuran untuk *index* 3 pada *Global CFP-Tree* yang dapat dilihat pada tabel 3.31

Tabel 3. 31 Penelusuran CFP-*Tree index* 3

No	Lintasan
1	3,2
2	3,2
3	3,2
4	3,2,1
5	3,2,1
6	3,2,1
7	3,2,1
8	3,1
9	3,1
10	3,1

Dari tabel 3.31 didapat jumlah *support count* dari setiap *node* yaitu. 1(7) dan 2(7). Karena *minimum supportnya* 4, maka *node* yang memenuhi *minimum support* adalah 1(7) dan 2(7). Selanjutnya *node* 1(7) dan 2(7) dimasukan ke *Local item Table*, dimulai dari *support count* terbesar ke terkecil dengan *index* baru yang dapat dilihat pada tabel 3.32

Tabel 3. 32 *Local Item Table*

Index Local	***Index Global***	**Nama Barang**	***Support Count***
1	1	Pac.Muffler Mio	7
2	2	Pac.R.Crank Cover Vega Z R	7

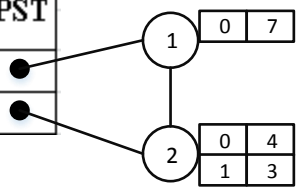
Setelah *local item* tabel terbentuk, maka tahapan selanjutnya adalah memetakan *node* yang memenuhi nilai minimum *support* dengan nomor *index* yang sudah diberikan. Berikut hasil *mapping* yang dapat dilihat pada tabel 3.33

Tabel 3. 33 Hasil Mapping Local CFP-*Tree*

No	Lintasan
1	2
2	2
3	2
4	1,2
5	1,2
6	1,2
7	1,2
8	1
9	1
10	1

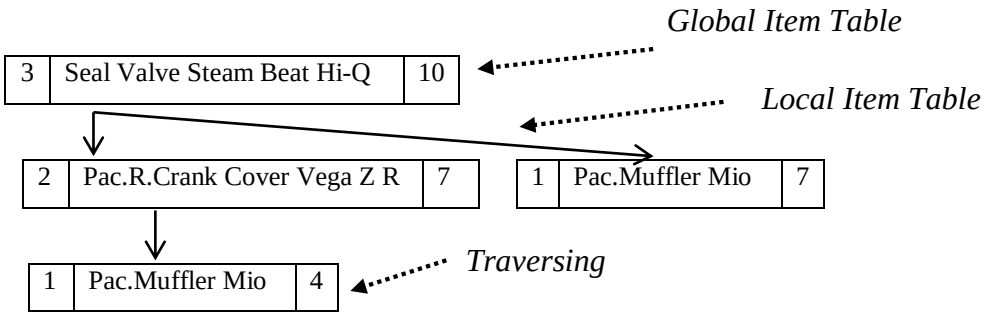
Setelah hasil *mapping* didapat, maka selanjutnya akan dimasukan ke *Local CFPTree*. Aturan pembentukan *Local CFP-Tree* sama dengan pembentukan *Global CFP-Tree*, yang membedakan adalah pada *Global CFP-Tree* yang digunakan dalam pembentukan *tree*-nya adalah *Global Item* tabel yang terbentuk dari *Global Item table*, sedangkan pada *Local CFP-Tree* yang digunakan dalam pembentukan *tree*-nya adalah *Local item* tabel yang terbentuk dari *Local Frequent item*. Berikut adalah *Local CFP-Tree* dari *index* 3 yang dapat dilihat pada gambar 3.28

Index	Nama Barang	*Support Count*	PST
1	Pac.Muffler Mio	7	
2	Pac.R.Crank Cover Vega Z R	7	



Gambar 3. 28 *Local CFP-Tree* Untuk *index* 3

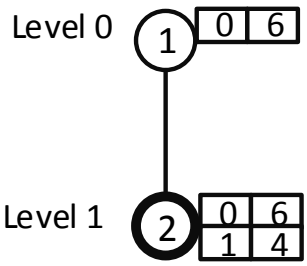
Setelah *Local CFP-Tree index* 3 terbentuk, langkah selanjutnya adalah membuat *Local CFP-Tree projection* di mana *index* 3 akan menjadi *root* dan setiap *index* di *Local CFP-Tree* akan menjadi anak dari *index* 3. Untuk menemukan *traversing*, lakukan penelusuran di *Local CFP-Tree index* 6 secara *bottom-up*, hasilnya dapat dilihat pada gambar 3.29



Gambar 3. 29 *Local CFP-Tree Projection* 3

Dari *local CFP-Tree projection* maka didapatkan *item frequent* untuk *index* 3 dengan nama barang Seal Valve Steam Beat Hi-Q adalah (3,2;7), (3,2,1;7) dan (3,1;7).

i. Pencarian *local Frequent Pattern Tree* untuk *index* 2. Lakukan penelusuran secara *bottom-up* pada *Global CFP-Tree* yang hanya berakhiran *index* 2, untuk jelasnya dapat dilihat pada gambar 3.30



Gambar 3. 30 CFP-*Tree* Berakhiran *index* 2

Berikut hasil penelusuran untuk *index* 2 pada *Global CFP-Tree* yang dapat dilihat pada tabel 3.34

Tabel 3. 34 Penelusuran CFP-*Tree* *index* 2

No	Lintasan
1	2
2	2
3	2
4	2
5	2,1
6	2,1
7	2,1
8	2,1
9	2,1
10	2,1

Dari tabel 3.34 didapat jumlah *support count* dari setiap *node* yaitu. 1(6) Karena *minimum supportnya* 4, maka *node* 1(6) memenuhi *minimum support*. Selanjutnya *node* 1(6) di masukan ke *Local item Table*, dimulai dari *support count* terbesar ke terkecil dengan *index* baru yang dapat dilihat pada tabel 3.35

Tabel 3. 35 *Local Item Table*

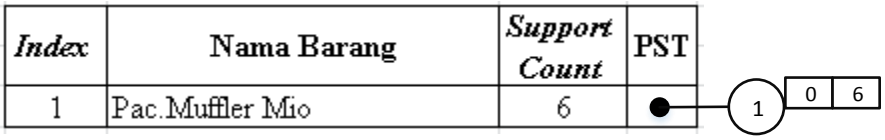
Index Local	***Index Global***	**Nama Barang**	***Support Count***
1	1	Pac.Muffler Mio	6

Setelah *local item* tabel terbentuk, maka tahapan selanjutnya adalah memetakan *node* yang memenuhi nilai minimum *support* dengan nomor *index* yang sudah diberikan. Berikut hasil *mapping* yang dapat dilihat pada tabel 3.36

Tabel 3. 36 Hasil Mapping Local CFP-*Tree*

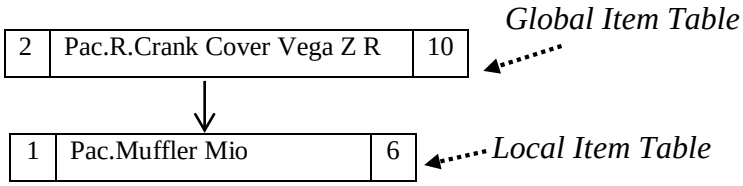
No	Lintasan
1	1
2	1
3	1
4	1
5	1
6	1

Setelah hasil *mapping* didapat, maka selanjutnya akan dimasukan ke *Local CFPTree*. Aturan pembentukan *Local CFP-Tree* sama dengan pembentukan *Global CFP-Tree*, yang membedakan adalah pada *Global CFP-Tree* yang digunakan dalam pembentukan *tree*-nya adalah *Global Item* tabel yang terbentuk dari *Global Item table*, sedangkan pada *Local CFP-Tree* yang digunakan dalam pembentukan *tree*-nya adalah *Local item* tabel yang terbentuk dari *Local Frequent item*. Berikut adalah *Local CFP-Tree* dari *index* 2 yang dapat dilihat pada gambar 3.31



Gambar 3. 31 *Local CFP-Tree* Untuk *index* 2

Setelah *Local CFP-Tree index* 2 terbentuk, langkah selanjutnya adalah membuat *Local CFP-Tree projection* di mana *index* 2 akan menjadi *root* dan setiap *index* di *Local CFP-Tree* akan menjadi anak dari *index* 2. Hasilnya dapat dilihat pada gambar 3.32



Gambar 3. 32 *Local CFP-Tree Projection* 2

Dari *local CFP-Tree projection* maka didapatkan *item frequent* untuk *index* 2 dengan nama barang Pac.R.Crank Cover Vega Z R adalah (2,1;6).

j. Untuk pencarian *local Frequent Pattern Tree* untuk *index* 1. Lakukan penelusuran secara *bottom-up* pada *Global CFP-Tree* yang hanya berakhiran *index* 1, untuk jelasnya dapat dilihat pada gambar 3.33



Gambar 3. 33 CFP-*Tree* Berakhiran *index* 1

Dari gambar 3.33 tidak ada *node* yang ditemukan, maka tidak terbentuk *local item tabel* sehingga untuk *Local CFP-Tree* dan *Frequent Pattern*-nya juga tidak terbentuk.

4. Pembentukan *Rules*

Frequent pattern yang didapat yaitu (9,1:4), (8,3;4), (8,3,1;4), (8,2;4), (8,2,1;4), (8,1;5), (7,2;4), (7,2,1;4), (7,1;5), (6,3;4), (6,2;5), (6,2,1;4), (6,1;5), (5,2;5), (5,1;5), (4,1;4), (3,2;7), (3,2,1;7), (3,1;7) dan (2,1;6).

Selanjutnya dari *frequent pattern* akan di-*generate* untuk mendapatkan *rule* dengan *minimum confidence* = 70%. Dihitung dengan persamaan

$$\text{Confidence} = (A \rightarrow B) = \frac{\text{Jumlah Transaksi Yang Mengandung A\&B}}{\text{Jumlah Transaksi Yang Mengandung A}} \text{x}100\%$$

Nilai *confidence* diperoleh dengan cara membagi jumlah transaksi yang mengandung *item* A dan *item* B dengan jumlah transaksi yang mengandung *item* A. Berikut adalah hasil *generate rule* yang diacu pada tabel 3.37.

Tabel 3. 37 Hasil *Generate Rule*

Frequent Pattern	*Confidence*
9→1	(4/4)*100% = 100 %
8→3	(4/5)*100% = 80 %
8→3,1	(4/5)*100% = 80 %
8→2	(4/5)*100% = 80 %
8→2,1	(4/5)*100% = 80 %
8→1	(5/5)*100% = 100 %
7→2	(4/6)*100% = 66,67 %
7→2,1	(4/6)*100% = 66,67 %
7→1	(5/6)*100% = 83,33 %
6→3	(4/7)*100% = 57,14 %
6→2	(5/7)*100% = 71,42 %
6→2,1	(4/7)*100% = 57,14 %
6→1	(5/7)*100% = 71,42 %
5→2	(5/8)*100% = 62,50 %
5→1	(5/8)*100% = 62,50 %
4→1	(4/8)*100% = 50 %
3→2	(7/10)*100% = 70 %
3→2,1	(4/10)*100% = 40 %
3→1	(7/10)*100% = 70 %
2→1	(6/10)*100% = 60 %

Setelah perhitungan *confidence* diketahui, maka selanjutnya dilakukan pengecekan. Jika tidak memenuhi minimum *confidence* barang tersebut akan dihapus. Setiap *id global item* akan dikembalikan sesuai dengan nama barang dari masing-masing *id global item,* agar pembacaan rule lebih jelas. Berikut hasil *rules* yang terpilih sementara.

Tabel 3. 38 Hasil Rules Yang Terpilih Sementara

Rules	Paket Barang	*Confidence*
9→1	Jika membeli Pac.Cyl Mio / Mio Soul maka membeli juga Seal Valve Steam Beat Hi-Q	100%
8→3	Jika membeli Pac.Muffler Beat / Scoopy / Spacy / Vario maka membeli juga Pac.Cyl Jupiter Mx	80%
8→3,1	Jika membeli Pac.Muffler Beat / Scoopy / Spacy / Vario maka membeli juga Pac.Cyl Jupiter Mx dan Pac.Muffler Mio	80%
8→2	Jika membeli Pac.Muffler Beat / Scoopy / Spacy / Vario maka membeli juga Seal Valve Steam Beat Hi-Q	80%
8→2,1	Jika membeli Pac.Muffler Beat / Scoopy / Spacy / Vario maka membeli juga Seal Valve Steam Beat Hi-Q dan Pac.Muffler Mio	80%
8→1	Jika membeli Pac.Muffler Beat / Scoopy / Spacy / Vario maka membeli juga Pac.Muffler Mio	100%
7→1	Jika membeli Pac.Cyl Jupiter Mx maka membeli juga Pac.Muffler Mio	83,33%
6→2	Jika membeli Seal Valve Steam Jupiter Mx Hi-Q maka membeli juga Ts Gasket Kit Vixion Hi-Q	71,42%
6→1	Jika membeli Seal Valve Steam Jupiter Mx Hi-Q maka membeli juga Ts Gasket Kit Mio / Mio Soul	71,42%
3→2	Jika membeli Seal Valve Steam Beat Hi-Q maka membeli juga Pac.R.Crank Cover Vega Z R	70%
3→1	Jika membeli Seal Valve Steam Beat Hi-Q maka membeli juga Pac.Muffler Mio	70%

5. Aturan Bisnis Pembentukan Paket

Dalam penentuan kombinasi barang yang akan ditawarkan dalam bentuk paket, pihak Cv. Bukit Manikam memiliki aturan bisnis pembentukan paket yaitu kombinasi barang dalam satu paket hanya terdapat satu *brand*, sebagai contoh yamaha dengan yamaha. Maka proses selanjutnya rules difilter sesuai dengan yang telah ditentukan, hasilnya dapat dilihat pada tabel berikut :

Tabel 3. 39 Hasil Rules Setelah Difilter

No	Paket Barang	*Confidence*
1	Pac.Muffler Beat / Scoopy / Spacy / Vario dan Seal Valve Steam Beat Hi-Q	80%
2	Pac.Cyl Jupiter Mx dan Pac.Muffler Mio	83,33%
3	Seal Valve Steam Jupiter Mx Hi-Q dan Ts Gasket Kit Vixion Hi-Q	71,42%
4	Seal Valve Steam Jupiter Mx Hi-Q dan Ts Gasket Kit Mio / Mio Soul	71,42%

3.1.3 Analisis Kebutuhan Non Fungsinal

Analisis non fungsional dan kebutuhan non fungsional dibagi menjadi tiga tahapan analisis yaitu analisis pengguna, analisis perangkat lunak dan analisis perangkat keras.

3.1.3.1 Analisis Pengguna

User atau pengguna yang akan mengoperasikan aplikasi ini merupakan supervisor yang berjanggung jawab dalam menjalankan bisnis. Analisis pengguna dapat dilihat pada tabel 3.40.

Tabel 3. 40 Analisis Pengguna

User	Tanggung Jawab	Hak Akses	Tingkat Keterampilan	Pengalaman
Supervisor	Mengecek kevalidan data dan sistem	Dapat mengolah data transaksi penjualan dan memperoleh informasi dari aplikasi *mining.*	Dapat mengikuti petunjuk yang ada pada sistem seperti penginputan data dan juga dapat mengolah informasi yang didapat.	Dapat mengoperasikan komputer dan bisa menggunakan program

3.1.3.2 Analisis Perangkat Lunak

Analisis kebutuhan perangkat lunak dalam membangun aplikasi digunakan untuk mengetahui dan mengoptimalkan implementasi dari aplikasi yang akan dibangun. Berikut merupakan analisis terhadap perangkat lunak dibagian supervisor Cv.Bukit Manikam.

1. Analisis spesifikasi perangkat lunak pada system yang sedang berjalan di bagian supervisor Cv.Bukit Manikam adalah:
   a. Sistem Operasi Windows 7
   b. *Microsoft Office* 2013
2. Analisis spesifikasi kebutuhan perangkat lunak untuk menjalankan system yang akan dibangun adalah:
   a. Sistem Operasi Windows 7
   b. *MySQL* sebagai *database server*
   c. *Microsoft Visual Studio* 2010
3. Evaluasi kebutuhan perangkat lunak

   Berdasarkan analisis spesifikasi perangkat luanak yang ada dibagian supervisor Cv.Bukit Manikam, spesifikasi perangkat lunak yang digunakan masih kurang. Untuk mengatasi hal tersebut akan diinstal *MySQL* dan *Visual Studio* pada komputer yang digunakan.

3.1.3.3 Analisis Perangkat Keras

Analisis kebutuhan perangkat keras digunakan untuk memudahkan proses perancangan dan implementasi. Berikut merupakan analisis terhadap perangkat keras dibagian supervisor Cv.Bukit Manikam.

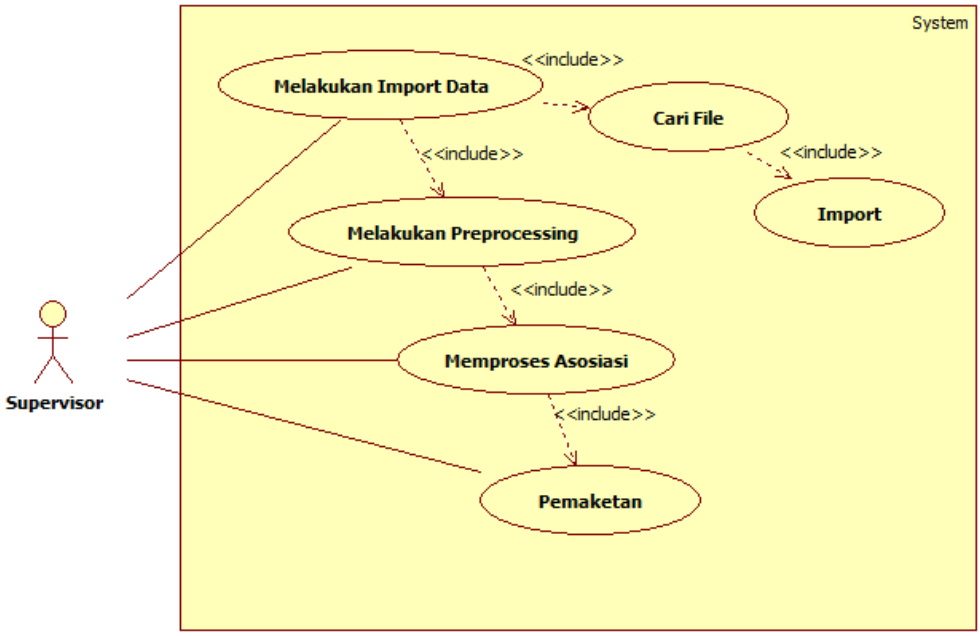
1. Analisis spesifikasi perangkat keras pada sistem yang sedang berjalan dibagian supervisor Cv.Bukit Manikam adalah:
   a. *Processor*: Intel Core i3 5010U-2.1 Ghz
   b. *Hard disk*: 500 GB
   c. RAM: 4 GB
2. Analisis spesifikasi kebutuhan perangkat keras untuk menjalankan sistem yang akan dibangun adalah:
   a. *Processor:* Dual Core 2GHz
   b. *Hard disk:* 500 MB untuk menyimpan *file*
   c. RAM: 2 GB

3.1.4 Analisis Kebutuhan Fungsional

Analisis kebutuhan fungsional adalah analisis terhadap kebutuhan secara fungsional baik dalam aliran data ataupun informasi. Analisis kebutuhan data fungsional yang digunakan dalam sistem ini digambarkan dengan analisis Pemrograman Berorientasi Objek dengan menggunakan konsep UML (*Unified Modeling Languages*).

3.1.4.1 Diagram Use case

Diagram *use case* merupakan pemodelan untuk perilaku (*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 3.34.



Gambar 3. 34 Diagram *Use case*

Berikut adalah tabel deskripsi *user* yang dapat dilihat pada tabel 3.41 dan tabel deskripsi *use case* yang dapat dilihat pada tabel 3.42.

Tabel 3. 41 Deskripsi User

User	Deskripsi
Supervisor	*User* bertugas untuk melakukan *import* data, *memasukkan minimum support, minimum confidence* dan memilih *filtering.*

Tabel 3. 42 Tabel Deskripsi *Use case*

No	*Use case*	Deskripsi
1	Melakukan *Import* Data	*Import* data harus dilakukan *user* terlebih dahulu, data yang diimport merupakan data transaksi yang selanjutnya data tersebut akan disimpan ke dalam *database*.
2	Cari *File*	*User* mencari *file* data transaksi yang akan dipakai dan selanjutnya data tersebut dipilih.
3	*Import*	Data transaksi yang telah dipilih dan tampil digrid selanjutnya diimport.
4	Melakukan *Preprocessing*	Data transaksi yang sudah tersimpan di dalam *database* kemudian akan dilakukan proses pemilihan atribut, pembersihan data dan hitung *support* count.
5	Memproses Asosiasi	*User* menginputkan nilai parameter yaitu nilai *Minimum Support* dan *Minimum Confidence* , kemudian sistem melakukan pencarian *frequent itemset,* melakukan proses eliminasi dan menghasilkan *rule*
6	Pemaketan	*User* memilih filter pemaketan, kemudian sistem melakukan proses *filtering rules* dan menghasilkan pemaketan.

3.1.4.2 Sekenario 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 3.43

Tabel 3. 43 Skenario *Use case* Import Data

Use case Name	***Import Data***	
Goal In Context	*Import* data transaksi Cv. Bukit Manikam dan menyimpannya ke dalam *database*	
Precondition	*User* menyiapkan data transaksi	
Successful End Condition	Data transaksi yang di *import* oleh *user* berhasil disimpan ke dalam *database*	
Failed End Condition	Data transaksi gagal disimpan ke dalam *database*	
Actors	*User*	
Triger	*User* memasukan data transaksi	
Included Cases	-	
Main Flow	Step	Action
	1	*User* memilih menu import data
	2	Sistem menampilkan menu import data
	3	*User* meng-klik tompol cari *file*
	4	Sistem menampilkan select *file* dialog
	5	*User* memilih *file* data transaksi
	6	Sistem menampilkan data transaksi di grid
	7	*User* meng-klik tompol import
	8	Sistem mengimport data transaksi ke *database*
Extension	Step	Branching Action
	1.1	Menampilkan pesan kesalahan dalam proses *import* data

2. Skenario *Use case* Cari *File*

Skenario *use case* cari *file* menggambarkan langkah-langkah aksi *user* terhadap sistem untuk melakukan pemilihan *file* data transaksi yang akan disimpan ke dalam *database*.

Tabel 3. 44 Skenario *Use case* Cari *File*

Use case Name	**Cari *File***	
Goal In Context	*User* dapat memilih *file* data transaksi.	
Precondition	*User* menyiapkan data transaksi.	
Successful End Condition	Data transaksi dapat dipilih user.	
Failed End Condition	-	
Actor	*User*	
Triger	*User* memilih data transaksi	
Included Class	-	
Main Flow	*Step*	*Action*
	1	*User* mengklik cari *file*
	2	Sistem menampilkan select *file* dialog
	3	*User* memilih *file* data transaksi
	4	Sistem menampilkan data transaksi di grid
Extension	*Step*	*Branching Action*
	-	

3. Skenario *Use case Import*

Skenario *use case import* menggambarkan langkah-langkah aksi *user* terhadap sistem untuk melakukan *import* data transaksi yang akan disimpan ke dalam *database*.

Tabel 3. 45 Skenario *Use case Import*

Use case Name	***Import***	
Goal In Context	*Import* data transaksi Cv. Bukit Manikam dan menyimpannya ke dalam *database*.	
Precondition	*User* menyiapkan data transaksi.	
Successful End Condition	Data transaksi yang di *input* oleh *user* berhasil disimpan ke dalam *database*.	
Failed End Condition	Data transaksi gagal disimpan ke dalam *database*.	
Actor	*User*	
Triger	*User* memasukan data transaksi	
Included Class	-	
Main Flow	*Step*	*Action*
	1	*User* mengklik tombol import
	2	Sistem mengimport data transaksi
Extension	*Step*	*Branching Action*
	-	

4. Skenario *Use case Preprocessing*

Skenario *use case preprocessing* menjelaskan langkah-langkah untuk melakukan proses Pemilihan Atribut, Pembersihan Data, dan Hitung *Support* Count. Skenario *use case preprocessing* dapat dilihat pada tabel 3.46

Tabel 3. 46 Skenario *Use case* Preprocesing

Use case Name	***Preprocesing***	
Goal In Context	Sistem melakukan proses pemilihan atribut, pembersihan data, dan hitung *support* count	
Precondition	Data transaksi sudah tersimpan didalam *database*	
Successful End Condition	Sistem berhasil melakukan proses pemilihan atribut, pembersihan data, dan hitung *support count*	
Failed End Condition	Sistem gagal untuk melakukan proses pemilihan atribut, pembersihan data dan hitung *support count*	
Actors	*User*	
Triger	*User* melakukan *request* proses pemilihan atribut, pembersihan data, dan hitung *support count*	
Included Cases	Pemilihan Atribut, Pembersihan Data, dan Hitung *Support* Count	
Main Flow	Step	Action
	1	User memilih menu preprocessing
	2	Sistem menampilkan menu preprocessing
	3	*User* mengklik tombol proses preprocessing
	4	Sistem melakukan proses Seleksi Atribut
	5	Sistem melakukan proses Pembersihan Data *item* tunggal
	6	Sistem melakukan proses Hitung *Support Count* tiap *item* pada data transaksi
Extension	Step	Branching Action
	1.1	Sistem menampilkan pesan kesalahan dalam proses preprocessing

5. Skenario *Use case* Proses Asosiasi

Skenario *use case* asosiasi menggambarkan langkah-langkah aksi *user* untuk melakukan proses asosiasi Input Parameter dan Lihat *Rule*. Skenario *use case* proses dapat dilihat pada tabel 3.47

Tabel 3. 47 Skenario *Use case* Proses Asosiasi

Use case Name	**Proses Asosiasi**	
Goal In Context	Sistem menghasilkan *rule*	
Precondition	Data sudah melalui proses *preprocessing*	
Successful End Condition	Sistem berhasil mendapatkan *rule* yang dihasilkan.	
Failed End Condition	Sistem gagal mendapatkan *rule* yang akan dihasilkan.	
Actors	*User*	
Triger	*User* menginputkan nilai parameter berupa nilai *Minimum Support* dan *Minimum Confidence*	
Included Cases	Input Parameter dan Lihat *Rules*	
Main Flow	Step	Action
	1	*User* memilih menu proses asosiasi
	2	Sistem menampilkan menu proses asosiasi
	3	*User* memasukkan nilai minimum *confidence* dan minimum *support*
	4	*User* mengklik tombol proses
	5	Sistem memproses seleksi data yang memenuhi minimum *confidence* dan minimum *support*
	6	Sistem menampilkan *generate rules*
	7	Sistem menampilkan hasil *rule*.
Extension	Step	Branching Action
	1.1	Sistem tidak mendapatkan *frequent itemset* karena nilai *minimum support* yang

		dimasukkan oleh *user* terlalu tinggi
	1.2	Sistem gagal melakukan *generate rule*, karena tidak ada *frequent itemset* yang d*item*ukan
	1.3	Sistem tidak mendapatkan *rule* karena nilai *Minimum Confidence* yang dimasukkan terlalu tinggi

6. Skenario *Use case* Pemaketan

Skenario *use case* pemaketan menggambarkan langkah-langkah aksi *user* untuk melakukan proses lihat informasi pembentukan paket. Skenario *use case* pemaketan dapat dilihat pada tabel 3.48

Tabel 3. 48 Skenario *Use case* Pemaketan

Use case Name	**Pemaketan**	
Goal In Context	Sistem menampilkan informasi paket barang hasil *filtering* paket	
Precondition	Sistem telah menghasilkan *rule*	
Successful End Condition	Sistem berhasil menampilkan informasi paket barang hasil *filtering* paket	
Failed End Condition	Sistem gagal menampilkan informasi paket barang hasil *filtering* paket	
Actors	*User* melakukan memilih *filtering* paket	
Triger	*User*	
Main Flow	Step	Action
	1	*User* memilih menu pemaketan
	2	Sistem menampilkan menu pemaketan
	3	*User* memilih filter *brand* motor
	4	*User* mengklik filter
	5	Sistem memproses filter paket yang dipilih *user*
	6	Sistem menampilkan paket hasil filter

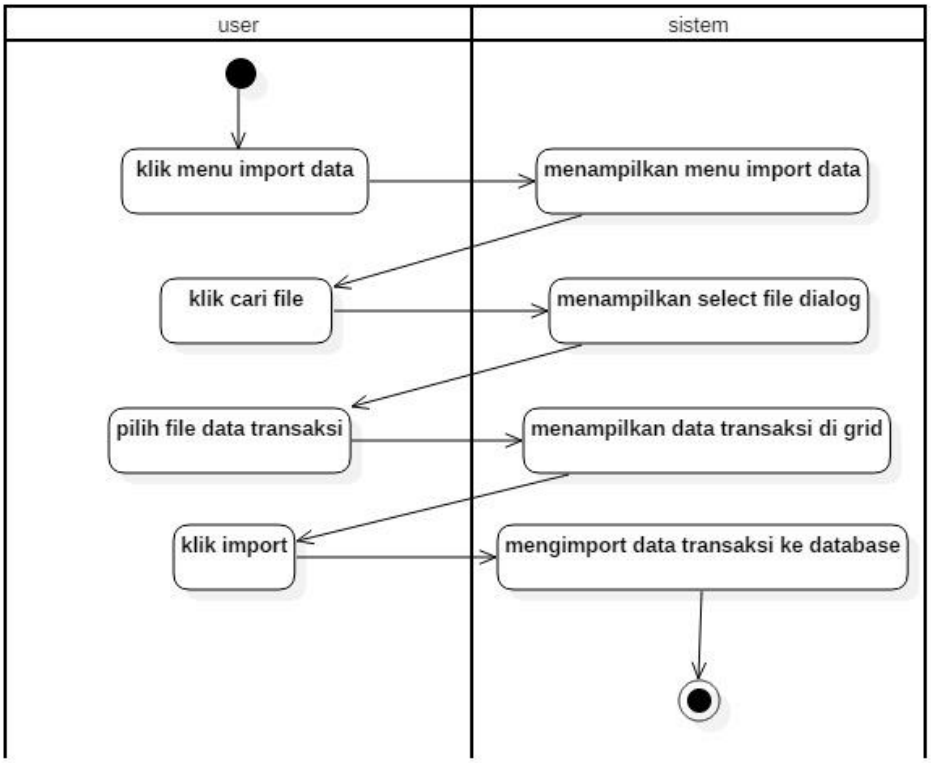
Extension	Step	Branching Action
	1.1	Sistem gagal mendapatkan filtering yang dipilih
	1.2	Sistem tidak menampilkan hasil karena filter yang dipilih user tidak d*item*ukan
	1.3	Sistem tidak memiliki informasi barang yang bisa dijual dalam bentuk paket

3.1.4.3 Diagram Activity

Diagram Activity menggambarkan rangkaian aliran dari aktivitas setiap proses. Aktivitas diagram dari setiap proses dapat dilihat pada gambar-gambar berikut :

1. *Activity Diagram* Import Data

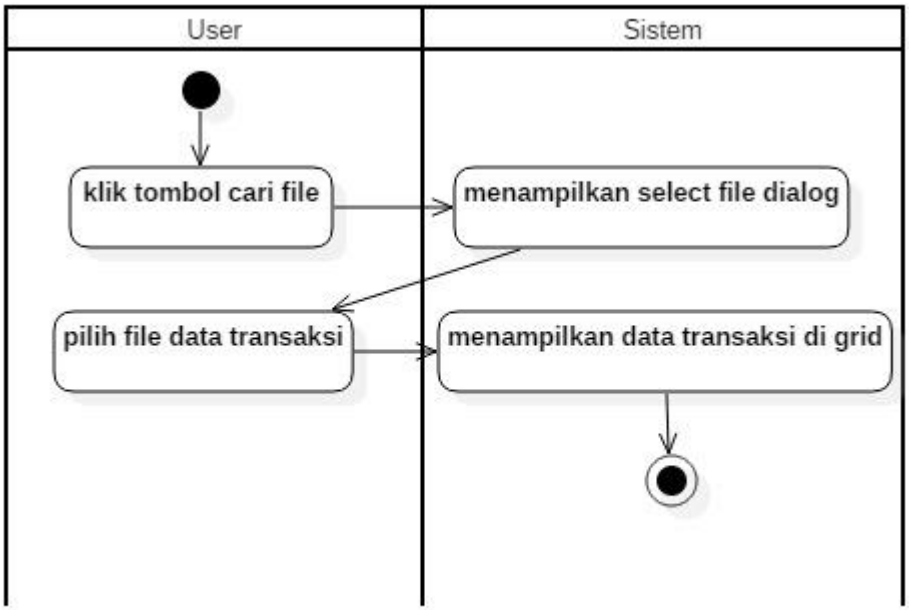
Activity diagram import data menjelaskan aktivitas aktor dalam melakukan proses import data. Penjelasan activity *diagram* import data diacu pada Gambar 3.35.



Gambar 3. 35 *Activity Diagram* Import Data

2. *Activity* Diagram Cari *File*

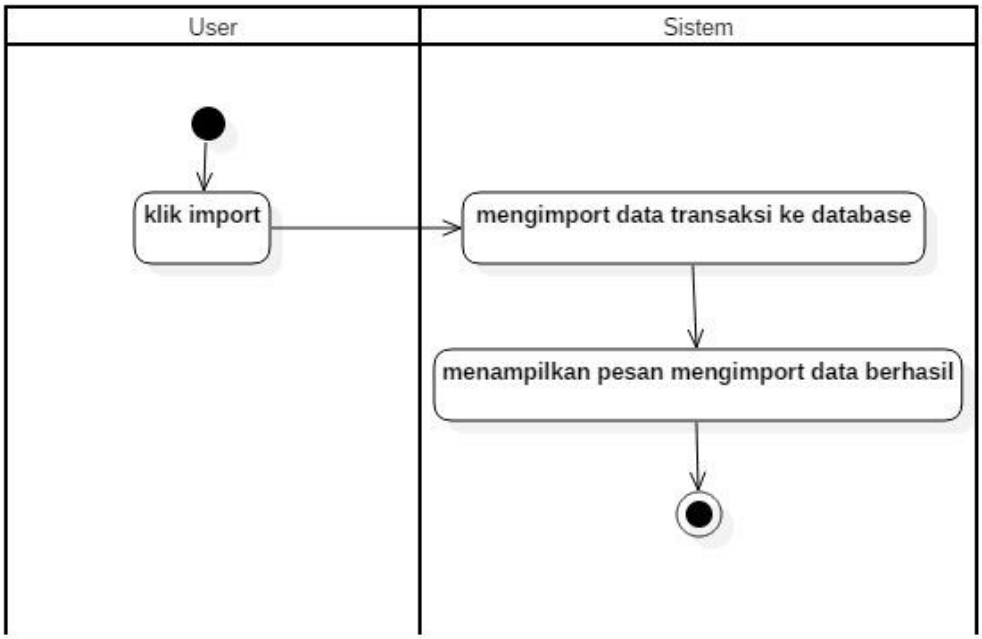
Activity diagram import data menjelaskan aktivitas aktor dalam melakukan proses pilih *file*. Penjelasan *activity* diagram pilih *file* diacu pada gambar 3.36.



Gambar 3. 36 Activity Diagram Cari *File*

3. *Activity* Diagram *Import*

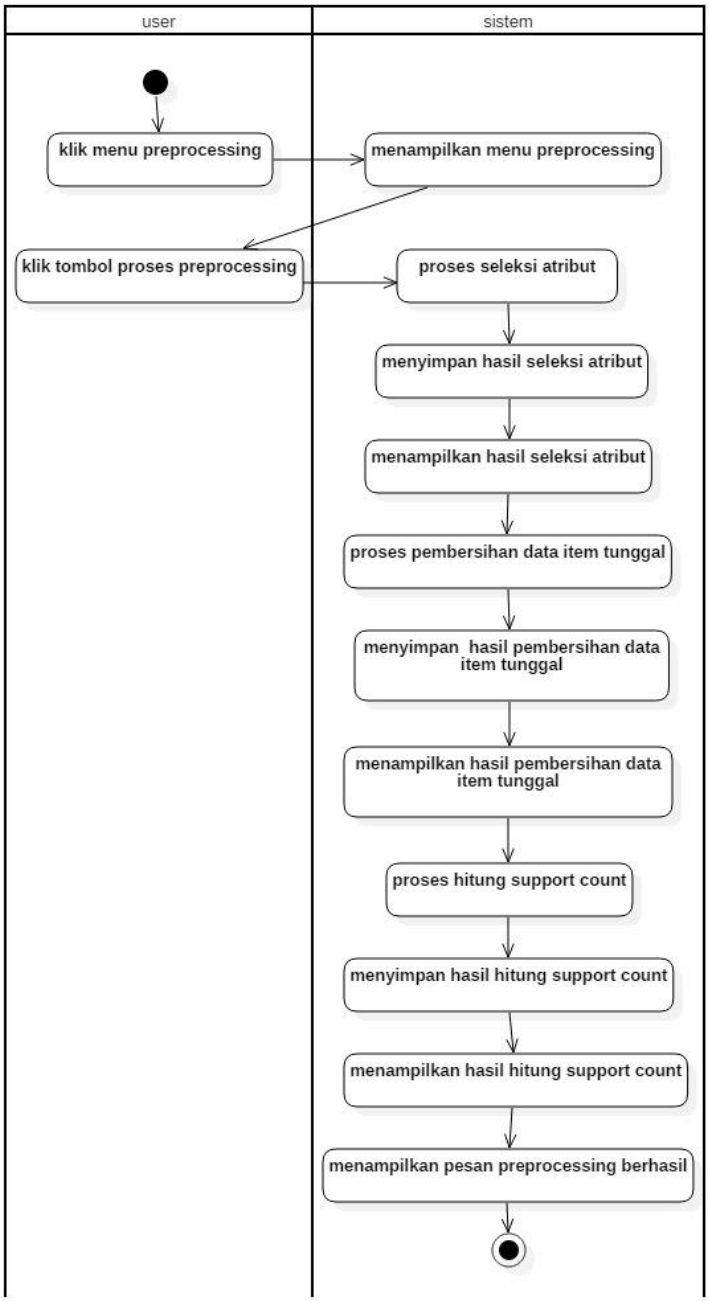
Activity diagram import menjelaskan aktivitas aktor dalam melakukan proses pilih import. Penjelasan activity diagram import diacu pada gambar 3.37.



Gambar 3. 37 Activity Diagram Import

4. Activity Diagram Preprocessing

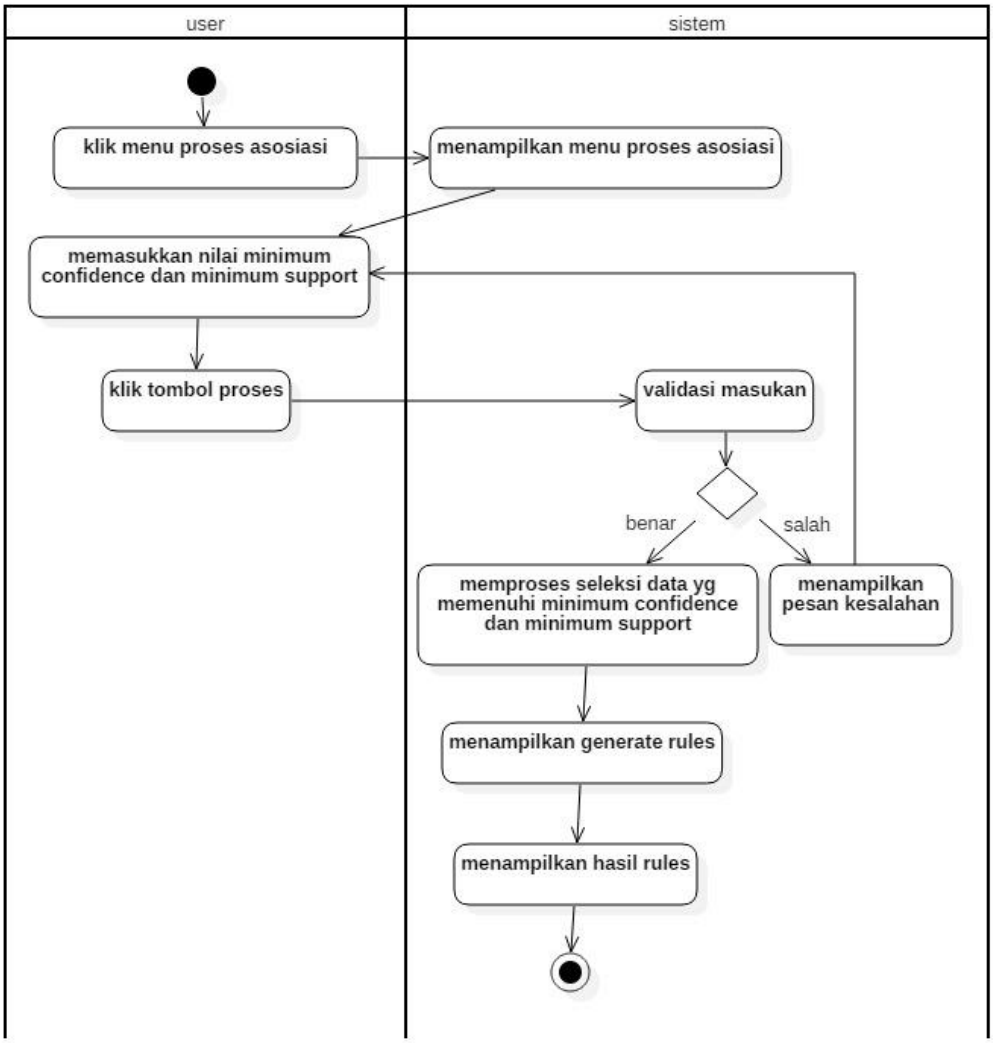
Activity diagram preprocessing menjelaskan aktivitas sistem dalam melakukan proses seleksi atribut, pembersihan data dan hitung frekuensi. *Activity diagram preprocessing* dapat dilihat pada gambar 3.36.



Gambar 3. 38 *Activity Diagram Preprocessing*

5. *Activity Diagram* Proses Asosiasi

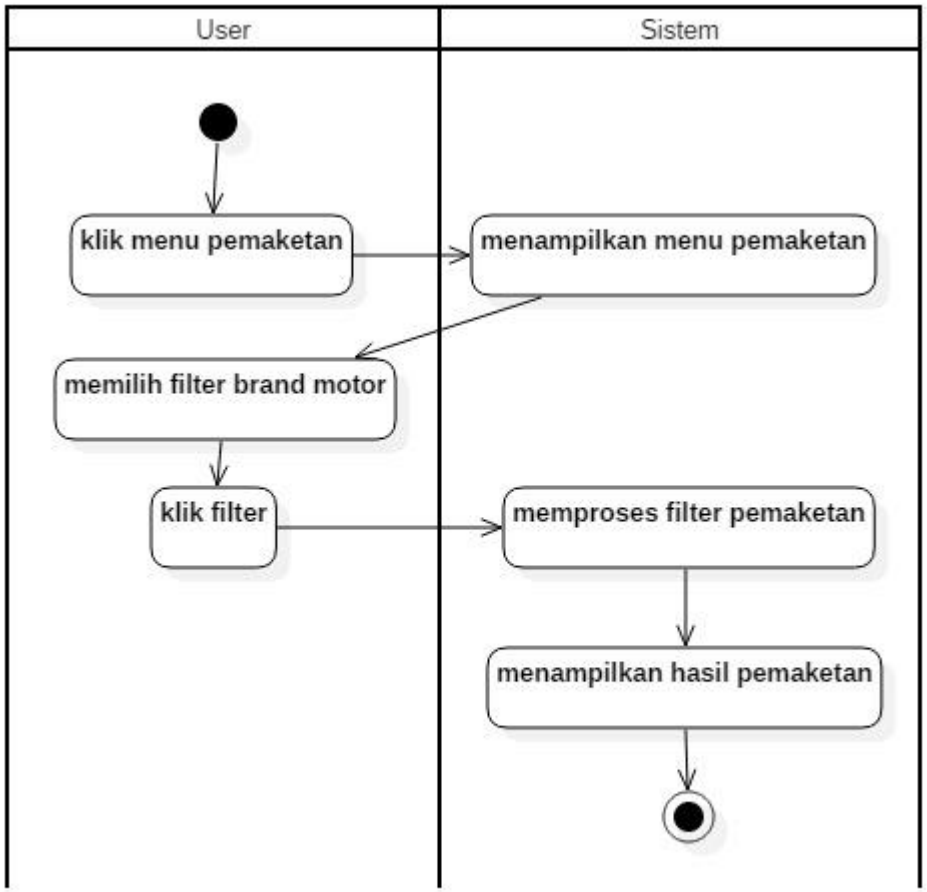
Activity diagram asosiasi menjelaskan aktivitas user dan sistem dalam melakukan proses asosiasi . *Activity diagram* asosiasi dapat dilihat pada gambar 3.37



Gambar 3. 39 *Activity Diagram* Asosiasi

6. *Activity Diagram* Pemaketan

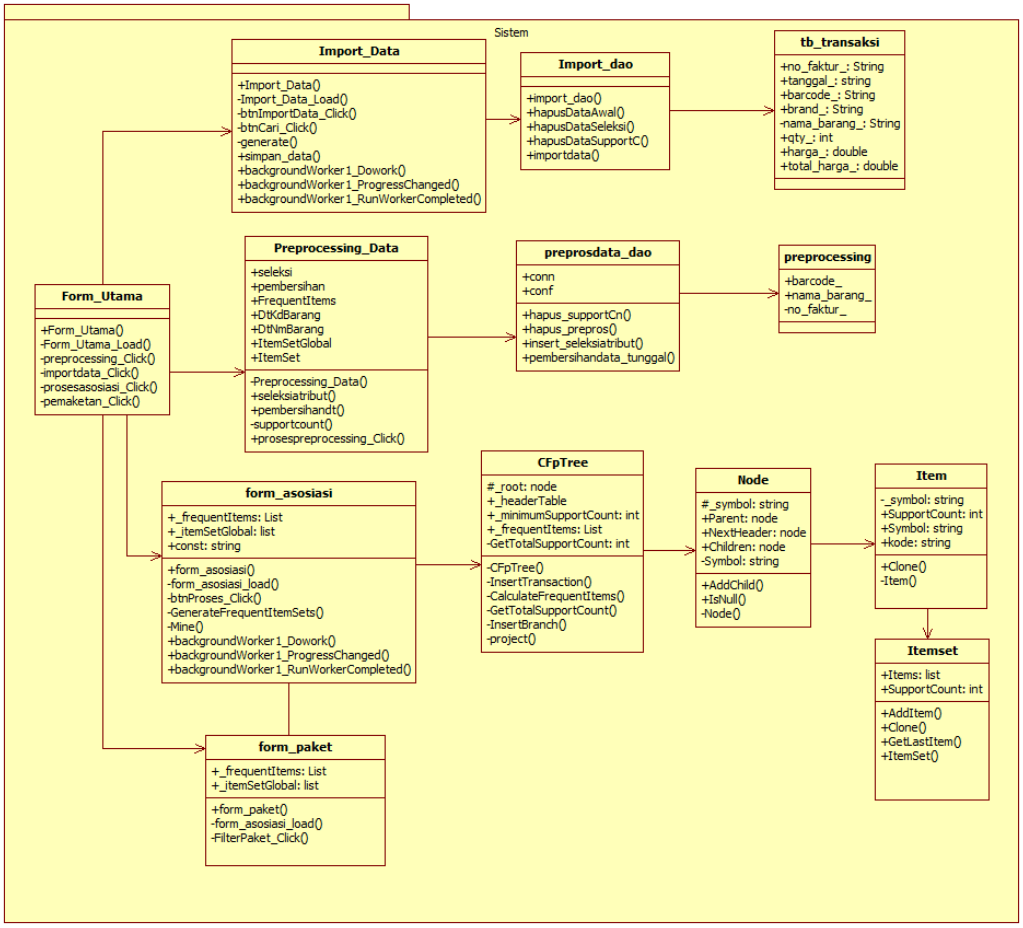
Activity diagram pemaketan menjelaskan aktivitas *user* dan sistem dalam menghasilkan pemaketan. *Activity diagram* pemaketan dapat dilihat pada gambar 3.38.



Gambar 3. 40 *Activity Diagram* Pemaketan

3.1.4.4 Class Diagram

Class diagram menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan di buat untuk membangun sistem. Kelas diagram dalam aplikasi yang akan dibangun ini digambarkan pada gambar 3.41.



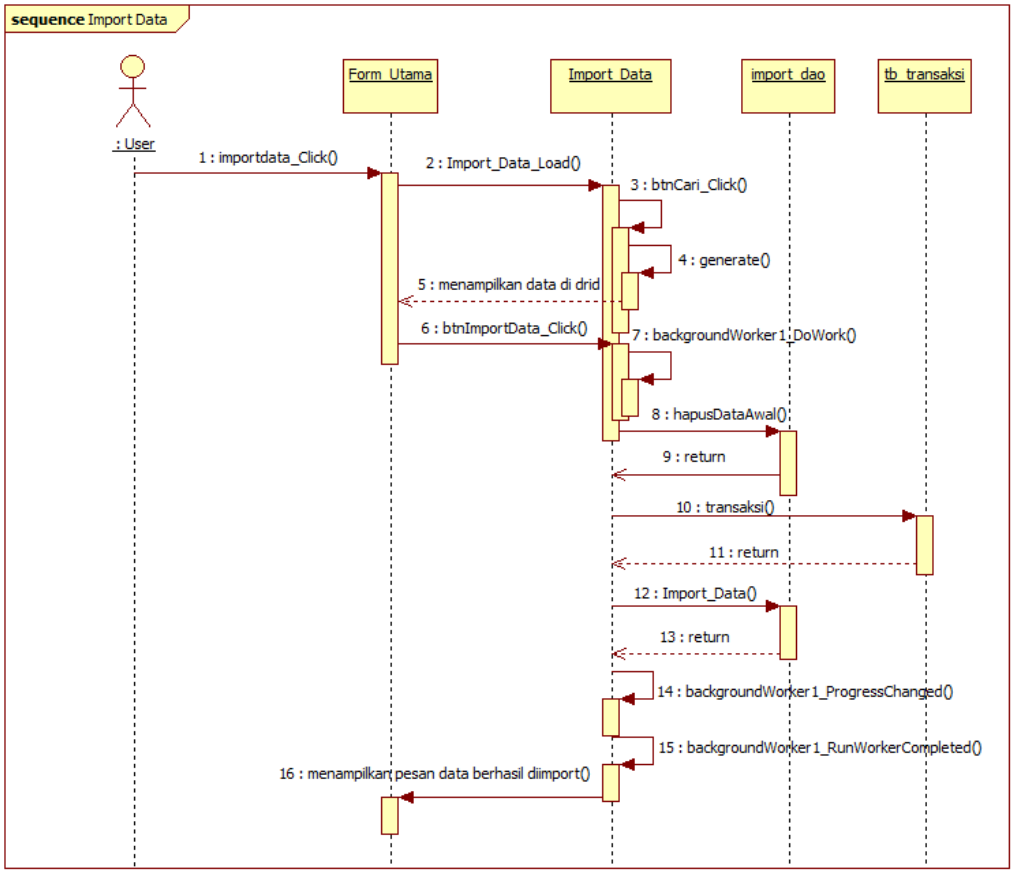
Gambar 3. 41 *Class Diagram*

3.1.4.5 Diagram Sequence

Diagram *sequence* menggambarkan interaksi objek pada *use case* dengan mendeksripsikan waktu hidup objek dan pesan yang dikirimkan dan di terima antar objek. Berikut adalah diagram squensial dari sistem yang akan dibangun :

1. *Sequence Diagram* Import Data

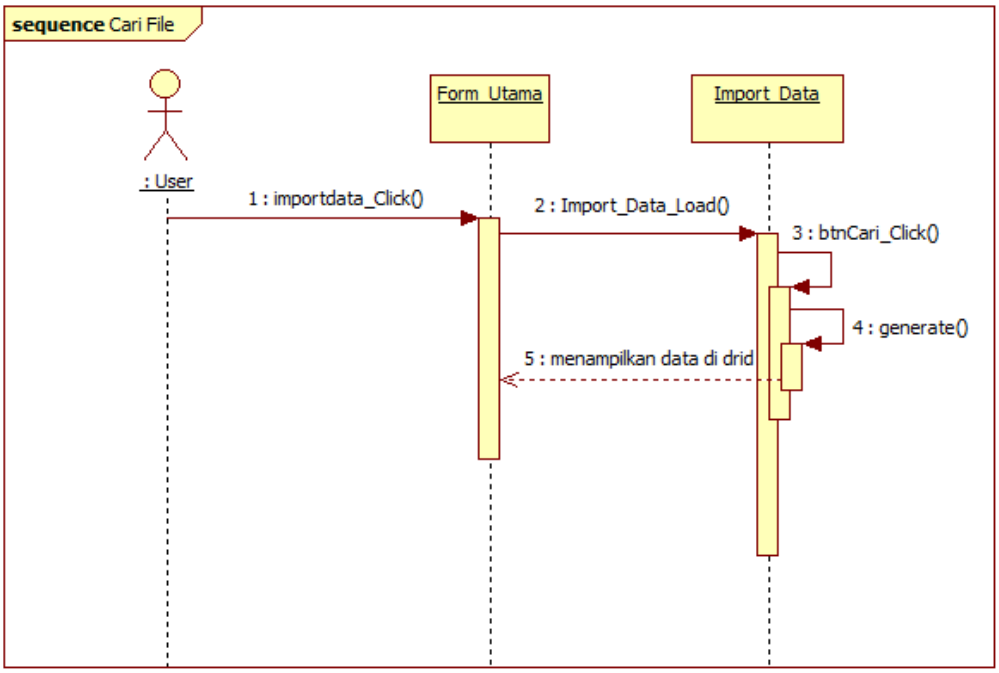
   Untuk *sequence diagram* pada proses import data dapat dilihat pada gambar 3.42.



Gambar 3. 42 *Sequence Diagram* Import Data

2. *Sequence Diagram* Cari *File*

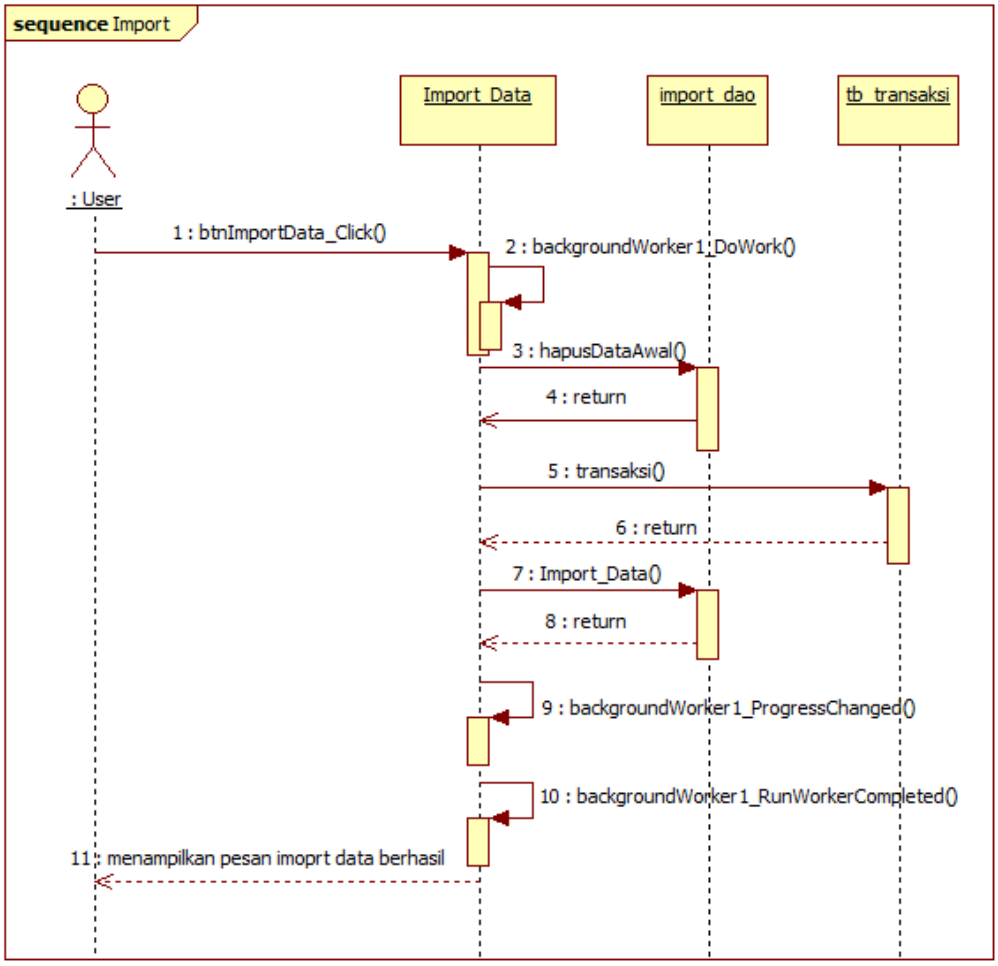
Untuk *sequence diagram* pada proses cari *file* dapat dilihat pada gambar 3.43.



Gambar 3. 43 *Sequence Diagram* Cari *File*

3. *Sequence Diagram* Import

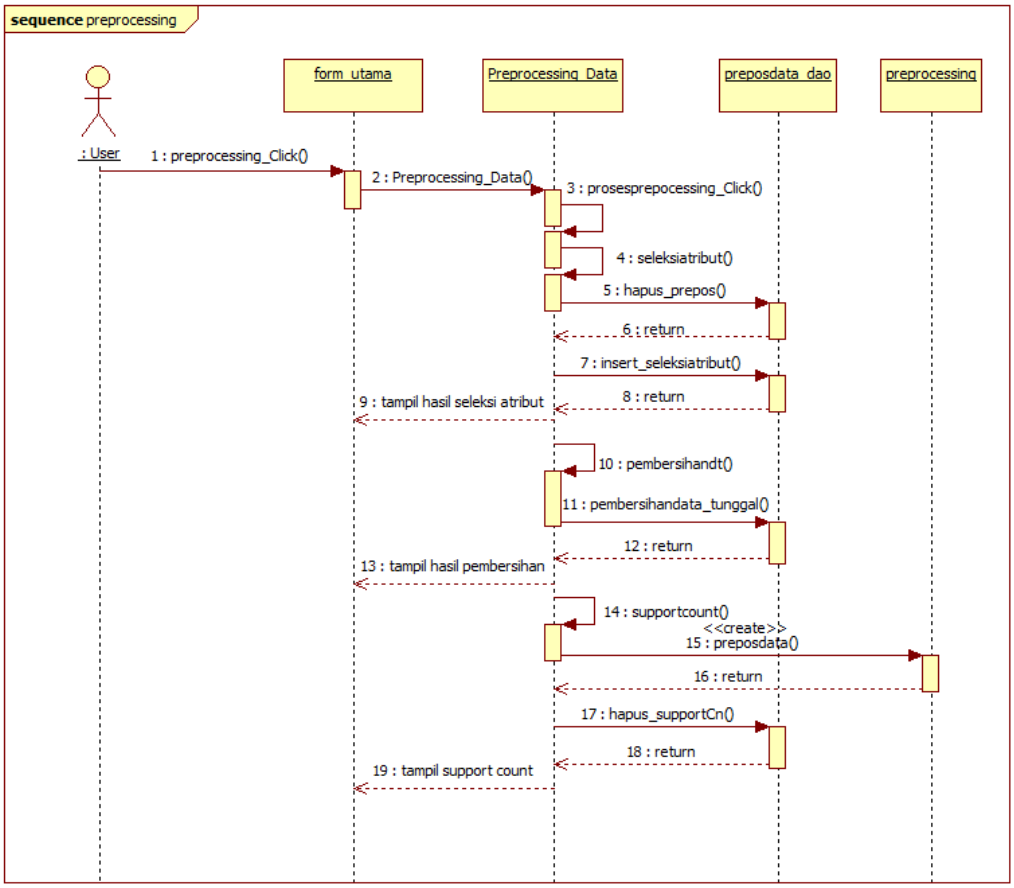
Untuk *sequence diagram* pada proses import dapat dilihat pada gambar 3.44.



Gambar 3. 44 *Sequence Diagram Import*

4. *Sequence Diagram Preprocessing*

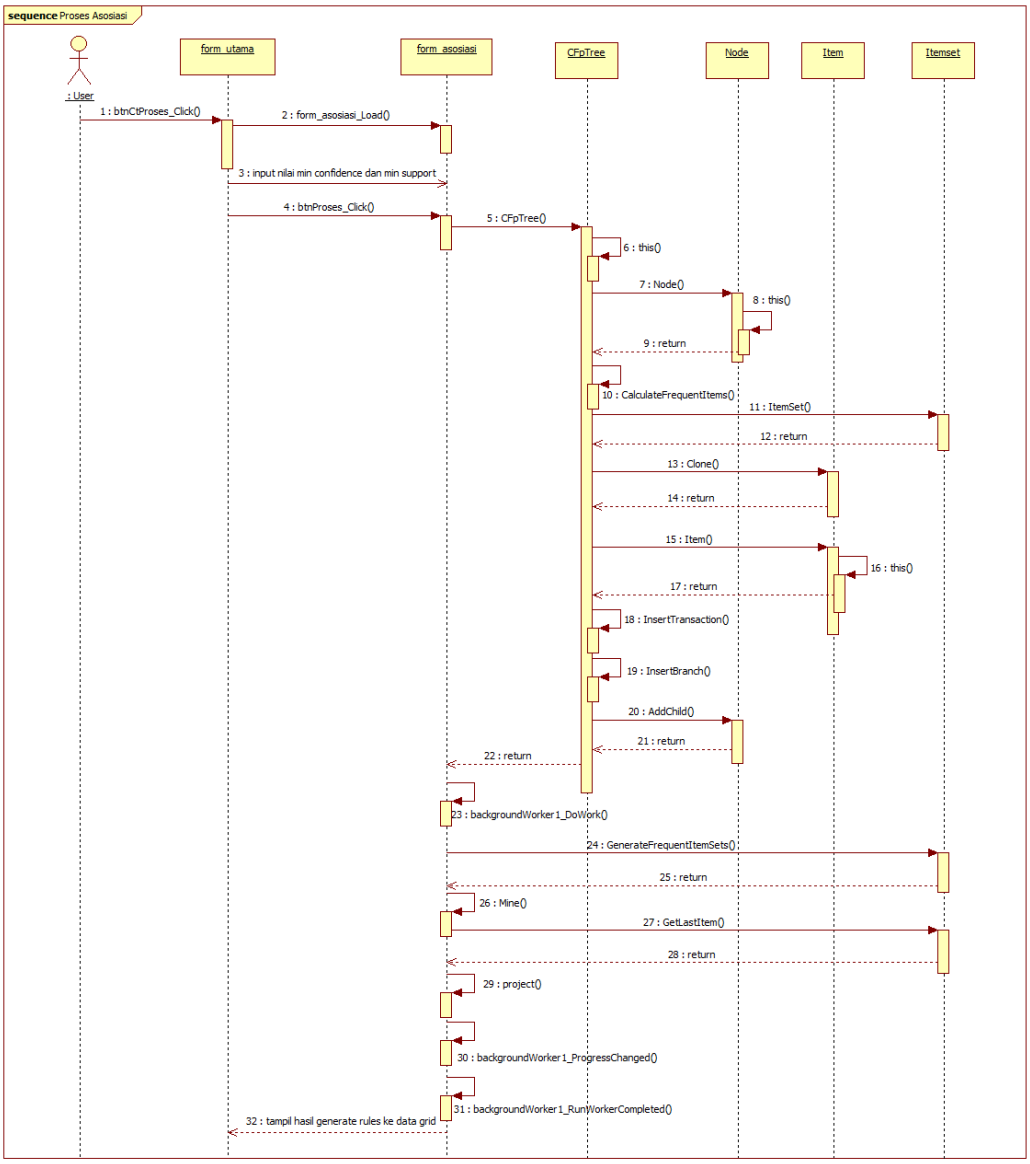
Untuk *sequence diagram* pada *preprocessing* dapat dilihat pada gambar 3.45.



Gambar 3. 45 *Sequence Diagram Preprocessing*

5. *Sequence Diagram* Proses Asosiasi

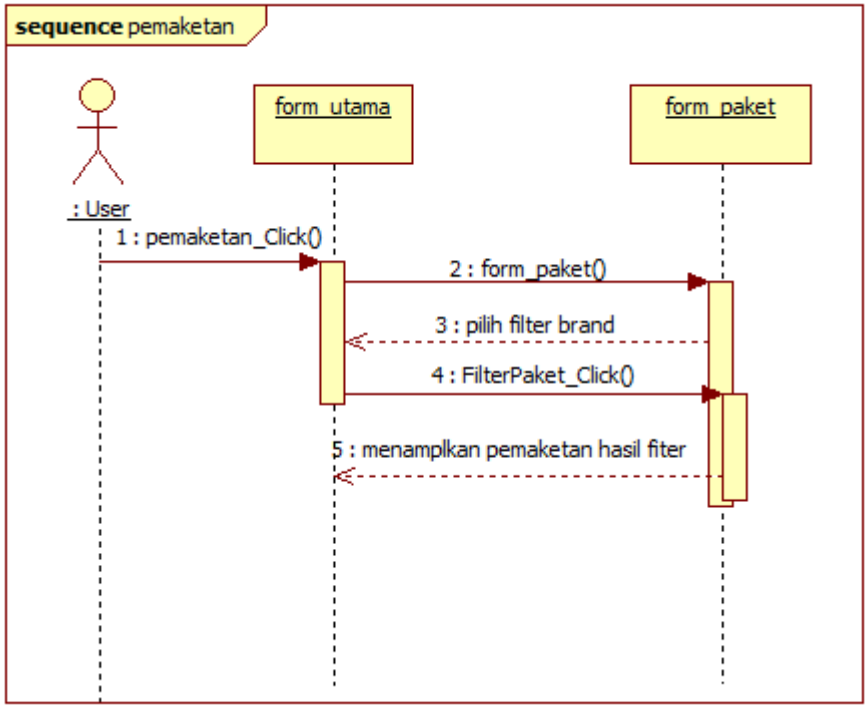
Untuk *sequence diagram* pada proses asoasiasi dapat dilihat pada gambar 3.46



Gambar 3. 46 *Sequence Diagram* Proses Asosiasi

6. *Sequence Diagram* Pemaketan

Untuk *sequence diagram* pada proses pemaketan dapat dilihat pada gambar 3.47.



Gambar 3. 47 *Sequence Diagram* Proses Lihat Rules

3.2 Perancangan Sistem

Perancangan sistem merupakan tahap untuk melakukan perencanaan rancangan terhadap sistem yang akan dibangun, tahapan taahapannya sebagai berikut ;

3.2.1 Perancangan Kelas

Berikut ini adalah perancangan kelas yang digunakan pada sistem yang akan dibangun.

Tabel 3.49 Struktur Kelas Form_Utama

Class : Form_Utama		
Nama	**Jenis**	**Type**
Form_Utama()	Method	Public
Form_Utama_Load()	Method	Private
sbImportData_Click()	Method	Private
sbPreprocessing_Click()	Method	Private
btn_Asosiasi_Click()	Method	Private
btn_bentuk_paket()	Method	Private

Tabel 3.50 Struktur Kelas Import_data

Class : Import_data		
Nama	Jenis	Type
Import_data()	Method	Public
btnCari_Click()	Method	Private
generate()	Method	Private
simpan_data()	Method	Private
backgroundWorker1_DoWork()	Method	Private
backgroundWorker1_ProgressChanged()	Method	Private
backgroundWorker1_RunWorkerCompleted()	Method	Private
Import_Data_Load()	Method	Private
btnSimpan_Click()	Method	Private
	Method	Private

Tabel 3.51 Struktur Kelas Import_dao

Class : Import_dao		
Nama	Jenis	Type
import_dao()	Atribut	Public
hapusDataAwal()	Atribut	Public
hapusDataKode()	Method	Private
hapusDataFrekuensi()	Method	Private
hapusDataJoin()	Method	Private
hapusDatacleaning()	Method	Public

Tabel 3.52 Struktur Kelas preprocessing

Class : preprocessing		
Nama	**Jenis**	**Type**
Preprocessing_Data()	Method	Public
btnAtribut_Click()	Method	Private
btnCleaning_Click()	Method	Private
btnFrequency_Click()	Method	Private
sortingData()	Method	Private

Tabel 3.53. Struktur Kelas Form_asosiasi

Class : form_asosiasi		
Nama	**Jenis**	**Type**
fp*Tree* Global: Fp*Tree*	Atribut	Private
_minConf: double	Atribut	Private
_conf: double	Atribut	Private
_frequent*Item*s: List	Atribut	Private
_*item*SetGlobal: List	Atribut	Private
minSupCount: int	Atribut	Private
totalAb: int	Atribut	Public
totaAb: int	Atribut	Public
form_asosiasi()	Atribut	Public
form_asosiasi_Load()	Method	Private

Nama	Jenis	Type
btnProses_Click()	Method	Private
GenerateFrequent*Item*Sets()	Method	Private
Mine()	Method	Private
backgroundWorker1_DoWork()	Method	Private
backgroundWorker1_ProgressChanged()	Method	Private
backgroundWorker1_RunWorkerCompleted()	Method	Private

Tabel 3.54 Struktur Kelas Form_paket

Class : form_paket		
Nama	**Jenis**	**Type**
Frequent*Item*s: List	Method	Private
*Item*SetGlobal: List	Method	Private
form_asosiasi_Load()	Method	Private
form_paket()	Method	Private
sbCari_Click()	Method	Private

Tabel 3.55 Struktur Kelas CFp*Tree*

Class : CFp*Tree*		
Nama	**Jenis**	**Type**
_root: *node*	Atribut	Private
_headerTable: IDictionary	Atribut	Private
minSupCount: int	Atribut	Private
Frequent*Item*s: List	Atribut	Public
trxList: List	Method	Private
GetTotal*Support*Count: Int	Method	Private
CFp*Tree* ()	Method	Private
InsertTransaction()	Method	Public
CalculateFrequent*Item*s()	Method	Private
InsertBranch()	Method	Private
GetTotal*Support*Count()	Method	Private

Tabel 3.56 Struktur Kelas *Node*

Class : *Node*		
Nama	**Jenis**	**Type**
symbol: string	Atribut	Public
FpCount: Int	Atribut	Public
Parent: *node*	Atribut	Public
NextHeader: *node*	Method	Private
Children: List	Method	Private
Symbol: string	Method	Public
Node ()	Method	Private
-*Node* ()	Method	Public
IsNull()	Method	Private
AddChild()	Method	Private

Tabel 3.57 Struktur Kelas *Item*

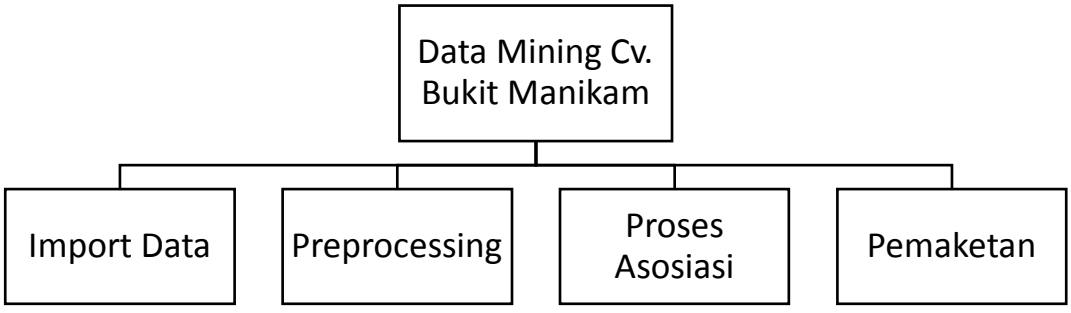
Class : *item*		
Nama	**Jenis**	**Type**
symbol: string	Atribut	Public
*Support*Count: int	Method	Public
Symbol: string	Method	Public
kode: String	Method	Public
-*Item*()	Method	Private
Item()	Method	Public
Clone()	Method	Public

Tabel 3.58 Struktur Kelas *Item*set

Class : *Item*set		
Nama	**Jenis**	**Type**
kode: String	Atribut	Public
*Support*Count: Int	Method	Public
Nobill: String	Method	Public
Coda: Int	Method	Public
*Item*Set()	Method	Public
Add*Item*()	Method	Public
GetLast*Item*()	Method	Public
Clone()	Method	Public

3.2.2 Perancangan Struktur Menu

Dalam pembangunan perangkat lunak diperlukan sebuah struktur menu untuk menunjukkan keterkaitan dan tingkatan antara menu utama dengan sub menu, sehingga lebih mudah dipahami. Berikut adalah struktur menu dari perangkat lunak yang akan dibangun yang diacu pada gambar 3.48



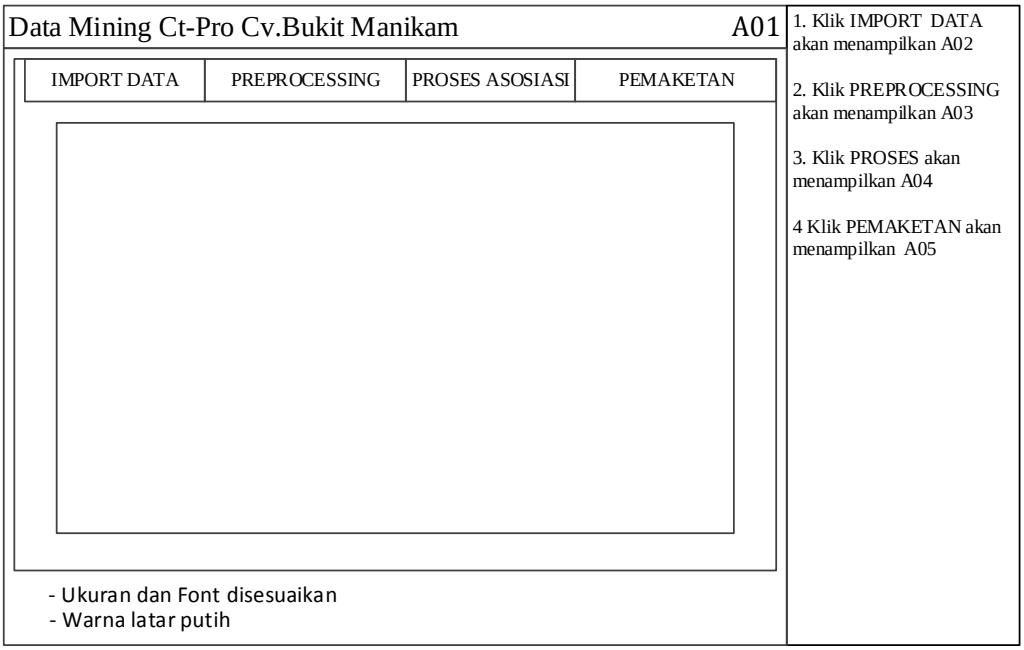
Gambar 3. 48 Struktur Menu

3.2.3 Perancangan Antar Muka

Perancangan antar muka berfungsi untuk memberi gambaran tentang pernagkat lunak yang akan dibangun, agar lebih memudahkan dalam pembuatan perangkat lunak nantinya. Adapun perancangan antar muka untuk perangakt lunak yang akan dibangun akan dijelaskan berikut ini :

1. Perancangan Antar Muka Menu Utama

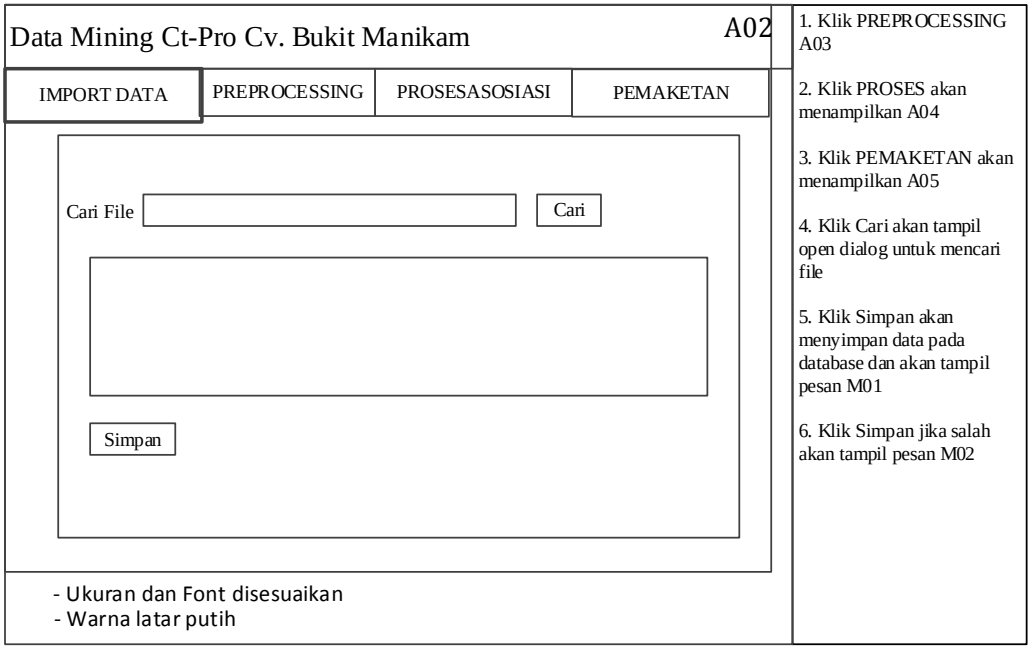
   Perancangan antar muka menu utama dapat dilihat pada gambar 3.49.



Gambar 3. 49 Perancangan Antar Muka Menu Utama

2. Perancangan Antar Muka Import Data

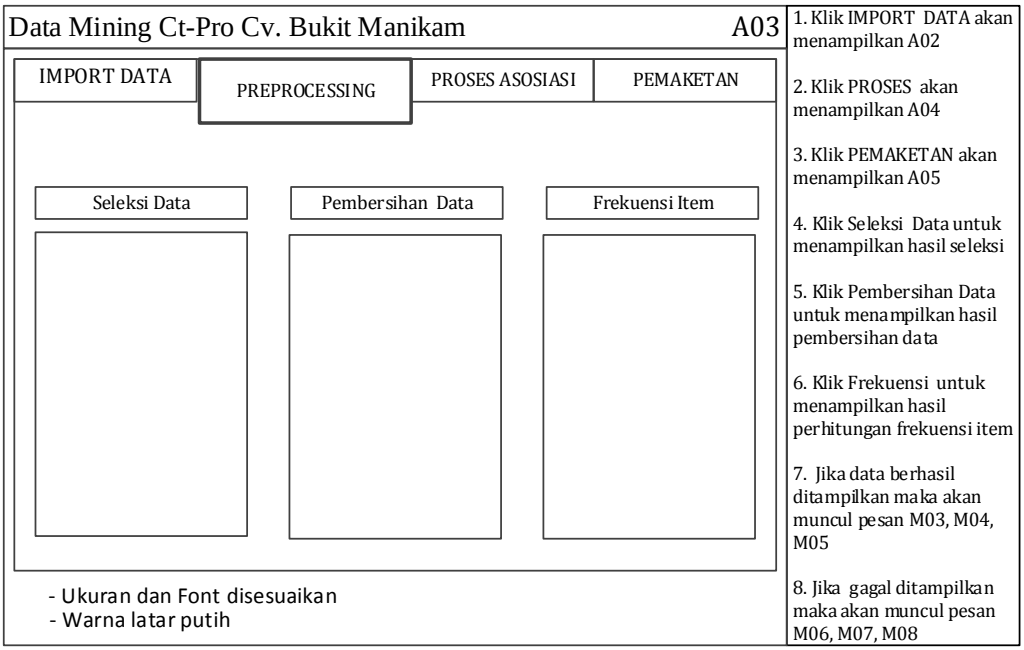
Perancangan antar muka import data dapat dilihat pada gambar 3.50



Gambar 3. 50 Perancangan Antar Muka Import Data

3. Perancangan Antar Muka *Preprocessing*

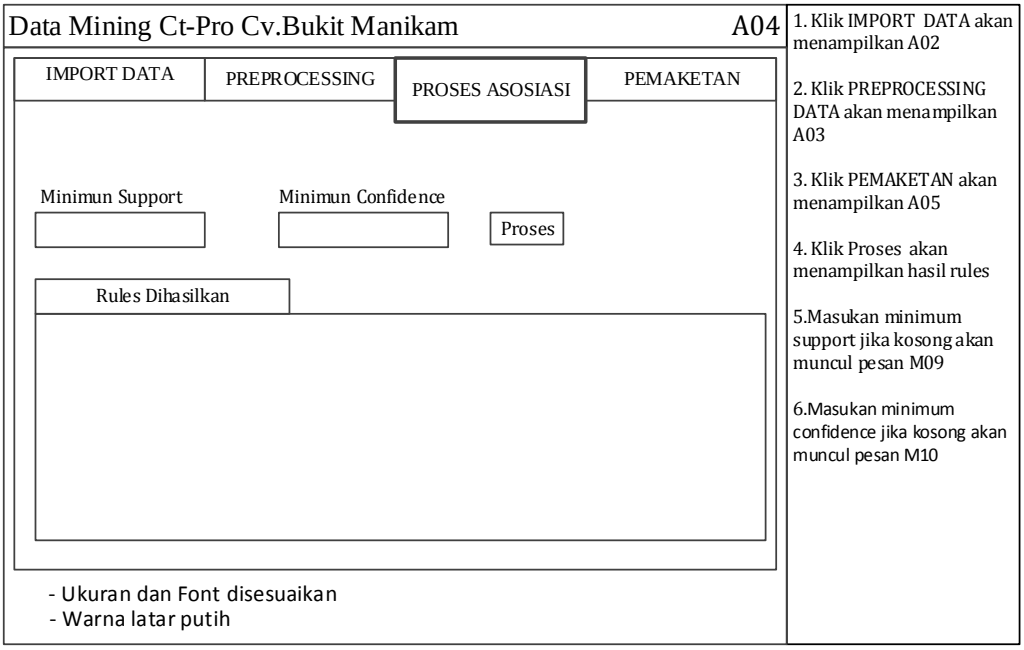
Perancangan antar muka *preprocessing* dapat dilihat pada gambar 3.51



Gambar 3. 51 Perancangan Antar Muka *Preprocessing*

4. Perancangan Antar Muka Asosiasi

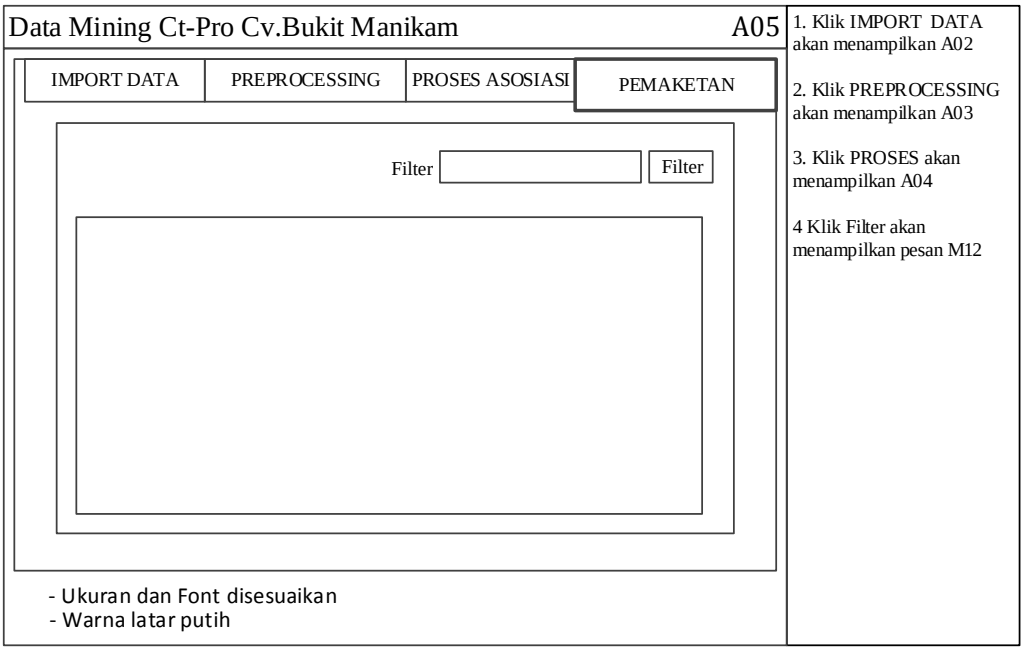
Perancangan antar muka asosiasi dapat dilihat pada gambar 3.52



Gambar 3. 52 Perancangan Antar Muka Asosiasi

5. Perancangan Antar Muka Pemaketan

Perancangan antar muka asosiasi dapat dilihat pada gambar 3.53



Gambar 3. 53 Perancangan Antar Muka Pemaketan

3.2.4 Perancangan Pesan

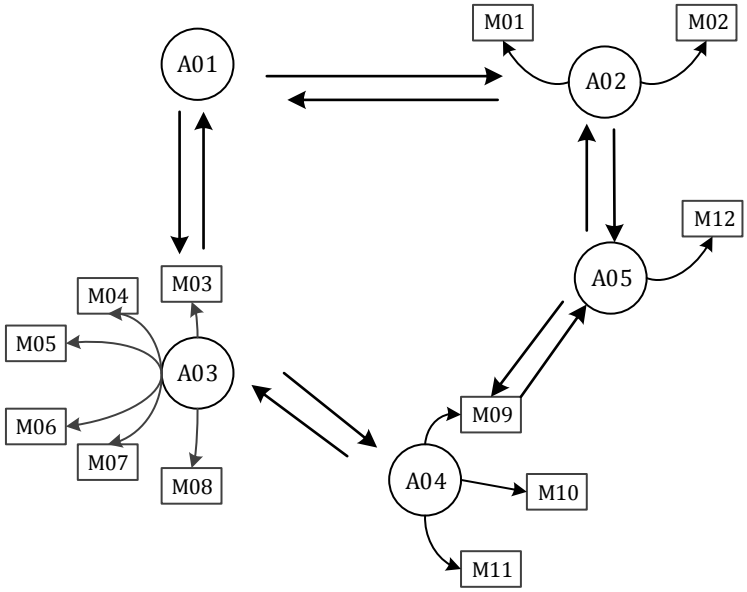
Adapun perancangan pesan dapat dilihat pada tabel 3.59

Tabel 3. 59 Perancangan Pesan

Kode Pesan	Pesan	No. Form
M01	Data Berhasil Diimport	A02
M02	Format *File* Salah	A02
M03	Seleksi Atribut Berhasil	A03
M04	Pembersihan Data Berhasil	A03
M05	Hasil Hitung *Support* Count Berhasil	A03
M06	Seleksi Atribut Gagal	A03
M07	Pembersihan Data Gagal	A03
M08	Hasil Hitung *Support* Count Gagal	A03
M09	*Minimum Support* tidak boleh kosong	A04
M10	*Minimum Confidence* tidak boleh kosong	A04
M11	Rules tidak d*item*ukan	A04
M12	Filtering Tidak d*item*ukan	A05

3.2.5 Jaringan Semantik

Jaringan semantik bertujuan untuk merepresentasikan pemetaan data untuk mencegak duplikasi data. Adapun jaringan semantik yang menghubungkan antarmuka pada aplikasi yang akan dibangun dapat dilihat pada gambar 3.54



Gambar 3. 54 Jaringan Semantik