BAB II
TINJAUAN PUSTAKA

II.1 Tinjauan Tempat Penelitian

Pada bagian tinjauan tempat penelitian ini akan dijelaskan beberapa hal mengenai tempat penelitian. Tinjauan penelitian meliputi profil perusahaan, logo perusahaan, visi dan misi, serta struktur organisasi perusahaan.

II.1.1 Profil Perusahaan

Maternal Disaster merupakan Perusahaan yang berjalan di bidang produksi dan penjualan pakaian jadi seperti T-Shirt, Jaket, Sweaters, Celana, Kemeja, Rompi, Dompet, Sabuk, Topi, Sandal, Tas, dll, dengan system penjualan retail, konsinyasi dan beli putus. berdiri sejak tahun 2003 hingga sekarang berkembang, tidak hanya pakaian jadi, maternal juga memproduksi merchandise, percetakan dan berbagai macam aksesoris. kualitas produk sangat diutamakan di dukung dengan sumber daya yang berkualitas dan berpengalaman, Maternal Disaster beralamat di Jl.Sidomukti No.26, Sukaluyu, Bandung. Saat ini Maternal beberapa outlet di kota besar di Indonesia.

Pada saat ini Maternal Disaster mengadakan perubahan manajemen dalam rangka lebih memfokuskan potensi dalam pengadaan produk dalam jumlah besar demi memenuhi permintaan pasar yang masih sangat potensial perkembangannya.

II.1.2 Logo Perusahaan

Logo dari Maternal Disaster dapat dilihat pada Gambar II.1



Gambar II. 1 Logo Perusahaan

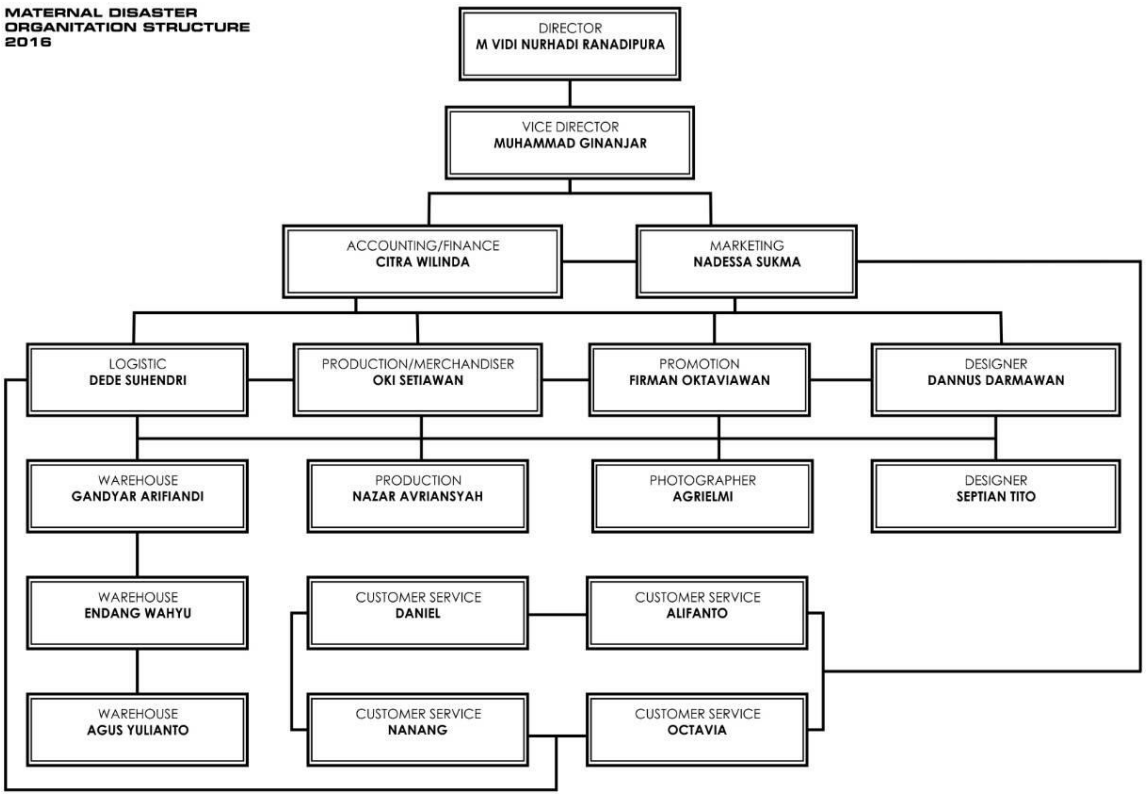
II.1.3 Visi dan Misi

Berikut ini merupakan visi dan misi dari Maternal Disaster :

- Visi distro adalah Memperluas dan mempertahankan usaha yang sedang di jalankan, serta mencari keuntungan sebesar mungkin dengan modal seminim mungkin.
- Misi distro adalah mepertahankan produk asli dan membuat produk sesuai dengan perkembangan jaman.

II.1.4 Struktur Organisasi Perusahaan

Struktur organisasi Maternal Disaster dapat dilihat pada Gambar II.2.



Gambar II. 2 Struktur Organisasi Perusahaan

Keterangan posisi :

1. *Director* atau Direktur Utama

   Direktur Utama mempunyai tugas untuk menawarkan visi dan misi, memimpin rapat umum, mengambil keputusan, menjalankan tanggung jawab, melakukan koordinasi peningkatan mutu pelayanan, penelitian, pengabdian dan kerja sama, bertanggung jawab atas semua proses bisnis perusahaan.
2. *Vice Director* atau Wakil Direktur Utama

   Wakil Dirut bertugas utama menyelesaikan permasalahan perusahaan yang tidak bisa di kerjakan oleh direktur utama seperti masalah-masalah internal lainnya.
3. *Accounting Finance* atau Keuangan

   Bertugas untuk mengatur pengeluaran dan pemasukan perusahaan dan membuat laporan yang bersangkutan dengan kegiatan perusahaan.
4. *Marketing*

   Bertugas untuk melakukan perencanaan strategi pemasaran dengan memperhatikan trend pasar dan sumber daya distro, serta mengikuti perkembangan pasar terutama pada produk sejenis dari perusahaan distro pesaing.
5. *Logistic*

   Melaksanakan tata administrasi penerimaan dan pengeluaran produk dari dan ke gudang sesuai dengan ketentuan dan prosedur yang ditetapkan.
6. *Production Marchendiser*

   Bertugas dalam perencanaan produksi, membuat rencana pembuatan produk, serta kegiatan lainnya untuk periode tertentu.
7. *Production*

   Menjalankan produksi yang sedang berjalan.
8. *Promotion*

   Bertugas untuk membuat strategi promosi yang baik untuk perusahaan, mendata dan menyimpan laporan dengan rapih setiap kebutuhan promosi

baik produk atau laporan biaya termasuk kontrak kerjasama dengan pihak *fiend* sesuai dengan format yang disepakati.

9. *Designer*

   Bertugas untuk membuat perancangan produk sesuai dengan trend pasar.

10. *Warehouse*

    Bertugas untuk mengatur pengiriman barang, stock barang yang terdapat di gudang.

11. *Customer Service*

    Bertugas untuk melayani pelanggan.

II.2 Landasan Teori

Pada bagian landasan teori ini akan menjelaskan teori-teori yang akan digunakan sebagai acuan dalam penelitian yang akan dilakukan.

II.2.1 Data

Data adalah suatu nilai yang merepresentasikan fakta di dunia nyata dan mewakili suatu objek seperti manusia (pegawai, siswa, pembeli, pelanggan), barang, hewan, peristiwa, konsep, keadaan, dsb. Nilai tersebut lalu direkam dalam bentuk angka, huruf, simbol, teks, gambar, atau kombinasinya [4].

II.2.2 Basis Data

Basis data adalah sekumpulan data yang saling terhubung dan mempunyai skema tertentu. Menurut Fathansyah [4], basis data sendiri dapat didefinisikan dalam sejumlah sudut pandang seperti :

1. Himpunan kelompok data (arsip) yang saling berhubungan yang diorganisasi sedemikian rupa agar kelak dapat dimanfaatkan kembali dengan cepat dan mudah.
2. Kumpulan data yang saling berhubungan yang disimpan secara bersama sedemikian rupa dan tanpa pengulangan (redudansi) yang tidak perlu, untuk memenuhi berbagai kebutuhan.
3. Kumpulan *file* atau tabel atau arsip yang saling berhubungan yang disimpan dalam media penyimpanan elektronis.

II.2.3 Database Management System

Database management system adalah sistem yang digunakan untuk membuat dan mengelola basis data. Tujuan dari *DBMS* adalah menyediakan lingkungan yang mudah dan aman untuk penggunaan dan perawatan *database*.

Dalam *DBMS* pengguna difasilitasi untuk mendefinisikan skema basis data yang disebut sebagai *data definition language (DDL)*. Selain itu pengguna juga dapat membuat, mengubah, dan membuat *query* data yang biasanya disebut dengan *data manipulation language* (DML) [5].

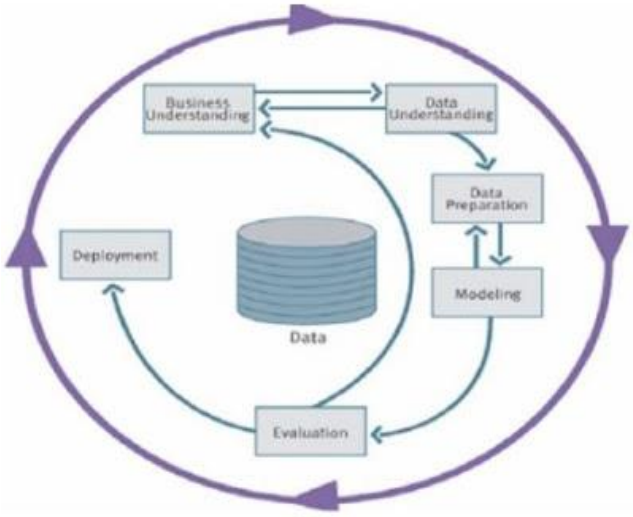
II.2.4 Data Mining

Data *mining* adalah kegiatan mengekstraksi atau menambang pengetahuan dari data yang berukuran atau berjumlah besar. Nama lain yang sering digunakan dan memiliki makna yang sama dengan data mining adalah *knowledge mining from data*, *knowledge extraction*, *pattern analysis*, *data archaeology*, *data dredging*.

Data *mining* berusaha memecahkan masalah dengan menganalisis sekumpulan data yang sudah ada dalam basis data. Sedangkan hasil dari proses dari data mining adalah penemuan pengetahuan, informasi, pola, atau model[6].

II.2.5 Metode CRISP-DM

CRISP – DM (*Cross – Industry Standard of Data Mining*). CRISP – DM merupakan metodologi data *mining* yang digambarkan dengan model proses dan di dalamnya terdapat tugas-tugas yang harus dilakukan untuk melakukan proyek data *mining* [2]. Tahapan dari CRISP – DM dapat dilihat pada gambar II.3 Model Proses CRISP-DM.



Gambar II. 3 Model Proses CRISP-DM

1. Pemahaman Bisnis

   Tahap pertama pada model proses CRISP-DM adalah tahap pemahaman bisnis (*business understanding*). Pada tahap ini, dilakukan pengumpulan informasi mengenai tujuan bisnis dan hasil yang diharapkan oleh manager distro Maternal Disaster. Kemudian dilakukan penyusunan rencana kegiatan yang akan dilakukan untuk mencapai tujuan yang telah ditentukan. Kegiatan pada tahap ini meliputi identifikasi tujuan bisnis, penilaian situasi, penentuan tujuan *data mining*, dan penyusunan rencana proyek.

   a. Identifikasi Tujuan Bisnis
      - Latar Belakang
      - Tujuan Bisnis
      - Kriteria Sukses Bisnis

   b. Penialian Situasi
      - Ketersediaan Sumber Daya
      - Ketersediaan Data
      - Ketersediaan Personel
      - Kebutuhan dan Asumsi

   c. Penentuan Tujuan Data *Mining*

- Tujuan Data *Mining*
- Kriteria Sukses Data *Mining*

2. Pemahaman Data

   Tahap kedua pada model proses CRISP-DM adalah tahap pemahaman data (*data understanding*). Pada tahap ini, dilakukan pengumpulan dan analisis terhadap data yang diperoleh dari distro Maternal Disaster. Kegiatan pada tahap ini meliputi pengumpulan data awal, pendeskripsian data, eksplorasi data, dan verifikasi kualitas data. Pada tahap ini akan dilakukan beberapa aktivitas, yaitu :

   a. Pengumpulan Data Awal

      Pada tahap pengumpulan data awal akan dikumpulkan data-data yang akan digunakan dalam proses *clustering*.

   b. Penjelasan Data

      Pada tahap ini data yang telah dikumpulkan akan dijelaskan apa saja isi dari data tersebut.

   c. Eksplorasi Data

      Pada tahap ini data tersebut akan di eksplorasi untuk dapat membantu tujuan dari data *mining*.

   d. Evaluasi Kualitas Data

      Pada tahap ini data tersebut akan di evaluasi kualitas datanya dengan pengecekan *missing value* dan *outlier*.

3. Persiapan Data

   Tahap ketiga pada model proses CRISP-DM adalah tahap persiapan data (data *preparation*). Pada tahap ini akan dilakukan pengolahan terhadap data yang telah diperoleh (data *preprocessing*). *Data preprocessing* perlu dilakukan untuk meningkatkan kualitas data agar dapat digunakan pada proses pemodelan, terdapat beberapa tugas dalam tahap ini, yaitu:

   a. Pemilihan Data

      Pada tahap ini akan dipilih atribut-atribut yang akan digunakan dalam proses data *mining*.

b. Pembersihan Data

Pada tahap ini data yang telah dipilih akan dibersihkan dari *missing value*.

c. Penyusunan Data

Pada tahap ini data yang telah dibersihkan dipersiapkan untuk digunakan pada pemodelan.

d. Integrasi Data

Pada tahap ini dilakukan penggabungan dua tabel atau lebih yang memiliki informasi berbeda tentang objek yang sama kedalam set data baru.

4. Pemodelan

Tahap pemodelan merupakan kegiatan utama dari proses penyelesaian masalah data *mining*. Data yang sudah disusun akan diproses menggunakan suatu metode sehingga dapat menghasilkan model. Kegiatan pada tahap pemodelan meliputi :

a. Teknik Pemodelan
b. Mekanisme Model
c. Penggunaan Teknik Pemodelan

5. Evaluasi

Evaluasi adalah tahap interpretasi terhadap hasil data *mining*. Evaluasi dilakukan secara mendalam denga tujuan agar hasil pada tahap *modelling* seuai dengan sasaran yang ingin dicapai dalam tahap *business understanding*.

a. Evaluasi Hasil
b. Peninjauan Proses
c. Menentukan Langkah Selanjutnya

6. Deployment

Deployment merupakan tahapan membuat laporan hasil kegiatan data *mining*. Laporan akhir mengenai pengetahuan yang didapat atau

pengenalan pola pada data dalam proses data *mining* dan dipresentasikan dalam bentuk grafik atau deskriptif yang mudah dipahami.

II.2.6 Data *Preprocessing*

Dalam data *mining*, kualitas data yang akan digunakan perlu diperhatikan. Ada beberapa faktor yang mempengaruhi kualitas data, antara lain keakuratan, keutuhan, konsistensi, aktualitas, dan penafsiran. Data *preprocessing* dapat memperbaiki kualitas data, sehingga dapat meningkatkan keakuratan dan efisiensi hasil dari data *mining* [6]. Beberapa kegiatan data *preprocessing* yang akan dilakukan dalam penelitian ini adalah sebagai berikut :

a. Penanganan *Missing value*

   Missing value pada dasarnya tidak bermasalah bagi keseluruhan data, apalagi jika jumlahnya hanya sedikit. Namun jika persentase data yang hilang tersebut cukup besar, maka perlu dilakukan pengujian apakah data yang mengandung banyak *missing value* masih layak diproses lebih lanjut ataukah tidak. Salah satu cara menangani *missing value* adalah dengan menghitung nilai rata-rata dari objek yang nilainya hilang, nilai rata-rata tersebut akan dijadikan nilai pengganti dari nilai yang hilang. Selain itu *missing value* juga bisa ditangani dengan memberikan nilai *default* pada setiap nilai yang hilang.

b. Pembangunan Atribut

   Pembangunan atribut (*atribute construction*) merupakan proses pembentukan atribut-atribut baru dan atribut yang sudah ada. Teknik tersebut digunakan untuk membantu dalam proses pembentukan model. Penurunan atribut juga berguna untuk menyeragamkan tipe data dan atribut, sehingga akan lebih mudah dalam proses pemodelan.

II.2.7 *Clustering*

Clustering adalah proses mengelompokkan objek berdasarkan informasi yang diperoleh dari data yang menjelaskan hubungan antar objek dengan prinsip untuk memaksimalkan kesamaan antar anggota satu kelas dan meminimumkan

kesamaan antar kelas atau *cluster* [7]. Tujuannya menemukan *cluster* yang berkualitas dalam waktu yang layak. *Clustering* dalam data *mining* berguna untuk menemukan pola distribusi di dalam sebuah data set yang berguna untuk proses analisa data. Kesamaan objek biasanya diperoleh dari kedekatan nilai-nilai atribut yang menjelaskan objek-objek data, sedangkan objek-objek data biasanya direpresentasikan sebagai sebuah titik dalam ruang multidimensi.

Dengan menggunakan *clustering*, dapat diidentifikasi daerah yang padat, pola-pola distribusi secara keseluruhan dan keterkaitan yang menarik antara atribut-atribut data. Dalam data *mining* usaha difokuskan pada metode-metode penemuan untuk *cluster* pada basis data berukuran besar secara efektif dan efisien. Secara garis besar ada beberapa kategori algoritma *clustering* yang dikenal yaitu:

a. Metode Partisi (*Partition-based Clustering*)

   Dalam metode ini pengguna harus menentukan jumlah k partisi yang diinginkan lalu setiap data dites untuk dimasukkan pada salah satu partisi sehingga tidak ada data yang *overlap* dan satu data hanya memiliki satu *cluster*. Contohnya: algoritma *K-Means*, *mixture modeling*.

b. Metode Hierarki (*Hierarchical Clustering*)

   Membuat hierarki berupa dendogram di mana data yang mirip akan disimpan di hierarki yang berdekatan. Metode ini terbagi menjadi dua yaitu *buttom-up* yang menggabungkan *cluster* kecil menjadi *cluster* lebih besar dan *top-down* yang memecah *cluster* besar menjadi *cluster* yang lebih kecil. Contohnya: *Agglomerative Hierarchical Clustering* (AHC), *Divisive Hierarchical Clustering* (DHC).

Dalam metode *clustering* hal yang penting adalah mengukur kemiripan atau ketidakmiripan data yang akan dikelompokkan. *Similaritas* (kemiripan) antara dua objek merupakan ukuran numerik dari seberapa mirip dua buah objek. *Similaritas* mempunyai nilai yang lebih tinggi jika objek-objek tersebut sama. *Range* nilai *similaritas* adalah [0,1]. *Dissimilaritas* (ketidakmiripan) antara dua objek merupakan ukuran numerik dari seberapa berbeda dua buah objek. *Dissimilaritas* bernilai lebih rendah jika objek-objek tersebut mirip. Minimum

dissimilaritas adalah 0 dan batas atasnya bervariasi. *Dissimiliratas* sering disebut juga dengan *distance* (jarak) [8]. Salah satu cara pengukuran jarak adalah menggunakan jarak *Euclidian* (*Euclidian Distance*), yang dinyatakan dengan persamaan (2.1) berikut :

$$d = \sqrt{\sum_{i=1}^{r}(xi - yi)^2} \qquad \text{Persamaan (2.1)}$$

Di mana :

d = jarak

r = jumlah fitur dalam vektor

xi = nilai fitur x ke i

yi = nilai fitur y ke i

II.2.7.1 Teknik *Clustering*

Tujuan utama metode *cluster* adalah pengelompokkan sejumlah data atau onjek kedalan *cluster* (kelompok) sehingga dalam setiap *cluster* akan berisi data yang semirip mungkin.

Terdapat dua pendekatan dalam teknik *clustering*, yaitu partisi dan hirarki. Dalam teknik partisi akan mngelompokkan onjek *X1, X2, X3,...Xn* kedalam k *cluster*. Ini bisa dilakukan dengan menentukan pusat *cluster* awal, lalu dilakukan relokasi objek berdasarkan kriteria tertentu sampai dicapai pengelompokkan yang optimum.

Dalam *cluster* hirarki dimulai dengan membuat m *cluster* dimana setiap *cluster* beranggotakan satu objek dan berakhir dengan satu *cluster* dimana anggotanya dalah m objek. Penghentian atau *cut-off* bisa dilakukan ketika *cluster* yang diinginkan sudah tercapai [4].

II.2.7.2 Analisis *Cluster* Berbasis Hirarki

Dalam statistik, pengelompokkan berbasis hirarki adalah metode analisis *cluster* yang berushaa untuk membangun sebuah hirarki *cluster*. Strategi untuk pengelompokkan berbasis hirarki pada umumnya terdapat dua jenis, yaitu *agglomerative* dan *divisive*.

Agglomerative merupakan sebuah pengelompokkan berbasis hirarki dengan pendekatan *bottom up,* yaitu proses pengelompokkan dimulai dari masing-masing data sebagai satu buah *cluster*, kemudian secacra rekursif mencari *cluster* terdekat sebagai pasangan untuk bergabung sebgai *cluster* yang llebih besar. Proses tersebut diulang terus menerus sehingga tampak bergerak keatas membentuk hirarki.

Divisive merupakan metode pengelompokkan data berbasis hirarki dengan pendekatan *top down*, yaitu proses pengelompokkan dimulai dari satu *cluster* yang berisi semua data, kemudian secara rekursif memecah *cluster* menjadi dua *cluster* sampai *clusteri* hanya berisi data tunggal (data itu sendiri). Untuk cara iini, yang dibutuhkan adlah keputusan *cluster* yang manakah yang akan dipecah pada setiap langkah dan bagaimana cara memecahkannya.

II.2.7.3 *Agglomerative Hierachical Clustering* (AHC)

Pengelompokan data menggunakan *agglomerative hierarchical clustering* (AHC) diawali dengan merepresentasikan setiap objek pada data sebagai satu kelompok, kemudian dilakukan perhitungan jarak (*distance measure*) antar kelompok tersebut. Setelah itu dua kelompok yang memiliki jarak terdekat digabungkan menjadi sebuah kelompok baru. Kemudian jarak antara kelompok yang baru dengan kelompok lain dihitung menggunakan salah satu metode perhitungan jarak antar kelompok (*single linkage*, *complete linkage*, *average linkage*, dll.) [9]. Proses perhitungan jarak dan penggabungan dua kelompok dilakukan secara iteratif. Berikut ini merupakan beberapa metode yang dapat digunakan untuk menghitung jarak antar kelompok :

a. *Single Linkage* (Jarak Terdekat)

   Single linkage disebut juga *minimum link*, di mana similaritas dari dua *cluster* didasarkan pada dua titik paling dekat dari dua *cluster* yang berbeda. Untuk mencari *single linkage* dapat menggunakan persamaan (2.2) berikut ini :

$$duv=\min\{duv\},duv \in D \qquad \text{Persamaan (2.2)}$$

b. *Complete Linkage* (Jarak Terjauh)

Complete linkage disebut juga *maximum link*, di mana jarak antara dua buah *cluster* dihitung dari jarak terjauh antara anggota *cluster* satu dengan *cluster* kedua. Untuk mencari *complete linkage* dapat menggunakan persamaan (2.3) berikut ini :

$$duv=\max\{duv\}, duv \in D \qquad \text{Persamaan (2.3)}$$

c. *Average Linkage* (Jarak Rata-rata)

Average linkage disebut juga *group average*, di mana jarak antara dua buah *cluster* dihitung dari rata-rata jarak antara anggota *cluster* satu dengan *cluster* yang kedua. Untuk mencari *average linkage* dapat menggunakan persamaan (2.4) berikut ini :

$$duv=\text{average}\{duv\}, duv \in D \qquad \text{Persamaan (2.4)}$$

Berikut ini merupakan langkah-langkah dalam algoritma *Agglomerative Hierarchical Clustering* (AHC) [10]:

1. Mempresentasikan setiap outlet sebagai satu kelompok, kemudian menghitung jarak (*distance measure*) antara kelompok-kelompok tersebut.
2. Cari dua *cluster* yang mempunyai jarak paling dekat dengan menggunakan metode *complite linkage* (terdapat pada persamaan (2.3).
3. Ulangi langkah kesatu dan kedua sampai tersisa satu kelompok (kumpulan semua outlet) dan berhenti ketika jumlah *cluster* yang terbentuk sama dengan 1.
4. Ketika jumlah *cluster* berjumlah 1, maka tahap ke-3 telah selesai.

Pemotongan dendrogram dapat dilakukan dengan berdasarkan pada *agglomeration schedule*. Jumlah *cluster* bisa didapatkan dengan mencari perbedaan koofisien jarak dari setiap iterasinya.

II.2.7.4 Contoh Penerapan *Clustering* Menggunakan Algoritma AHC

Contoh diberikan 5 data dan pengukuran jarak menggunakan jarak *Euclidean*, dengan menggunakan metode *single linkage.*

Data ke –i	Fitur x	Fitur y
1	1	1
2	4	1
3	6	1
4	1	2
5	2	3

Yang pertama dilakukan adalah menghitung jarak dengan menggunakan *euclidean* pada semua pasangan dua data. Hasil dari perhitungan dengan menggunakan *euclidean* adalah sebagai berikut :

$$D = d(U,V) = \begin{bmatrix} 0 & 3 & 1 & 5 & 7 \\ 3 & 0 & 4 & 4 & 4 \\ 1 & 4 & 0 & 4 & 6 \\ 5 & 4 & 4 & 0 & 2 \\ 7 & 4 & 6 & 2 & 0 \end{bmatrix}$$

Dengan memperlakukan setiap data sebagai *cluster* maka *cluster* yang terbentuk pada saat ini adalah sebanyak 5 *cluster* atau jumlah *cluster K* = jumlah data *n*, tahapan selanjutnya adalah memilih jarak dua *cluster* yang paling kecil. min$\{d(U,V)\}$=d13=1

Dari proses perhitungan dengan menggunakan metode *single linkage* maka didapatkan bawha jarak yang paling minimum adalah *cluster* 1 dan *cluster* 3, maka kedua *cluster* ini akan digabung. Untuk melanjutkan tingkat pengelompokan berikutnya maka jarak – jarak antara *cluster* (13) dengan *cluster* yang tersisia 2,4, dan 5 dihitung kembali dengan menggunakan metode *single linkage*. Jarak – jarak yang didapatkan adalah:

d(13)2 = min {d12 ,d(32)} = min {3,4} = 3

d(13)4 = min {d14 ,d(34)} = min {5,4} = 4

d(13)5 = min {d15 ,d(35)} = min {7,6} = 6

Dengan menghapus baris – baris dan kolom – kolom matriks D yang bersesuaian dengan *cluster* 1 dan 3 dan menambahkan baris dan kolom untuk *cluster* 1 dan 3, dan menambahkan baris dan kolom untuk *cluster* (13), maka didapatkan matriks yang baru :

$$D = d(U,V) = \begin{matrix} (13) \\ 2 \\ 4 \\ 5 \end{matrix} \begin{bmatrix} 0 & 3 & 4 & 6 \\ 3 & 0 & 4 & 4 \\ 4 & 4 & 0 & 2 \\ 6 & 4 & 2 & 0 \end{bmatrix}$$

Tahapan selanjutnya adalah memilih kembali jarak dua *cluster* yang paling kecil.

$$\min\{d(U,V)\} = d45 = 2$$

Maka *cluster* yang terpilih adalah *cluster* 4 dan 5, maka *cluster* 4 dan *cluster* 5 digabung. Kemudian untuk menghitung jarak – jarak antara *cluster* (45) dengan *cluster* lain yang tersisa yaitu (13) dan 2 dihitung kembali dengan menggunakan metode *single linkage*. Jarak – jarak yang didapatkan adalah :

d(45)(13) = min {d41 ,d43, d51, d53)} = min {5,4,7,6} = 4

d(45)2 = min {d42 ,d(52)} = min {4,4} = 4

Dengan menghapus baris – baris dan kolom – kolom matriks D yang bersesuaian dengan *cluster* 4 dan *cluster* 5 dan menambahkan baris dan kolom untuk *cluster* (45), maka didapatkan matriks jarak yang baru :

$$D = d(U,V) = \begin{matrix} (45) \\ (13) \\ 2 \end{matrix} \begin{bmatrix} 0 & 4 & 4 \\ 4 & 0 & 3 \\ 4 & 3 & 0 \end{bmatrix}$$

Selanjutnya dipilih kembali jarak dua *cluster* yang paling kecil. min{$d(U,V)$}=d(13)2=3

Terpilih *cluster* (13) dan 2, maka *cluster* (13) dan 2 digabung. Untuk melanjutkan tingkat *clustering* berikutnya, maka jarak – jarak antara *cluster* (13) dan 2 dengan *cluster* yang lain yang tersisisa yaitu (45) dihitung kembali dengan menggunakan metode *single linkage*. Jarak – jarak yang didapatkan adalah :

d(123)(45) = min {d14 ,d43, d24, d25, d34, d35)} = min {5,7,4,4,4,6} = 4

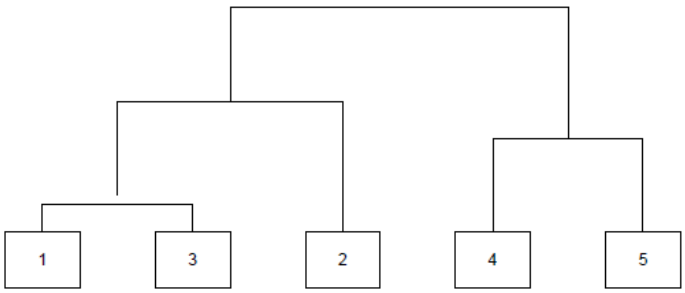
Dengan menghapus baris – baris dan kolom – kolom matriks D yang bersesuaian dengan *cluster* (13) dan 2, dan menambahkan baris dan kolom untuk *cluster* (123), maka matriks yang baru yang didapatkanya adalah :

$$D = d(U,V) = \begin{matrix}(123)\\(45)\end{matrix}\begin{bmatrix}0 & 4\\4 & 0\end{bmatrix}$$

Jadi *cluster* (123) dan (45) digabung membentuk *cluster* tunggal dari semua 5 data, (12345), ketika jarak terdekat mencapai 4. Hasil *clustering* yang didapatkan dapat ditampilkan dalam bentuk *dendogram*.

II.2.7.5 *Dendogram*

Dendogram atau struktur pohon biasanya digunakan untuk merepresentasikan proses *clustering* berbasis hirarki, yang menunjukan bagaimana suatu objek dikelompokan dalam setiap langkahnya. Berikut ini merupakan contoh dari *dendogram* dapat dilihat pada Gambar II.4 :



Gambar II. 4 Dendogram Hasil *Clustering* Berbasis Hirarki

II.2.8 *Agglomerative Schedule*

Agglomeration schedule adalah ringkasan numerik dari solusi *cluster* yang dihasilkan dari proses *clustering* menggunakan metode hierarki. [11] Berikut ini merupakan contoh dari *agglomeration schedule* :

Stage	Cluster Combined		Coefficients	Stage Cluster First Appears		Next Stage
	Cluster 1	Cluster 2		Cluster 1	Cluster 2	
1	8	11	1.260	0	0	7
2	6	7	1.579	0	0	4
3	2	9	1.625	0	0	7
4	1	6	2.318	0	2	6
5	3	5	2.619	0	0	8
6	1	10	3.670	4	0	10
7	2	8	4.420	3	1	8
8	2	3	4.505	7	5	9
9	2	4	4.774	8	0	10
10	1	2	5.718	6	9	0

Gambar II. 5 Contoh *Agglomeration Schedule*

Penjelasan tabel *agglomeration schedule* :

1. *Stage* merupakan tahapan atau iterasi yang telah dilakukan pada proses *clustering* menggunakan metode hierarki.
2. *Cluster combined* merupakan *cluster* yang digabungkan pada setiap *stage*nya.
3. *Cooficients* merupakan jarak dari dua *cluster* yang digabungkan pada setiap *stagenya* yang dicari dengan menggunakan rumus jarak.
4. *Stage cluster first appear* merupakan *stage* apabila *cluster* yang digabungkan telah digabungkan dengan *cluster* lain pada *stage* sebelumnya. Jika *cluster* yang digabungkan belum pernah digabungkan pada *stage* sebelumnya, maka akan diisi dengan nilai 0.

Next stage merupakan *stage* selanjutnya saat *cluster* akan digabungkan dengan *cluster* lain. [11].

II.2.9 Object Oriented Programing

Pemograman berorientasi objek adalah suatu strategi pembangunan perangkat lunak yang mengorganisasikan perangkat lunak sebagai kumpulan objek yang berisi data dan operasi yang diberlakukan terhadapnya. Menggunakan pendekatan berorientasi objek akan memandang sistem yang akan dikembangkan sebagai suatu kumpuln objek yang berkorespondensi dengan objek dunia nyata.

Dalam rekayasa perangkat lunak, konsep pendekatan berorientasi objek dapat diterapkan pada tahap analisis, perancangan, pemograman, dan pengujian perangkat lunak. Berikut adaah beberapa konsep dasar yang harus dipahami pemograman berorientasi objek [12] :

1. Kelas (*Class*)

   Kelas adalah kumpulan objek-objek dengan karakteristik yang sama. Kelas merupakan definisi statik dan himpunan objek yang sama yang mungkin lahir atau diciptakan dari kelas tersebut. Sebuah kelas akan mempunyai sifat (atribut), kelakuan (operasi/metode), hubungan (*relationship*) dan arti. Suatu kelas dapat diturunkan dan kelas semula dapat diwariskan ke kelas yang baru.

2. Objek (*Object*)

   Objek adalah abstraksi dan sesuatu yang mewakili dunia nyata seperti benda, manusia, satuan organisasi, tempat, kejadian, strutur, status, atau hal-hal lain yang bersifat abstrak. Objek merupakan suatu entitas yang mampu menyimpan informasi (status) dan mempunyai operasi (kelakuan) yang dapat diterapkan atau dapat berpengaruh pada status objeknya. Objek mempunyai siklus hidup yaitu diciptakan, dimanipulasi, dan dihancurkan.

3. Metode (*Method*)

   Operasi atau metode pada sebuah kelas hampir sama dengan fungsi atau prosedur pada terstruktur. Sebuah kelas boleh memiliki lebih dari satu metode atau operasi. Metode atau operasi yang berfungsi untuk memanipulasi objek itu sendiri.

4. Atribut (*Attribute*)

   Atribut dari sebuah kelas adalah variabel global yang dimiliki sebuah kelas. Atribut dapat berupa nilai atau elemen-elemen data yang dimiliki oleh objek dalam kelas objek. Atribut secara individual oleh sebuah objek, misalnya berat, jenis, nama, dan sebagainya.

5. Abstraksi (*Abstraction*)

   Prinsip untuk merepresentasikan dunia nyata yang kompleks menjadi satu bentuk model yang sederhana dengan mengabaikan aspek-aspek lain yang tidak sesuai dengan permasalahan.

6. Enkapulasi (*Encapsulation*)

   Pembungkusan atribut data dan layanan (operasi-operasi) yang dipunyai objek untuk menyembunyikan implementasi dan objek sehingga objek lain tidak mengetahui cara kerja.

7. Pewarisan (*Inheritance*)

   Mekanisme yang memungkinkan satu objek mewarisi sebagian atau seluruh definisi dan objek lain sebagai bagian dari dirinya.

8. Antarmuka (*Interface*)

   Antarmuka atau interface sangat mirip dengan kelas, tetapi tanpa atribut kelas dan tanpa memiliki metode yang dideklarasikan. Antarmuka biasanya digunakan agar kelas lain tidak langsung mengakses ke suatu kelas.

9. *Reusability*

   Pemanfaatan kembali objek yang sudah didefinisikan untuk suatu permasalahan pada permasalahan lainnya yang melibatkan objek tersebut.

10. Generalisasi dan Spesialisasi

    Menunjukkan hubungan antara kelas dan objek yang umum dengan kelas dan objek yang khusus. Misalnya kelas yang lebih umum (generalisasi) adalah kendaraan darat dan kelas khususnya (spesialisasi) adalah mobil, motor, dan kereta.

11. Komunikasi antar objek

    Komunikasi antar-objek dilakukan lewat pesan (*message*) yang dikirim dan satu objek ke objek lainnya.

12. Polimorfisme (*Polymorphism*)

    Kemampuan suatu objek untuk digunakan dibanyak tujuan yang berbeda dengan nama yang sama sehingga menghemat baris program.

13. *Package*

    Package adalah sebuah kontainer atau kemasan yang daoat digunakan untuk mengelompokkan kelas-kelas sehingga memungkinkan beberapa kelas yang bernama sama disimpan dalam package yang berbeda.

II.2.10 *Unified Modeling Language* (UML)

Unified modeling language adalah bahasa pemodelan untuk sistem atau perangkat lunak yang berorientasi objek [12]. Beberapa model yang akan digunakan untuk menggambarkan sistem yang akan dibangun dalam penelitian ini adalah :

1. *Use Case Diagram*

   Use case diagram digunakan untuk memodelkan fungsionalitas-fungsionalitas sistem atau perangkat lunak dilihat dari pengguna yang ada di luar sistem (yang sering dinamakan sebagai *actor*). Kegunaan dari *use case* diagram adalah untuk mendaftarkan aktor-aktor dan *use case* serta memperlihatkan aktor mana yang berpartisipasi dalam masing-masing *use case*. Aktor merupakan representasi dari orang yang berada di luar sistem dan akan berinteraksi dengan sistem. *Use case* adalah fungsionalitas sistem atau perangkat lunak.

2. *Class Diagram*

   Class diagram merupakan diagram yang menggambarkan hubungan antara class yang satu dengan class yang lain. *Class* didefinisikan dengan tiga area pokok, yaitu nama *class*, atribut, dan operasi atau *method*. Pada *class*, atribut dan *method* akan dilakukan *encapsulation*, jenis-jenis *encapsulation* adalah *public*, *private*, dan *protected*.

3. *Activity Diagram*

   Activity diagram bertujuan memodelkan komputasi-komputasi dan aliran-aliran kerja yang terjadi dalam sistem atau perangkat lunak. Suatu diagram aktivitas memuat didalamnya *activity state* dimana suatu *activity state* merepresentasikan eksekusi pernyataan dalam suatu prosedur atau kinerja suatu aktivitas dalam satu aliran kerja.

4. *Sequence Diagram*

   Sequence diagram digambarkan secara *vertical* dan horizontal. Pada bagian *vertical* digambarkan sumbu waktu (*lifeline*), waktu bertambah dari atas ke bawah. Pada bagian horizontal memperlihatkan objek-objek yang terlibat. Selama objek ada, peran digambarkan menggunakan garis tegas. Selama aktivitas prosedur pada objek aktif, garis waktu digambarkan dengan garis ganda. Pesan-pesan digambarkan sebagai suatu tanda panah dari garis waktu suatu objek ke garis waktu objek lainnya.

II.2.11 Java

Java adalah bahasa pemrograman yang dapat membuat seluruh bentuk aplikasi, *desktop*, *web*, *mobile* dan lainnya, sebagaimana dibuat dengan menggunakan bahasa pemrograman konvensional yang lain. Bahasa Pemrograman Java ini berorientasi objek (OOP-*Object Oriented Programming*), dan dapat dijalankan pada berbagai *platform* sistem operasi. Perkembangan Java tidak hanya terfokus pada satu sistem operasi, tetapi dikembangkan untuk berbagai sistem operasi dan bersifat *open source*. Dengan slogannya "*Write once, run anywhere*"[13].

II.2.12 XAMPP (*Explorer Apache MySQL PHP PHPMyAdmin*)

XAMPP ialah perangkat lunak bebas yang mendukung banyak sistem operasi, merupakan campuran dari beberapa program. Yang mempunyai fungsi sebagai *server* yang berdiri sendiri (localhost), yang terdiri dari program MySQL database, *Apache* HTTP *Server*, dan penerjemah ditulis dalam bahasa pemrograman PHP dan *Perl*.

Nama XAMPP merupakan singkatan dari X (empat sistem operasi), *Apache*, MySQL, PHP dan *Perl*. Program ini tersedia di bawah GNU (*General Public License*) dan bebas, adalah mudah untuk menggunakan *web server* yang dapat melayani tampilan halaman *web* yang dinamis. Jika ingin mendapatkan xampp dapat mengunggah langsung dari situs resminya

- **htdoc** adalah folder di mana Anda meletakkan file yang akan dijalankan, seperti file PHP, HTML dan script lainnya.
- **phpMyAdmin** adalah bagian untuk mengelola *database* MySQL yang dikomputer.
- Untuk membukanya, membuka *browser* dan ketik alamat http: // localhost / phpMyAdmin, halaman phpMyAdmin akan muncul.

Control Panel yang berfungsi untuk mengelola layanan (service) XAMPP. Seperti *stop service* (berhenti), atau mulai (mulai).

II.2.13 MySQL

Database MySQL merupakan sistem manajemen basis data SQL yang sangat terkenal dan bersifat *open source*. MySQL dibangun, didistribusikan, dan didukung oleh MySQL AB. MySQL AB merupakan perusahaan komersial yang dibiayai oleh pengembang MySQL. MySQL mempunyai dua macam lisensi yaitu lisensi yang bersifat *open source* dengan menggunakan GNU (*General Public License*) dan lisensi kedua berupa *Standard Commercial License* yang dapat dibeli dari MySQL AB.

Dalam konteks bahasa SQL, informasi disimpan dalam tabel-tabel yang secara logis merupakan struktur dua dimensi yang tersimpan atas baris-baris data (*row* atau *record*) yang berada dalam satu atau lebih kolom (*column*). Baris pada tabel disebut *insance* dari data sedangkan kolom sering disebut sebagai *atributes* atau *field*. MySQL yang berjalan di sistem operasi *Windows* terbukti sangat stabil. MySQL versi *Windows* mempunyai fitur yang sama dengan MySQL versi *Unix* [13].

II.2.14 *Web*

World Wide Web atau WWW atau juga dikenal dengan *WEB* adalah salah satu layanan yang didapat oleh pemakai komputer yang terhubung ke internet. *Web* ini menyediakan informasi bagi pemakai komputer yang terhubung ke internet dari sekedar informasi kecil atau informasi yang tidak berguna sama sekali sampai informasi yang serius, dari informasi yang gratisan sampai informasi yang komersial.

Adapun cara kerja *web* adalah sebagai berikut:

a. Informasi *web* disimpan dalam dokumen dalam bentuk halaman-halaman *web* atau *web* page.
b. Halaman *web* tersebut disimpan dalam *server web* komputer.
c. Sementara dipihak pemakai ada komputer yang bertindak sebagai komputer klien dimana ditempatkan program untuk membaca halaman *web* yang ada di *server web* (*browser*).
d. Browser membaca halaman *web* yang ada di *server web*.

II.2.15 Blackbox

Pengujian *blackbox* berfokus pada persyaratan fungsional perangkat lunak. Dengan demikian, pengujian *blackbox* memungkinkan perekayasa perangkat lunak mendapatkan serangkaian kondisi input yang sepenuhnya menggunakan semua persyaratan fungsional untuk suatu program. Pengujian *blackbox* bukan merupakan alternatif dari teknik *whitebox*, tetapi merupakan pendekatan komplementer yang kemungkinan besar mampu mengungkap kelas kesalahan daripada metode *whitebox*.

Pengujian blackbox berusaha menemukan kesalahan dalam kategori sebagai berikut :

(1) Fungsi-fungsi yan tidak benar atau hilang,
(2) Kesalahan *interface*,
(3) Kesalahan dalam struktur data atau akses *database* eksternal,
(4) Kesalahan kinerja,
(5) Inisialisasi dan kesalahan terminasi.

Tidak seperti pengujian *whitebox*, yang dilakukan pada saat awal proses pengujian, penguhian *blackbox* cenderung diaplikasikan selama tahap akhir pengujian. Karena pengujian *blackbox* memperhatikan struktur kontrol, maka perhatian berfokus pada domain informasi.

II.2.16 Pengujian Beta (*beta testing*)

Pada jenis pengujian ini, perangkat lunak didistribusikan sebagai sebuah versi beta dengan pengguna yang menguji aplikasi di situs mereka. Pengecualian/cacat yang terjadi akan dilaporkan kepada pengembang. Pengujian beta dilakukan setelah pengujian alfa. Versi perangangkat lunak yang dikenal dengan sebutan versi beta dirilis untuk pengguna yang terbatas diluar perusahaan. Perangkat lunak dilepaskan ke kelompok masyarakat agar dapat memastikan bahwa perangkat lunak dilepaskan ke kelompok masyarakat agar dapat memastikan bahwa perangkat lunak tersebut memiliki beberapa kesalahan atau *bug*[14].