BAB II

STUDI PUSTAKA

2.1 Tinjauan Pustaka

Untuk penelitian berkaitan dengan peringkasan teks otomatis pernah dilakukan sebelumnya oleh DwijaWisnu dan Hetami. (2015) dengan judul Perancangan *Information Retrieval (Ir)* Untuk Pencarian Ide Pokok Teks Artikel Berbahasa Inggris Dengan Pembobotan Vector Space Model. Pada penelitian yang dilakukan oleh beliau melakukan peringkasan pada dokumen atau teks artikel yang tidak terdiri dari beberapa sub bab dan artikel yang digunakan menggunakan bahasa inggris dengan menggunakan metode *vector space model*. Hasil kesimpulan yang didapatkan dari sistem pencarian ide pokok otomatis ini memberikan nilai recall 66,68%, precision 72,29%, dan f-measure sebesar 70,38% [2].

Putra, dkk (2011) dengan judul Peringkas Dokumen Tunggal Berbahasa Indonesia Menggunakan Metode Sentences Clustering dan Frequent Term. Penelitian yang dilakukan oleh beliau menggunakan metode *faktorisasi matriks nonnegatif* pada dokumen terstruktur. Ringkasan yang dihasilkan aplikasi dibandingkan dengan ringkasan manual dari manusia mempunyai rata-rata *Precision* 80 % dan *Recall* 70 %, hasil yang sama juga dihasilkan dengan aplikasi sama namun tanpa Stemming, bedanya adalah banyak kalimat yang dihasilkan. Keseluruhan hasil pengujian ini menunjukkan bahwa aplikasi mampu memberikan hasil yang hampir sama dengan ringkasan manusia [4].

Mustaqhfiri, dkk dengan judul Peringkasan Teks Otomatis Berita Berbahasa Indonesia Menggunakan Metode Maximum Marginal Relevance. Penelitian yang dilakukan oleh beliau yaitu melakukan peringkasan pada artikel berita,

menggunakan metode *maximum marginal relevance.* Data uji coba diambil dari surat kabar berbahasa Indonesia online sejumlah 30 berita. Hasil pengujian dibandingkan dengan ringkasan manual yang menghasilkan rata-rata *recall* 60%, *precision* 77%, dan *f-measure* 66% [5].

Nugraha, dkk (2013) dengan judul Implementasi Peringkasan Otomatis Pada Dokumen Terstruktur Dengan Metode Faktorisasi Matriks Nonnegatif. Penelitian yang dilakukan oleh beliau yaitu melakukan *clustering* untuk peringkasan tujuannya untuk membuat lead (teras) berita. Dari uji coba penentuan rentang bilangan acak yang menghasilkan nilai Frobenius Norm paling kecil diperoleh rentang bilangan acak terbaik adalah antara 0.05 hingga 0.25. Namun berdasar waktu eksekusi terkecil pada proses peringkasan dokumen, rentang bilangan acak yang diperoleh adalah antara 0.1 hingga 0.25 [6].

Karmila, dkk (2013) dengan judul Aplikasi *Automated Text Summarization* (ATS) Pembuat *Lead* (Teras) Berita dengan *Text to Speech* (TTS) Menggunakan Algoritma TF-IDF dan *Vector Space Model*. Penelitian yang dilakukan oleh beliau menggunakan metode *sentence clustering* dan *frequent term*, Hasil pengujian menunjukkan bahwa aplikasi pembuat lead berita telah dapat melakukan fungsinya dengan baik dari segi fungsional maupun konseptual/ struktural. Kualitas ringkasan yang dihasilkan cukup memuaskan dengan persentase kesesuaian tertinggi mencapai 100% dan terendah sebesar 64,29% [3].

Pada penelitian ini akan menggunakan metode *vector space model.* Penelitiannya dilakukan untuk peringkasan pada dokumen jurnal bahasa indonesia yang memiliki beberapa subbab di dalamnya. Tujuan dari peringkasan ini adalah untuk mendapatkan abstrak dari jurnal secara otomatis.

Berikut ini adalah hasil penelitian yang telah dilakukan dan memilki hubungan dengan penelitian yang akan dilakukan antara lain:

Tabel 2.1 Tabel *State of The Art*

NO	PENELITI	METODE	DATA	LAYANAN/FITUR
1	DwijaWisnu, Hetami. (2015)	*Vector Space Model*	Teks	Sistem ini melakukan pencarian ide pokok teks artikel berbahasa inggris secara otomatis.
2	Yuliawati, dkk. (2011)	*Faktorisasi Matriks Nonnegatif*	Teks	Aplikasi ini meringkas teks otomatis pada dokumen terstruktur
3	Mustaqhfiri, dkk.	*Maximum Marginal Relevance*	Teks	Aplikasi ini meringkas *single* dokumen secara otomatis dengan menggunakan judul artikel berita sebagai *query*
4	Karmila, dkk. (2013)	*Vector Space Model*	Teks	Aplikasi peringkasan untuk membuat *lead* (teras) berita
5	Nugraha, dkk. (2013)	*Sentences Clustering* dan *Frequent Term*	Teks	Aplikasi peringkasan teks otomatis pada dokumen tunggal bahasa indonesia

2.2 Landasan Teori

2.2.1 Peringkasan Teks Otomatis

Peringkasan teks adalah proses pemampatan teks sumber ke dalam versi lebih pendek namun tetap mempertahankan informasi yang terkandung didalamnya [2].

Hovy Eduard dalam mendefinisikan ringkasan sebagai berikut : Ringkasan adalah teks yang diproduksi dari satu atau lebih teks, yang mengandung suatu porsi yang siknifikan dari informasi dalam teks asli, dan ringkasan tidak lebih dari setengahnya teks asli [10].

Terdapat dua tipe dari pembuatan suatu ringkasan yang mengambil bagian terpenting dari teks aslinya, yaitu [11]:

a. Abstrak, menghasilkan sebuah interpretasi terhadap teks aslinya. Dimana sebuah kalimat akan ditransformasikan menjadi kalimat yang lebih singkat. Contoh kalimat: ”Ani menyukai apel, jambu, jeruk, mangga, dan pepaya” akan diubah menjadi kalimat ”Ani menyukai buah-buahan”.
b. Ekstrak, metode ini menggunakan *statistical*, *linguistical*, dan*heuristic* atau kombinasi dari semuanya dalam menetapkan ringkasan dari suatu teks.

Otomatis dapat didefinisikan sebagai suatu teknologi dimana suatu proses atau prosedur dijalankan tanpa bantuan manusia. Itu semua dapat diimplementasikan dengan menggunakan suatu program instruksi yang dikombinasikan dengan sistem kendali yang menjalankan instruksi [11].

Peringkasan teks otomatis yang dimaksud dalam tugas akhir ini adalah aplikasi yang mampu meringkas suatu teks dan menghasilkan peringkasan yang menghasilkan abstrak.

2.2.2 Abstrak

Abstrak merupakan penyajian singkat mengenai isi tulisan sehingga pada tulisan, ia menjadi bagian tersendiri. Abstrak berfungsi untuk menjelaskan secara singkat kepada pembaca [12]. Secara umum abstrak terdiri dari 150 hingga 250 kata [1].

a. Fungsi Abstrak

Fungsi abstrak adalah untuk memberikan informasi kepada masyarakat perihal hasil penelitian yang telah dibuat. Uraian yang hanya satu halaman tersebut memudahkan abstrak dimasukkan dalam jaringan internet. Hal ini dimaksudkan memudahkan anda mengetahui hasil penelitian tanpa harus membaca keseluruhan penelitian yang berlembar lembar. Sehingga abstrak membantu anda dalam mencari referensi dalam penelitian yang anda cari [12].

Adanya abstrak akan menghindari tindakan plagiasi oleh pihak yang tidak bertanggung jawab. Sebuah penelitian akan terlindungi jika hanya abstraknya saja yang ditampilkan dan diperluas di internet [12].

2.2.3 *Data Mining*

Tan mendifinisikan *data mining* sebagai proses untuk mendapatkan informasi yang berguna dari gudang basis data yang besar. *Data mining* juga dapat diartikan sebagai pengekstrakan informasi baru yang diambil dari bongkahan data besar yang membantu dalam mengambil keputusan.

Salah satu teknik yang dibuat dalam *data mining* adalah bagaimana menelusuri data yang ada untuk membangun sebuah model, kemudian menggunakan model tersebut agar dapat mengenali pola data yang lain yang tidak berada dalam basis data yang tersimpan. Kebutuhan untuk prediksi juga dapat memanfaatkan teknik ini. Dalam *data mining*, pengelompokan data juga bisa dilakukan. Tujuannya adalah agar kita dapat mengetahui pola universal data-data yang ada [13].

a. Tahapan *Data Mining*

Secara umum tahapan data mining terdiri dari 7 bagian, diantaranya :

1. *Data Cleaning*, adalah suatu proses pembersihan data dari informasi yang tidak berguna yang dapat memperlambat proses *query* ataupun memperburuk kualitas hasilnya. Pada tahap ini data dikonfigurasi ulang sehingga mempunyai format yang konsisten, hal ini dikarenakan sumber data yang mungkin berbeda sehingga perlu diselaraskan.
2. *Data Integration*, adalah suatu tahap menggabungkan data dari berbagai sumber.
3. *Data Selection*, tahap dimana dilakukan pemilihan data yang relevan dengan *analysis stack*.

4. *Transformation*, transfomasi atau konsolidasi data kedalam bentuk yang lebi baik untuk mining, dengan mewujudkan operasi-operasi *summary* dan *agreggation*.
5. *Data Mining*, dilakukan pencarian pola dan melakukan ekstraksi jika telah ditentukan dengan menerapkan "*intelligent methods* ".
6. *Pattern Evaluation*, pola yang telah diidentifikasi oleh sistem diinterpretasikan mejadi *knowledge* yang kemudian digunakan sebagai informasi guna mengambil keputusan.
7. *Knowledge Presentation*, penggunaan teknik visualisasi dan representasi untuk menyajikan pengetahuan yang telah diperoleh kepada pengguna.

2.2.4 *Teks Mining*

Text mining merupakan salah satu bidang khusus dari data mining. Text mining dapat didefiniskan sebagai suatu proses menggali informasi dimana seorang user berinteraksi dengan sekumpulan dokumen menggunakan *tool* analisis yang merupakan komponen-komponen dalam data mining [14].

Dalam *text mining* berbeda dengan dengan *data mining* dimana data mining yang digunakan adalah structured data sementara dalam *text mining* umumnya data yang ditemui adalah *semi-structured* atau *unstructured*. Sementara keduanya memiliki permasalahan yang sama yaitu jumlah data yang besar, dimensi yang tinggi, dan data juga struktur yang terus berubah. Struktur teks yang kompleks dan tidak lengkap, arti yang tidak jelas dan tidak standar, dan bahasa yang berbeda ditambah terjemahan yang tidak akurat memberikan tantangan tambahan pada *text mining*.

Text mining dalam prakteknya mencari pola-pola tertentu, mengasosiasikan satu bagian teks dengan lain berdasar aturan-aturan tertentu, kata-kata yang dapat mewakili sehingga dapat dilakukan analisa keterhubungan antar satu dengan lain, dalam

kumpulan dokumen yang sangat banyak. Dokumen yang ada bisa bersifat statis, yaitu dokumen yang tidak akan di perbarui lagi ataupun dinamis yaitu dokumen yang akan selalu diperbarui dalam rentang waktu tertentu.

2.2.5 *Text Preprocessing*

Text preprocessing adalah tahapan untuk mempersiapkan teks menjadi data yang akan diolah di tahapan berikutnya. Inputan awal pada proses ini adalah berupa dokumen [5]. *Text preprocessing* pada penelitian ini terdiri dari beberapa tahapan, yaitu: proses pemecahan kalimat, proses *case folding*, proses *tokenizing* kata, proses *filtering*, dan proses *stemming*.

a. Pemecahan Kalimat

Memecah dokumen menjadi kalimat-kalimat merupakan langkah awal tahapan *text preprocessing*. Pemecahan kalimat yaitu proses memecah *string* teks dokumen yang panjang menjadi kumpulan kalimat-kalimat. Dalam memecah dokumen menjadi kalimat-kalimat menggunakan fungsi **split()**, dengan tanda titik ".", tanda tanya "?" dan tanda seru "!" sebagai delimeter untuk memoting *string* dokumen.

b. Case Folding

Case folding adalah tahapan proses mengubah semua huruf dalam teks dokumen menjadi huruf kecil, serta menghilangkan karakter selain a-z.

c. Tokenizing

Tokenizing adalah proses pemotongan string input berdasarkan tiap kata yang menyusunnya. Pemecahan kalimat menjadi kata-kata tunggal dilakukan dengan men-*scan* kalimat dengan pemisah (*delimiter*) *white space* (spasi, tab, dan *newline*).

d. Filtering

Filtering merupakan proses penghilangan *stopword*. *Stopword* adalah kata - kata yang sering kali muncul dalam dokumen namun artinya tidak deskriptif dan tidak memiliki keterkaitan dengan tema tertentu. Didalam bahasa Indonesia *stopword* dapat disebut sebagai kata tidak penting, misalnya “di”, ”oleh”, “pada”, ”sebuah”, ”karena” dan lain sebagianya.

e. Stemming

Stemming merupakan proses mencari akar (*root)* kata dari tiap *token* kata yaitu dengan pengembalian suatu kata berimbuhan ke bentuk dasarnya (*stem*). Pada penelitian ini menggunakan *porter stemming* untuk bahasa indonesia. Terdapat lima aturan pada proses *stemming* untuk bahasa Indonesia menggunakan *porter stemmer*, yaitu ada lima aturan tahap dalam proses *stemming* pada bahasa Indonesia, yaitu :

1) Penanganan terhadap partikel inflek sional, yaitu: lah, tah, kah. Contoh: duduklah, makanlah.
2) Penanganan terhadap kata ganti infleksional, yaitu: ku, mu, dan nya. Contoh: sepedaku, bukunya.
3) Penanganan terhadap prefiks derivasional pertama, yaitu : meng dan semua variasinya, peng dan semua variasinya, di, ter, dan ke. contoh : membakar, pengukur, kekasih.
4) Penanganan terhadap prefix derivasional kedua, yaitu : ber dan semua variasinya serta per dan semua variasinya. Contoh: berlari, belajar, perkata.
5) Penanganan terhadap Sufiks derivasional, yaitu kan, am dan i. Contoh: ambilkan, janjian dan dapati.

Karena struktur morfologi bahasa Indonesia yang rumit maka kelima tahap aturan tidak cukup untuk menangani proses stemming bahasa Indonesia. Kesulitan

membedakan kata yang mengandung imbuhan baik prefiks maupun sufiks dari kata dasar yang salah satu suku katanya merupakan bagian imbuhan, terutama dengan kata dasar yang mempunyai suku kata lebih besar dari dua.

2.2.6 TF/IDF (*Term Frequency-Inversed Document Frequency*)

Pada algoritma TF/IDF digunakan rumus untuk menghitung bobot (W) masing-masing dokumen terhadap kata kunci [2].

Dengan rumus sebagai berikut:

$$W_{dt} = tf_{dt} * IDF_t \quad (1)$$

Dimana :

d = dokumen ke-d

t = kata ke-t dari kata kunci

W = bobot dokumen ke-d terhadap kata ke-t

tf = banyaknya kata yang dicari pada sebuah dokumen

IDF = Inversed Document Frequency

IDF = log2 (D/df)

D = total dokumen

df = banyak dokumen yang mengandung kata yang dicari

Setelah bobot (W) masing-masing dokumen diketahui, maka dilakukan proses sorting/pengurutan dimana semakin besar nilai W, semakin besar tingkat similaritas dokumen tersebut terhadap kata kunci, demikian sebaliknya. Contoh implementasi sederhana dari TF-IDF adalah sebagai berikut:

Kata kunci (kk) = pengetahuan logistik

Dokumen 1 (D1) = manajemen transaksi logistik

Dokumen 2 (D2) = pengetahuan antar individu

Dokumen 3 (D3) = dalam manajemen pengetahuan terdapat transfer pengetahuan logistik

Jadi jumlah dokumen (D) = 3

Setelah dilakukan tahap tokenizing dan proses filtering, maka kata antar pada dokumen 2 serta kata dalam dan terdapat pada dokumen 3 dihapus. Berikut ini adalah tabel perhitungan TF/IDF

Tabel 2.2 Tabel Contoh Perhitungan TF/IDF

Token	tf				df	D/df	IDF = Log10(D/df)	W			
	kk	D1	D2	D3				kk	D1	D2	D3
Manajemen	0	1	0	1	2	1.5	0.176	0	0.176	0	0.176
Transaksi	0	1	0	0	1	3	0.477	0	0.477	0	0
Logistic	1	1	0	1	2	1.5	0.176	0.176	0.176	0	0.176
Transfer	0	0	0	1	1	3	0.477	0	0	0	0.477
Pengetahuan	1	0	2	2	2	1.5	0.176	0	0	0.176	0.352
individu	0	0	0	1	1	3	0.477	0	0	0.477	0
Total								0.352	0.829	0.653	1.181

bobot (W) untuk D1 = logistic + pengetahuan

= 0.176 + 0 = 0.176

bobot (W) untuk D2 = logistic + pengetahuan

= 0 + 0.176 = 0.176

bobot (W) untuk D3 = logistic + pengetahuan

= 0.176 + 0.352 = 0.528

Dari contoh studi kasus di atas, dapat diketahui bahwa nilai bobot (W) dari D1 dan D2 adalah sama. Apabila hasil pengurutan bobot dokumen tidak dapat mengurutkan secara tepat, karena nilai W keduanya sama, maka diperlukan proses perhitungan dengan algoritma *vector-space model*. Ide dari metode ini adalah dengan menghitung nilai cosinus sudut dari dua vektor, yaitu W dari tiap dokumen dan W dari kata kunci.

2.2.7 *Vector Space Model*

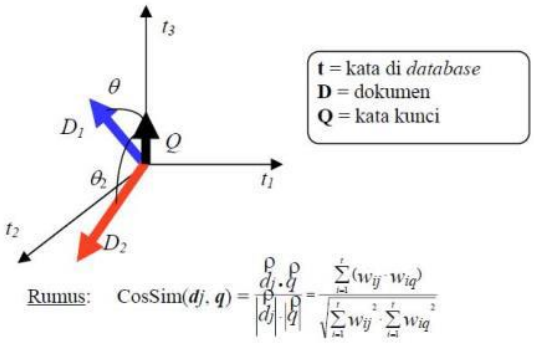
Model ruang vektor dibuat berdasarkan pemikiran bahwa isi dari dokumen ditentukan oleh kata-kata yang digunakan dalam dokumen tersebut. Model ini

menentukan kemiripan (*similarity*) antara dokumen dengan query dengan cara merepresentasikan dokumen dan query masing-masing ke dalam bentuk vektor. Tiap kata yang ditemukan pada dokumen dan query diberi bobot dan disimpan sebagai salah satu elemen vektor.

Kemiripan antar dokumen didefinisikan berdasarkan representasi *bag-of-words* dan dikonversi ke suatu model ruang vektor (*vector space model, VSM*). Pada VSM, setiap dokumen di dalam database dan query pengguna direpresentasikan oleh suatu vektor multi-dimensi. Dimensi sesuai dengan jumlah term dalam dokumen yang terlibat pada model ini:

1. *Vocabulary* merupakan kumpulan semua term berbeda yang tersisa dari dokumen setelah *preprocessing* dan mengandung t term index. Term-term ini membentuk suatu ruang vektor.
2. Setiap *term* i di dalam dokumen atau query j, diberikan suatu bobot (*weight*) bernilai real Wij.
3. Dokumen dan query diekspresikan sebagai vektor t dimensi dj = (W1, W2, ..., Wtj) dan terdapat n dokumen di dalam koleksi, yaitu j = 1, 2, ..., n.

Contoh dari model ruang vektor tiga dimensi untuk dua dokumen D1 dan D2, satu query pengguna Q1, dan tiga term T1, T2 dan T3 diperlihatkan pada gambar 1.2



Gambar 2.1 *vector space model* [2]

Dalam model ruang vektor, koleksi dokumen direpresentasikan oleh matriks term-document (atau *matriks term-frequency*). Setiap sel dalam matriks bersesuaian dengan bobot yang diberikan dari suatu term dalam dokumen yang ditentukan. Nilai nol berarti bahwa term tersebut tidak hadir di dalam dokumen. Contoh matrik term-document untuk database dengan n dokumen dan t term berikut adalah gambar matriks term document:

Gambar 2.2 Contoh matriks *vsm* [2]

Dokumen-dokumen yang panjang sering dianggap lebih relevan dibandingkan dokumen yang pendek, padahal belum tentu demikian. Untuk mengurangi pengaruh perbedaan panjang dokumen ini, pada pembobotan kata digunakan satu faktor lagi yang disebut sebagai normalisasi panjang dokumen. Normalisasi yang digunakan adalah normalisasi kosinus. Berdasarkan rumus normalisasi ,kosinus yaitu :

$$w(word_i) = \frac{w(word_i)}{\sqrt{w^2(word_1)+w^2(word_2)+\cdots+w^2(word_n)}} \quad \text{..... (2)}$$

Dengan W adalah bobot dari query dan dokumen.

Setelah mendapatkan nilai cosine tiap-tiap dokumen, maka hasil bobot dari kata kunci diurutkan. Bobot yang besar menjadi prioritas sebagai dokumen yang memiliki hubungan dengan kata kunci.

2.2.8 Basis Data

Sistem basis data adalah sistem terkomputerisasi yang tujuan utamanya adalah untuk memelihara data yang sudah diolah atau informasi dan membuat informasi tersedia saat dibutuhkan. Pada intinya basis data adalah media untuk menyimpan data agar dapat diakses dengan mudah dan cepat [9].

a. CDM (*Conceptual Data Model*)

CDM atau model konsep data merupakan konsep yang berkaitan dengan pandangan pemakai terhadap data yang disimpan dalam basis data. CDM dibuat sudah dalam bentuk tabel-tabel tanpa tipe data yang menggambarkan relasi antar tabel untuk keperluan implementasi ke basis data [9].

b. PDM (*Physical Data Model*)

PDM atau model rasional adalah model yang menggunakan sejumlah tabel untuk menggambarkan data serta hubungan antara data. Setiap tabel mempunyai sejumlah kolom di mana setiap kolom memiliki nama yang unik beserta tipe datanya. PDM merupakan konsep yang menerangkan detail dari bagaimana data di simpan di dalam basis data [9].

c. MySQL

MySQL adalah sistem manajemen *database SQL* yang bersifat *Open Source* dan paling populer saat ini. Sistem *Database MySQL* mendukung beberapa fitur seperti *multithreaded, multi-user*, dan *SQL database managemen sistem (DBMS). Database* ini dibuat untuk keperluan sistem *database* yang cepat, handal dan mudah digunakan.

Ulf Micheal Widenius adalah penemu awal versi pertama *MySQL* yang kemudian pengembangan selanjutnya dilakukan oleh perusahaan *MySQL* AB. *MySQL* AB yang merupakan sebuah perusahaan komersial yang didirikan oleh para

pengembang *MySQL*. *MySQL* sudah digunakan lebih dari 11 millar instalasi saat ini [15].

Kelebihan *MySQL*

Berikut ini beberapa kelebihan *MySQL* sebagai *database server* antara lain [15] :

1. Source *MySQL* dapat diperoleh dengan mudah dan gratis.
2. Sintaksnya lebih mudah dipahami dan tidak rumit.
3. Pengaksesan *database* dapat dilakukan dengan mudah.
4. *MySQL* merupakan program yang *multithreaded*, sehingga dapat dipasang pada *server* yang memiliki *multiCPU*.
5. Didukung programprogram umum seperti *C, C++, Java, Perl, PHP, Python,* dsb.
6. Bekerja pada berbagai *platform*. (tersedia berbagai versi untuk berbagai sistem operasi).
7. Memiliki jenis kolom yang cukup banyak sehingga memudahkan konfigurasi sistem *database*.
8. Memiliki sistem sekuriti yang cukup baik dengan verifikasi *host*.
9. Mendukung ODBC untuk sistem operasi *Windows*.
10. Mendukung *record* yang memiliki kolom dengan panjang tetap atau panjang bervariasi.

MySQL dan *PHP* merupakan sistem yang saling terintegrasi. Maksudnya adalah pembuatan *database* dengan menggunakan sintak *PHP* dapat di buat. Sedangkan *input* yang di masukkan melalui aplikasi *web* yang menggunakan *script serverside* seperti *PHP* dapat langsung dimasukkan ke *database MySQL* yang ada di *server* dan tentunya *web* tersebut berada di sebuah *web server* [15].

2.2.9 PHP

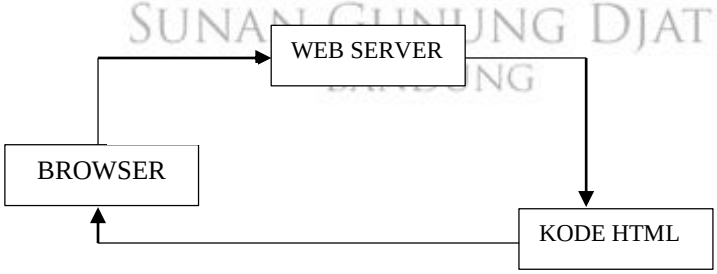
PHP atau yang memiliki kepanjangan *PHP Hypertext Preprocessor* merupakan suatu bahasa pemrograman yang difungsikan untuk membangun suatu *website* dinamis. PHP menyatu dengan kode HTML. HTML digunakan sebagai pembangun atau pondasi dari kerangka *layout web*, sedangkan PHP difungsikan sebagai prosesnya, sehingga dengan adanya PHP tersebut, sebuah *web* tersebut akan sangat mudah di-*maintenance* [16].

PHP berjalan pada sisi *server* sehingga PHP disebut juga sebagai bahasa *Server Side Scripting*, artinya bahwa dalam setiap/untuk menjalankan PHP, wajib membutuhkan *web server* dalam menjalankannya.

a. Cara Menggunakan PHP

PHP merupakan bahasa *Server Side Scripting*, di mana PHP selalu membutuhkan *web server* dalam menjalankannya.

Secara prinsip, *server* akan bekerja apabila ada permintaan dari *client*, yaitu kode-kode PHP. *Client* tersebut akan dikirimkan ke *server*, kemudian *server* akan mengembalikan pada halaman sesuai instruksi yang diminta.



Gambar 2.3 Cara Kerja PHP [16]

2.2.10 Perancangan Terstruktur

Pendekatan perancangan terstruktur dimulai dari awal 1970. Pendekatan terstruktur dilengkapi dengan alat-alat (*tools*) dan teknik-teknik (*techniques*) yang dibutuhkan dalam

pengembangan sistem, sehingga hasil akhir dari sistem yang dikembangkan akan diperoleh sistem yang strukturnya didefinisikan dengan baik dan jelas .

Melalui pendekatan terstruktur, permasalahan yang komplek diorganisasi dapat dipecahkan dan hasil dari sistem akam mudah untuk dipelihara, fleksibel, lebih memuaskan pemakainya, mempunyai dokumentasi yang baik, tepat waktu, sesuai dengan anggaran biaya pengembangan, dapat meningkatkan produktivitas dan kualitasnya akan lebih baik (bebas kesalahan).

2.2.11 *Data Flow Diagram (DFD)*

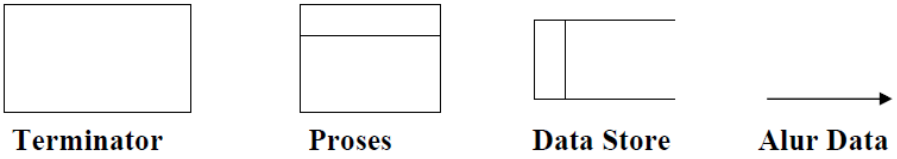
Data Flow Diagram (DFD) adalah alat pembuatan model yang memungkinkan profesional sistem untuk menggambarkan sistem sebagai suatu jaringan proses fungsional yang dihubungkan satu sama lain dengan alur data, baik secara manual maupun komputerisasi. DFD ini sering disebut juga dengan nama *Bubble chart*, *Bubble diagram*, model proses, diagram alur kerja, atau model fungsi [8].

DFD ini adalah salah satu alat pembuatan model yang sering digunakan, khususnya bila fungsi-fungsi sistem merupakan bagian yang lebih penting dan kompleks dari pada data yang dimanipulasi oleh sistem. Dengan kata lain, DFD adalah alat pembuatan model yang memberikan penekanan hanya pada fungsi sistem.

DFD ini merupakan alat perancangan sistem yang berorientasi pada alur data dengan konsep dekomposisi dapat digunakan untuk penggambaran analisa maupun rancangan sistem yang mudah dikomunikasikan oleh profesional sistem kepada pemakai maupun pembuat program.

a. **Komponen *Data Flow Diagram***

Berikut ini merupakan komponen dari data flow diagram yang digambarkan oleh Gambar 2.4



Gambar 2.4 Komponen *Data Flow Diagram*

1. Terminator

Terminator mewakili entitas eksternal yang berkomunikasi dengan sistem yang sedang dikembangkan. Biasanya terminator dikenal dengan nama entitas luar (*external entity*).

Komponen terminator ini perlu diberi nama sesuai dengan dunia luar yang berkomunikasi dengan sistem yang sedang dibuat modelnya, dan biasanya menggunakan kata benda, misalnya Bagian Penjualan, Dosen, Mahasiswa.

2. Proses

Komponen proses menggambarkan bagian dari sistem yang mentransformasikan input menjadi output. Proses diberi nama untuk menjelaskan proses/kegiatan apa yang sedang/akan dilaksanakan. Pemberian nama proses dilakukan dengan menggunakan kata kerja transitif (kata kerja yang membutuhkan obyek).

3. Data Store

Komponen ini digunakan untuk membuat model sekumpulan paket data dan diberi nama dengan kata benda jamak, misalnya Mahasiswa.

Data store ini biasanya berkaitan dengan penyimpanan - penyimpanan, seperti file atau *database* yang berkaitan dengan penyimpanan secara komputerisasi,

misalnya file disket, file harddisk, file pita magnetik. *Data store* juga berkaitan dengan penyimpanan secara manual seperti buku alamat, file folder, dan agenda.

4. Alur Data

Suatu *data flow* / alur data digambarkan dengan anak panah, yang menunjukkan arah menuju ke dan keluar dari suatu proses. Alur data ini digunakan untuk menerangkan perpindahan data atau paket data/informasi dari satu bagian sistem ke bagian lainnya.

Selain menunjukkan arah, alur data pada model yang dibuat oleh profesional sistem dapat merepresentasikan bit, karakter, pesan, formulir, bilangan real, dan macam-macam informasi yang berkaitan dengan komputer. Alur data juga dapat merepresentasikan data/informasi yang tidak berkaitan dengan komputer.

Alur data perlu diberi nama sesuai dengan data/informasi yang dimaksud, biasanya pemberian nama pada alur data dilakukan dengan menggunakan kata benda, contohnya Laporan Penjualan.

b. Penggambaran DFD

Tidak ada aturan baku untuk menggambarkan DFD. Tapi dari berbagai referensi yang ada, secara garis besar langkah untuk membuat DFD adalah :

1. Identifikasi terlebih dahulu semua entitas luar yang terlibat di sistem.
2. Identifikasi semua input dan output yang terlibat dengan entitas luar.
3. Buat Diagram Konteks (*diagram context*). Diagram ini adalah diagram level tertinggi dari DFD yang menggambarkan hubungan sistem dengan lingkungan luarnya.
4. Buat Diagram Level Zero. Diagram ini adalah dekomposisi dari diagram konteks.
5. Buat Diagram Level Satu. Diagram ini merupakan dekomposisi dari diagram level zero.