BAB 2
LANDASAN TEORI

2.1. *Automated Essay Scooring*

Automated essay scooring (AES) adalah sebuah sistem penilaian otomatis dengan cara membandingkan dua jawaban kemudian dikalkulasi hasil dari perbandingan menggunakan metode-metode yang ada. Contoh metode yang bisa digunakan untuk penerapan AES adalah *Corpus based similarity* dan *String based similarity*. *Corpus Based Similarity* (CBS) adalah metode yang mengukur kesamaan arti dari setiap kata berdasarkan sebuah kamus yang telah disediakan. *Latent Semantic Analysis* (LSA) merupakan teknik yang popular dari CBS. *String Based Similarity* (CBS) adalah metode yang mengukur kesamaan string antara dua text tanpa melihat arti dari text tersebut. Teknik yang bisa digunakan untuk CBS adalah *Cosine Similarity* [9].

2.2. Kalimat Definitif atau Definsi

Kata definisi berasal dari kata '*definition*' (bahasa Inggris) yang bermakna keterangan yang berupa penjelasan atas suatu objek. Sehingga kalimat definisi dapat diartikan sebagai kalimat yang memuat keterangan atau penjelasan atas suatu objek. Objek disini dapat berupa benda, orang, proses atau aktivitas [11].

Ciri Ciri Kalimat Definisi

Kalimat definisi dapat dikenali melalui ciri-ciri berikut:

1. Menjelaskan makna atau arti dari suatu objek
2. Penjelasan bersifat umum atau tidak mengarah kepada ciri khusus objek tersebut
3. Biasanya ditemukan pada laporan ilmiah.
4. Jika kalimat ini dibalik maka tidak mengubah makna atau arti dari kalimat tersebut

Contoh Kalimat Definisi

Agar lebih memahami tentang kalimat definisi, berikut disajikan beberapa contoh kalimat definisi.

1. Suaka margasatwa adalah kawasan hutan yang memiliki keanekaragaman dalam jenis satwanya.
2. Warga negara adalah sekumpulan orang yang tinggal di wilayah (negara) tertentu.
3. Survei adalah penelitian yang dilakukan dengan menyebar kuesioner atau melakukan wawancara untuk pengumpulan data.
4. Kebudayaan adalah segala hal yang berkaitan dengan hasil akal dan budi manusia.

2.3. *Typograpihical Error*

Typographical error atau sering disingkat menjadi typo adalah kesalahan yang dibuat dalam proses pengetikan seperti ejaan atau meninggalkan suatu kata. Salah satu yang paling umum dari kesalahan ketik atau *typo* adalah transposisi huruf, dimana semua huruf yang diperlukan untuk mengeja kata yang hadir, tetapi dalam urutan yang salah [4]. Ini jenis kesalahan yang sering terjadi ketika seseorang mengetik dengan cepat dan tidak cukup memperhatikan apa yang ditulis. Juga tidak jarang melihat substitusi huruf disebabkan oleh slip jari pada keyboard. Kesalahan tersebut dapat terjadi karena kegagalan mekanis atau slip dari tangan atau jari. Contoh-contoh *typo* seperti, duplikasi sederhana, ketertinggalan, dan transposisi atau tertukarnya posisi huruf [4].

2.3.1. Jenis-jenis *typo*

Berikut jenis-jenis *typo* yang sering terjadi dalam pengetikan:

Kekurangan salah satu huruf, kelebihan, dan kesalahan penempatan [5]. Contoh kekurangan huruf adalah seperti kata ‘nasi’ menjadi ‘nsi’ [4]. Contoh kelebihan huruf seperti kata ‘nasi’ menjadi ‘nasii’, dan contoh kesalahan penempatan huruf seperti kata ‘nasi’ menjadi kata ‘nsai’.

2.4. *Preprocessing*

Tahap text *Preprocessing* adalah tahapan dimana aplikasi melakukan seleksi data yang akan diproses pada setiap dokumen. Proses preprocessing ini

meliputi (1) *case folding*, (2) *filtering*, (3) *tokenizing*, dan (4) *stopword removal* [12].

2.4.1. *Case Folding*

Case Folding dibutuhkan dalam mengkonversi keseluruhan teks dalam dokumen menjadi suatu bentuk standar (biasanya huruf kecil atau lowercase). Sebagai contoh, user yang ingin mendapatkan informasi "KOMPUTER" dan mengetik "KOMPUTER", "KomPUter", atau "komputer", maka diberikan hasil *retrieval* yang sama yakni "komputer". *Case folding* adalah mengubah semua huruf dalam dokumen menjadi huruf kecil. Hanya huruf 'a' sampai dengan 'z' yang diterima. Karakter selain huruf dihilangkan dan dianggap delimiter. [12]

2.4.2. *Filtering*

Filtering digunakan untuk mendapatkan kalimat yang mempunyai arti saja dengan menghilangkan simbol-simbol. Dengan kata lain *filtering* adalah proses membuang kata-kata serta tanda-tanda yang tidak bermakna secara signifikan, yang tersisa hanya huruf a..z dan angka 1..9 [13].

2.4.3. *Tokenizing*

Tokenizing digunakan untuk membuat susunan kata dalam bentuk token-token yang diperlukan untuk proses *stopword removal*. Tokenisasi secara garis besar memecah sekumpulan karakter dalam suatu teks ke dalam satuan kata. [12].

2.4.4. *Stopword Removal*

Stopword removal merupakan proses lanjutan dari tokenizing di dalam preprocessing kalimat. Proses *stopword removal* merupakan proses untuk menghilangkan kata yang 'tidak relevan' pada hasil token sebuah dokumen teks dengan cara membandingkannya dengan stoplist yang ada. *Stoplist* berisi sekumpulan kata yang 'tidak relevan', namun sering sekali muncul dalam sebuah dokumen. Dibawah ini tabel 2.1 daftar *stoplist* [14].

Tabel 2. 1 Daftar Stoplist [14]

Stoplist				
‘yang’	‘untuk’	‘ini’	‘telah’	‘begitu’
‘pada’	‘ke’	‘karena’	‘dari’	‘maka’
‘menurut’	‘namun’	‘kepada’	‘di’	‘lagi’
‘antara’	‘dia’	‘oleh’	‘serta’	‘tentang’
‘ia’	‘dua’	‘saat’	‘bagi’	‘demi’
‘seperti’	‘tidak’	‘harus’	‘sekitar’	‘dimana’
‘jika’	‘dan’	‘sementara’	‘kami’	‘kemana’
‘sehingga’	‘kembali’	‘setelah’	‘belum’	‘sampai’
‘sebagai’	‘ada’	‘mereka’	‘anda’	‘sedangkan’
‘masih’	‘juga’	‘sudah’	‘itulah’	‘selagi’
‘hal’	‘akan’	‘saya’	‘daripada’	‘sementara’
‘ketika’	‘dengan’	‘terhadap’	‘yakni’	‘sebelum’
	n’	p’		‘
‘adalah’	‘kita’	‘secara’	‘yaitu’	‘tetapi’
‘itu’	‘hanya’	‘agar’	‘kenapa’	‘apakah’
‘dalam’	‘atau’	‘lain’	‘mengapa’	‘supaya’
‘bisa’	‘bahwa’	‘anda’	‘begitu’	‘dll’

2.5. Algoritma *Jaro Winkler*

Jaro Winkler Distance adalah merupakan varian dari *Jaro Distance* metrik yaitu sebuah algoritma untuk mengukur kesamaan antara dua string, biasanya algoritma ini digunakan di dalam pendeteksian duplikat. Semakin tinggi *Jaro-Winkler distance* untuk dua string, semakin mirip dengan string tersebut. *Jaro-Winkler distance* terbaik dan cocok untuk digunakan dalam perbandingan string singkat seperti nama orang. Skor normalnya seperti 0 menandakan tidak ada kesamaan, dan 1 adalah sama persis. Algoritma *Jaro Winkler Distance* memiliki kompleksitas waktu *quadratic runtime complexity* yang sangat efektif pada string pendek dan dapat bekerja lebih cepat dari algoritma *edit distance* [15]. Dasar dari algoritma ini memiliki tiga bagian:

1. Menghitung panjang string,
2. Menemukan jumlah karakter yang sama di dalam dua string, dan
3. Menemukan jumlah transposisi.

Pada algoritma Jaro digunakan rumus untuk menghitung jarak (dj) antara dua string yaitu |s1| dan |s2| adalah:

$$\boldsymbol{dj} = \frac{\mathbf{1}}{\mathbf{3}}\left(\frac{\boldsymbol{m}}{|\boldsymbol{s1}|} + \frac{\boldsymbol{m}}{|\boldsymbol{s2}|} + \frac{\boldsymbol{m}-\boldsymbol{t}}{\boldsymbol{m}}\right) \qquad 2.\ 1$$

Dimana:

m = jumlah karakter yang sama persis

|s1| = panjang string 1

|s2| = panjang string 2

t = jumlah transposisi

Jarak toritis duah buah karakter yang disamakan dapat dibenarkan jika tidak melebihi:

$$\left(\frac{\mathbf{max}(|\boldsymbol{s1}|,|\boldsymbol{s2}|)}{\boldsymbol{s}}\right) < -\mathbf{1} \qquad 2.\ 2$$

Akan tetapi bila mengacu kepada nilai yang akan dihasilkan oleh algoritma Jaro-Winkler maka nilai jarak maksimalnya adalah 1 yang menandakan kesamaan string yang dibandingkan mencapai seratus persen atau sama persis. Biasanya |s1| digunakan sebagai acuan untuk urutan di dalam mencari transposisi. Yang dimaksud transposisi di sini adalah karakter yang sama dari string yang dibandingkan akan tetapi tertukar urutannya. Sebagai contoh, dalam membandingkan kata CRATE dengan TRACE, bila dilihat seksama maka dapat dikatakan semua karakter yang ada di |s1| ada dan sama dengan karakter yang ada di |s2| , tetapi dengan urutan yang berbeda. Dengan mengganti C dan T, dapat dilihat perubahan kata CRATE menjadi TRACE. Pertukaran dua elemen string inilah adalah contoh nyata dari transposisi yang dijelaskan [15].

Dalam pencocokkan DwAyNE dan DuANE memiliki urutan yang sama D-A-N-E, jadi tidak ada transposisi. aro-Winkler distance menggunakan prefix scale (p) yang memberikan tingkat penilaian yang lebih, dan prefix length yang menyatakan panjang awalan yaitu panjang karakter yang sama dari string yang dibandingkan sampai ditemukannya ketidaksamaan. Bila string |s1| dan |s2| yang diperbandingkan, maka Jaro-Winkler distancenya (dw) adalah [15]:

$$\boldsymbol{dw} = \boldsymbol{dj} + (\boldsymbol{lp}(\mathbf{1} - \boldsymbol{dj})) \qquad 2.\ 3$$

Dimana:

dj = Jaro distance untuk string si, *l* = Panjang prefiks umum di awal string nilai maksimalnya 4 karakter (panjang karakter yang sama sebelum ditemukan ketidaksamaan max 4), *p* = Konstanta scaling factor. Nilai standar untuk konstanta ini menurut Winkler adalah *p* = 0.1.

Berikut ini adalah contoh pada perhitungan Jaro- Winkler distance. Jika string s1 MARTHA dan s2 MARHTA maka: *m* = 6, |s1| = 6, |s2| = 6
Karakter yang tertukar hanyalah T dan H. Maka t = 1. Maka nilai Jaro distance adalah:

$$dj = \frac{1}{3} \times \left(\frac{6}{6} + \frac{6}{6} + \frac{6-1}{6}\right) = 0.944$$

Kemudian bila diperhatikan susunan s1 dan s2 dapat diketahui nilai l = 3, dan dengan nilai konstan p = 0.1. Maka nilai Jaro-Winkler distance adalah [15]:
dw = 0.944 + (3 × 0.1 (1 − 0.944)) = 0.961 . Jika string s1 DWAYNE dan s2 DUANE maka: m = 4, |s1| = 6, |s2| = 5, t = 0, hal ini dikarenakan tidak ada karakter yang sama tapi tertukar urutannya. Karakter seperti D, A, N, E dianggap dalam urutan yang sama. Maka nilai Jaro distance adalah [15]:

$$dj = \frac{1}{3} \times \left(\frac{4}{6} + \frac{6}{5} + \frac{4-1}{64}\right) = 0.822$$

Kemudian bila diperhatikan susunan |s1| dan |s2| dapat diketahui nilai *l =1*, dan dengan nilai konstan *p= 0.1*. Maka nilai Jaro-Winkler distance adalah:

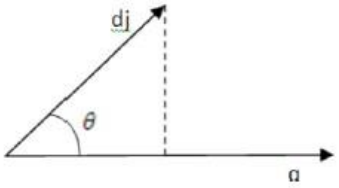
$$dw = 0.822 + (1 \times 0.1\ (1 - 0.8.22)) = 0.961$$

2.6. *Vector Space Model* (VSM)

Vector Space Model (VSM) adalah metode untuk melihat tingkat kedekatan atau kesamaan *similarity term* dengan cara pembobotan *term*. Dokumen dipandang sebagi sebuah vektor yang memiliki *magnitude* (jarak) dan *direction* (arah). Pada *Vector Space Model,* sebuah istilah direpresentasikan dengan sebuah dimensi dari ruang vektor. Relevansi sebuah dokumen ke sebuah *query* didasarkan pada similaritas diantara vektor dokumen dan vektor *query* [16].

Vector space model digunakan untuk mengukur kemiripan antara suatu dokumen dengan suatu query. Konsep dasar dari *Vector Space Model* adalah menghitung jarak antar dokumen kemudian mengurutkan berdasarkan tingkat kedekatannya. Seluruh kata dalam dokumen dibentuk menjadi satu yang disebut sebagai term. Tiap dokumen ditampilkan sebagai vektor yang akan dibandingkan dengan term yang telah dibentuk. Similarity Analysis untuk mengukur kemiripan dokumen dilakukan dengan menghitung cosinus jarak antara dokumen tersebut [16].

Sebuah dokumen *dj* dan sebuah query *q* direpresentasikan sebagai vektor dimensi seperti pada gambar 2.1.



Gambar 2. 1 Representasi Dokumen dan Query pada Ruang Vektor [16]

Keterangan:

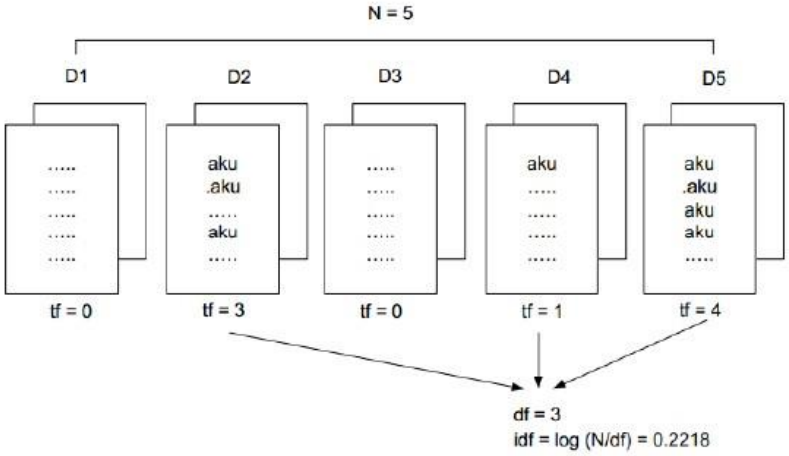
dj = dokumen, θ = sudut dan *q* = query.

2.6.1. *Term Frequency-Inverse Document Frequency* (TF-IDF) *Weighting*

Term Frequency (tf) merupakan frekuensi kemunculan suatu kata (*term*) dalam dokumen. Oleh sebab itu, *tf* memiliki nilai yang bervariasi dari satu dokumen ke dokumen lain tergantung dari tingkat kepentingan sebuah *term* dalam sebuah dokumen. Semakin sering suatu *term* muncul dalam suatu dokumen, *term* tersebut akan memiliki nilai *tf* yang lebih besar daripada *term-term* lain yang jarang muncul.

Penggunaan faktor *tf* belum mencukupi dalam menentukan pembobotan. Untuk itu,diperlukan faktor *Inverse Document Frequency* (*idf*) yang merupakan

sebuah statistik “global” yang mengkarakteristikkan sebuah *term* dalam keseluruhan koleksi dokumen. *idf* merupakan sebuah perhitungan dari bagaimana *term* didistribusikan secara luas pada koleksi dokumen yang bersangkutan. Semakin sedikit dokumen yang mengandung *term* yang dimaksud, maka nilai *idf* semakin *besar*. Jika setiap dokumen dalam koleksi mengandung *term* yang bersangkutan, maka nilai *idf* dari term tersebut adalah nol (0). Hal ini menunjukkan bahwa setiap *term* yang muncul pada dokumen dalam koleksi tidak berguna untuk membedakan dokumen berdasarkan topik tertentu. Ilustrasi algoritma *tf-idf* [17] ditunjukkan pada gambar 2.2 berikut:



Gambar 2. 2 Ilustrasi Algoritma *tf-idf*

Keterangan :

D1 - D5 = dokumen

tf = banyaknya term yang dicari pada setiap dokumen

N = total dokumen

Proses perhitungan *Vector space model* melalui tahapan perhitungan *term frequency (tf)* menggunakan persamaan 2.4 [16].

$$tf = tfij \qquad 2.4$$

Dengan *tf* adalah *term frequency*, dan *tfij* adalah banyaknya kemunculan term *ti* dalam dokumen *dj*, *Term frequency (tf)* dihitung dengan menghitung banyaknya kemunculan term *ti* dalam dokumen *dj* [16].

Perhitungan *Inverse Document Frequency (idf)*, menggunakan persamaan 2.5.

$$\boldsymbol{idfi = log \frac{N}{dfi}} \qquad 2.5$$

Dengan *idfi* adalah *inverse document frequency*, *N* adalah jumlah dokumen yang terambil oleh sistem, dan *dfi* adalah banyaknya dokumen dalam koleksi dimana term *ti* muncul di dalamnya, maka Perhitungan *idfi* digunakan untuk mengetahui banyaknya term yang dicari *(dfi)* yang muncul dalam dokumen lain yang ada pada *database*.

Perhitungan *term frequency Inverse Document Frequency* (*tfidf*), menggunakan persamaan 2.6.

$$\boldsymbol{Wij = tfij.log\frac{N}{dfi}} \qquad 2.6$$

Dengan *Wij* adalah bobot dokumen, *N* adalah Jumlah dokumen yang terambil oleh sistem, *tfij* adalah banyaknya kemunculan term *ti* pada dokumen *dj*, dan *dfi* adalah banyaknya dokumen dalam koleksi dimana term *ti* muncul di dalamnya. Bobot dokumen *(Wij)* dihitung untuk didapatkannya suatu bobot hasil perkalian atau kombinasi antara *term frequency (tfij)* dan *Inverse Document Frequency (idf)* [16].

Perhitungan Jarak *query*, menggunakan persamaan 2.7.

$$|q| = \sqrt{\sum_{i=1(Wi,q)}^{t} \mathbf{2}} \qquad 2.7$$

Dengan |q| adalah Jarak query, dan Wiq adalah bobot query dokumen ke-i, maka Jarak query (|q|) dihitung untuk didapatkan jarak query dari bobot query dokumen (Wiq) yang terambil oleh sistem. Jarak query bisa dihitung dengan persamaan akar jumlah kuadrat dari query [16].

Perhitungan jarak dokumen, menggunakan persamaan 2.8.

$$|dj| = \sqrt{\sum_{i=1(Wi,q)}^{t} \mathbf{2}} \qquad 2.8$$

Dengan |*dj*| adalah jarak dokumen, dan *Wij* adalah bobot dokumen ke-*i*, maka Jarak dokumen *(|dj |)* dihitung untuk didapatkan jarak dokumen dari bobot

dokumen dokumen *(Wij)* yang terambil oleh sistem. Jarak dokumen bisa dihitung dengan persamaan akar jumlah kuadrat dari dokumen.

Menghitung *index terms* dari dokumen dan query *(q,dj).* menggunakan persamaan 2. 9 **[16]**.

$$\boldsymbol{q, dj = \sum_{i=1}^{t} Wi, q, W, j} \qquad 2.\ 9$$

Dengan *Wij* adalah bobot *term* dalam dokumen, *Wiq* adalah bobot *query*.

Pengukuran *Cosine Similarity* menghitung nilai *kosinus* sudut antara dua *vector* menggunakan persamaan 2.10 [16].

$$\boldsymbol{Sim(q, dj) = \frac{q.dj}{|q| * |dj|}} \qquad 2.\ 10$$

Similaritas antara *query* dan dokumen atau *Sim(q,dj)* berbanding lurus terhadap jumlah bobot *query (q)* dikali bobot dokumen *(dj)* dan berbanding terbalik terhadap akar jumlah kuadrat *q (|q|)* dikali akar jumlah kuadrat dokumen *(|dj|).* Perhitungan similaritas menghasilkan bobot dokumen yang mendekati nilai 1 atau menghasilkan bobot dokumen yang lebih besar dibandingkan dengan nilai yang dihasilkan dari perhitungan *inner product* [16].

2.7. Pemodelan

Skema pemodelan data adalah metode yang digunakan untuk memodelkan atau menggambarkan *database* pada aplikasi perbaikan kesalahan ejaan pada sistem essay scoring yang akan dibangun.

2.7.1. *Entity Relationship* Diagram

ERD merupakan suatu model jaringan yang menggunakan susunan data yang disimpan pada sistem secara abstrak. Tujuan dari *entity relationship* adalah untuk menunjukkan objek data dan relationship yang ada pada objek tersebut. Di samping itu model ER ini merupakan salah satu alat untuk perancangan dalam basis data [18].

Komponen-komponen ERD adalah sebagai berikut:

1. Entitas

Entitas menunjukkan objek-objek dasar yang terkait didalam sistem. Objek dasar dapat berupa orang, benda atau hal lain yang keterangannya perlu disimpan dalam basis data.

2. Atribut

Atribut sering juga disebut sebagai properti merupakan keterangan-keterangan yang terkait pada sebuah entitas yang perlu disimpan sebagai basis data. Atribut berfungsi sebagai penjelas sebuah entitas.

Jenis-jenis *Kardinalitas*:

a. Satu ke Satu (1 : 1)

   Yaitu perbandingan antara entity pertama dengan entity kedua berbanding satu berbanding satu.

b. Satu ke Banyak (1 : N)

   Yaitu perbandingan antara entity pertama dengan entity kedua berbanding satu berbanding banyak.

c. Banyak ke Satu (N : 1)

   Yaitu perbandingan antara entity pertama dengan entity kedua berbanding banyak berbanding satu.

d. Banyak ke Banyak (N : N)

   Yaitu perbandingan antara entity pertama dengan entity kedua berbanding banyak berbanding banyak.

3. Relasi

Adalah karakteristik dari entity atau relationship yang menyediakan penjelasan detail tentang entity atau relationship tersebut. Relasi atau hubungan adalah kejadian atau transaksi yang terjadi di antara dua entitas yang keterangannya perlu disimpan dalam basis data.

4. Derajat Relasi (*Kardinalitas*)

Kardinalitas relasi menunjukkan maksimum entitas yang dapat berelasi dengan entitas pada himpunan entitas yang lain. Kardinalitas relasi yang terjadi di antara dua himpunan entitas (misalkan A dan B) dapat berupa satu ke satu (*one to one*), satu ke banyak (*one to many*), banyak ke satu (*many to one*), dan banyak ke banyak (*many to many*).

2.7.2. Diagram Konteks

Diagram Konteks adalah sebuah diagram sederhana yang menggambarkan hubungan antara *entity* luar, masukan dan keluaran dari sistem. Diagram konteks direpresentasikan dengan lingkaran tunggal yang mewakili keseluruhan sistem [8]. Diagram konteks dimulai dengan penggambaran terminator, aliran data, aliran kontrol penyimpanan, dan proses tunggal yang menunjukan keseluruhan sistem. Ada beberapa aturan dalam menggambarkan diagram konteks, berikut di bawah ini menunjukan aturan tersebut [18].

1. Menggunakan hanya satu simbol proses
2. Memberi label simbol proses tersebut untuk menggambar seluruh sistem, biasanya berupa kata kerja di tambah objek
3. Tidak memberi nomor pada simbol proses
4. Menyertakan semua terminator dari sistem
5. Menunjukan semua arus data antara terminator dan sistem

2.7.3. DFD (Data *Flow Diagram*)

DFD adalah suatu model logika data atau proses yang dibuat untuk menggambarkan darimana asal data dan kemana tujuan data yang keluar dari sistem, dimana data disimpan, proses apa yang menghasilkan data tersebut dan interaksi antara data yang tersimpan dan proses yang dikenakan pada data tersebut [18].

1. Arus Data (*Data Flow*)

   Menunjukan arus dari data yang dapat berupa masukan untuk sistem atau dari proses sistem.
2. Proses

   Proses adalah kegiatan yang dilakukan oleh orang, mesin atau komputer dari hasil arus data yang masuk ke dalam proses untuk dihasilkan arus data yang akan keluar dari proses.
3. Kesatuan Luar (*External Entity*)

   Kesatuan luar merupakan kesatuan di lingkungan luar sistem yang dapat berupa orang, organisasi atau sistem lain yang akan memberikan masukan atau menerima keluaran dari sistem.

4. File

   Kumpulan data yang disimpan dengan cara tertentu. Data yang mengalir disimpan dalam file. Aliran data di-update atau ditambahkan ke dalam file.

2.7.4. Kamus Data

Kamus data adalah kumpulan daftar elemen data yang mengalir pada sistem perangkat lunak sehingga masukan (*input*) dan keluaran (*output*) dapat dipahami secara umum [18]. Kamus data mendefinisikan elemen data dengan menjelaskan arti aliran data dan penyimpanan data dalam DFD, mendeskripsikan komposisi paket data yang bergerak melalui aliran, mendeskripsikan komposisi penyimpanan data, menspesifikasikan nilai dan satuan yang relevan bagi penyimpanan dan aliran, mendeskripsikan hubungan detil antar penyimpanan.

Kamus data berfungsi untuk membantu pelaku sistem untuk mengartikan alokasi secara detail dan mengorganisasikan semua elemen data yang digunakan dalam sistem secara persis sehingga baik pemakai atau penganalisis sistem mempunyai dasar pengertian yang sama tentang masukan, keluaran, penyimpanan dan proses.

Kamus data biasanya berisi:

1. Nama-nama dari data
2. Digunakan pada – merupakan proses-proses yang terkait data
3. Deskripsi – merupakan deskripsi data
4. Informasi tambahan – seperti tipe data, nilai data, batas nilai data, dan komponen yang membentuk data.

Kamus data memiliki simbol untuk menjelaskan informasi tambahan seperti pada tabel 2.2 berikut.

Tabel 2. 2 Kamus Data

Simbol	Keterangan
=	disusun atau terdiri dari
+	Dan
[|]	baik ... atau....
$\{\}^n$	n kali diulang/ bernilai banyak
()	data opsional
...	batas komentar

2.8. Bahasa Pemrograman

Bahasa pemrograman adalah kode-kode yang mempunyai sebuah aturan dan fungsi yang digunakan untuk membangun sebuah program aplikasi. Bahasa pemerograman ini memiliki beberapa jenis bahasa seperti, HTML, Java, PHP, C, C++, C#, PHP, dan lain-lain. Bahasa yang dipakai untuk membangun aplikasi perbaikan kesalahan ejaan pada sistem essay scoring adalah menggunakan bahasa PHP dan JavaScript.

2.8.1. PHP

PHP *Hypertext Preprocessor* atau disingkat dengan PHP ini adalah suatu bahasa *scripting* khususnya digunakan untuk *web development*. Karena sifatnya yang *server side scripting*, maka untuk menjalankan PHP harus menggunakan web server. PHP juga dapat diintegrasikan dengan HTML, JavaScript, Jquery, Ajax. Namun, pada umumnya PHP lebih banyak digunakan bersamaan file bertipe HTML [19].

2.8.2. *Java Script*

*Java scrip**t*** adalah bahasa script berdasar pada objek yang memperbolehkan pemakai untuk mengendalikan banyak aspek interaksi pemakai pada suatu dokumen HTML. Dimana objek teresbut dapat berupa suatu window, frame, dokumen, URL, form, button, atau item yang lain. Yang semuanya itu mempunyai properti yang saling berhubungan dengannya, dan masing-masing memiliki nama, lokasi, warna nilai, dan atribut lain.
[20].

2.9. Pengelola *Database*

Banyak definisi harfiah soal database dan orang awam pun dapat memahami bahwa database adalah sebuah tempat penyimpanan data. Database adalah sebuah tempat penyimpanan data dari kumpulan informasi di dalam komputer yang dihimpun secara sistematis sehingga dapat diolah oleh program komputer yang dapat menghasilkan sebuah data untuk tujuan tertentu. Hasil olahan yang dihasilkan dari database biasanya bisa digunakan untuk mengambil sebuah keputusan penting bagi sebuah perusahaan, database juga bisa digunakan untuk menyimpan data history sebuah perusahaan dan masih banyak lagi fungsinya [21].

Software yang digunakan untuk mengelola dan permintaan panggilan (*query*) basis data yang disebut sistem manajemen *database* (*database management system*, DBMS). Dalam pengelolaan *database* peneliti menggunakan MySQL dengan menggunakan bahasa SQL. Penjabaran MySQL dan SQL akan dijelaskan seperti berikut.

2.9.1 SQL (Structured Query Language)

SQL adalah kependekan dari *Structured Query Language*. SQL adalah bahasa yang digunakan untuk mengakses data dalam basis data relasional. Dalam penggunaannya SQL memiliki beberapa aturan yang telah distandarkan oleh asosiasi yang bernama ANSI. Saat ini hampir semua produk RDBMS menggunakan SQL sebagai standar bahasa Query nya seperti Oracle, MySQL, MariaDB, SQL Server dan lainnya [22].

2.9.2. 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 [23].

2.9.3. Diagram Alir

Diagram alir sistem merupakan diagram yang menggambarkan sistem secara keseluruhan. Dapat dikatakan diagram alir sistem menggambarkan sistem secara umum. Diagram alir sistem tidak digunakan untuk menggambarkan urutan langkah untuk memecahkan masalah hanya untuk menggambarkan prosedur dalam sistem yang dibangun.

2.10. Perangkat Lunak Pembangunan

Perangkat lunak adalah istilah khusus dalam membangun sebuah program aplikasi. Perangkat lunak yang dibutuhkan dalam pembangunan aplikasi perbaikan kesalahan ejaan pada sistem essay scoring seperti XAMPP, Web Browser, Sublime Text 3. Berikut dijelaskan dibawah.

2.10.1. XAMPP

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 [24]. Dalam penelitian ini XAMPP digunakan sebagai server *localhost* untuk membuka aplikasi perbaikan kesalahan ejaan pada sistem essay scoring.

2.10.2. Web Browser

Web Browser adalah sebuah perangkat lunak yang digunakan sebagai tempat mesin pencari informasi. Informasi-informasi yang kita inginkan bisa kita cari dengan bantuan perangkat lunak ini. Peneliti juga menggunakan perangakt lunak ini untuk megakses program aplikasi perbaikan kesalahan ejaan pada sistem essay scoring yang dibangun.

2.10.3. Sublime Text 3

Sublime Text 3 adalah sebuah aplikasi untuk menuliskan dan mengedit kode-kode progam yang akan dibuat, seperti HTML, PHP, Java, dan lain-lain. Peneliti menggunkan sublime text 3 untuk menulisankan program-program yang akan dibuat untuk aplikasi perbaikan kesalahan ejaan pada sistem essay scoring.

2.11. *Recall and Precision*

Recall adalah proporsi jumlah dokumen yang dapat ditemukan-kembali oleh sebuah proses pencarian di sistem IR. Rumusnya: Jumlah dokumen relevan yang ditemukan /jumlah semua dokumen relevan di dalam koleksi. Lalu, *precision*

adalah proporsi jumlah dokumen yang ditemukan dan dianggap relevan untuk kebutuhan si pencari informasi. Rumusnya: Jumlah dokumen relevan yang ditemukan / Jumlah semua dokumen yang ditemukan. Sedangkan *Precision* dapat diartikan sebagai kepersisan atau kecocokan (antara permintaan informasi dengan jawaban terhadap permintaan itu). Jika seseorang mencari informasi di sebuah sistem, dan sistem menawarkan beberapa dokumen, maka kepersisan ini sebenarnya juga adalah relevansi. Artinya, seberapa persis atau cocok dokumen tersebut untuk keperluan pencari informasi, bergantung pada seberapa relevan dokumen tersebut bagi si pencari [25].

Tabel 2. 3 Recall and Precision [25]

		Nilai Sebenarnya	
		TRUE	FALSE
Nilai Prediksi	TRUE	TP (True Positive) Correct result	FP (False Positive) Unexpected result
	FALSE	FN (False Negative) Missing result	TN (True Negative) Correct absence of result

$$precision = \frac{TP}{TP + FP}$$

$$recall = \frac{TP}{TP + FN}$$

$$acuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

2.11.1. *Black Box Testing*

Pengujian *black box* merupakan pendekatan komplementer dari teknik *white box*, karena pengujian *black box* diharapkan mampu mengungkap kelas kesalahan yang lebih luas dibandingkan teknik white box. Pengujian *black box* berfokus pada pengujian persyaratan fungsional perangkat lunak, untuk

mendapatkan serangkaian kondisi input yang sesuai dengan persyaratan fungsional suatu program.

Pengujian black box adalah pengujian aspek fundamental sistem tanpa memperhatikan struktur logika internal perangkat lunak. Metode ini digunakan untuk mengetahui apakah perangkat lunak berfungsi dengan benar. Pengujian black box merupakan metode perancangan data uji yang didasarkan pada spesifikasi perangkat lunak. Data uji dibangkitkan, dieksekusi pada perangkat lunak dan kemudian keluaran dari perangkat lunak dicek apakah telah sesuai dengan yang diharapkan [26].

Pengujian black box berusaha menemukan kesalahan dalam kategori:

1. fungsi-fungsi yang tidak benar atau hilang,
2. kesalahan interface,
3. kesalahan dalam struktur data atau akses database eksternal,
4. kesalahan kinerja, serta

inisialisasi dan kesalahan terminasi.