BAB I

PENDAHULUAN

1.1. Latar belakang masalah

Mengetik adalah suatu kegiatan untuk menuliskan sebuah tulisan dalam bentuk cetakkan yang sudah dirangkai oleh mesin ketik. Didalam rangkaian mesin ketik terdapat cetakkan huruf yang sudah diatur sesuai dengan aturan posisi keyboard yang ada pada mesin ketik. Dalam pembuatan dokumen, mesin ketik sangat cepat untuk menuliskan sebuah teks dengan rapi dan teratur, jika dibandingkan dengan penulisan manual yang memiliki kekurangan dalam banyak hal, seperti bentuk huruf yang tidak bisa dibaca, ukuran huruf yang tidak sama ukurannya, tata letak yang acak acakkan, dan masih banyak lagi.

Zaman pasti berubah begitu juga dengan teknologi, kemajuan teknologi setiap tahun berkembang. Mesin ketik terkena dampak dari perubahan teknologi ini, mesin ketik digitalisasikan oleh komputer karena lebih efisien dalam membuat sebuah tulisan atau teks. Sebuah teks atau tulisan yang mudah dipahami dalam penyampaian informasi tidak memiliki kesalahan kata atau *typo* yang mengakibatkan ketidakjelasan dalam informasi yang disampaikan oleh karena itu banyak aplikasi pengolahan kata muncul.

Salah satu aplikasi pengolah kata seperti microsoft office word, memiliki fitur *suggestions word* dan *autocorrect word* yang dimana sangat berguna dalam mengecek sebuah tulisan yang terdapat kesalahan kata dalam penulisannya. Fungsi dari *suggestions word* dan *autocorrect word* sangat berguna dalam

menuliskan sebuah tulisan, terlebih lagi dalam penulisan dokumen penting seperti jurnal ilmiah, skripsi, naskah dan masih banyak lagi.

Didalam sebuah website memiliki *text* editor yang dimana digunakan untuk mengolah kata sama halnya seperti microsoft office dan aplikasi lainnya, tetapi fitur di *text* editor yang ada di website terbatas, terlebih lagi soal mendeteksi kata yang salah atau *typo* diinputkan oleh user. Dengan kesalahan kata ini banyak artikel yang diterbitkan memiliki kata yang tidak sesuai dengan bahasa yang baku. Ini akan membuat user yang mengunjungi website sulit untuk menerima informasi yang disajikan oleh website tersebut.

Dalam penelitian ini penulis membangun sebuah *library* yang meng *handle* masalah tersebut, penulis akan menamakan *library* ini saltikjs. *Library* ini memiliki fitur mendeteksi kata yang salah dan menggantinya dengan kata yang benar, dan saltikjs ini akan tulis dengan javascript sebagai *base language* nya. dalam pembuatan *library* ini akan dibangun dari awal tanpa menggunakan *library* lain yang artinya *pure* dari javascript itu sendiri agar saltikjs ini *standalone* yaitu berdiri sendiri tanpa membutuhkan *library* lain sebagai tambahannya.

Untuk mendeteksi kata yang salah dibutuhkan metode pencarian yang memungkinkan waktu yang efisien dalam pencarian kata pada suatu teks, Dan membuat rekomendasi Kata Yang tepat untuk mencari kata yang benar dari kata yang salah tersebut memerlukan algoritma yang dapat mendeteksi kesamaan antara kata yang benar dan kata yang salah. Dalam Penelitian ini menggunakan *Regular Search Expression* sebagai metode pencarian kata yang salah dan

algoritma *Levenshtein Distance* sebagai algoritma yang mendeteksi kesamaan kata.

Berdasarkan latar belakang yang diatas, bahwasanya metode R*egular Search Expression* banyak digunakan dalam *search engine* favorit yang sering kita gunakan seperti Google, Yahoo, dst. dan penggunaan dari metode ini juga sangat luas dan tidak banyak digunakan dalam *search engine*, tetapi metode ini bisa ditemukan hampir dalam setiap fitur atau aplikasi yang memiliki fitur "*find*" atau "*replace*" [1, pp. 170]. Algoritma *Levenshtein Distance* adalah suatu pengukuran untuk menghitung jumlah perbedaan antara dua kata. Perhitungan jarak antara dua kata ditentukan dari jumlah minimum operasi perubahan untuk mengubah kata A menjadi kata B [3, pp. 130].

1.2. Rumusan masalah

a. Apakah algoritma *Levenshtein Distance* dapat diterapkan pada *library* saltikjs sebagai algoritma yang mencocokan kesamaan kata?
b. Apakah metode *Regular Search Expression* dapat diterapkan pada *library* saltikjs sebagai metode pencarian kata dalam teks?
c. Apakah Algoritma *Levenshtein Distance* bisa diterapkan pada fitur Rekomendasi Kata untuk kata yang salah?
d. Apakah metode *Regular Search Expression* dapat mencari kata yang salah berdasarkan data kamus yang ada?

1.3. Batasan masalah

a. *Library* diimplementasikan menggunakan javascript murni.

b. *Library* dibangun untuk mencari kata yang salah dan mencari kata yang benar berdasarkan data pada kamus yang tersedia.

c. Data kamus yang digunakan penulis dalam penelitian adalah data kamus *dummy* bahasa indonesia.

d. Hanya menggunakan element html yang beratributkan *contenteditable plaintext-only* sebagai editor teks.

e. *Library* saltikjs hanya diterapkan di website.

1.4. Maksud Dan Tujuan Penelitian

a. Membangun *library* javascript yang berfungsi untuk mendeteksi salah ketik atau kata yang *typo*.

b. Membuat fitur rekomendasi kata pada saltikjs.

c. Mengimplementasikan algoritma *Levenshtein Distance* untuk mendeteksi kesamaan kata.

d. Mengimplementasikan *Regular Search Expression* dalam pencarian kata pada *library* saltikjs.

1.5. Manfaat Penelitian

a. Membantu web developer untuk mendeteksi kata yang salah dan merekomendasikan kata yang benar untuk pengguna web.

b. Memudahkan pengguna web mengecek salah kata pada tulisan atau laporan.

1.6. Metode Penelitian

Metode Penelitian yang dilakukan dalam pembuatan *library* saltikjs ini adalah sebagai berikut :

1.6.1. Metode Pengumpulan Data

a. Mempelajari jurnal yang mengangkat metode *Regular Search Expression* sebagai metode dalam penelitian jurnal tersebut.
b. Mempelajari jurnal yang mengangkat algoritma *Levenshtein Distance* sebagai algoritma dalam penelitian jurnal tersebut.
c. Mempelajari Javascript dan metode *object oriented programming* pada javascript.

1.6.2. Metode Pengembangan Saltikjs

a. Menggunakan javascript sebagai *base language* dari *library* saltikjs.
b. Menggunakan Metode *Regular Search Expression* ke dalam *library* saltikjs.
c. Menggunakan Algoritma *Levenshtein Distance* ke dalam *library* saltikjs.
d. Menggunakan *object oriented programming* pada saltikjs.
e. Implementasi saltikjs pada website.

1.7. Sistematika Penulisan

Laporan penelitian ini, penulis susun menjadi 5 bab, dengan masing - masing bab diuraikan sebagai berikut :

BAB I : PENDAHULUAN

Bab ini berisi Latar Belakang Masalah, Rumusan Masalah, Batasan Masalah, Maksud Dan Tujuan Penelitian, Manfaat Penelitian, Metode Penelitian, dan Sistematika Penulisan.

BAB II : LANDASAN TEORI

Bab ini menjelaskan teori-teori yang mendukung penelitian ini, yang berupa tinjauan pustaka seperti pengertian javascript, pengertian *library*, pengertian *object oriented programming*, pengertian algoritma, pengertian algoritma *Levenshtein Distance*, dan penjelasan metode *Regular Search Expression*.

BAB III : ANALISIS DAN PERANCANGAN SISTEM

Bab ini menjelaskan tentang analisis dan perancangan *library* yang akan dibuat sekaligus batasan batasan *library* serta untuk menguraikan perancangan *library* yang dibuat.

BAB IV : IMPLEMENTASI DAN PEMBAHASAN

Bab ini berisi mengenai tentang implementasi dan pembahasan *library* yang telah dibuat berdasarkan hasil perancangan pada bab 3 sebelumnya.

BAB V : KESIMPULAN DAN SARAN

Bab ini berisi kesimpulan dan saran seluruh penelitian yang telah dilakukan.