BAB 3
METODOLOGI PENELITIAN

3.1 Studi Literatur

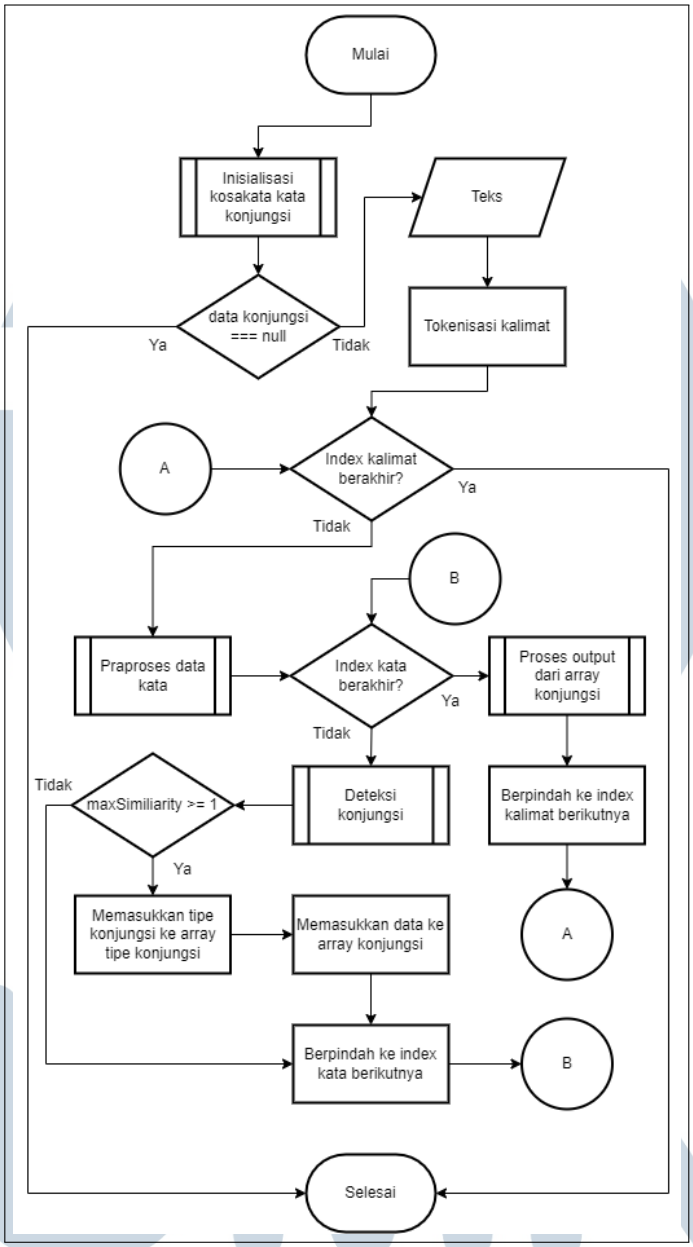
Pada tahap studi literatur, dilakukan pengumpulan informasi seperti teori konsep maupun teknis yang relevan dengan kebutuhan yang diperlukan dalam penelitian ini. Informasi yang dikumpulkan sebagian besar menjelaskan mengenai berita yang disajikan oleh portal berita Tribun, algoritma *cosine similarity*, tata berbahasa yang tepat terutama penggunaan kata konjungsi, dan *confusion matrix*. Studi literatur dilakukan untuk memahami lebih dalam mengenai teknologi yang digunakan dan memperkuat fondasi konsep dalam peracangan sistem yang ingin dikembangkan.

3.2 Perancangan Sistem

Tahap perancangan sistem dilakukan sebagai upaya untuk merancang arsitektur yang akan diimplementasi untuk mendukung kerja sistem agar dapat bekerja sesuai dengan ekspektasi dari penelitian ini. Sistem yang dibangun pada penelitian ini berbentuk *website* menggunakan bahasa pemrograman Javascript dengan *framework* Next.JS. Perancangan sistem dilakukan menggunakan diagram alir atau *flowchart* sedangkan rancangan tampilan antarmuka dilakukan menggunakan desain prototipe.

3.2.1 Flowchart

Diagram alir atau *flowchart* digunakan untuk memvisualisasikan tahap perancangan sistem.

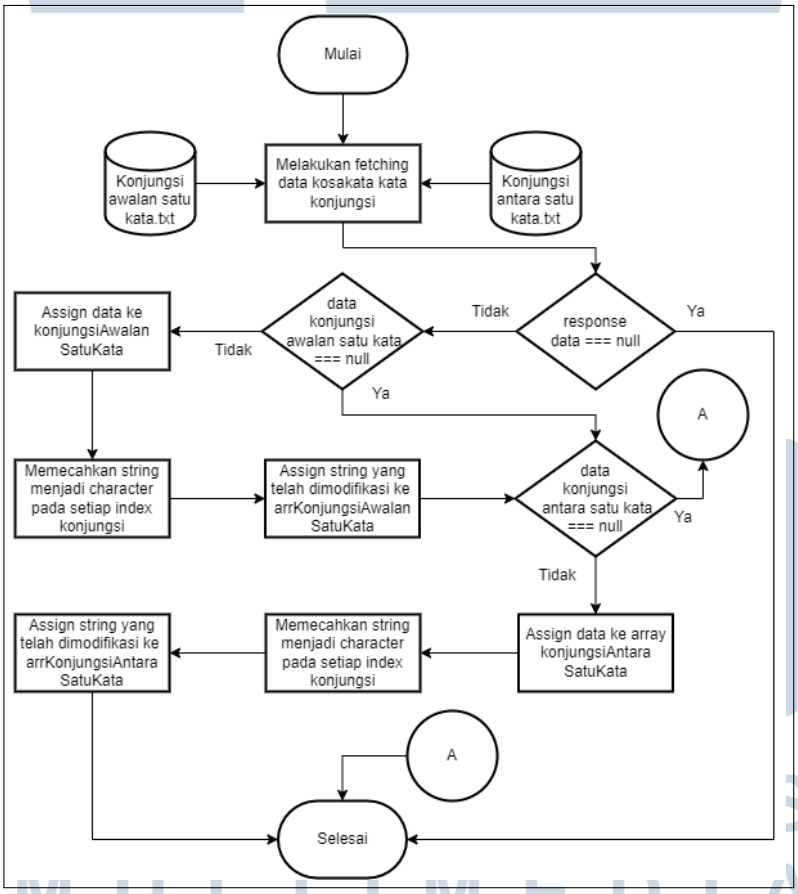


Gambar 3.1. Diagram Alir *High Level* Sistem Pendeteksi Konjungsi

Gambar 3.1 merupakan diagram alir *high level* yang menjabarkan proses utama pada rancangan sistem. Rancangan sistem dimulai dengan inisialisasi data kosakata kata konjungsi. Inisialisasi dilakukan agar sistem dapat memahami macam konjungsi yang ingin dideteksi. Tahap inisialisasi dilanjutkan dengan penerimaan input berupa teks yang bertipe data *string*. Input teks bersifat fleksibel yang dapat berupa kalimat, paragraf, ataupun teks artikel. Berdasarkan input yang didapatkan, data input diolah kembali melalui tahap praproses data agar data menjadi lebih terstruktur dan mudah untuk diproses pada tahap berikutnya. Tahap praproses

data mengembalikan data baru yang telah disaring dan diolah. Proses perhitungan *cosine similarity* mengembalikan nilai *similarity* dengan rentang nilai 0 hingga 1. Kemudian, dilakukan pengecekkan tingkat similaritas dengan menguji nilai *similarity* yang diperoleh melalui proses perhitungan *cosine similarity*. Apabila angka *similarity* lebih besar sama dengan 1, maka data akan dimasukkan ke dalam *array* konjungsi. Pada proses output yang dilakukan dari *array* konjungsi menggunakan data yang telah dicek dan diolah untuk menampilkan hasil deteksi dan nilai *similarity* untuk setiap iterasi data.

A. Inisialisasi Kosakata Kata Konjungsi

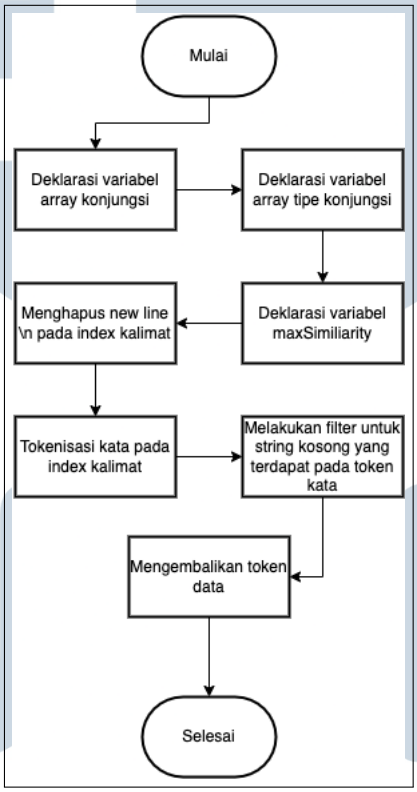


Gambar 3.2. Diagram Alir Proses Inisialisasi Kosakata Kata Konjungsi

Gambar 3.2 merupakan diagram alir yang menggambarkan proses inisialisasi kosakata kata konjungsi. Data kata konjungsi akan digunakan sebagai acuan sistem dalam mendeteksi kata konjungsi pada proses selanjutnya. Proses dimulai dengan *fetch* data *text file* konjungsi awalan satu kata dan konjungsi antara

satu kata yang memiliki format txt. Apabila *response* mengembalikan data, maka proses akan berlanjut. Namun jika *response* tidak mengembalikan data, maka proses akan berakhir. Setelah data *response* diterima, data akan dicek apabila data merupakan data konjungsi awalan satu kata dan konjungsi antara satu kata, masing-masing data akan disimpan ke dalam array tersendiri. Data kata konjungsi akan diolah dengan memecah *string* pada *array* menjadi *character*, hasil dari olah data ini menghasilkan *array* yang berisikan *array of character*

B. Praproses Data

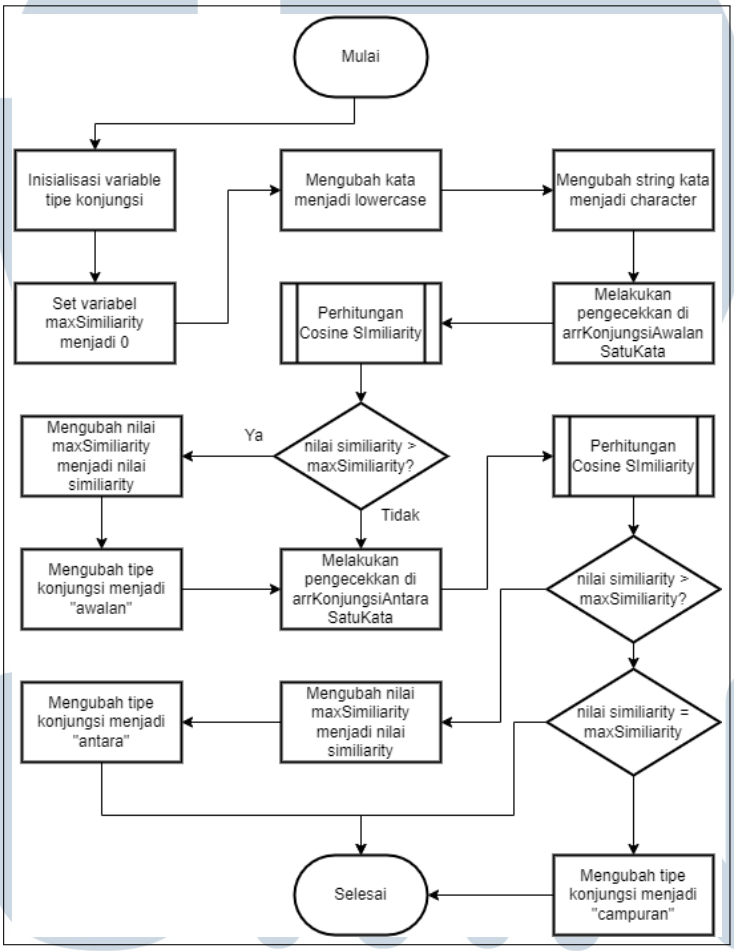


Gambar 3.3. Diagram Alir Praproses Data

Gambar 3.3 merupakan diagram alir yang menggambarkan tahap praproses data. Pada tahapan ini dilakukan beberapa deklarasi variabel yaitu *array* konjungsi, *array* tipe konjungsi, dan *maxSimilarity* yang akan digunakan pada tahap deteksi konjungsi. Praproses data dimulai dengan mengubah *string* yang didapatkan melalui input teks menjadi huruf kecil atau *lowercase*. Dikarenakan data input yang diterima berupa teks atau artikel maka perlu dilakukan penghapusan *new line* pada teks agar data dapat diolah lebih mudah. Teks yang telah dipraproses akan

dipecah menjadi satuan token dan dilakukan *filtering* untuk string kosong setelah tahap tokenisasi. Hasil yang dikembalikan oleh proses merupakan token yang telah melalui tahap praproses data.

C. Deteksi Konjungsi

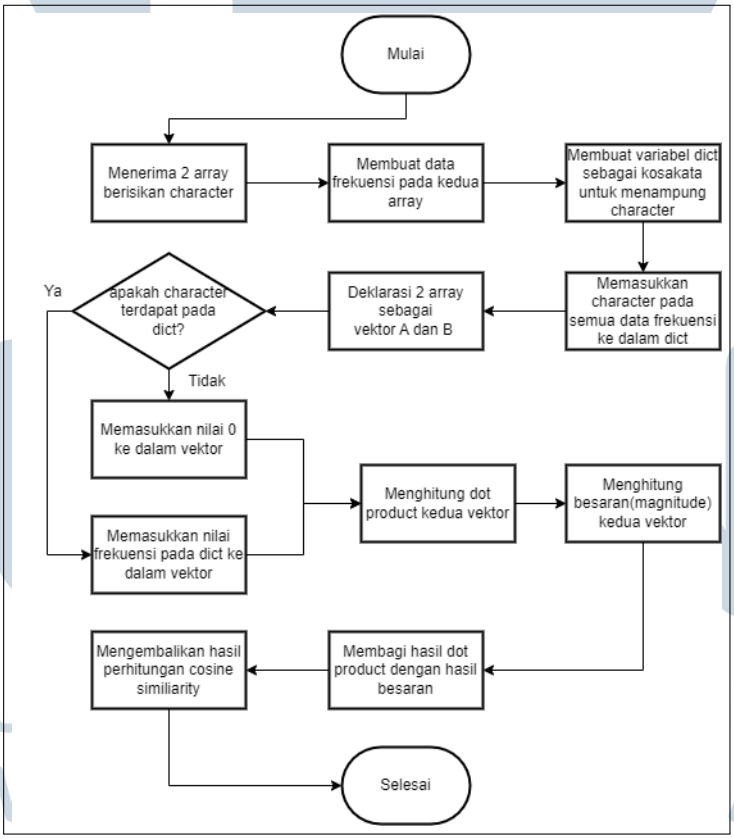


Gambar 3.4. Diagram Alir Proses Deteksi Konjungsi

Gambar 3.4 merupakan diagram alir yang menggambarkan proses sistem dalam mendeteksi konjungsi. Proses dimulai dengan inisialisasi data yaitu deklarasi variabel tipe konjungsi dan *assign* variabel *maxSimilarity* menjadi 0. Input teks diubah dari *string* menjadi *character* untuk memudahkan proses perhitungan dengan data kata konjungsi. Pengecekkan pertama dilakukan dengan data yang terdapat di *array* konjungsi awalan satu kata. Setelah proses perhitungan *cosine similarity* mengembalikan nilai *similarity*, nilai tersebut akan

dibandingkan dengan nilai *maxSimilarity*. Apabila nilai similarity lebih besar daripada nilai *maxSimilarity*, maka nilai *maxSimilarity* akan diubah menjadi nilai *similarity*. Kemudian, nilai variabel tipe konjungsi akan diubah menjadi ”awalan”. Pengecekkan selanjutnya dilakukan dengan data yang terdapat di *array* konjungsi antara satu kata. Setelah proses perhitungan *cosine similarity* mengembalikan nilai *similarity*, dilakukan pengecekan apabila nilai *similarity* sama dengan nilai *maxSimilarity*. Apabila benar, maka tipe konjungsi akan diubah menjadi ”campuran” dan proses akan berakhir. Apabila salah, dilakukan pengecekkan kembali apabila nilai *similarity* besar daripada nilai *maxSimilarity*. Kemudian, nilai variabel tipe konjungsi akan diubah menjadi ”antara”.

C.1 Cosine Similarity

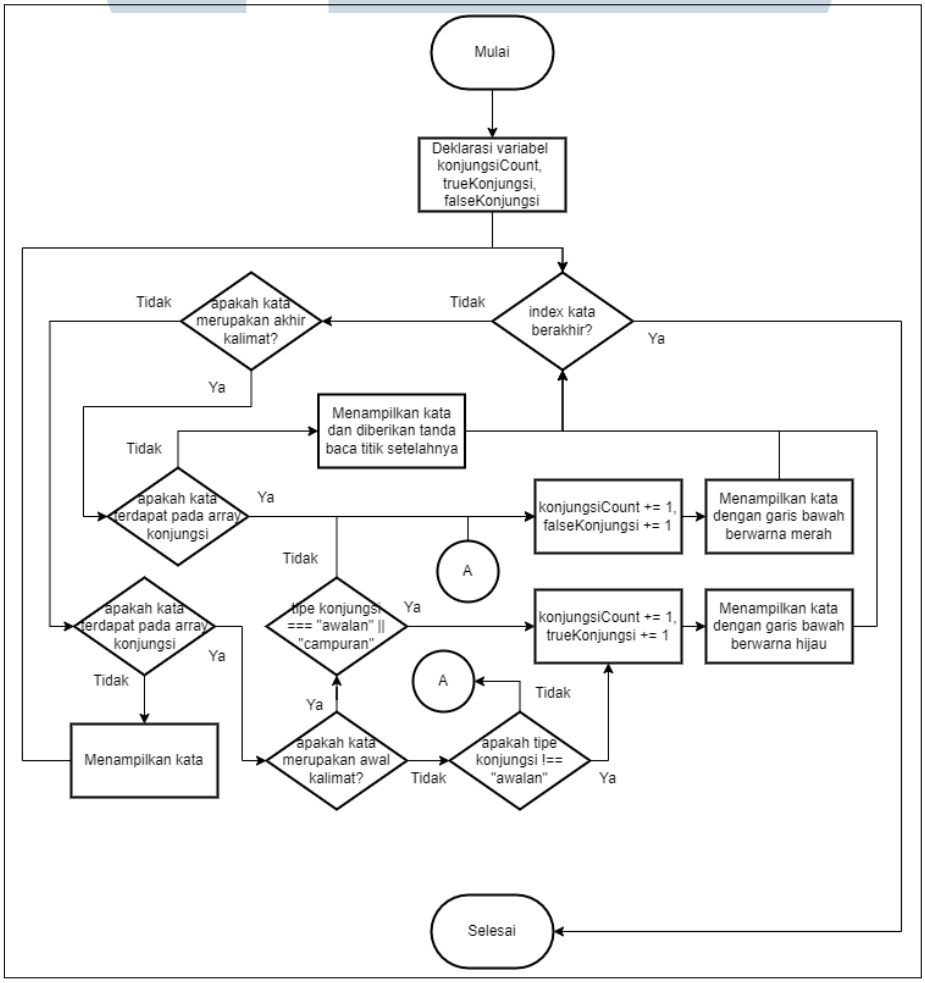


Gambar 3.5. Diagram Alir Proses Perhitungan *Cosine Similarity*

Gambar 3.5 merupakan diagram alir yang menggambarkan proses perhitungan *cosine similarity*. Proses perhitungan *cosine similarity* menerima 2 input berupa *array* yang berisikan *character*. Berdasarkan input yang diterima,

dibuat distribusi frekuensi untuk setiap *character* ke dalam *array* terpisah. Kemudian, dilakukan deklarasi variabel dict yang digunakan sebagai kosakata yang menyimpan *character* yang terdapat pada kedua *array*. Kedua *array* tersebut selanjutnya diubah menjadi vektor dengan memanfaatkan data frekuensi dan variabel dict. Proses dilanjutkan dengan mengimplementasikan *formula cosine similarity* dengan parameter yang telah diubah menjadi vektor yang didapatkan. Formula *cosine similarity* adalah *dot product* kedua vektor dibagi dengan *magnitude* vektor A dikalikan dengan *magnitude* vektor B. Proses diakhiri dengan mengembalikan hasil perhitungan *cosine similarity*.

D. Proses Output dari *Array* Konjungsi



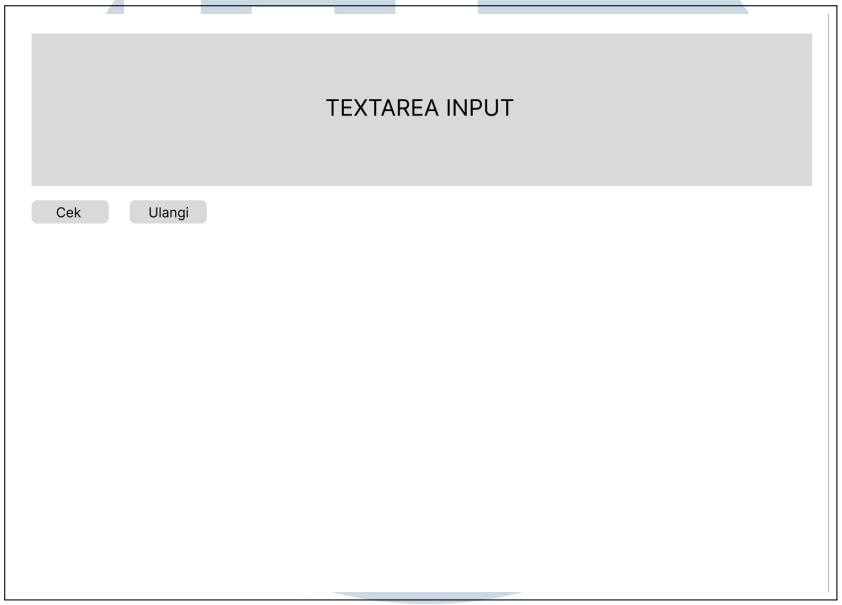
Gambar 3.6. Diagram Alir Proses Output dari *Array* Konjungsi

Gambar 3.6 merupakan diagram alir yang menggambarkan proses output deteksi konjungsi. Proses dimulai dengan melakukan deklarasi beberapa variabel yaitu konjungsiCount, trueKonjungsi, dan falseKonjungsi, Variabel konjungsiCount digunakan sebagai indikator jumlah konjungsi yang terdeteksi. Variabel trueKonjungsi digunakan sebagai indikator jumlah konjungsi terdeteksi yang sesuai dengan format penggunaaan kata konjungsi. Variabel falseKonjungsi digunakan sebagai indikator jumlah konjungsi terdeteksi yang tidak sesuai dengan format penggunaan kata konjungsi.

Proses output dilakukan dengan iterasi untuk setiap kata yang terdapat pada kalimat. Pada setiap iterasi, kata akan dicek dan ditampilkan sesuai dengan validasi yang berlaku. Pengecekkan pertama adalah mengidentifikasi apabila kata terletak di akhir kalimat. Apabila benar, maka dilakukan pengecekkan kedua yaitu mengidentifikasi apakah kata terdapat di *array* konjungsi. Apabila benar, maka variabel konjungsiCount dan falseKonjungsi ditambahkan dan kata tersebut ditampilkan dengan garis bawah berwarna merah. Hal ini dilakukan karena kata konjungsi memiliki tata penulisan yang tidak sesuai dengan kaidah yang diterapkan. Apabila salah, maka kata tersebut ditampilkan dan diberikan tanda baca berupa titik setelahnya. Apabila kata tidak terletak di akhir kalimat, maka dilakukan pengecekkan yaitu mengidentifikasi apakah kata terdapat di *array* konjungsi. Apabila salah, maka kata akan ditampilkan. Apabila benar, maka dilakukan pengecekkan selanjutnya yaitu mengidentifikasi apabila kata terletak di awal kalimat. Apabila benar, maka dilakukan pengecekkan apabila tipe konjungsi adalah ”awalan” atau ”campuran”. Apabila benar, variabel konjungsiCount dan trueKonjungsi akan ditambahkan dan kata ditampilkan dengan garis bawah berwarna hijau. Hal ini dilakukan karena kata konjungsi memiliki tata penulisan yang sesuai dengan kaidah yang diterapkan. Apabila salah, maka variabel konjungsiCount dan falseKonjungsi ditambahkan dan kata tersebut ditampilkan dengan garis bawah berwarna merah. Apabila kata tidak terletak di awal kalimat, maka dilakukan pengecekkan yaitu mengidentifikasi apabila tipe konjungsi bukan merupakan ”awalan”. Apabila benar, variabel konjungsiCount dan trueKonjungsi akan ditambahkan dan kata ditampilkan dengan garis bawah berwarna hijau. Apabila salah, maka variabel konjungsiCount dan falseKonjungsi ditambahkan dan kata tersebut ditampilkan dengan garis bawah berwarna merah. Proses berakhir apabila iterasi kata telah berakhir.

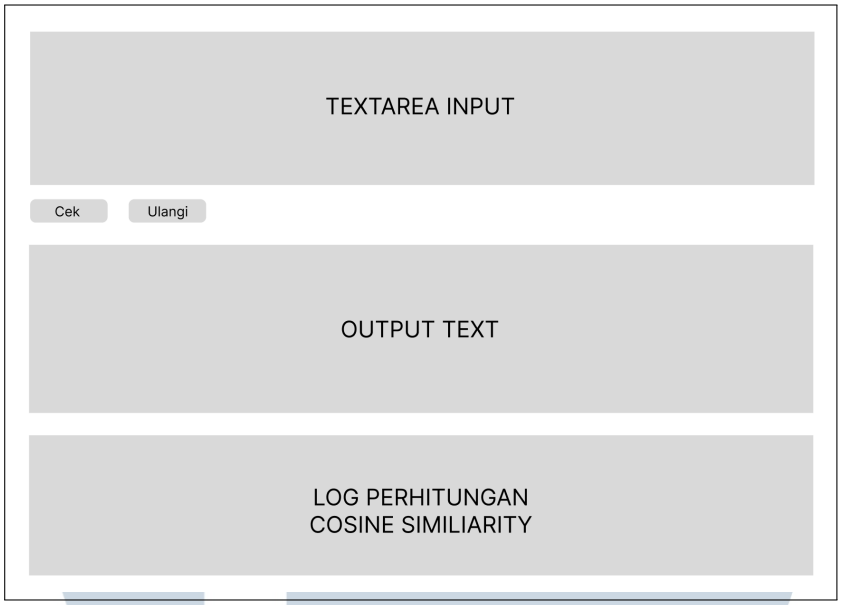
3.3 Perancangan Tampilan Antarmuka

Berikut merupakan perancangan tampilan antarmuka yang digunakan sebagai media untuk memudahkan pengguna dalam menggunakan deteksi konjungsi yang dibangun.



Gambar 3.7. Prototipe Deteksi Konjungsi Sebelum Pengecekkan

Gambar 3.7 merupakan prototipe tampilan antarmuka awal deteksi konjungsi. Pada prototipe tersebut terdapat *textarea input* yang sebagai input yang dapat digunakan pengguna untuk memasukkan teks ataupun artikel. Terdapat juga tombol ”Cek” dan ”Ulangi” pada prototipe tersebut. Tombol ”Cek” digunakan pengguna untuk memulai proses deteksi konjungsi dengan input yang dimasukkan ke *textarea input*. Sedangkan tombol ”Ulangi” digunakan untuk mengulang tahapan deteksi dengan memuat ulang tampilan.



Gambar 3.8. Prototipe Deteksi Konjungsi Setelah Pengecekkan

Gambar 3.8 merupakan prototipe tampilan antarmuka setelah dilakukan proses deteksi konjungsi. Pada prototipe tersebut terdapat kolom *output text* yang digunakan untuk menampilkan hasil deteksi dari teks yang dimasukkan ke *textarea input*. Selain itu, terdapat *log* perhitungan *cosine similarity* yang menampilkan hasil perhitungan *cosine similarity* untuk setiap kata.