BAB III

METODOLOGI DAN PERANCANGAN SISTEM

3.1 Metodologi Penelitian

Dalam penelitian ini, metodologi penelitian yang dilakukan adalah sebagai berikut.

1. Studi Literatur

   Dalam proses ini akan dipelajari dasar teori, perancangan, dan cara implementasi aplikasi *text mining*, seperti tahap membuat aplikasi dengan *framework* PHP CodeIgniter, *preprocessing* data, implementasi algoritma Nazief Adriani untuk proses *stemming*, Manhattan Distance, dan K-Medoids *clustering*.

2. Perancangan Sistem

   Tahap ini akan dilakukan perancangan untuk *user interface* aplikasi yang terbagi menjadi 2 bagian, yaitu *front-end* dan *back-end*. Pada bagian *back-end*, dirancang sebuah halaman yang akan digunakan *admin* aplikasi untuk melakukan *input* data skripsi mahasiswa, menambah atau menghapus kumpulan data *stopword list*, memberikan infomasi tambahan pada kata baru untuk dipelajari sistem, menambah jurusan dan tahun *input* baru, dan megolah data skripsi mahasiswa yang terdapat pada database. Berbeda dengan *back-end*, bagian *front-end* akan digunakan untuk proses analisa dan pencarian topik skripsi. Perancangan ini dibuat sedemikian rupa dengan mengususng konsep *user-friendly*, seperti tampilan yang sederhana dan

proses analisis yang terbagi menjadi beberapa langkah dahulu sebelum dieksekusi.

3. Pemrograman Sistem

Pemrograman dilakukan dengan NetBeans IDE versi 8.0.2 dengan bahasa pemograman PHP dan menggunakan *framework* CodeIgniter. Digunakan *software* ekstraktor Xpdf versi 3.04 (www.foolabs.com) untuk memecah file PDF menjadi *plain text*.

4. Pengujian dan Survei

Proses pengujian aplikasi akan dilakukan menggunakan *notebook* Asus A46CA dengan sistem operasi Windows 10. Pengujian dibagi menjadi 2 jenis, yaitu pengujian terhadap kualitas dari *cluster* dan pengujian terhadap aplikasi yang telah dibangun. Pengujian *cluster* akan menggunakan data objek yang terdapat dalam masing-masing *cluster* hasil dari proses *clustering*, sedangkan pengujian aplikasi akan dilakukan dengan metode survei. Survei ini akan dilakukan pada mahasiswa Fakultas Teknologi Informasi dan Komunikasi Universitas Multimedia Nusantara untuk angkatan 2013, 2014, dan 2015 sebanyak 30 orang. Menurut Gay dan Diehl (1992) jumlah sampel minimum dalam melakukan sebuah penelitian adalah sebanyak 30 sampel. Setelah koresponden mencoba aplikasi, mereka akan diberikan kuesioner untuk diisi sebagai bahan evaluasi penelitian.

5. Evaluasi

Tahap ini akan dilakukan evaluasi terhadap data yang telah dikumpulkan. Evaluasi untuk menguji kualitas *cluster* akan menggunakan metode *purity*. Penggunaan metode *purity* menghasilkan sebuah nilai yang digunakan

untuk mengetahui apakah suatu dokumen dalam *cluster* sudah sesuai atau benar pengelompokannya. Evaluasi terhadap aplikasi yang dibangun akan menggunakan metode Likert Scale berdasarkan data dari kuesioner yang telah diberikan kepada responden. Metode Likert Scale akan menghasilkan sebuah presentase keberhasilan aplikasi dalam membantu mahasiswa mencari rekomendasi dari tren topik skripsi.

3.2 Tahapan Preprocessing Data

Ada pendekatan yang berbeda antara terori dan implementasi dalam melakukan tahap awal dari sebuah proses *text mining* atau dikenal dengan istilah tahap *preprocessing* data. Nantinya implementasi akan dilalukan dengan bahasa pemograman *web*, yaitu PHP *Hypertext Prepocessor* (PHP). Pada subbab ini akan dijelaskan beberapa fungsi atau teknik yang akan digunakan dalam melakukan *preprocessing* data, yaitu tahap *case folding*, *tokenizing, filtering*, dan *stemming*.

1. Case Folding dan Tokenizing

   Pada tahap ini, proses akan mendapatkan sebuah *plain text* hasil ekstraksi dari file PDF. Data tersebut akan melalui tahap *case folding*, yaitu membuang delimeter-delimeter karakter *string* dan membuat semua kata menjadi huruf kecil. Pendekatan yang dapat dilakukan adalah menggukana fungsi *preg_replace*() untuk menghapus delimeter, *str_replace*() untuk mengganti kata, dan *strlower*() untuk membuat *text* menjadi huruf kecil. Selanjutnya adalah tahap *tokenizing*. Tahap ini digunakan untuk memecah *plain text* ke dalam array. Pendekatan yang dapat dilakukan untuk tahap ini adalah penggunaan fungsi *explode*() untuk memecah *text* menjadi array dan

fungsi *array_count_values*() untuk menghitung frekeuensi kemunculan dan pengelompokan kata agar tidak terjadi perulangan kemunculan kata.

2. Filtering

Tahap ini merupakan proses untuk menghapus kata-kata yang tidak penting berdasarkan dari kumpulan *stopword list*. Pendekatan pada tahap ini terbilang mudah karena hanya membandingkan antara kata hasil proses *tokenizing* dan kumpulan kata dari *stopword list*. Jika kata yang dibandingkan terdapat dalam *stopword list,* maka kata tersebut dibuang. Sebaliknya jika tidak ditemukan, maka kata tersebut akan masuk ke tahap selanjutnya. Hal terpenting dalam tahap ini adalah mengetahui kualitas dari kata-kata yang termasuk dalam *stopword list*. Penelitian ini menggunakan kumpulan kata *stopword* dari beberapa sumber penelitian dan *web* terpercaya yang menyediakan *stopword list*, seperti *stopword* bahasa Indonesia (Tala, 2003) dan *www.ranks.nl/stopwords/* untuk *stopword* bahasa Inggris.

3. Stemming

Setelah melewati proses *filtering,* selanjutnya akan masuk ke dalam tahap stemming. Kata yang berhasil lolos ke tahap ini akan dicarikan kata dasarnya. Tahap ini akan menggunakan metode Nazief Adriani, dipilih karena memiliki performa yang baik untuk melakukan *stemming* teks berbahasa Indonesia. Metode ini berfungsi menghilangkan awalan dan akhiran imbuhan kata untuk dicarikan kata dasarnya dalam kamus. Pendekatannya dapat menggunakan *if-else statement* untuk memilah berbagai kondisi kata sesuai dengan kaidah dasar bahasa Indoensia yang

baik dan benar. Selain itu, penggunaan fungsi *preg_match*() untuk mencocokan kata, fungsi *preg_replace*() untuk menghapus imbuhan yang terdapat pada kata, dan *substr*() untuk menukar penggalan kata lama dengan yang baru.

3.3 Perancangan *Sistem*

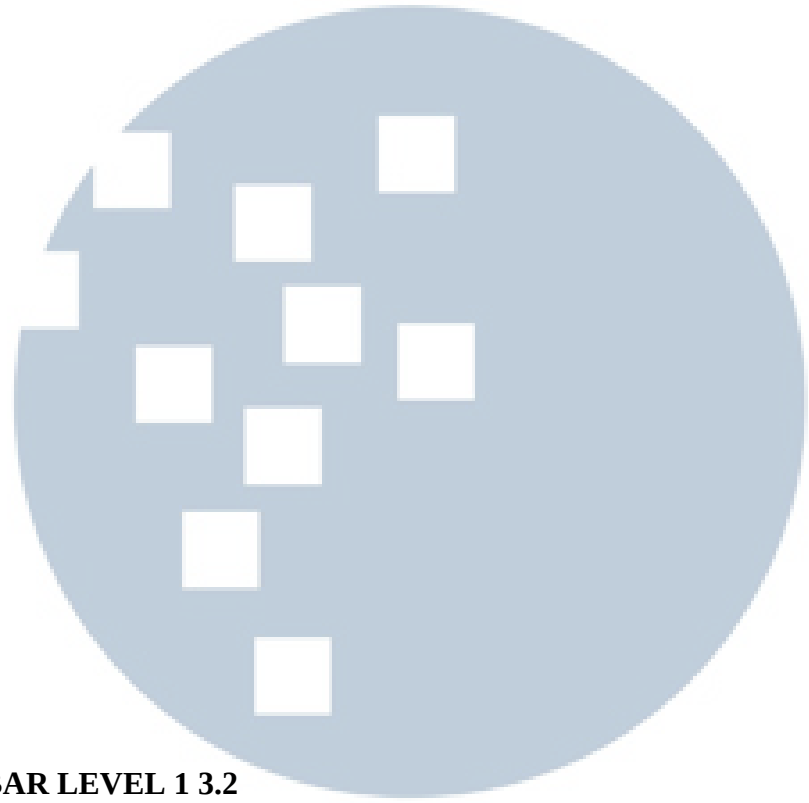
Tahap ini akan dijelaskan apa saja yang menjadi bagian dari perancangan sistem, seperti *Data Flow Diagram*, *Site Map Diagram*, *flowchart*, *Entitiy Relation Diagram*, dan struktur tabel.

3.3.1 Data Flow Diagram

Alur data selama proses sistem aplikasi akan dijabarkan melalui *Data Flow Diagram* (DFD). Gambar 3.1 merupakan *context diagram* yang menunjukkan secara garis besar *flow* data yang terjadi pada sistem. *Context diagram* bisa diakatakan sebagai level 0 dari DFD yang dirancang. Terdapat satu proses besar *Text Mining System* dan 6 entitas, yaitu *website ranks*, Tala (peneliti), Bahtera (kamus), mahasiswa, *user*, dan *admin*. Entitas *user* mewakili aktivitas yang terjadi pada bagian *front-end*, sedangkan entitas *website ranks*, Tala (peneliti), Bahtera (kamus), mahasiswa, dan *admin* mewakili aktivitas yang terjadi pada bagian *back-end*. Entitas *admin* juga mewakili aktivitas pada proses *main menu*.

GAMABAR CONTEXT DIAGRAM 3.1

Kemudian dilanjutkan dengan level 1 yang terdapat 4 proses. Ada 3 proses yang menjadi milik entitas *admin*, yaitu *login, back-end*, dan *main menu*. Entitas *website ranks*, Tala (peneliti), Bahtera (kamus), dan mahasiswa hanya memiliki satu proses saja, yaitu *back-end*. Entitas *user* juga hanya memiliki satu proses saja, yaitu *front-end*. Proses *login* berfungsi sebagai proses autentikasi pengguna (*admin*) untuk masuk ke dalam halaman *back-end*. Proses *back-end* adalah proses yang berjalan pada bagian belakang aplikasi dan dikelola oleh seorang *admin*, berfungsi untuk untuk mengolah data mentah yang di-*input* menjadi data siap pakai pada bagian *front-end*. Proses *main_menu* berfungsi untuk memanipulasi informasi penting yang akan ditampilkan pada *menu header front-end*. Proses *front-end* adalah proses yang bekerja pada halaman depan tempat pengguna menggunakan aplikasi, berfungsi untuk melakukan analisis *clustering* dengan metode K-Medoids dan pencarian data skripsi. Terdapat 13 tabel yang digunakan dalam proses *flow* data aplikasi *text mining*. Tabel yang digunakan antara lain tabel *admin*, data, *filtering, final_word*, hasil_analisis, jurusan, kata_dasar, *main_menu, stemming*, *stopword*, tabel_medoids, tahun, dan *tokenizing*. Berikut dijabarkan tabel apa saja yang digunakan setiap proses. Proses *login* menggunakan satu tabel, yaitu tabel *admin*. Proses *back-end* menggunakan sembilan tabel, yaitu tabel data, *tokenizing, filtering, stemming, final_word*, kata_dasar, *stopword*, tahun, dan jurusan. Proses *main_menu* menggunakan satu tabel, yaitu tabel *main_menu*. Proses *front-end* menggunakan dua tabel, yaitu tabel tabel_medoids dan hasil_analisis. Semua hasil di atas digambarkan dalam DFD level 1 berikut ini.

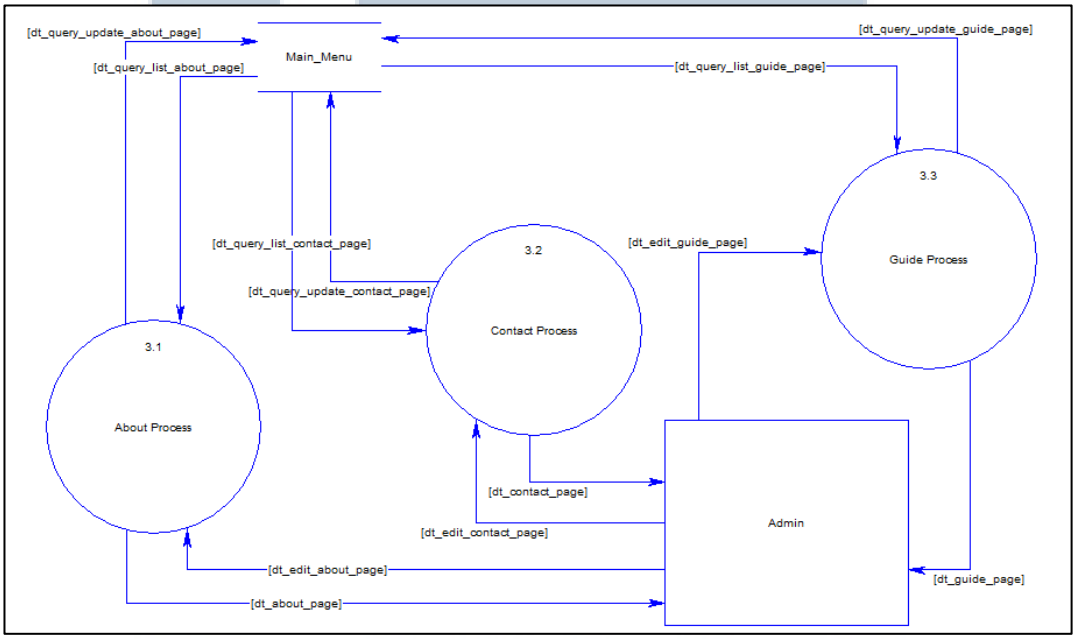


GAMBAR LEVEL 1 3.2

GAMBAR LEVEL 2 BACK END 3.3

Pada level 2 terdapat 11 subproses dari 3 proses pada level 1. Gambar 3.3 menunjukkan bahwa proses *back-end* memiliki 6 subproses, yaitu subproses *input data*, *stopword*, kata baru, tambah tahun, tambah jurusan, dan *all data*. Subproses *input data* digunakan untuk proses memasukan data baru yang berasal dari *file* abstrak milik mahasiswa. Selain menyimpan data mahasiswa, subproses ini juga melakukan proses mengolah *file* data abstrak mahasiswa dengan metode yang dikenal dengan sebutan *preprocessing* data. Subproses ini menggunakan tabel data untuk menampung semua *list* data skripsi mahasiswa, tabel *tokenizing* untuk menampung dan mengambil data yang telah memalui tahap *case folding* dan *tokenizing*, tabel *filtering* untuk menampung dan mengambil data yang telah melalui tahap *filtering*, tabel *stemming* untuk menampung dan mengambil data yang telah malalui tahap *stemming*, tabel *final_word* digunakan untuk menampung hasil kata yang telah melewati tahap *filtering* kedua, tabel kata_dasar berisikan kata dasar yang didapatkan dari Bahtera (kamus) untuk melakukan pengecekan dengan cara membandingakan kata dasar apakah kata hasil stemming terdapat di dalam kamus bahasa Indonesia atau tidak , tabel *stopword* digunakan untuk membuang kata tidak penting selama proses *filtering* dengan data yang berasal dari *website rank*, Tala (peneliti), dan *admin*, tabel tahun untuk memberikan *list* tahun terbit yang terdapat di dalam database sebagai salah satu syarat untuk *input* data baru, dan tabel jurusan untuk memberikan *list* jurusan yang terdapat di dalam database yang juga sebagai salah satu syarat untuk *input* data baru. Subproses *stopword* digunakan untuk mengatur kata-kata yang termasuk ke dalam *stopword list*, seperti menambahkan *stopword* baru dan menghapus *stopword*. Subproses ini menggunakan tabel *stopword* untuk menambah, menghapus, dan menapilkan *stopword list*. Subproses

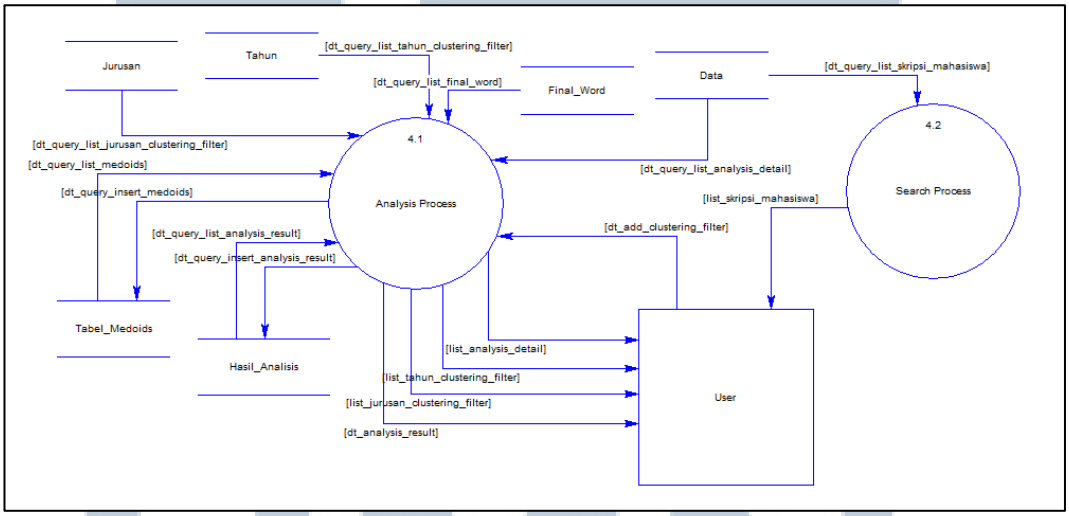
kata baru digunakan untuk menampilkan dan melakukan penambahan keterangan baru pada suatu kata, menggunakan tabel *final_word* untuk menampilkan dan *update* keterangan baru kata. Subproses tambah tahun digunakan untuk menambah tahun terbit skripsi yang baru, menggunakan tabel tahun untuk menambah tahun terbit yang baru. Subproses tambah jurusan digunakan untuk menambah jurusan mahasiswa yang baru, menggunakan tabel jurusan untuk menambah jurusan mahasiswa yang baru. Subproses *all data* digunakan untuk menampilkan dan menghapus *list* data skripsi mahasiswa dari database, menggunakan tabel data untuk menghapus dan menampilkan semua *list* data skripsi mahasiswa.



Gambar 3.4 DFD Level 2 Subproses *Main Menu*

Gambar 3.4 adalah DFD level 2 subproses *main menu*. Proses *main menu* memiliki 3 subproses, yaitu subproses *about*, *contact*, dan *guide*. Subproses *main menu* digunakan untuk mengatur informasi apa saja yang akan disampaikan atau ditampilkan pada pilihan halaman *menu header* yang terdapat pada bagian *front-end*. Subproses *about* digunakan untuk mengatur informasi yang terdapat pada

halaman *about*, menggunakan tabel *main_menu* untuk menampilkan dan mengubah informasi pada halaman *about*. Subproses *contact* digunakan untuk mengatur informasi yang terdapat pada halaman *contact*, menggunakan tabel *main_menu* untuk menampilkan dan mengubah informasi pada halaman *contact*. Subproses *guide* digunakan untuk mengatur informasi yang terdapat pada halaman *guide*, menggunakan tabel *main_menu* untuk menampilkan dan mengubah informasi pada halaman *guide*.



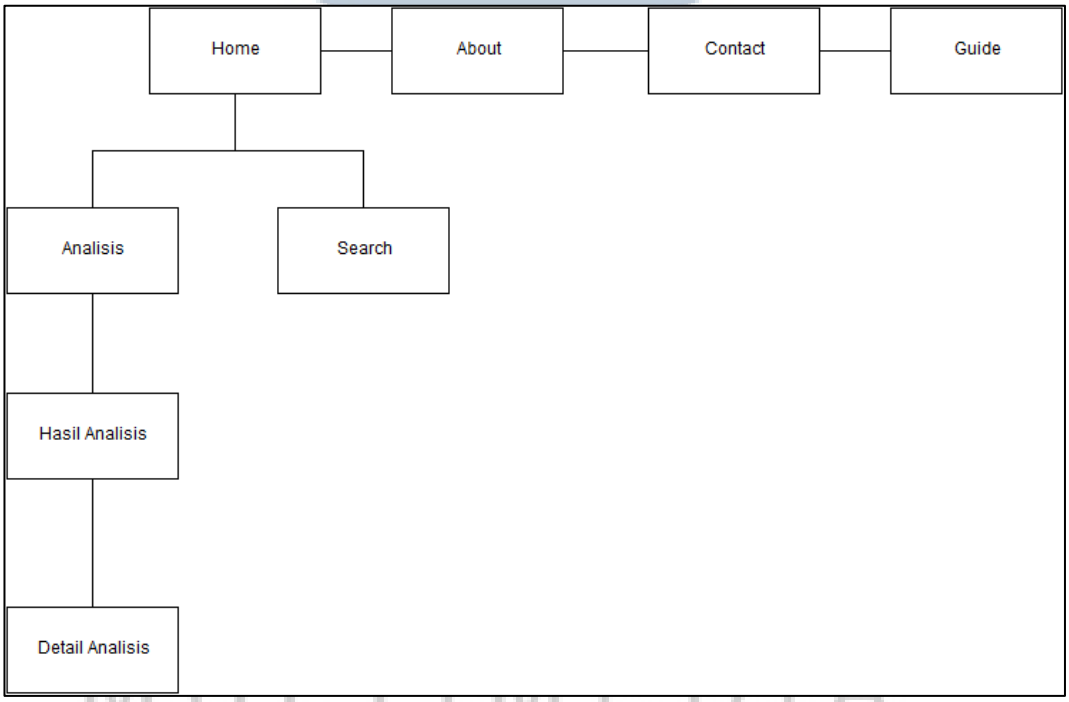
Gambar 3.5 DFD Level 2 Subproses *Front-End*

Gambar 3.5 adalah DFD level 2 subproses *front-end*. Proses *front-end* memiliki 2 subproses, yaitu subproses analisis dan *search*. Subproses analisis digunakan untuk proses analisis *clustering* mulai dari *input* filter *cluster* hingga menampilkan hasil analisis, menggunakan tabel *final_word* untuk mengambil kata yang akan di proses selama *clustering*, tabel tabel_medoids untuk membuat *array* frekuensi sebagai salah satu syarat proses metode K-Medoids dan proses penentuan pengelompokan kata ke dalam *cluster*, tabel hasil_analisis digunakan untuk menampung dan menampilkan data hasil analisis, tabel data digunakan untuk

menampilkan data skripsi mahasiswa sesuai dengan hasil analisis pada halaman detail analisis, tabel tahun digunakan untuk menampilkan *list* tahun terbit skripsi sebagai salah satu proses *filter clustering*, dan tabel jurusan digunakan untuk menampilkan *list* jurusan mahasiswa sebagai salah satu proses *filter clustering*.

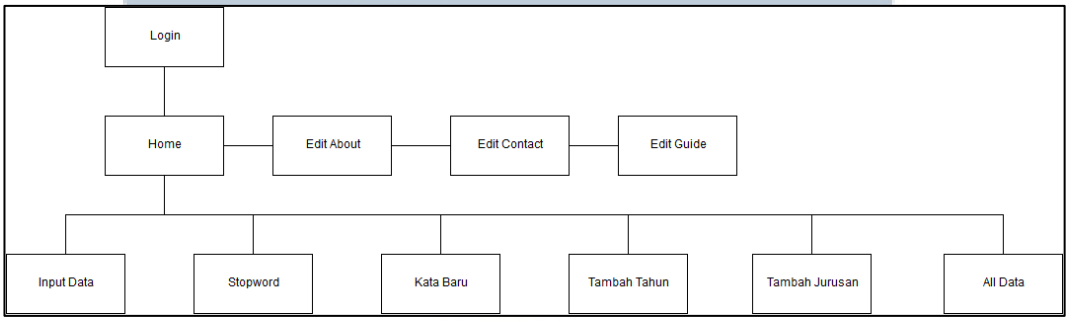
3.3.2 Site Map

Pembangunan aplikasi dilakukan dengan basis *web* sehingga untuk lebih mudah mengetahui susunan halaman apa saja yang terdapat di dalam aplikasi, maka dibuatkan *site map*. Diagram *site map* untuk aplikasi *text mining* untuk penentuan tren topik skripsi dibagi menjadi 2 bagian, yaitu bagian *front-end* dan bagian *back-end*.



Gambar 3.6 *Site Map Front-End*

Pada *site map* bagian *front-end* memiliki halaman utama, yaitu halaman *home*, *about*, *contact*, dan *guide*. Dari halaman *home* dapat berpindah ke halaman analisis dan *search*. Halaman analisis berfungsi sebagai halaman yang digunakan untuk proses analisis *clustering* K-Medoids, sedangkan halaman *search* digunakan untuk melakukan pencarian data skripsi mahasiswa. Setelah proses analisis selesai, maka pengguna aplikasi akan berpindah ke halaman hasil analisis dan jika ingin melihat detail dari hasil analisis, maka dapat berpindah ke halaman detail analisis. Halaman *about*, *contact*, dan *guide* tidak memiliki sub halaman.



Gambar 3.7 *Site Map Back-End*

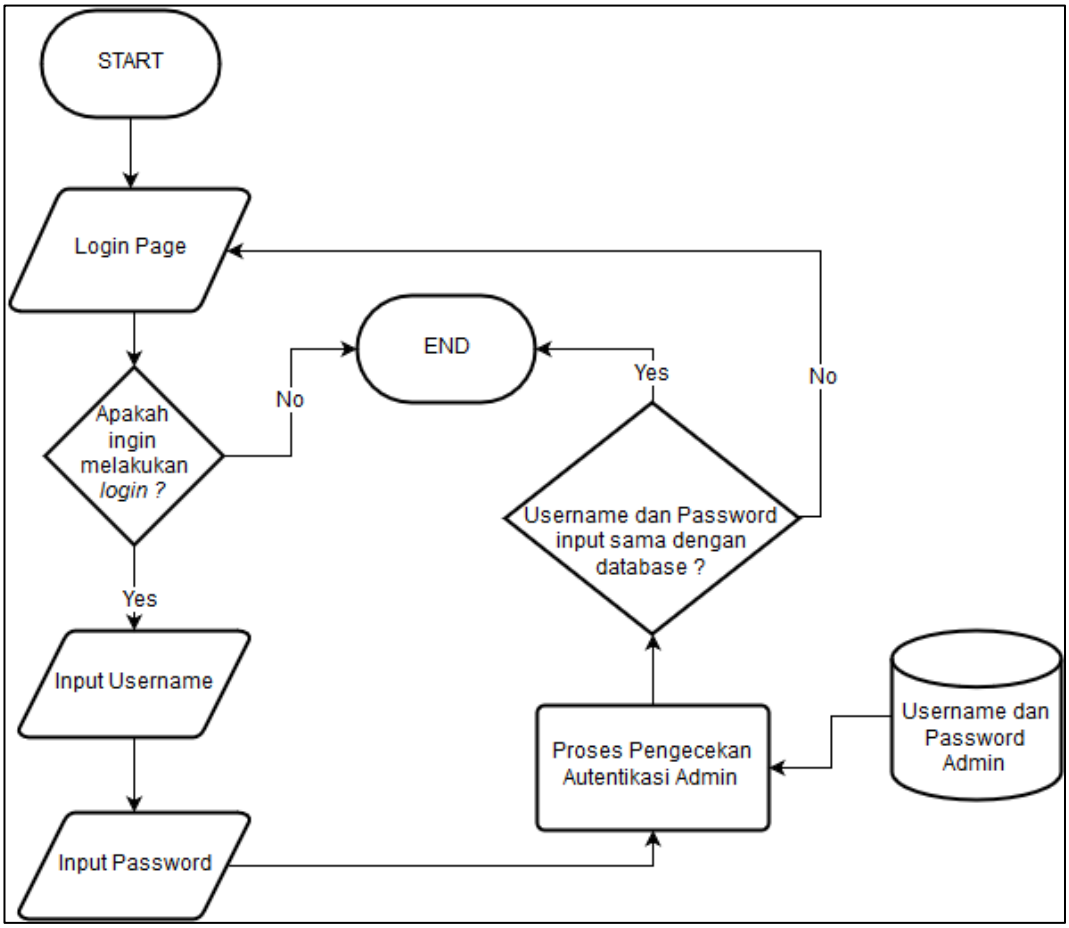
Pada *site map back-end* dimulai dengan halaman *login*. Setelah proses *login* berhasil, maka pengguna akan dialihkan ke halaman utama *back-end*. Bagian ini memiliki 4 halaman utama, yaitu halaman *home*, *edit about*, edit *contact*, dan *guide*. Dari halaman *home* dapat berpindah ke beberapa sub halaman lain, yaitu *input data*, *stopword*, kata baru, tambah tahun, tambah jurusan, dan *all data*. Sub halaman *input data* digunakan sebagai tempat tampilan untuk memasukan data skripsi mahasiswa yang baru. Sub halaman *stopword* digunakan sebagai tempat tampilan untuk menambah dan menghapus kata *stopword*. Sub halaman kata baru digunakan sebagai tempat tampilan untuk menampilkan dan memberikan keterangan baru pada kata-kata yang akan dipelajari oleh sistem. Sub halaman tambah tahun

digunakan sebagai tempat tampilan untuk menambah tahun terbit skripsi yang baru. Sub halaman tambah jurusan digunakan sebagai tempat tampilan untuk menambah jurusan mahasiswa. Sub halaman *all data* digunakan sebagai tempat tampilan untuk melihat detail dan menghapus data skripsi mahasiswa. Halaman *edit about, edit contact,* dan *edit guide* tidak memiliki sub halaman.

3.3.3 Flowchart

Tahap awal yang dilakukan sebelum melakukan pemograman sistem, yaitu membuat *flowchart* sebagai dasar alur yang akan dibangun pada aplikasi *text mining* secara keseluruhan. Pada tahap ini akan dijabarkan proses masing-masing yang terjadi pada aplikasi *text* mining untuk penentuan tren topik skripsi, seperti proses *login, input data*, kata baru, tambah tahun, tambah jurusan, *all data*, *edit about*, *edit contact, edit guide*, analisis, dan *search*.

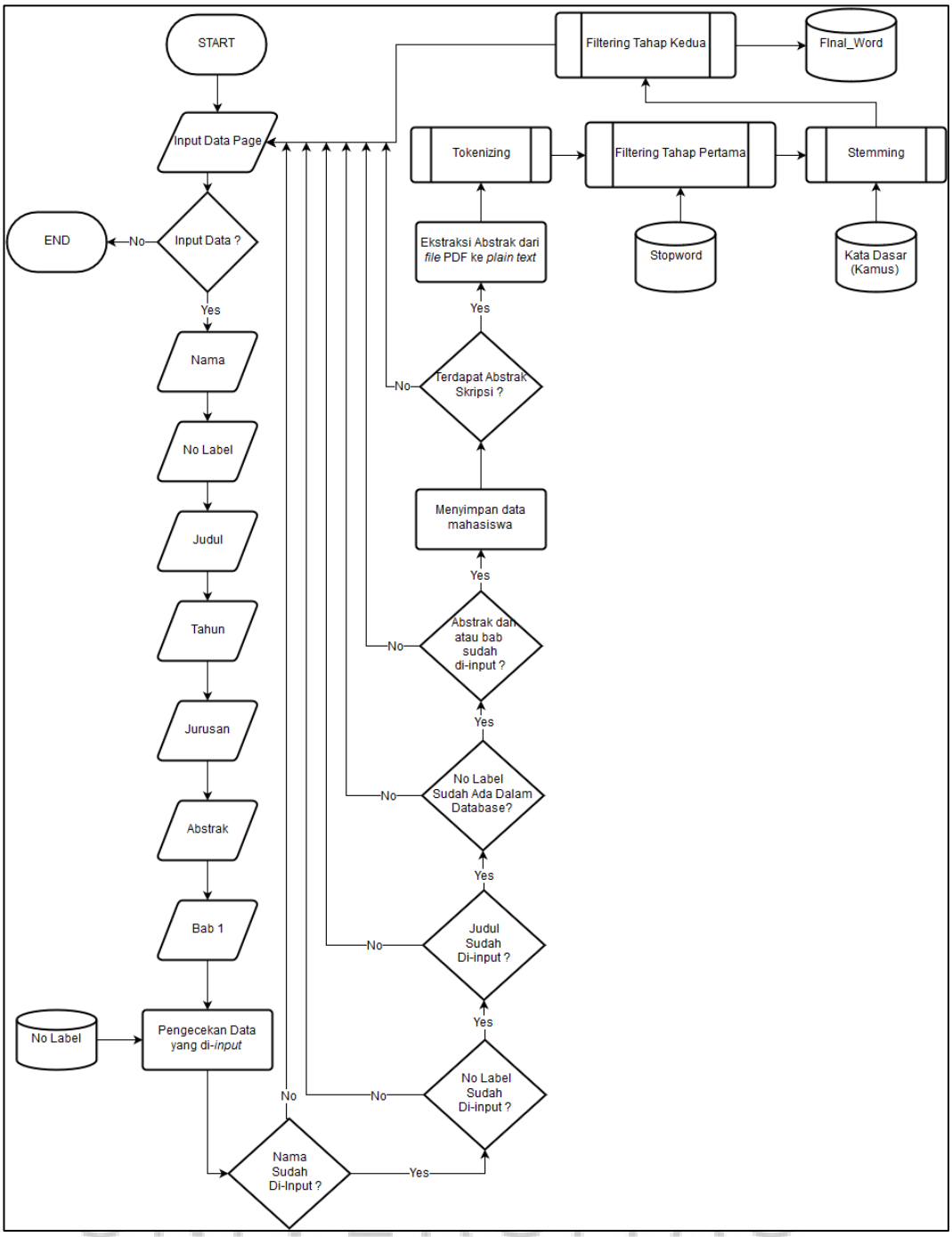
Gambar 3.8 adalah *flowchart* proses *login*. Proses diawali dengan menampilkan *display* halaman *login*. Selanjutnya, apakah pengguna ingin melakukan *login*, jika benar pengguna diminta melakukan input *username* dan *password*, sedangakan jika tidak, maka proses selesai. Selanjutnya jika pengguna ingin melakukan *login*, dilakukan proses pengecekan autentikasi dengan mencocokan antara data *input* dan data yang ada pada database *admin*. Apabila ditemukan kesamaan, maka proses selesai dan jika tidak sama, maka akan kembali ke *display* halaman *login*.



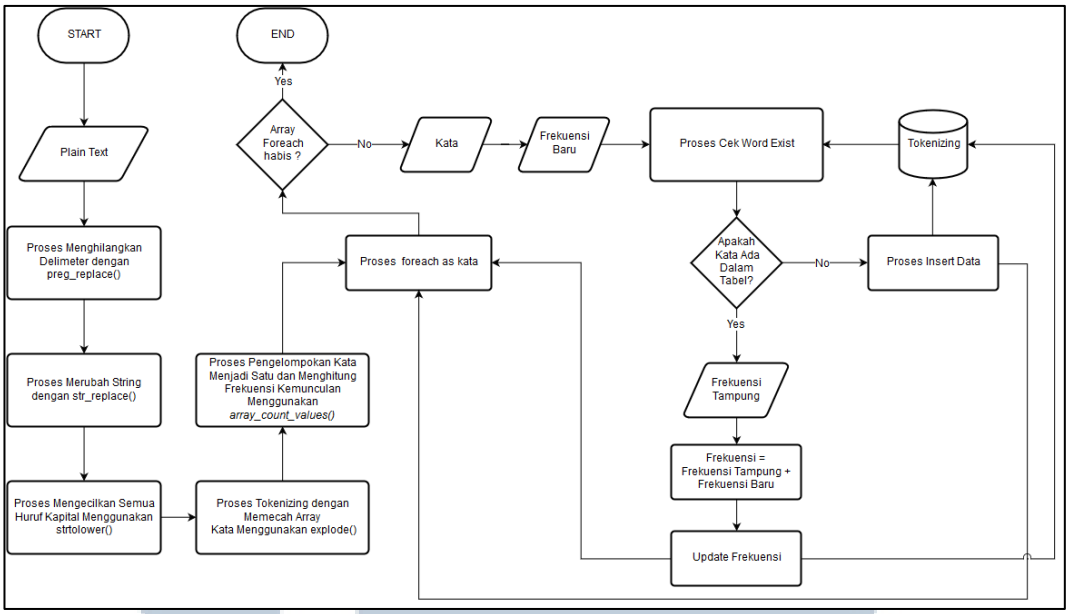
Gambar 3.8 *Flowchart* Proses *Login*

Gambar 3.9 adalah *flowchart* untuk proses *input data*. Proses ini dimulai dengan menampilkan *display* halaman *input data*. Kemudian apakah pengguna ingin melakukan *input data*, jika benar maka pengguna akan diminta untuk memberikan *input data*, seperti nama mahasiswa, no label, judul, tahun terbit, jurusan, abstrak, dan bab1, sedangkan jika tidak maka proses *input data* berakhir. Selanjutnya jika pengguna telah melakukan *input data*, maka akan menjalani proses pengecekan hasil *input*. Apakah nama mahasiswa sudah di-*input*. Jika belum akan dikembalikan ke *display input data*, sedangkan jika sudah maka lanjut ke langkah selanjutnya. Apakah no label mahasiswa sudah di-*input*. Jika belum akan dikembalikan ke *display input data*, sedangkan jika sudah lanjut ke langkah

selanjutnya. Apakah judul skripsi mahasiswa sudah di-*input*. Jika belum akan dikembalikan ke *display input* data, sedangkan jika sudah akan lanjut ke langkah selanjutnya. Apakah no label yang dimasukkan sudah terdapat di dalam database. Hal ini dilakukan dengan mencocokan *input* no label dari pengguna dengan data no label yang ada di dalam database. Jika sudah akan kembali ke *display input data*, sedangakan jika belum akan lanjut ke tahap selanjutnya, yaitu dilakukan pengecekan apakah pengguna sudah input *file* abstrak dan atau bab 1, jika belum akan kembali ke *display input data* dan jika sudah akan dilakukan proses penyimpanan data mahasiswa. Langkah proses selanjutnya adalah dilakukan pengecekan apakah ada *input* abstrak. Jika tidak ada, maka proses penyimpanan hanya sampai di sini dan kembali ke halaman *display input data* untuk melakukan *input data* yang lain atau proses berakhir. Jika terdapat *file* abstrak, maka akan dilakukan proses *tokenizing*. Hasil dari proses ini akan dilanjutkan dengan proses *filtering* dan proses ini dibantu dengan *stopword list* untuk membuang kata-kata tidak penting yang berasal dari database. Kemudian setelah tahap ini selesai, akan dilanjutkan dengan proses *stemming* dan proses ini akan dibantu oleh kata dasar yang didapatkan dari database kata_dasar untuk dicocokan dengan hasil keluaran proses *stemming*. Terakhir, dilakukan kembali proses *filtering* dengan *input* kata hasil *stemming*. Setelah semua proses telah dilalaui, hasilnya akan ditampung di dalam database *final_word* dan pengguna akan dikembalikan ke *display input data* untuk melakukan *input data* baru yang lain atau proses berakhir. Proses *tokenizing*, *filtering*, dan *stemming* akan dijabarkan lagi pada *flowchart* berikutnya.

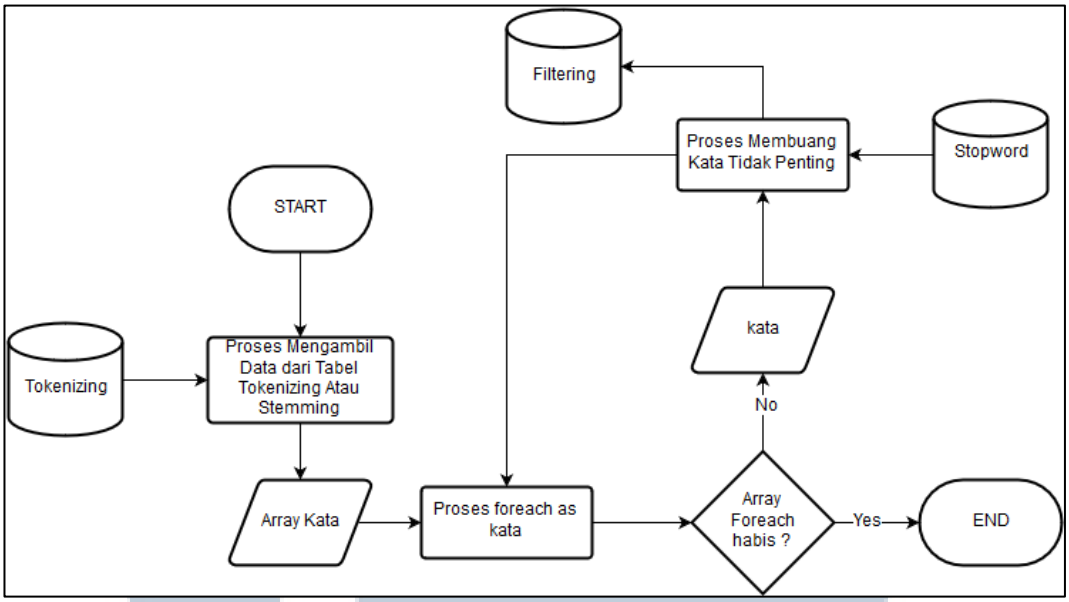


Gambar 3.9 *Flowchart* Proses *Input Data*

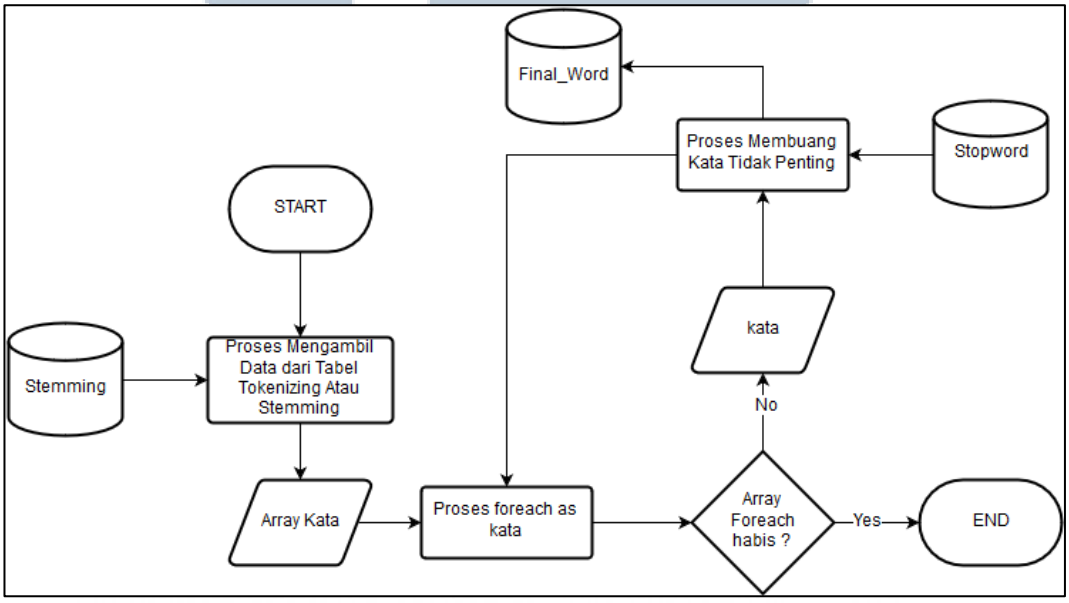


Gambar 3.10 *Flowchart* Proses *Tokenizing*

Gambar 3.10 adalah *flowchart* untuk proses *tokenizing*. Proses ini dimulai dengan *input plain text* hasil dari ekstraksi *file* abstrak. Kemudian akan masuk ke dalam proses *case folding*, yaitu untuk menghilangkan delimeter-delimeter yang terdapat di dalam *plain text*. Setelah selesai, masuk ke proses untuk mengecilkan semua huruf kapital. Selanjutnya masuk ke proses *tokenizing* untuk memecah *plain text* menjadi *array* kata. Terakhir, akan dilakukan proses penglompokan kata yang sama, dihitung frekuensi kemunculannya, hasilnya disimpan di dalam database, dan proses berakhir.



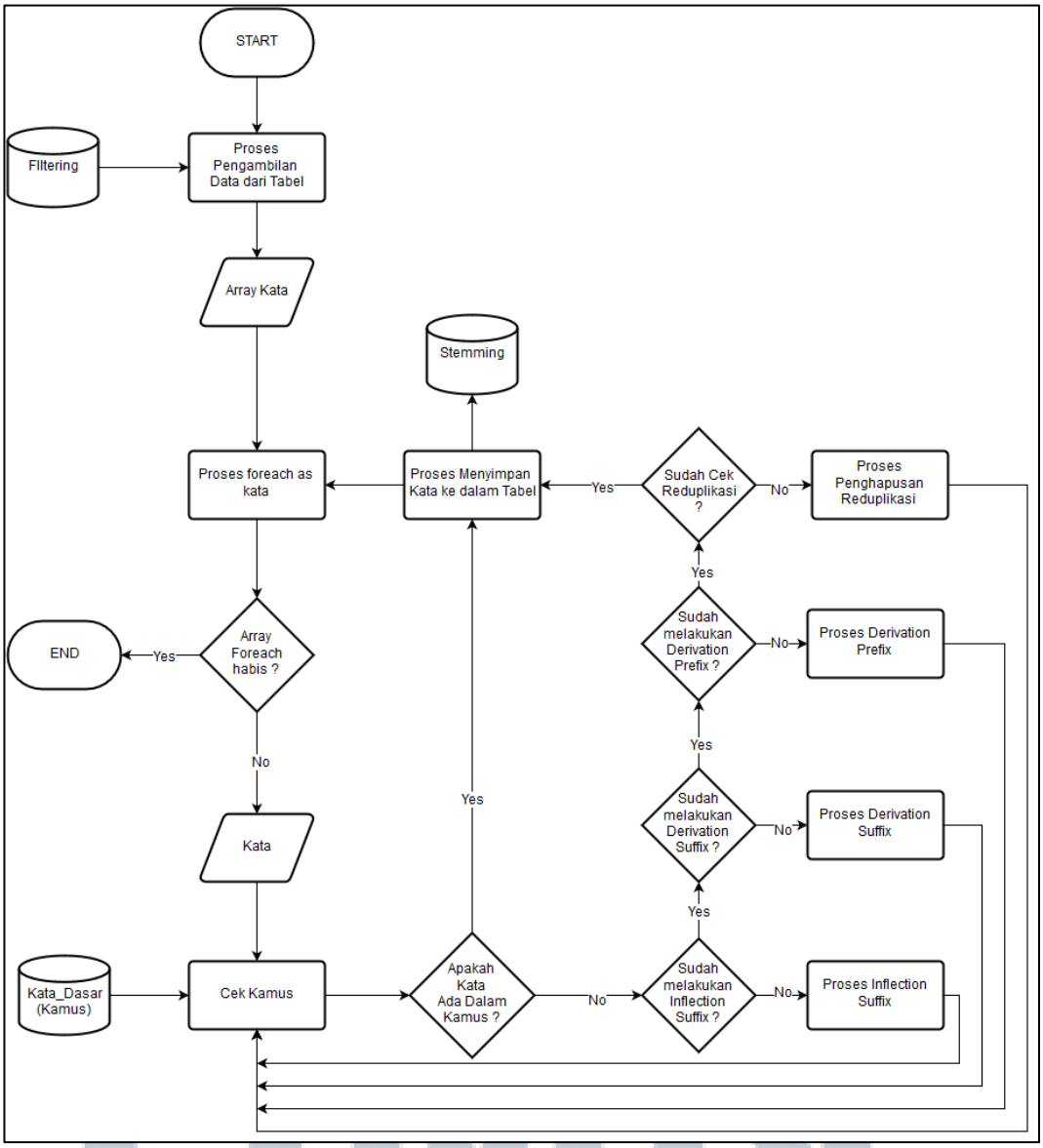
Gambar 3.11 *Flowchart* Proses *Filtering* Tahap Pertama



Gambar 3.12 *Flowchart* Proses *Filtering* Tahap Kedua

Gambar 3.11 adalah *flowchart* untuk proses *filtering* tahap pertama, sedangkan tahap kedua *filtering* dapat dilihat pada Gambar 3.12. Berdasarkan *flowchart* proses *input data*, terdapat dua kali terjadi proses *filtering*. Oleh karena itu, untuk proses pengambilan data dapat berasal dari database dengan tabel

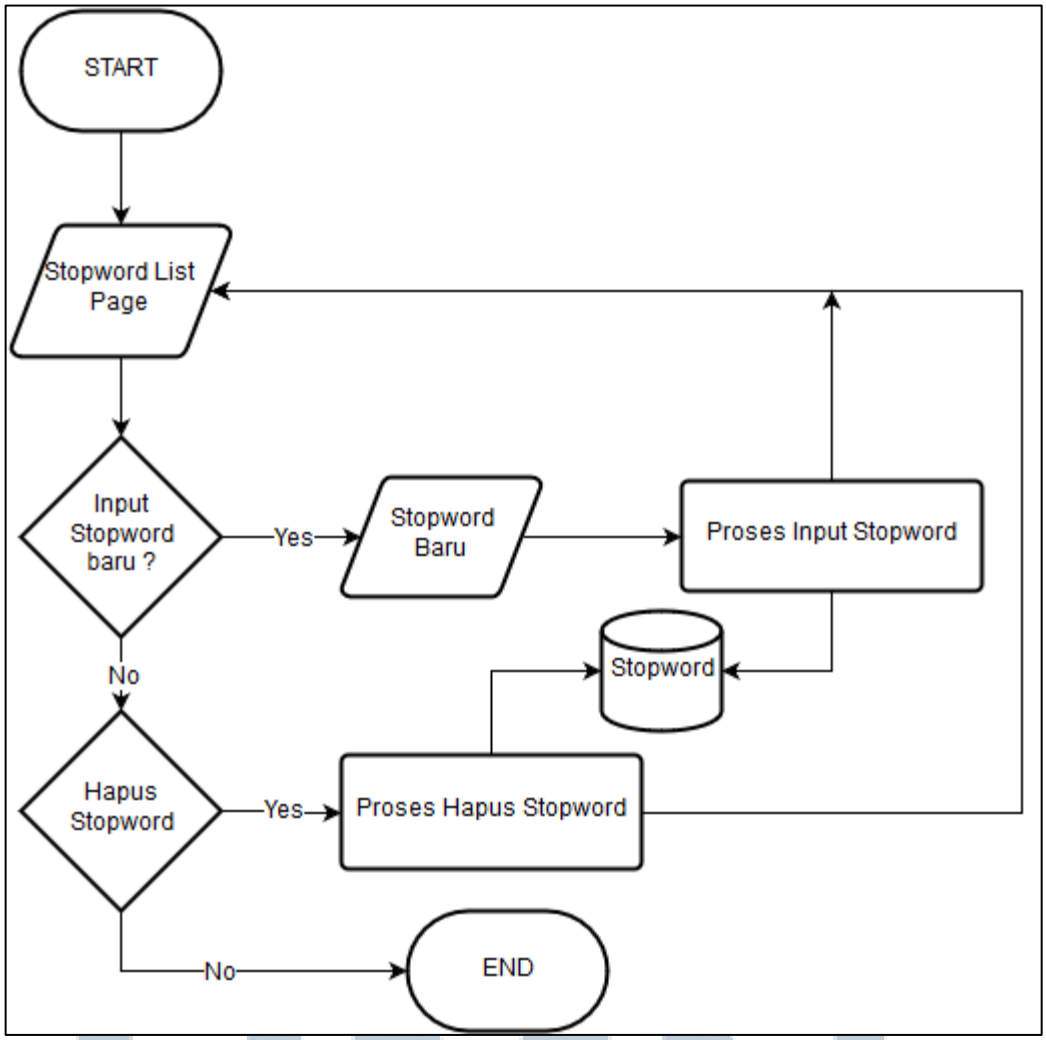
tokenizing atau *stemming*. Tabel *tokenizing* digunakan untuk proses *filtering* yang pertama, sedangkan tabel *stemming* digunakan untuk proses *filtering* yang kedua. Setelah proses pengambilan data, akan mengeluarkan *input* berupa *array* kata. *Array* kata ini akan dijabarkan menjadi per kata menggunakan fungsi *foreach*(). Dicek apakah *array foreach* sudah habis. Jika sudah habis, maka proses berakhir. Jika belum habis, maka *output* berupa kata akan masuk ke proses pembuangan kata tidak penting. Proses pembuangan kata tidak penting dibantu dengan data yang berasal dari database dengan tabel *stopword*. Apabila kata yang di-*input* tidak termasuk *stopword*, maka kata tersebut akan disimpan di dalam database dengan tabel *filtering* atau *final_word*. Tabel *filtering* digunakan untuk proses *filtering* tahap pertama, sedangkan tabel *final_word* digunakan untuk proses *filtering* tahap kedua. Jika ternyata kata tersebut termasuk kata yang tidak penting, maka tidak ada proses penyimpanan dan langsung kembali ke fungsi *foreach*() untuk kata selanjutnya.



Gambar 3.13 *Flowchart* Proses *Stemming*

Gambar 3.13 adalah *flowchart* untuk proses *stemming*. Proses dimulai dengan mengambil data dari database berupa kumpulan kata hasil *filtering*. Hasil *output* berupa *array* kata. Proses dilanjutkan dengan menjabarkan *array* kata menjadi per kata menggunakan fungsi *foreach*(). Dicek apakah *array foreach* sudah habis. Jika sudah habis, maka proses berakhir. Jika belum habis, maka *output* kata akan masuk ke tahap *stemming* dengan algoritma Nazief Adriani. Pertama, apakah

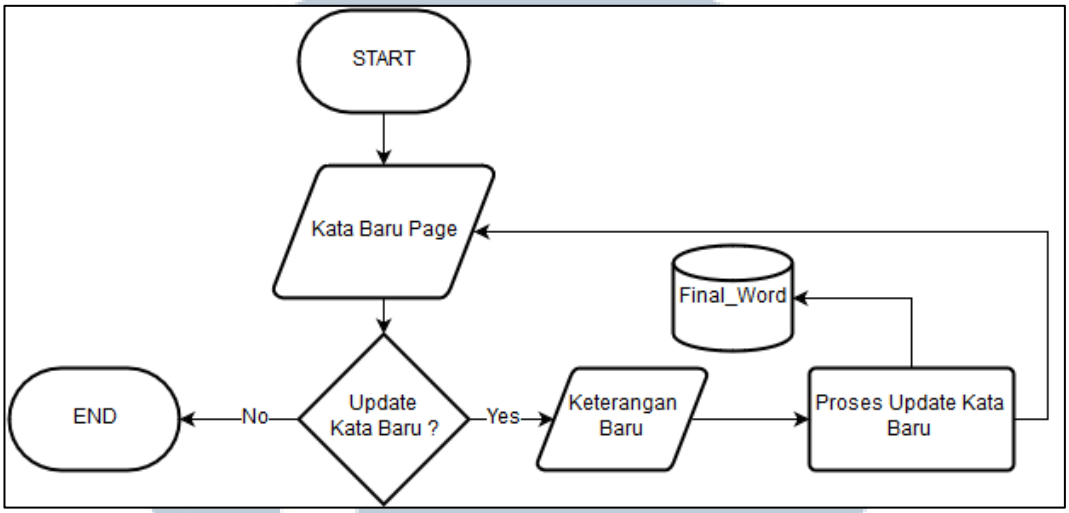
kata yang masuk terdapat di dalam kamus. Jika ditemukan, maka kata akan langsung disimpan di dalam database dan kembali ke fungsi *foreach*(). Jika tidak ditemukan, maka akan dilakukan tahap kedua, yaitu penghapusan imbuhan Inflection Suffix dan hasilnya dicek di dalam kamus kembali. Jika masih belum ditemukan, maka akan masuk ke langkah ketiga, yaitu penghapusan imbuhan Derivation Suffix dan hasilnya dicek di dalam kamus kembali. Jika belum juga ditemukan di dalam kamus, maka dilakukan tahap keempat, yaitu penghapusan imbuhan Derivation Prefix dan hasilnya dicek di dalam kamus kembali. Terakhir, jika belum juga ditemukan, maka akan dilakukan tahap pengecekan reduplikasi dan hasilnya dicek di dalam kamus. Apabila semua tahap telah dicoba, tetapi masih saja belum ditemukan kata dasarnya, maka kata tersebut akan dianggap sebagi *root word* yang baru.



Gambar 3.14 *Flowchart* Proses *Stopword*

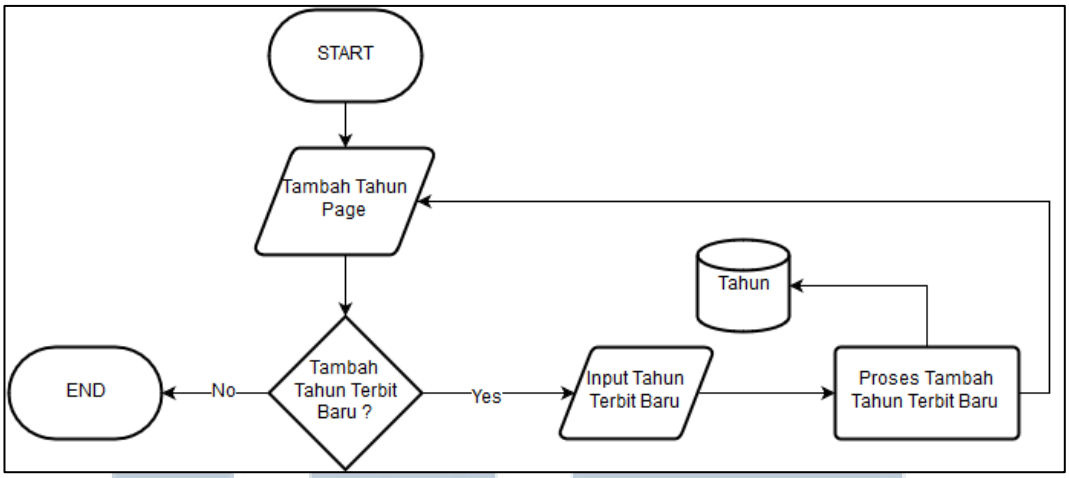
Gambar 3.14 adalah *flowchart* untuk proses *stopword*. Proses dimulai dengan menampilkan *display* halaman *stopword list*. Tahap selanjutnya apakah pengguna akan melakukan *input stopword* baru. Jika benar, maka pengguna diminta untuk melakukan *input* kata *stopword* yang baru, dilanjutkan proses penyimpanan data dan setelah proses selesai akan dikembalikan ke *display stopword list* untuk melakukan tindakan yang lain. Jika tidak melakukan *input stopword*, maka apakah pengguna akan menghapus kata *stopword*. Jika benar, maka akan menjalankan proses menghapus kata *stopword* dan setelah proses selesai akan dikembalikan ke

display stopword list untuk melakukan tindakan lain. Jika tidak melakukan penghapusan *stopword*, maka proses berakhir.



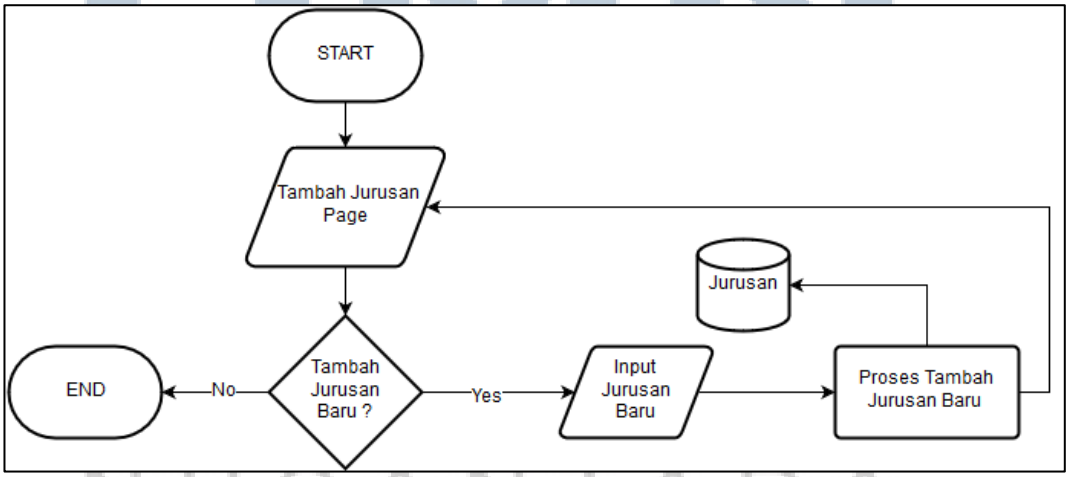
Gambar 3.15 *Flowchart* Proses Kata Baru

Gambar 3.15 adalah *flowchart* untuk proses kata baru. Proses dimulai dengan menampilkan *display* halaman kata baru. Kemudian apakah pengguna akan melakukan *update* terhadap kata baru yang akan dipelajari oleh sistem. Jika benar, maka pengguna akan diminta untuk memberikan *input* keterangan baru, dilanjutkan dengan proses *update* keterangan kata baru, dan setelah proses selesai, pengguna akan dikembalikan ke *display* kata baru untuk melakukan *update* pada kata baru yang lain atau proses berakhir.



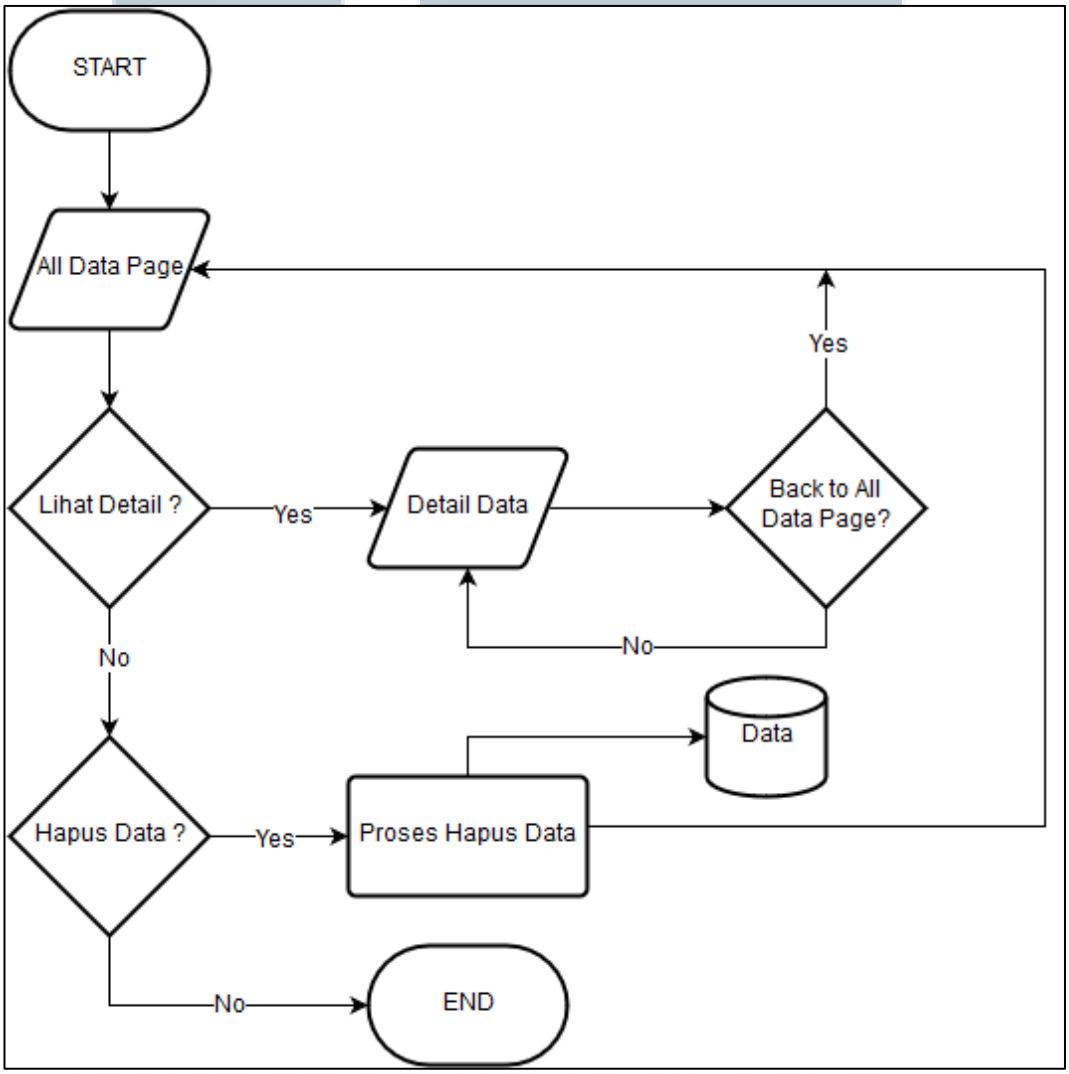
Gambar 3.16 *Flowchart* Proses Tambah Tahun

Gambar 3.16 adalah *flowchart* untuk proses tambah tahun. Proses dimulai dengan menampilkan *display* halaman tambah tahun. Kemudian apakah pengguna akan menambah tahun terbit baru pada sistem. Jika benar, maka pengguna akan diminta untuk memberikan *input* tahun terbit baru, dilanjutkan dengan proses tambah tahun terbit baru, dan setelah proses selesai, pengguna akan dikembalikan ke *display* tambah tahun untuk menambah tahun terbit baru yang lain atau proses berakhir.



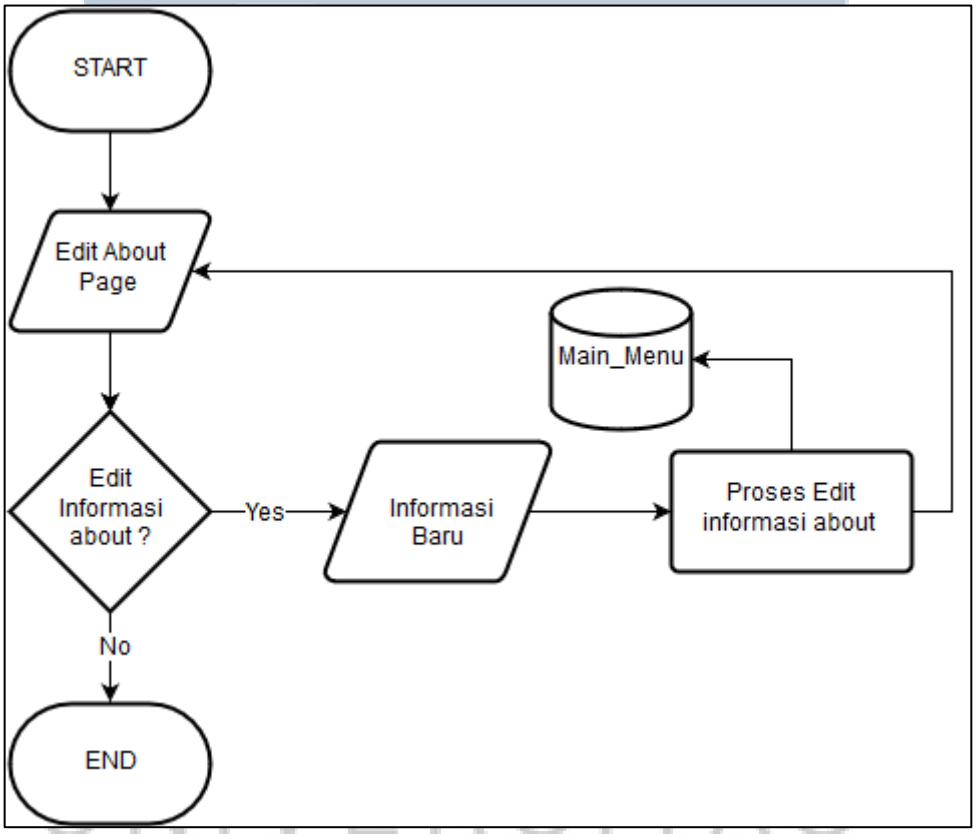
Gambar 3.17 *Flowchart* Proses Tambah Jurusan

Gambar 3.17 adalah *flowchart* untuk proses tambah jurusan. Proses dimulai dengan menampilkan *display* halaman tambah jurusan. Kemudian apakah pengguna akan menambah jurusan baru pada sistem. Jika benar, maka pengguna akan diminta untuk memberikan *input* jurusan baru, dilanjutkan dengan proses tambah jurusan baru, dan setelah proses selesai, pengguna akan dikembalikan ke *display* tambah jurusan untuk menambah jurusan baru yang lain atau proses berakhir.



Gambar 3.18 *Flowchart* Proses *All Data*

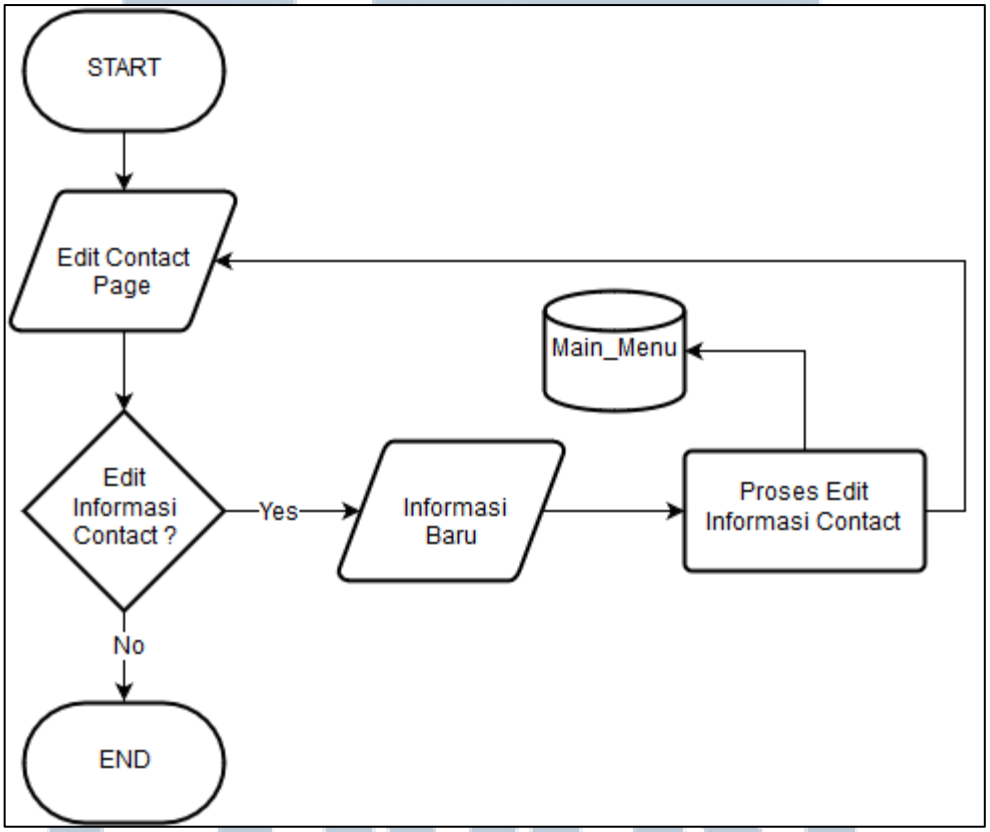
Gambar 3.18 adalah *flowchart* untuk proses *all data*. Proses dimulai dengan menampilkan *display* halaman *all data*. Kemudian apakah pengguna akan melihat detail dari data mahasiswa. Jika benar, maka akan ditampikan *display* detail data dan untuk kembali ke *display all data* pengguna dapat menekan tombol *back*. Jika tidak melihat detail, apakah pengguna akan menghapus data mahasiswa. Jika benar, akan masuk ke dalam proses menghapus data mahasiswa dan setelah proses selesai akan dikembalikan ke *display all data* untuk melakukan tindakan lain atau proses berakhir.



Gambar 3.19 *Flowchart* Proses *Edit About*

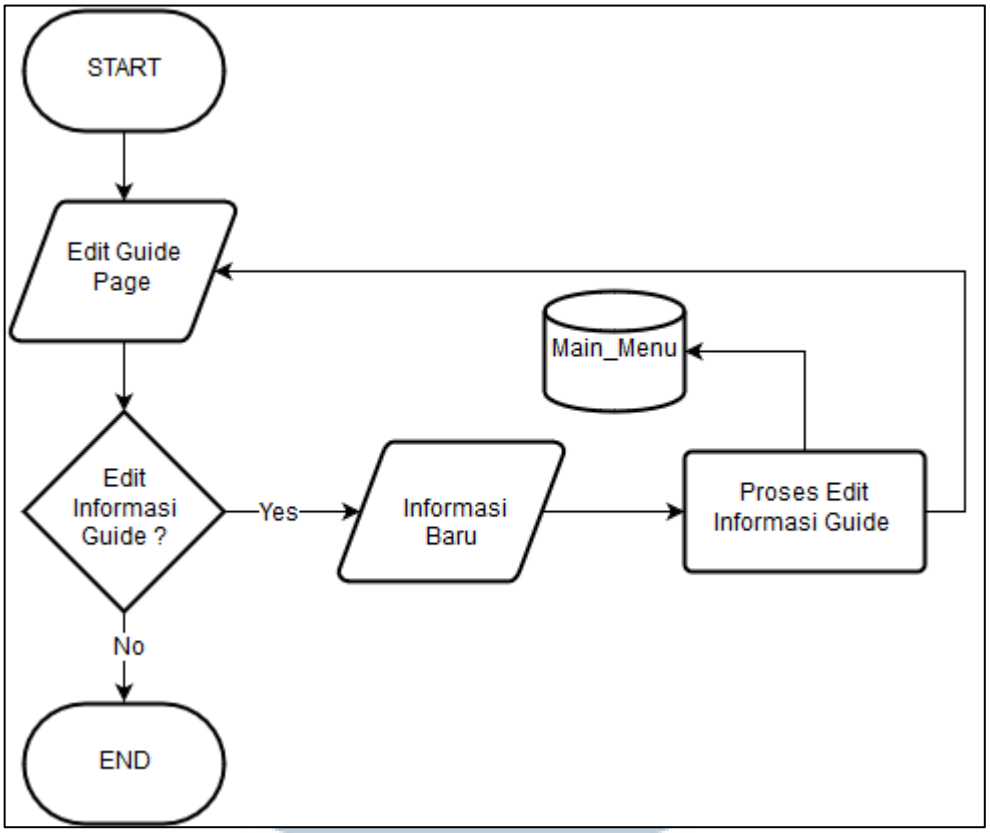
Gambar 3.19 adalah *flowchart* untuk proses *edit about*. Proses dimulai dengan menampilkan *display* halaman *edit about*. Kemudian apakah pengguna akan melakukan *update* pada informasi *about*. Jika benar, maka pengguna akan

diminta untuk memberikan *input* informasi *about* yang baru, dilanjutkan dengan proses *update* informasi *about* baru, dan setelah proses selesai, pengguna akan dikembalikan ke *display edit about* untuk menambah *update* infromasi *about* yang lain atau proses berakhir.



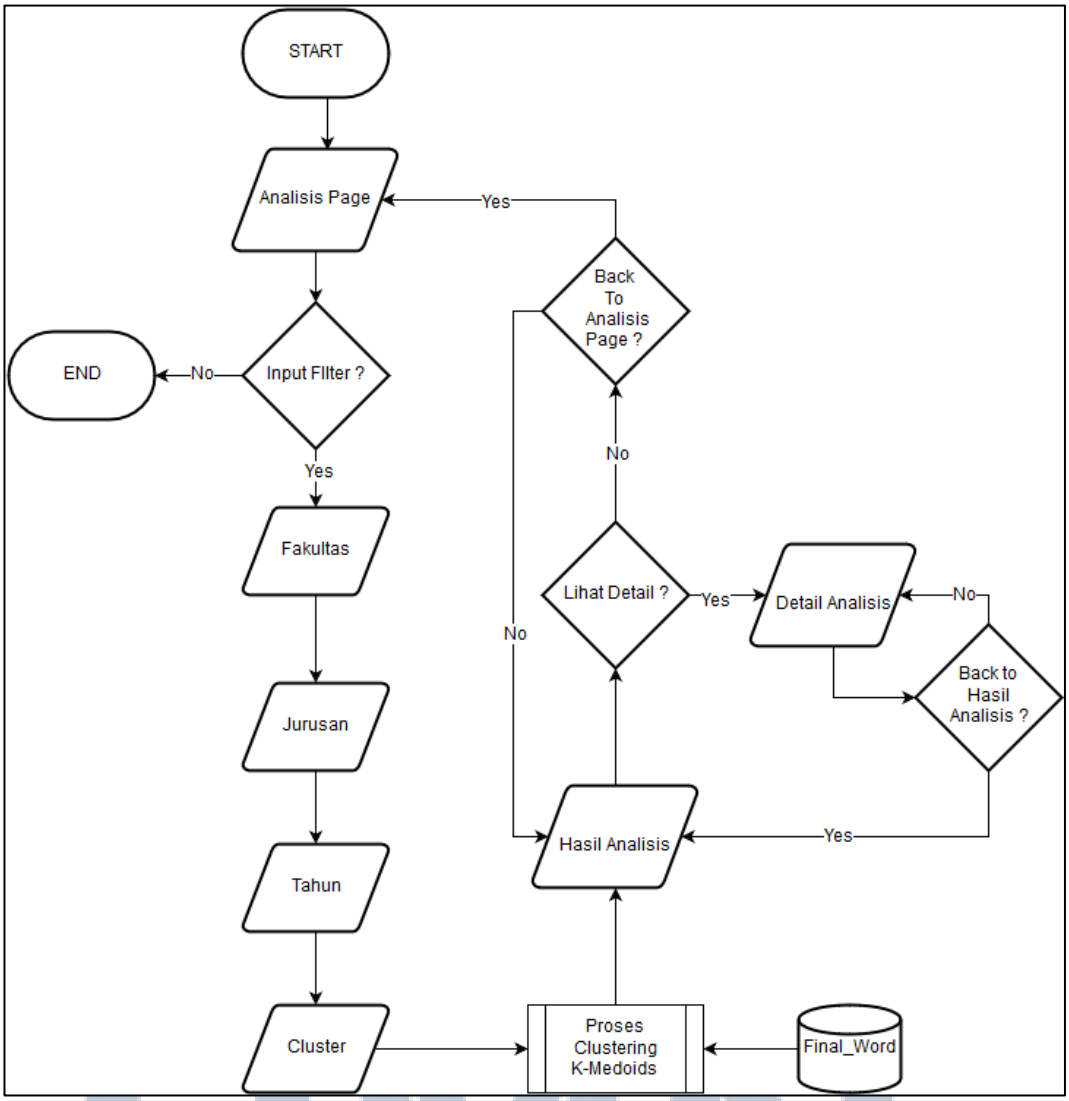
Gambar 3.20 *Flowchart* Proses *Edit Contact*

Gambar 3.20 adalah *flowchart* untuk proses *edit contact*. Proses dimulai dengan menampilkan *display* halaman *edit contact*. Kemudian apakah pengguna akan melakukan *update* pada informasi *contact*. Jika benar, maka pengguna akan diminta untuk memberikan *input* informasi *contact* yang baru, dilanjutkan dengan proses *update* informasi *contact* baru, dan setelah proses selesai, pengguna akan dikembalikan ke *display edit contact* untuk menambah *update* infromasi *contact* yang lain atau proses berakhir.



Gambar 3.21 *Flowchart* Proses *Edit Guide*

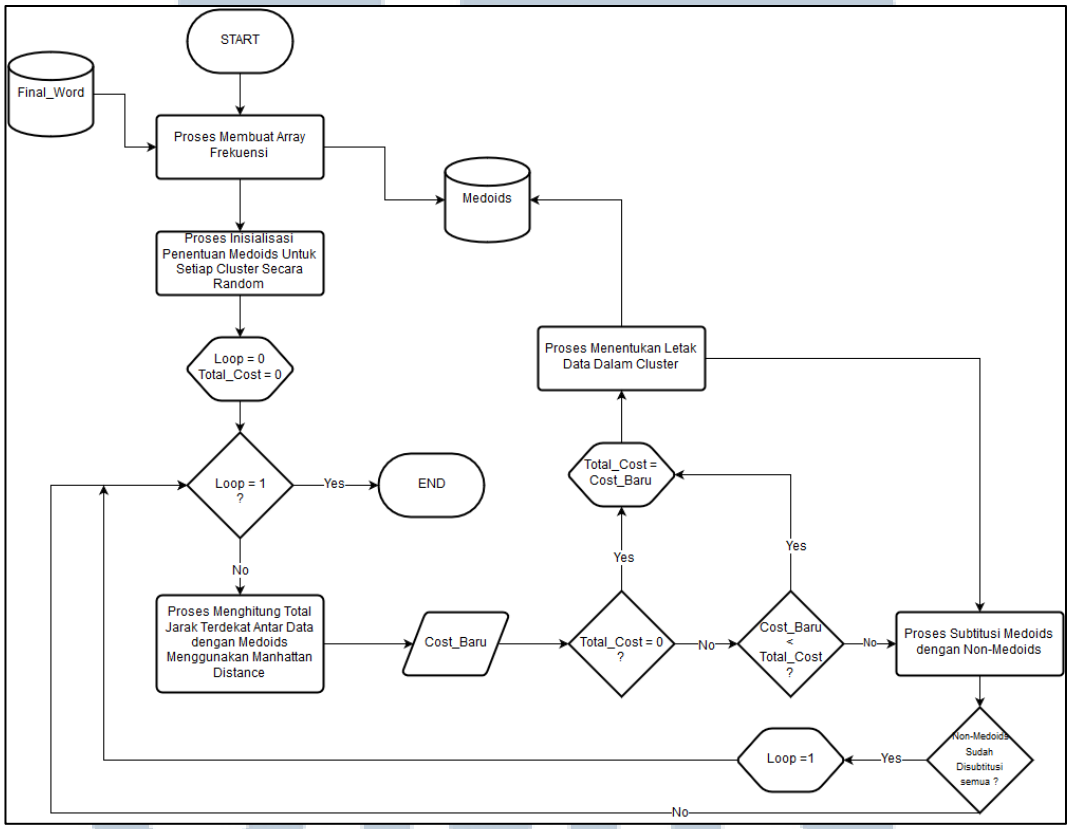
Gambar 3.21 adalah *flowchart* untuk proses *edit guide*. Proses dimulai dengan menampilkan *display* halaman *edit guide*. Kemudian apakah pengguna akan melakukan *update* pada informasi *guide*. Jika benar, maka pengguna akan diminta untuk memberikan *input* informasi *guide* yang baru, dilanjutkan dengan proses *update* informasi *guide* baru, dan setelah proses selesai, pengguna akan dikembalikan ke *display edit guide* untuk menambah *update* infromasi *guide* yang lain atau proses berakhir.



Gambar 3.22 *Flowchart* Proses Analisis

Gambar 3.22 adalah *flowchart* untuk proses analisis. Proses dimulai dengan menampilkan *display* halaman analisis. Kemudian apakah pengguna akan melakukan *input* filter. Jika tidak, maka proses akan berakhir, sedangkan jika benar, maka pengguna akan diminta untuk memberikan *input* fakultas, jurusan, tahun, dan *cluster*. Setelah semua data telah di-*input*, maka dilanjutkan dengan proses *clustering* K-Medoids. Proses *clustering* ini membutuhkan kata yang akan di analisis dari database *final_word* dan penjelasannya untuk tahap ini akan dijabarkan pada *flowchart* berikutnya. Saat proses *clustering* sudah selesai akan

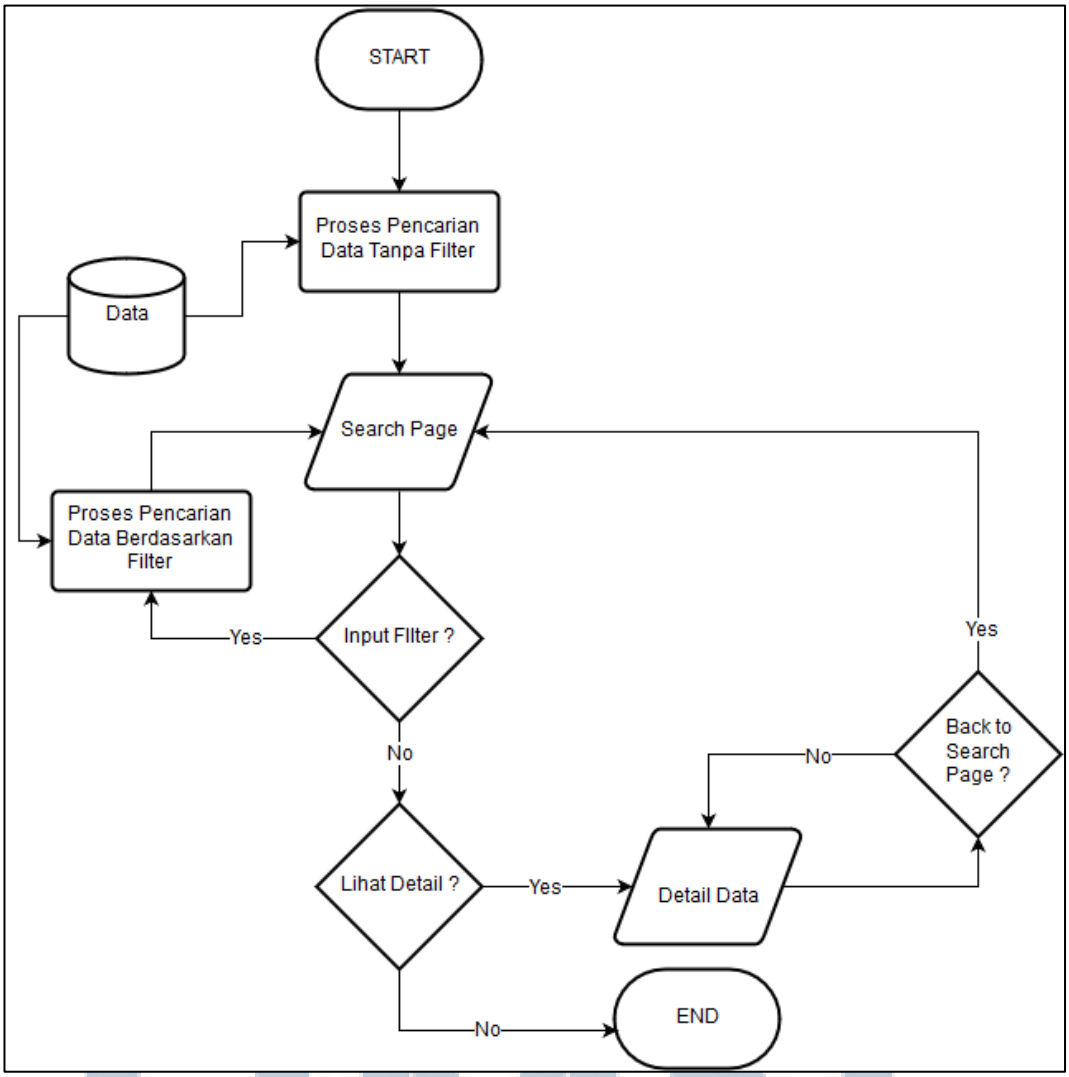
ditampilkan hasil analisis. Kemudian apakah pengguna akan melihat detail analisis. Jika benar, maka akan ditampilkan detail analisis. Setelah melihat detail analisis pengguna dapat menekan tombol *back* untuk kembali ke halaman analisis. Jika pengguna tidak ingin melihat detail analisis, maka pengguna dapat menekan tombol *back* untuk kembali ke halaman analisis.



Gambar 3.23 *Flowchart* Proses *Clustering* K-Medoids

Gambar 3.23 adalah *flowchart* untuk proses *clustering* K-Medoids. Dimulai dengan melakukan proses pembuatan *array* frekuensi kemunculan kata dari setiap dokumen dan data didapatkan dari database *final_word*. Selanjutnya, dilakukan proses untuk menentukan medoids dari setiap *cluster* secara acak. Kemudian,

tetapkan inisialisasi dua variabel yaitu *loop* = 0 dan *total_cost* = 0. Setelah ditentukan, maka selanjutnya akan masuk ke proses *looping* untuk menghitung total jarak dan menentukan pengelompokan *cluster*. Dicek apakah variabel *loop* = 1. Jika benar, maka proses *clustering* berakhir. Jika tidak benar, maka akan melakukan proses menghitung total jarak antara *medoids* dan *non-medoids.* Proses ini akan menghasilkan sebuah nilai *cost_baru* dan dicek apakah *total_cost* sekarang bernilai 0. Jika benar, maka akan dilakukan inisialisasi bahwa *total_cost = cost_baru*, melakukan proses penentuan letak data dalam *cluster,* dan lanjut ke proses subtitusi antara salah satu *medoids* dengan *non-medoids*. Jika tidak benar, akan dicek apakah nilai *cost_baru* kurang dari *total_cost*. Jika benar, maka akan akan dilakukan inisialisasi bahwa *total_cost = cost_baru*, melakukan proses penentuan letak data dalam *cluster*, dan lanjut ke proses subtitusi antara salah satu *medoids* dengan non-*medoids*. Jika nilai *cost_baru* tidak kurang dari *total_cost*, maka langsung lanjut ke proses subtitusi antara salah satu medoids dengan *non-medoids*. Kemudian apakah kata *non-medoids* sudah semuanya disubtitusi. Jika tidak benar, maka proses akan diulang dari perhitungan total jarak kembali hingga semua kata *non-medoids* sudah disubtitusi semua. Jika benar bahwa kata *non-medoids* sudah dilakukan subsitusi semua, maka *loop* akan diset menjadi *loop = 1* dan proses untuk *clustering* ini berakhir.



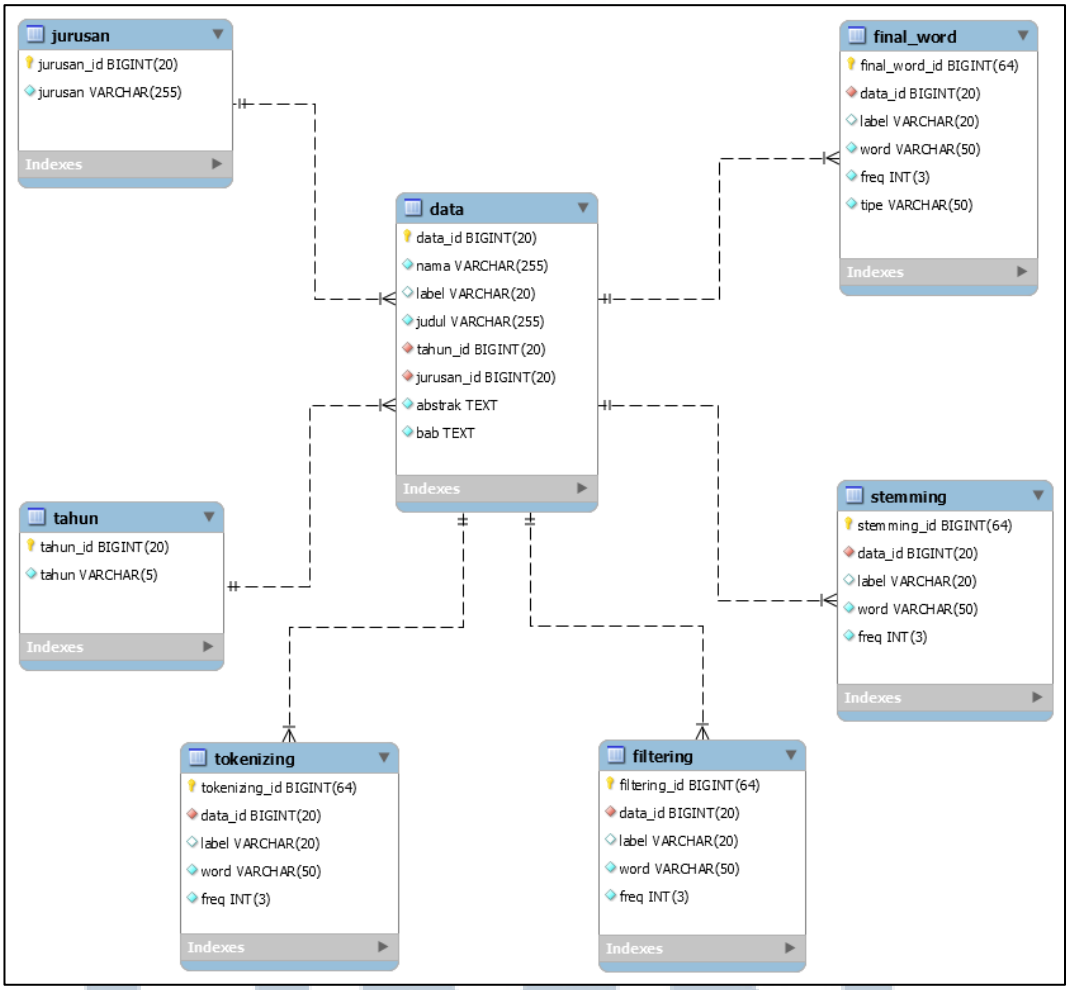
Gambar 3.24 *Flowchart* Proses *Search*

Gambar 3.24 adalah *flowchart* untuk proses *search*. Proses ini dimulai dengan pencarian semua data tanpa ada filter dari pengguna dan hasilnya ditampilkan pada *display* halaman *search*. Kemudian apakah pengguna ingin melakukan *input filter*. Jika benar, akan dilakukan proses pencarian data skripsi mahasiswa berdasarkan *input filter* dan hasilnya akan ditampilkan pada *display search*. Jika pengguna tidak melakukan *input filter*, apakah pengguna ingin melihat detail data skripsi mahasiswa. Jika benar, akan ditampilkan pada *display* detail data

dan apabila pengguna ingin kembali ke *display search* dapat menakan tombol *back*. Jika pengguna tidak melakukan lihat detail, maka proses berakhir.

3.3.4 Entity Relation Diagram

Entity Relation Diagram (ERD) merupakan suatu model yang digunakan untuk menggambarkan dan menjelaskan hubungan antar data dalam basis data berdasarkan objek-objek dasar data yang mempunyai hubungan antar relasi satu sama lain. ERD berfungsi untuk memodelkan struktur data dan hubungan antar data pada aplikasi *text mining* dan untuk menggambarkannya digunakan beberapa notasi dan simbol. Gambar 3.21 adalah ERD dari sistem yang dibuat untuk aplikasi *text mining* penentu tren topik skripsi. Dari 13 tabel yang digunakan hanya ada 7 tabel yang saling berhubungan. Hal ini disebabkan karena banyak tabel yang fungsinya hanya untuk keperluan tunggal atau dengan kata lain variabelnya tidak memiliki ketergantungan dengan tabel lain. Tabel yang tidak berhubungan seperti tabel kata_dasar, *stopword*, *main_menu*, *admin*, tabel_medoids, dan hasil analisis.



Gambar 3.25 ERD Aplikasi *Text Mining*

Gambar 3.25 terdapat 7 tabel yang saling berhubungan, yaitu tabel data, tahun, jurusan, *tokenizing*, *filtering*, *stemming*, dan *final_word*. Pada hubungan pertama terdapat *primary key* tahun_id dari tabel tahun yang berhubungan dengan tahun_id milik tabel data sebagai *foreign key*. Pada hubungan kedua terdapat *primary key* jurusan_id dari tabel jurusan yang berhubungan dengan jurusan_id milik tabel data sebagai *foreign_key*. Pada hubungan ketiga terdapat *primary key* data_id dari tabel data yang berhubungan dengan data_id milik tabel *tokenizing*, *filtering*, *stemming*, dan *final_word*.

3.3.4 Struktur Tabel

Database yang digunakan dalam penelitian ini adalah MySQL. Berikut truktur tabel yang digunakan adalah sebagai berikut.

1. Nama Tabel : admin

   Fungsi : Tabel ini digunakan untuk menyimpan informasi data dari *admin*

Tabel 3.1 Struktur Tabel admin

Field Name	Type	Length	Information
admin_id	bigint	20	*primary key*
username	varchar	255	
password	varchar	64	
name	varchar	255	
address	varchar	255	
telephone	varchar	20	
last_online	datetime		

Tabel *admin* memiliki 7 variabel, yaitu admin_id, username, password, name, address, telephone, dan last_online. admin_id sebagai *primary key* yang berfungsi sebagai entitas unik untuk menandakan seorang *admin*. username digunakan untuk menandakan identitas *admin* saat melakukan akses ke dalam sistem. password digunakan sebagai kata sandi untuk masuk ke dalam sistem. name digunakan untuk menyimpan nama asli *admin*. address digunakan untuk menyimpan alamat tinggal *admin*. telephone digunakan untuk menyimpan nomor milik *admin* yang dapat dihubungi. last_online digunakan untuk menyimpan waktu kapan terakhir *admin* melakukan akses ke dalam sistem.

2. Nama Tabel : data

   Fungsi : Tabel ini digunakan untuk menyimpan informasi data mahasiswa

Tabel 3.2 Struktur Tabel data

Field Name	Type	Length	Information
data_id	bigint	20	*primary key*
nama	varchar	255	
label	varchar	20	
judul	varchar	255	
tahun_id	bigint	20	*reference key* dari *tahun_id* milik tabel *tahun*
jurusan_id	bigint	20	*reference key* dari *jurusan_id* milik tabel *jurusan*
abstrak	text		
bab	text		

Tabel data memiliki 8 variabel, yaitu data_id, nama, nim, judul, tahun_id, jurusan_id, abstrak, dan bab. data_id sebagai *primary key* yang berfungsi sebagai entitas unik untuk menandakan informasi dari setiap mahasiswa. nama digunakan untuk menyimpan nama asli dari mahasiswa. label digunakan untuk menyimpan kode skripsi mahasiswa. judul digunakan untuk menyimpan judul skripsi milik mahasiswa. tahun_id merupakan *reference key* tahun_id dari tabel tahun untuk menunjukkan kode unik dari tahun terbit skripsi. jurusan_id merupakan *reference key* jurusan_id dari tabel jurusan untuk menunjukkan kode unik dari jurusan mahasiswa. abstrak digunakan untuk menyimpan *plain text* hasil dari ekstraksi file abstrak skripsi mahasiswa. bab digunakan untuk menyimpan *plain text* hasil dari ekstraksi file bab 1 skripsi mahasiswa.

3. Nama Tabel : filtering

Fungsi : Tabel ini digunakan untuk menampung hasil proses *filtering*

Tabel 3.3 Struktur Tabel filtering

Field Name	Type	Length	Information
filtering_id	bigint	64	*primary key*
data_id	bigint	20	*reference key* dari *data_id* milik tabel *data*
label	varchar	20	
word	varchar	50	
freq	int	3	

Tabel data memiliki 5 variabel, yaitu *filtering_id,* data_id, label, word, dan freq. *filtering_id* sebagai *primary key* yang berfungsi sebagai entitas unik untuk setiap data yang disimpan label digunakan untuk menyimpan kode skripsi dari mahasiswa. data_id merupakan *reference key* data_id dari tabel data untuk menunjukkan kode unik dari data mahasiswa. label digunakan untuk menyimpan kode skripsi mahasiswa. word digunakan untuk menyimpan kata yang tersaring dari proses *filtering*. freq digunakan untuk menyimpan hasil akumulasi dari kemunculan setiap kata yang sama.

4. Nama Tabel : final_word

   Fungsi : Tabel ini digunakan untuk menampung hasil akhir kata setelah semua proses *preprocessing* dilakukan.

Tabel 3.4 Struktur Tabel final_word

Field Name	Type	Length	Information
final_word_id	bigint	64	*primary key*
label	varchar	20	
word	varchar	50	
freq	int	3	
tipe	varchar	50	

Tabel data memiliki 5 variabel, yaitu filtering_id, label, *word*, freq, dan tipe. *final_word_id* sebagai *primary key* yang berfungsi sebagai entitas unik untuk setiap data yang disimpan. label digunakan untuk menyimpan kode skripsi dari mahasiswa. word digunakan untuk menyimpan kata yang tersaring dari proses *filtering*. freq digunakan untuk menyimpan hasil akumulasi dari kemunculan setiap kata yang sama. tipe digunakan untuk menandakan informasi mengenai kata yang disimpan.

5. Nama Tabel : hasil_analisis

   Fungsi : Tabel ini digunakan untuk menampung semua hasil akhir dari proses *clustering*.

Tabel 3.5 Struktur Tabel hasil_analisis

Field Name	Type	Length	Information
hasil_analisis_id	bigint	64	*primary key*
word	varchar	50	
cluster	int	3	
total_freq	int	10	
freq_array	text		
jurusan_filter	varchar	50	*reference key* dari *jurusan_id* milik tabel *jurusan*
tahun_filter	varchar	50	*reference key* dari *tahun_id* milik tabel *tahun*
cluster_filter	int	3	
execution_time	text		

Tabel data memiliki 9 variabel, yaitu hasil_analisis_id, word, cluster, total_freq, freq_array, jurusan_filter, tahun_filter, cluster_filter, dan execution_time. hasil_analisis_id sebagai *primary key* yang berfungsi sebagai entitas unik untuk setiap data yang disimpan. word digunakan untuk menyimpan

kata yang tersaring dari proses *filtering*. cluster digunakan untuk menandakan lokasi kata pada suatu *cluster*. total_freq digunakan untuk menyimpan akumulasi frekuensi kemunculan suatu kata dari semua dokumen. freq_array digunakan untuk menyimpan kemunculan setiap kata pada semua dokumen. jurusan_filter merupakan *reference key* jurusan_id dari tabel jurusan untuk menunjukkan kode unik dari jurusan mahasiswa. tahun_filter merupakan *reference key* tahun_id dari tabel tahun untuk menunjukkan kode unik dari tahun terbit skripsi. cluster_filter digunakan untuk menyimpan jumlah *cluster* yang digunakan dalam analisis. execution_time digunakan untuk menyimpan waktu eksekusi yang diperlukan dari membuat array frekuensi hingga proses *clustering* selesai.

6. Nama Tabel : jurusan

   Fungsi : Tabel ini digunakan untuk menyimpan nama-nama jurusan mahasiswa

Tabel 3.6 Struktur Tabel jurusan

Field Name	Type	Length	Information
jurusan_id	bigint	20	*primary key*
Jurusan	varchar	255	

Tabel data memiliki 2 variabel, yaitu jurusan_id, dan juruan. jurusan_id sebagai *primary key* yang berfungsi sebagai entitas unik untuk setiap data jurusan yang disimpan. jurusan digunakan untuk menyimpan nama-nama jurusan mahasiswa.

7. Nama Tabel : kata_dasar

   Fungsi : Tabel ini digunakan untuk menyimpan kata dasar bahasa Indonesia berdasarkan Kamus Besar Bahasa Indonesia (KBBI)

Tabel 3.7 Struktur Tabel kata_dasar

Field Name	Type	Length	Information
kata_dasar_id	bigint	20	*primary key*
kata_dasar	varchar	50	
tipe	varchar	50	

Tabel data memiliki 3 variabel, yaitu kata_dasar_id, kata_dasar, dan tipe. kata_dasar_id sebagai *primary key* yang berfungsi sebagai entitas unik untuk setiap kata dasar yang disimpan. kata_dasar digunakan untuk menyimpan kata dasar bahasa Indonesia berdasarkan KBBI. Tipe digunakan untuk menandakan informasi mengenai kata yang disimpan.

8. Nama Tabel : main_menu

   Fungsi : Tabel ini digunakan untuk menyimpan data *about*, *contact*, dan *guide* aplikasi

Tabel 3.8 Struktur Tabel main_menu

Field Name	Type	Length	Information
main_menu_id	bigint	20	*primary key*
about	text		
contact	text		
guide	text		

Tabel data memiliki 4 variabel, yaitu *main_menu_id*, about, contact, dan guide. *main_menu_id* sebagai *primary key* yang berfungsi sebagai entitas unik untuk setiap data yang disimpan. about digunakan untuk menyimpan data tentang latar belakang aplikasi dibuat. Contact digunakan untuk menyimpan data tentang profil pembuat aplikasi. Guide digunakan untuk menyimpan data tentang cara menggunakan aplikasi.

9. Nama Tabel : stemming

Fungsi : Tabel ini digunakan untuk menampung kata yang telah melalui proses *stemming*

Tabel 3.9 Struktur Tabel stemming

Field Name	Type	Length	Information
stemming_id	bigint	64	*primary key*
data_id	bigint	20	*reference key* dari *data_id* milik tabel *data*
label	varchar	20	
word	varchar	50	
freq	int	3	

Tabel data memiliki 5 variabel, yaitu stemming_id, data_id, label, word, dan freq. stemming_id sebagai *primary key* yang berfungsi sebagai entitas unik untuk setiap data hasil *stemming* yang disimpan. data_id merupakan *reference key* data_id dari tabel data untuk menunjukkan kode unik dari data mahasiswa. label digunakan untuk menyimpan kode skripsi dari mahasiswa. word digunakan untuk menyimpan kata hasil proses *stemming*. freq digunakan untuk menyimpan frekuensi kemunculan kata pada suatu dokumen.

10. Nama Tabel : stopword

Fungsi : Tabel ini digunakan untuk menyimpan kumpulan *stopword list*

Tabel 3.10 Struktur Tabel stopword

Field Name	Type	Length	Information
stopword_id	bigint	20	*primary key*
word	varchar	20	

Tabel data memiliki 2 variabel, yaitu stopword_id, dan word. stopword_id sebagai *primary key* yang berfungsi sebagai entitas unik untuk setiap kata *stopword* yang disimpan. word digunakan untuk menyimpan kata yang termasuk ke dalam *stopword list*.

11. Nama Tabel : tabel_medoids

    Fungsi : Tabel ini digunakan untuk menampung data selama menjalani proses *clustering*

Tabel 3.11 Struktur Tabel tabel_medoids

Field Name	Type	Length	Information
tabel_medoids_id	bigint	64	*primary key*
word	varchar	50	
freq_array	text		
total_freq	int	10	
flag	int	3	
flag_2	int	3	
cluster	int	3	

Tabel data memiliki 7 variabel, yaitu tabel_medoids_id, word, freq_array, total_freq, flag, flag_2, dan cluster. tabel_medoids_id sebagai *primary key* yang berfungsi sebagai entitas unik untuk setiap data yang disimpan. word digunakan untuk menyimpan kata yang akan digunakan pada proses *clustering*. freq_array digunakan untuk menyimpan data array frekuensi kemunculan setiap kata pada setiap dokumen. total_freq digunakan untuk menyimpan akumulasi frekuensi kemunculan kata dari semua dokumen. flag digunakan untuk merubah status kata menjadi tersedia di dalam tabel agar tidak terjadi duplikasi dengan kata yang sama. flag_2 digunakan untuk menandai status subtitusi kata yang digunakan selama proses *clustering*. Cluster digunakan untuk menandakan lokasi kata pada suatu *cluster*.

12. Nama Tabel : tahun

Fungsi : Tabel ini digunakan untuk menyimpan tahun terbit skripsi

Tabel 3.12 Struktur Tabel tahun

Field Name	Type	Length	Information
tahun_id	bigint	20	*primary key*
tahun	varchar	5	

Tabel data memiliki 2 variabel, yaitu tahun_id, dan tahun. tahun_id sebagai *primary key* yang berfungsi sebagai entitas unik untuk data tahun yang disimpan. tahun digunakan untuk menyimpan tahun terbit skripsi.

13. Nama Tabel : tokenizing

Fungsi : Tabel ini digunakan untuk menampung kata yang telah melalui proses *case folding* dan *tokenizing*

Tabel 3.13 Struktur Tabel tokenizing

Field Name	Type	Length	Information
tokenizing_id	bigint	64	*primary key*
data_id	bigint	20	*reference key* dari *data_id* milik tabel *data*
label	varchar	20	
word	varchar	50	
freq	int	3	

Tabel data memiliki 5 variabel, yaitu tokenizing_id, data_id, label, word, dan freq. tokenizing_id sebagai *primary key* yang berfungsi sebagai entitas unik untuk setiap data hasil tahap *case folding* dan *tokenizing* yang disimpan. data_id merupakan *reference key* data_id dari tabel data untuk menunjukkan kode unik dari data mahasiswa. label digunakan untuk menyimpan kode skripsi dari mahasiswa.

word digunakan untuk menyimpan kata hasil proses *case folding* dan *tokenizing*. freq digunakan untuk menyimpan frekuensi kemunculan kata pada suatu dokumen.

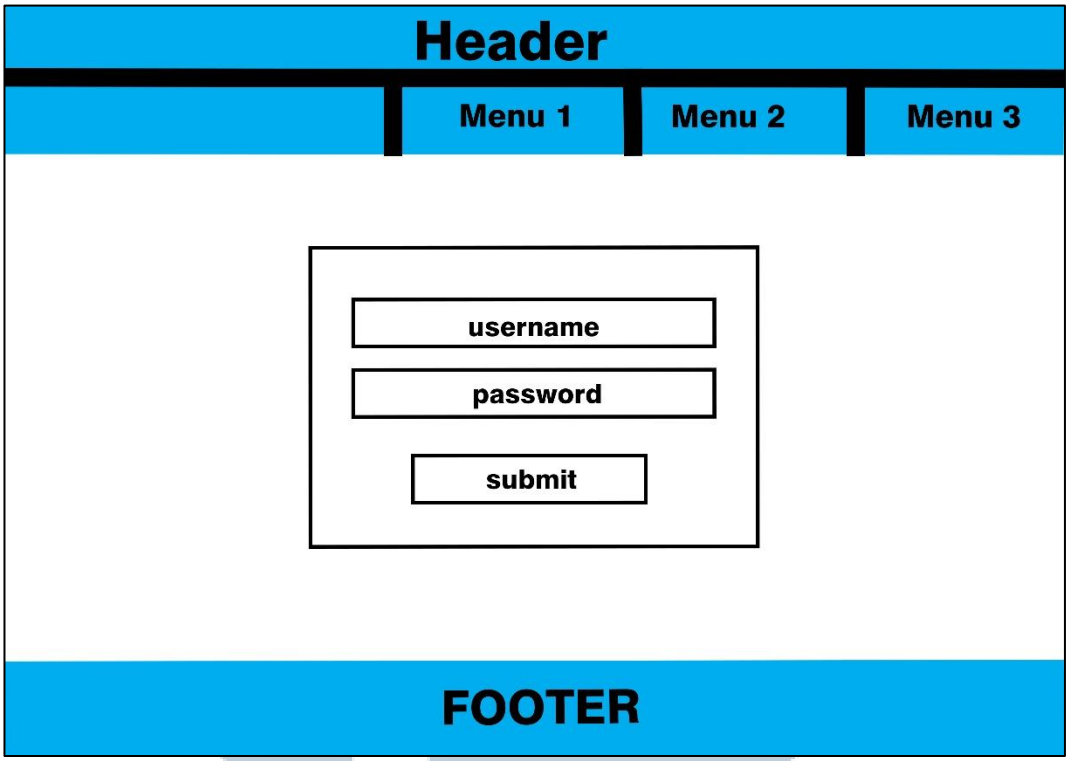
3.4 Perancangan Tampilan Antarmuka

Pada rancang bangun aplikasi *text mining* untuk penentuan tren topik skripsi, terdapat perancangan tampilan antarmuka yang digunakan sebagai penggambaran awal sistem yang akan dibangun pada penelitian ini. Perancangan akan dibagi menjadi 2 bagian, yaitu *back-end* dan *front-end*. Bagian *back-end* akan difokuskan pada keperluan *admin* yang nantinya sebagai pengelola aplikasi ini. Maksud keperluan tersebut adalah hal-hal yang berhubungan dengan pemrosesan data mentah, seperti melakukan *input* data skripsi mahasiswa, menambah jurusan dan tahun terbit, menambah dan menghapus *stopword list*, memberikan informasi tambahan pada kata baru untuk dipelajari sistem, dan megolah data skripsi mahasiswa yang terdapat pada database. Tahap *back-end* baik perancangan maupun implementasinya akan dilakukan terlebih dahulu karena bagian ini berfungsi sebagai dasar dari pengolahan data mentah dan hasil akhirnya baru akan digunakan dalam proses *clustering* yang terdapat pada bagian *front-end*. Pada bagian *front-end* akan dirancang sebuah halaman yang *user friendly* untuk proses *clustering* menacari tren topik skripsi. Proses *clustering* akan dibuat menjadi beberapa tahapan (*step by step*) agar pengguna cepat terbiasa menggunakan aplikasi ini. Selain untuk analisis *clustering*, bagian *front-end* juga terdapat halaman *search* yang difungsikan untuk pencarian informasi judul skripsi mahasiswa. Perancangan untuk semua halaman nantinya akan dibagi menjadi 3 bagian seperti di atas, yaitu *header*, *body*, dan *footer*. Pada bagian *header* akan terdapat pemasangan logo aplikasi, informasi waktu, dan 3 buah tombol menu. Pada bagian *body* akan digunakan untuk

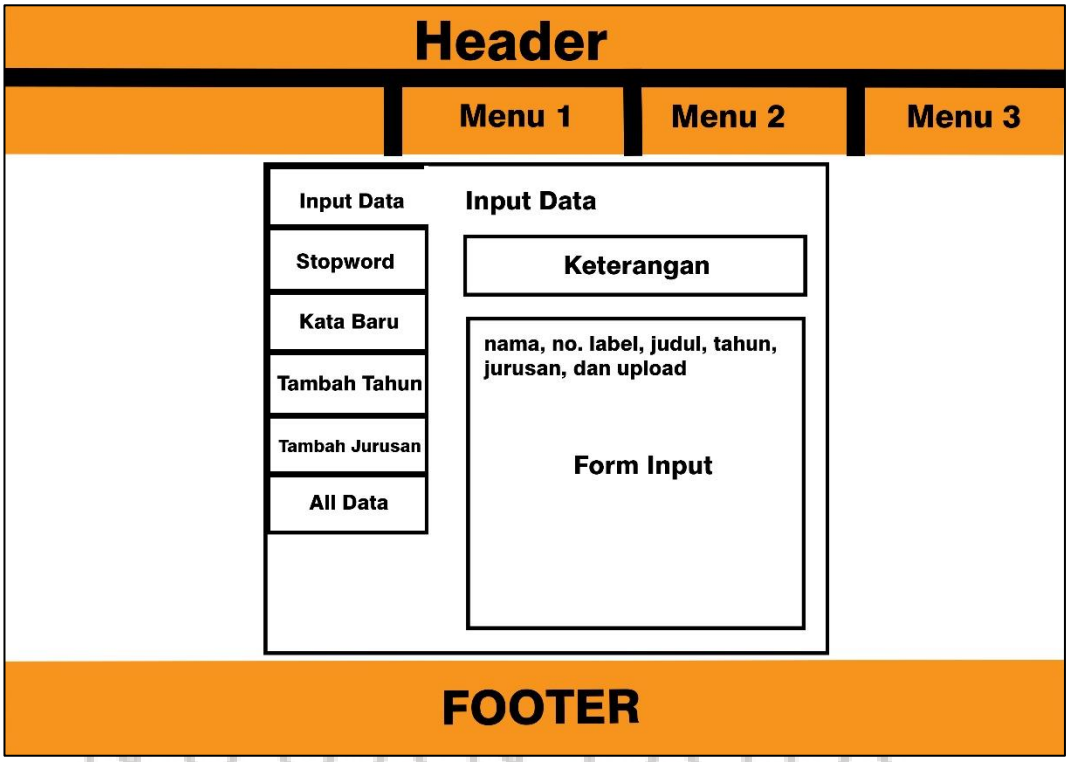
menampilkan isi dari aplikasi sesuai dengan halaman yang dituju. Pada bagian *footer* akan ada sebuah tombol yang nantinya digunakan pada proses *login* dan *logout* untuk berpindah halaman. Gambaran dan penjelasan yang lebih jelas untuk perancangan awal *back-end* dan *front-end* dapat dilihat pada subbab di bawah ini.

3.4.1 Back-End

Perancangan *back-end* dimulai dengan membuat sebuah halaman *login* yang berfungsi untuk membatasi hak akses pengguna biasa dengan *admin*. Halaman *login* dibuat tersendiri dengan halaman lain dan tampilannya cukup sederhana, hanya ada 2 *input* yang harus diisikan oleh pengguna (*admin*), yaitu *username* dan *password*. Terdapat sebuah tombol *submit* yang digunakan untuk melakukan proses verifikasi. Halaman *login* ini dapat dilihat pada Gambar 3.26.

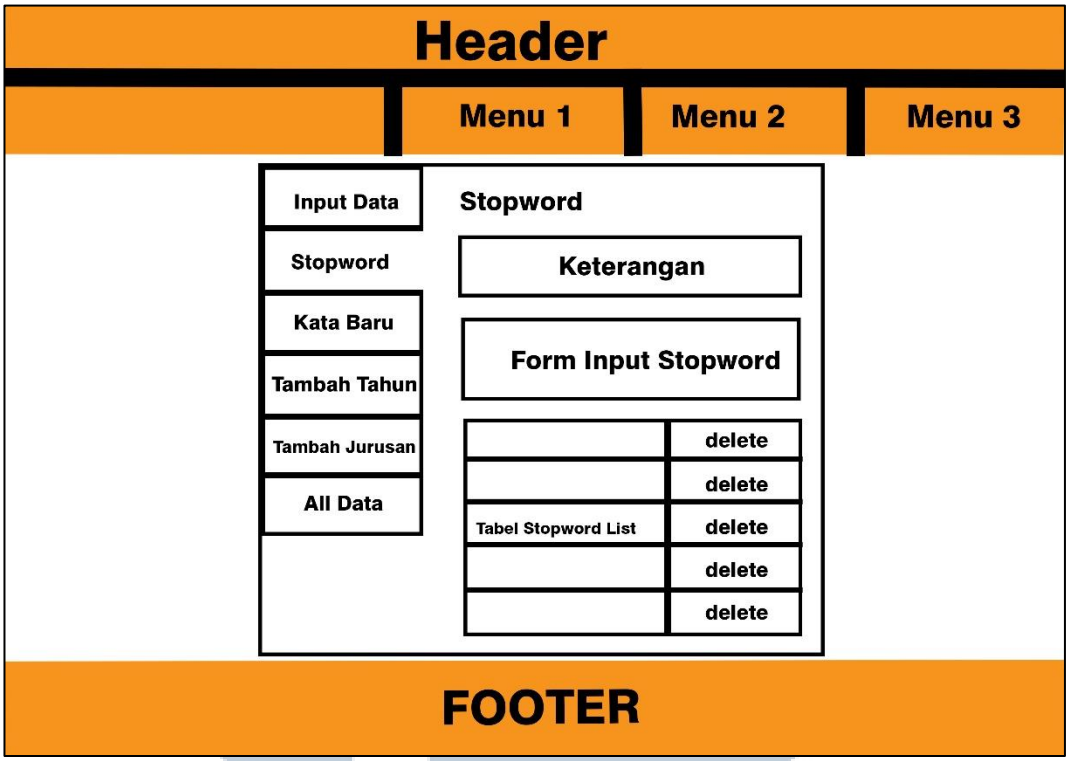


Gambar 3.26 Perancangan Antarmuka Halaman *Login*



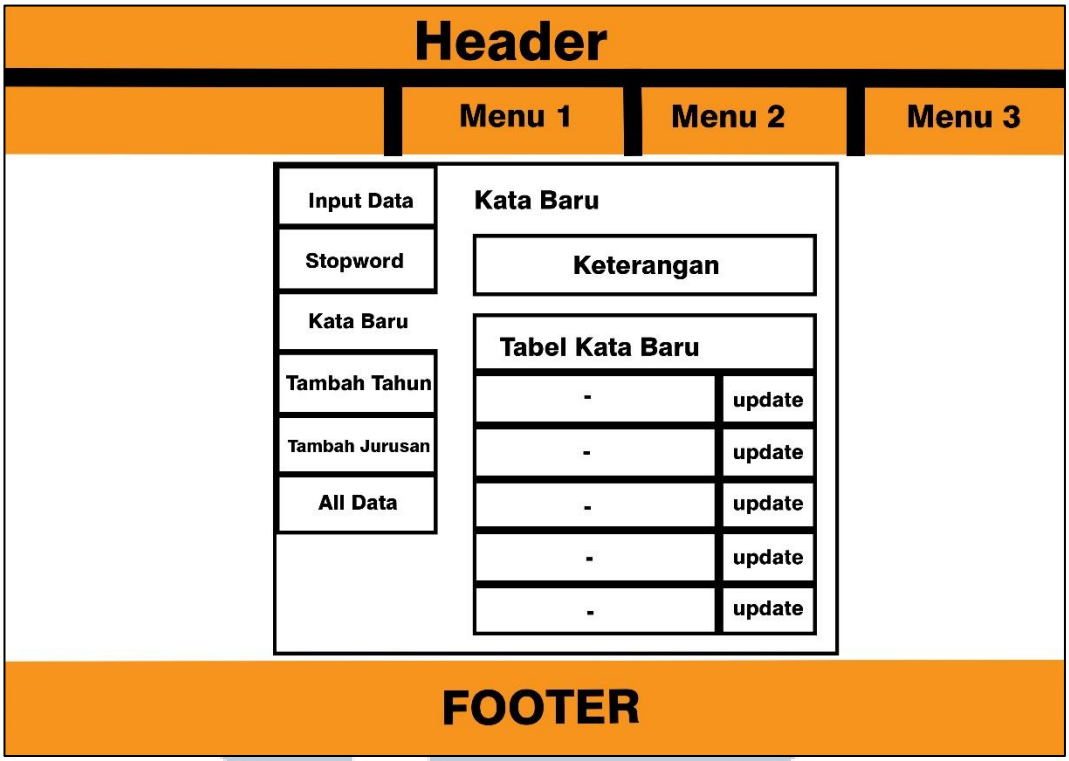
Gambar 3.27 Perancangan Antarmuka Back-End Halaman Input Data

Gambar 3.27 adalah perancangan antarmuka *back*-end halaman input data. Perancangan dari halaman *back-end input* data, menampilkan 2 sisi. Sisi kiri terdapat *left side bar* yang berisi berbagai pilihan tombol, sedangkan sisi kanan dijadikan tampilan isi dari tombol yang sedang aktif atau ditekan. Pembagian sisi ini akan berlaku untuk semua jenis perancangan pada halaman *back-end*. Tujuan dibuat pembagian ini adalah ingin memberikan kesan sederhana dan kemudahan akses ke halaman *back-end* lainnya tanpa harus berpindah halaman (*load page*). Pada bagian isi halaman *input* data dibagi menjadi 2 bagian, yaitu bagian keterangan dan bagian *form input*. Bagian keterangan akan berisi informasi penting mengenai hal-hal yang perlu diperhatikan dan dipersiapkan sebelum melakukan *input* data, sedangkan pada bagian *form input* nantinya akan terdiri dari beberapa *form* untuk pengisian data mahasiswa (nama, no label, judul, tahun terbit, jurusan, upload abstrak, dan upload bab 1) dan tombol *submit* untuk menjalankan proses *input* data.



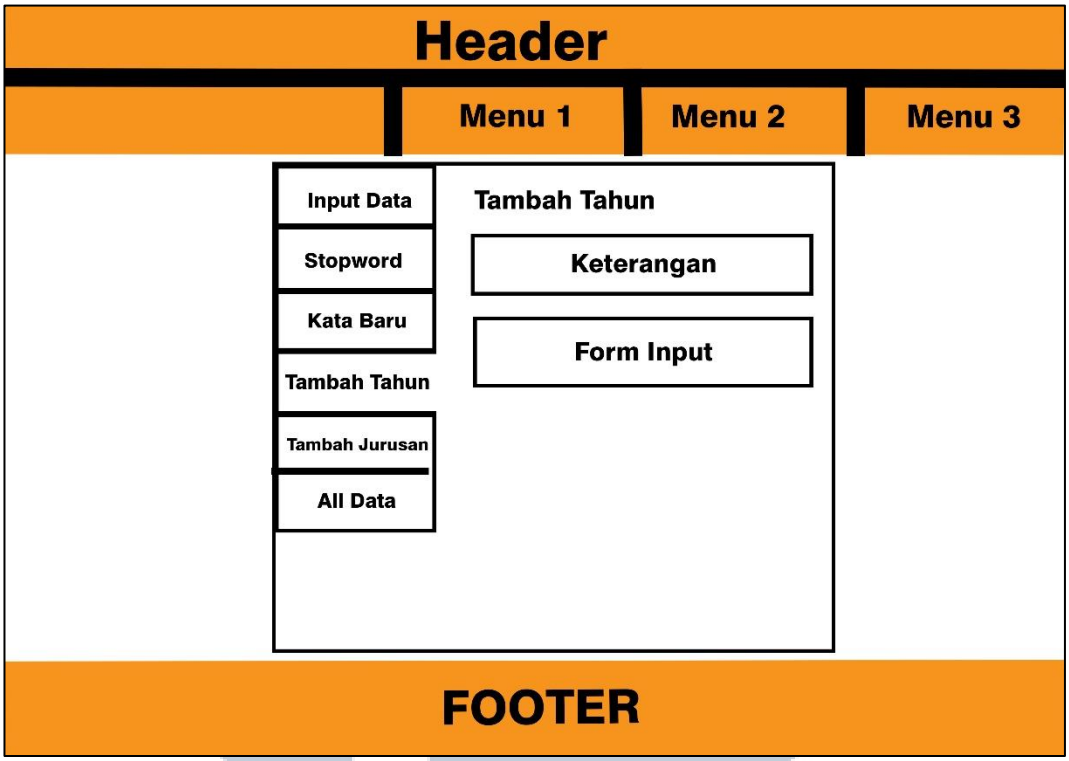
Gambar 3.28 Perancangan Antarmuka Back-End Halaman *Stopword*

Gamabar 3.28 adalah perancangan antarmuka *back-end* halaman *stopword*. Pembagian sisi kiri dan kanan halaman sama dengan halaman *back-end* lainnya. Pada halaman *stopword* dibagi menjadi 3 bagian, yaitu bagian keterangan, bagian *form input stopword*, dan bagian tabel *stopword list*. Pada bagian keterangan akan berisikan informasi penting mengenai apa saja hal yang perlu diperhatikan dan dipersiapkan saat menambah atau menghapus *stopword*. Pada bagian *input form stopword* akan terdiri dari sebuah *form input* dengan tipe *text* dan sebuah tombol *submit* yang digunakan untuk menambahkan *stopword* baru. Pada bagian tabel *stopword list* akan ditampilkan semua *stopword* dari database dan terdapat sebuah tombol *delete* untuk masing-masing baris yang digunakan untuk menghapus *stopword* dari database.



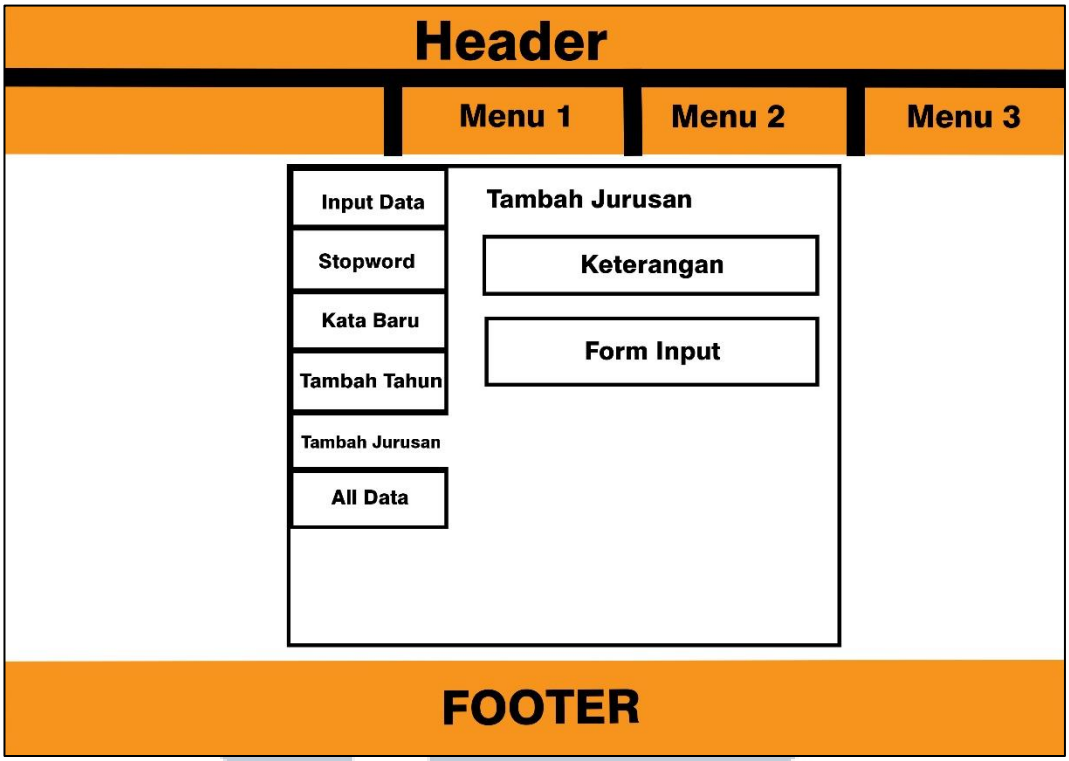
Gambar 3.29 Perancangan Antarmukam Back-End Halaman Kata Baru

Gamabar 3.29 adalah perancangan antarmuka *back-end* halaman kata baru. Pembagian sisi kiri dan kanan halaman sama dengan halaman *back-end* lainnya. Pada halaman kata baru dibagi menjadi 2 bagian, yaitu bagian keterangan dan bagian tabel kata baru. Pada bagian keterangan digunakan untuk menampilkan informasi penting mengenai hal apa saja yang perlu diperhatikan dan dipersiapkan saat melakukan *update* keterangan baru kata untuk dipelajari oleh sistem. Pada bagian tabel kata baru, akan ditampilkan semua daftar kata dari database yang akan dipelajari oleh sistem. Terdapat tombol *update* untuk masing-masing baris yang digunakan untuk melakukan perubahan atau menambahkan informasi baru pada keterangan kata yang lama.



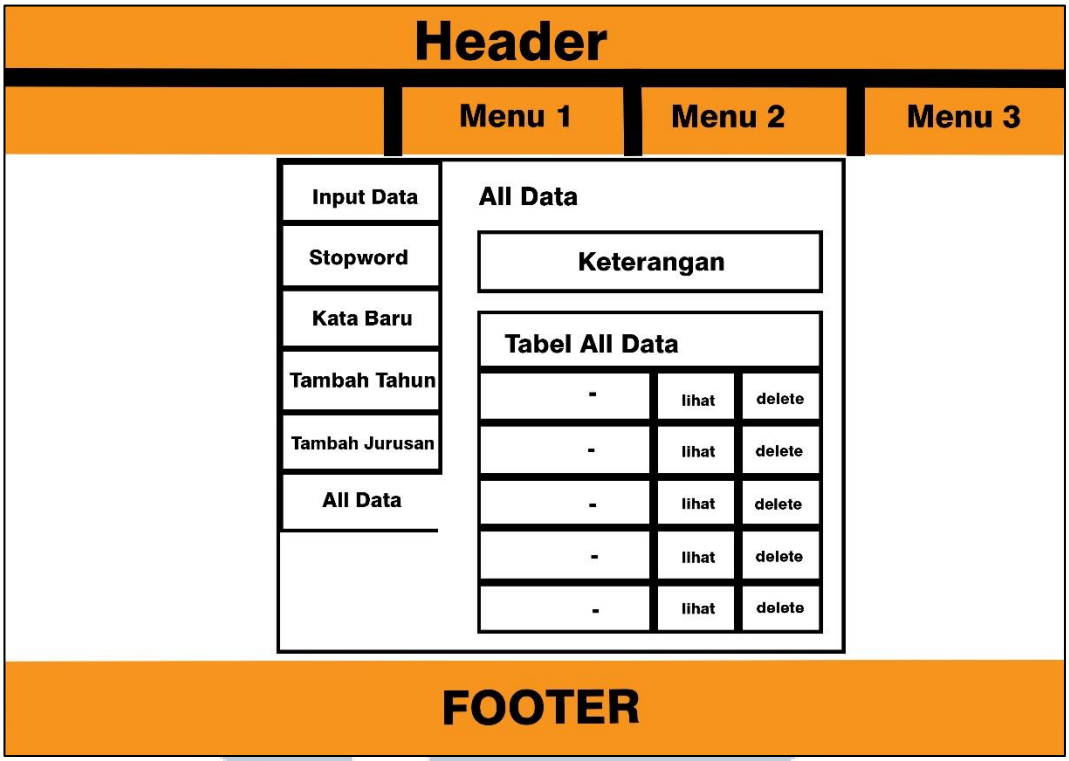
Gambar 3.30 Perancangan Antarmuka Back-End Halaman Tambah Tahun

Gambar 3.30 adalah perancangan antarmuka *back-end* halaman tambah tahun. Pembagian sisi kiri dan kanan halaman sama dengan halaman *back-end* lainnya. Pada halaman tambah tahun dibagi menjadi 2 bagian, yaitu bagian keterangan dan bagian *form input*. Pada bagian keterangan digunakan untuk menampilkan informasi penting mengenai hal apa saja yang perlu diperhatikan sebelum melakukan penambahan tahun terbit. Pada bagian *form input* nantinya akan terdapat sebuah *form* untuk mengisi tahun terbit baru dan sebuah tombol *submit* untuk melakukan proses penambahan tahun terbit baru.



Gambar 3.31 Perancangan Antarmuka Back-End Halaman Tambah Jurusan

Gambar 3.31 adalah perancangan antarmuka *back-end* halaman tambah jurusan. Pembagian sisi kiri dan kanan halaman sama dengan halaman *back-end* lainnya. Halaman tambah jurusan ini pembagian dan fungsinya hampir sama dengan sama halaman tambah tahun, yaitu dibagi menjadi bagian keterangan dan bagian *form input*. Bagian keterangan akan berisikan informasi penting mengenai hal apa saja yang perlu diperhatikan dan dipersiapkan sebelum melakukan penambahan jurusan baru ke dalam database. Pada bagian *form input* nantinya juga akan terdapat sebuat *form* untuk mengisi jurusan yang baru dan sebuah tombol *submit* untuk memulai proses penambahan jurusan baru.

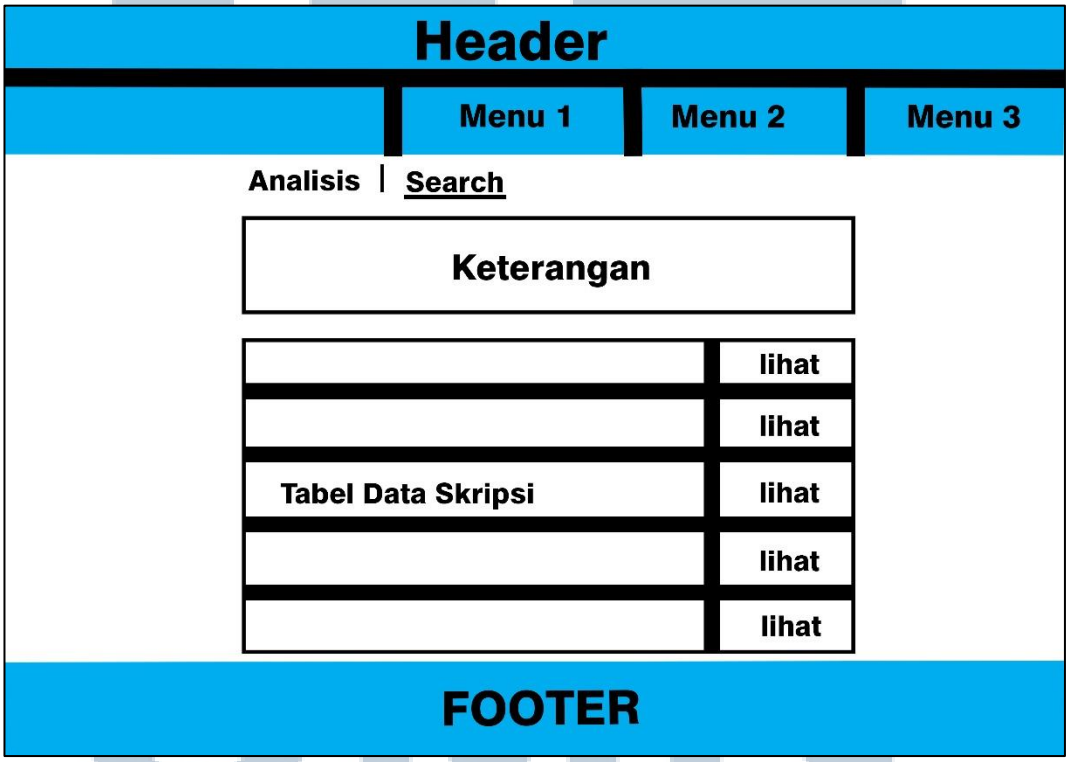


Gambar 3.32 Perancangan Antarmuka Back-End Halaman *All Data*

Gambar 3.32 adalah perancangan antarmuka *back-end* halaman *all data*. Pembagian sisi kiri dan kanan halaman sama dengan halaman *back-end* lainnya. Pada halaman *all data* dibagi menjadi 2 bagian, yaitu bagian keterangan dan bagian tabel *all data*. Pada bagian keterangan akan digunakan untuk menampilkan informasi penting mengenai hal apa saja yang perlu diperhatikan dan dipersiapkan saat melakukan pencarian atau menghapus data dari database. Pada bagian tabel *all data* digunakan untuk menampilkan semua data informasi skripsi mahasiswa. Tabel ini juga disediakan 2 tombol, yaitu tombol lihat dan tombol *delete*. Tombol lihat digunakan untuk menampilkan keterangan yang lebih spesifik terhadap data yang ingin dilihat dan dari sini pengguna juga dapat mengunduh abstrak atau bab 1 dari skripsi mahasiswa yang bersangkutan.

3.4.2 Front-End

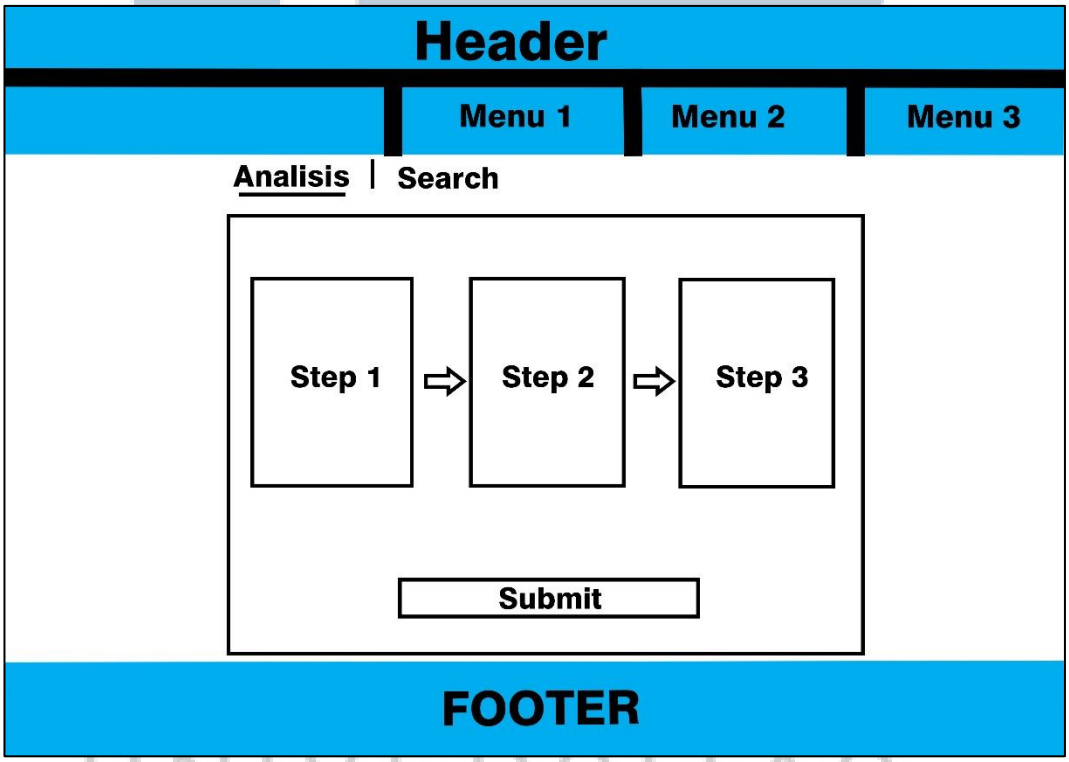
Perancangan bagian *front-end* berisikan halaman-halaman utama yang nantinya akan digunakan oleh pengguna untuk melakukan analisis atau pencarian data skripsi mahasiswa yang tersedia di dalam database. Fokus pada *front-end* dibagi menjadi 4 halaman, yaitu halaman *search,* halaman analisis, halaman hasil analisis, dan halaman detail analisis.



Gambar 3.33 Perancangan Antarmuka Front-End Halaman *Search*

Gambar 3.33 adalah perancangan antarmuka *front-end* halaman *search*. Halaman ini berfungsi untuk menampilkan semua data skripsi mahasiswa kepada pengguna. Halaman *search* dibagi menjadi 2 bagian, yaitu bagian keterangan dan bagian tabel data skripsi. Bagian keterangan digunakan untuk memberikan informasi penting kepada pengguna aplikasi mengenai hal-hal yang perlu diperhatikan saat melakukan pencarian. Pada bagian tabel data skripsi digunakan

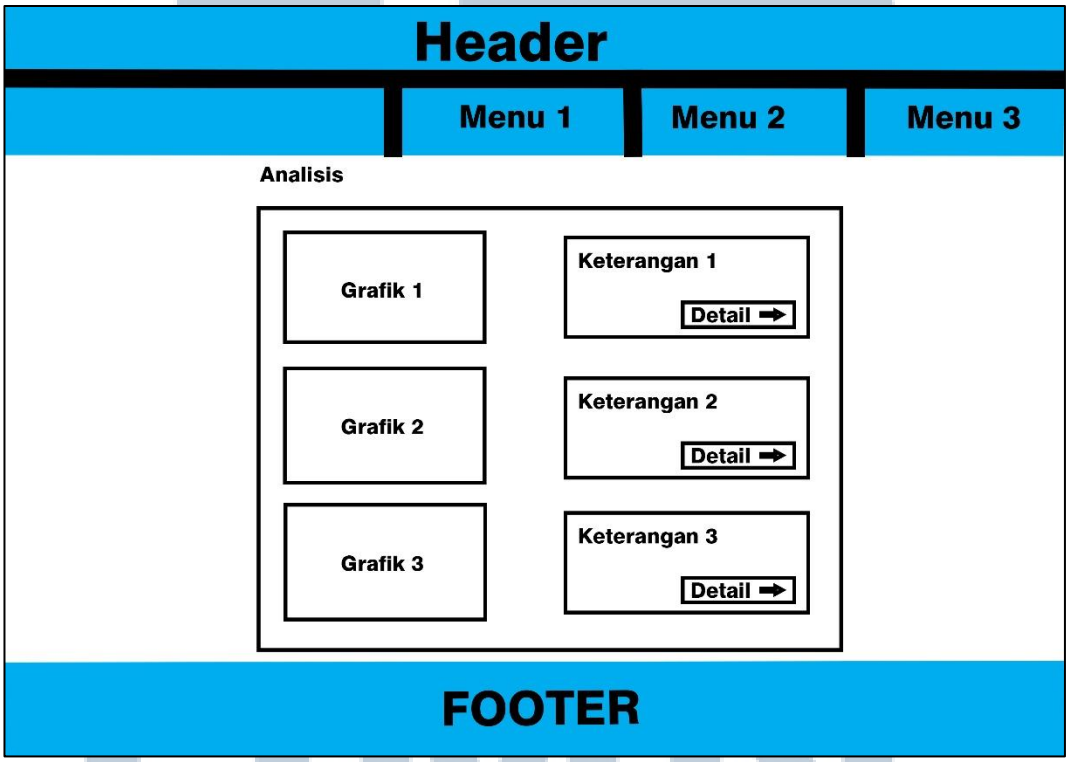
untuk menampilkan semua data skripsi mahasiswa yang terdapat di dalam database. Hampir sama dengan halaman *all data* pada *back-end*, tetapi yang membedakan adalah tidak disediakan tombol *delete* yang berfungsi untuk menghapus data dari database. Pengguna hanya dapat melihat detail data dan mengunduh abstrak atau bab 1 dari skripsi mahasiswa yang bersangkutan. Pada halaman ini, pengguna dapat melakukan pencarian berdasarkan nama penulis, judul, no label, tahun terbit, dan jurusan. Untuk memudahkan navigasi diberikan juga tombol *link* untuk berpindah ke halaman analisis atau halaman *search* pada *body* bagian atas.



Gambar 3.34 Perancangan Antarmuka Front-End Halaman Analisis

Gambar 3.34 adalah perancangan antarmuka *front-end* halaman analisis. Halaman ini berfungsi sebagai tahapan *input* yang dilakukan oleh pengguna dan menjadi salah satu proses awal *clustering*. Perancangan ini akan menggunakan pendekatan pengisian *step by step* dengan maksud agar pengguna tidak melewati

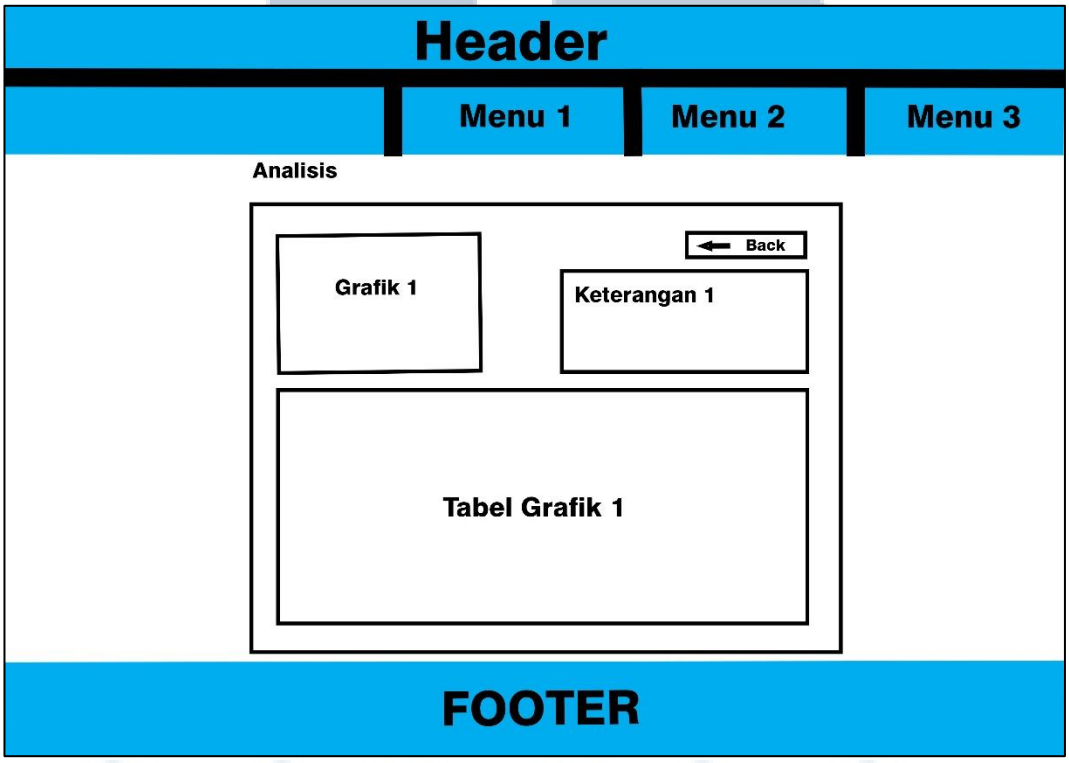
salah satu tahap penting untuk memenuhi proses *clustering*. Terdapat sebuah tombol *submit* yang digunakan untuk melakukan proses analisis setelah semua tahap talah dilalui oleh pengguna. Sama seperti halaman *search* disediakan juga sebuah tombol *link* untuk berpindah ke halaman analisis atau *search* pada *body* bagian atas.



Gambar 3.35 Perancangan Antarmuka Front-End Halaman Hasil Analisis

Gambar 3.35 adalah perancangan antarmuka *front-end* halaman hasil analisis. Halaman ini berfungsi untuk menampilkan hasil dari proses analisis yang telah dilakukan pada halaman sebelumnya. Halaman ini dibagi menjadi 2 bagian sisi. Sisi kiri halaman akan menampilkan grafik (*bar chart*) yang mendeskripsikan hasil dari analisis, sedangakan pada sisi kanan akan menampilkan keterangan untuk memperjelas grafik pada sisi kiri dan diberikan juga sebuah tombol *detail* untuk berpindah ke halaman detail analisis. Informasi hasil analisis pada halaman ini akan

dibagi menjadi 3 bagian berdasarkan semua kata pada setiap *cluster*, 5 kata dengan frekuensi kemunculan terbanyak dari semua *cluster*, dan 3 kata dengan frekuensi kemunculan terbanyak dari setiap *cluster*.



Gambar 3.36 Perancangan Antarmuka Front-End Halaman Detail Analisis

Gambar 3.36 adalah perancangan antarmuka *front-end* halaman detail analisis. Halaman ini berfungsi untuk menampilkan informasi yang lebih detail sesuai dengan pembagian yang telah disebutkan pada penjelasan halaman hasil analisis. Terdapat juga grafik dan keterangan padi sisi bagian atas halaman. Pada bagian sisi bawah diberikan tabel untuk menampikan data skripsi mahasiswa yang berhubungan dengan hasil grafik. Tabel ini nantinya akan sama dengan tabel yang terdapat pada halaman *search*, yaitu pengguna dapat melihat detail dan mengunduh abstrak atau bab 1 dari skripsi mahasiswa yang bersangkutan.