Hak cipta dan penggunaan kembali:

Lisensi ini mengizinkan setiap orang untuk menggubah, memperbaiki, dan membuat ciptaan turunan bukan untuk kepentingan komersial, selama anda mencantumkan nama penulis dan melisensikan ciptaan turunan dengan syarat yang serupa dengan ciptaan asli.

Copyright and reuse:

This license lets you remix, tweak, and build upon work non-commercially, as long as you credit the origin creator and license it on your new creations under the identical terms.

Team project ©2017
Dony Pratidana S. Hum | Bima Agus Setyawan S. IIP

BAB III

METODE DAN PERANCANGAN

3.1 Metode Penelitian

Metode penelitian yang digunakan dalam penelitian ini adalah sebagai berikut.

1. Studi Pustaka

   Melakukan studi terhadap sumber-sumber seperti jurnal, laporan, aplikasi, atau artikel yang telah dipublikasikan oleh peneliti lain yang bekaitan dengan *diff*, *longest common subsequence problem*, *software configuration management*, algoritma Myers Diff, dan lain-lain yang berguna untuk membangun aplikasi *data comparison* dalam penelitian ini.

2. Perancangan

   Melakukan perancangan aplikasi *data comparison* dengan mengimplementasikan algoritma Myers Diff ke dalam kode program berbasis web sesuai dengan tujuan dan batasan yang sudah ditentukan sebelumnya.

3. Implementasi

   Mengimplementasikan algoritma dan rancangan yang sudah dibuat sebelumnya ke dalam kode program untuk membangun aplikasi *data comparison*.

4. Survei

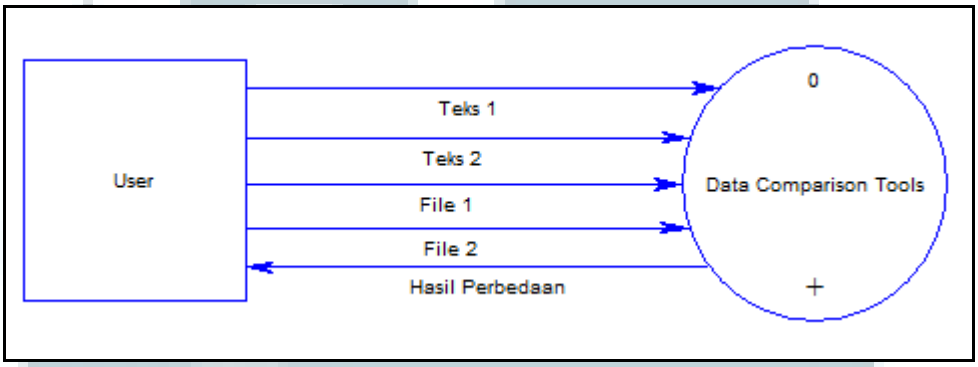
   Melakukan survei untuk mengetahui format keluaran seperti apa yang paling nyaman untuk dilihat dan dibaca, sehingga mempercepat pendeteksian perubahan yang terjadi terhadap dua buah *file* kode sumber.

5. Penulisan Laporan

Menulis laporan penelitian mulai dari proses pembuatan aplikasi *data comparison* sampai hasil dan kesimpulan dari penelitian yang dilakukan.

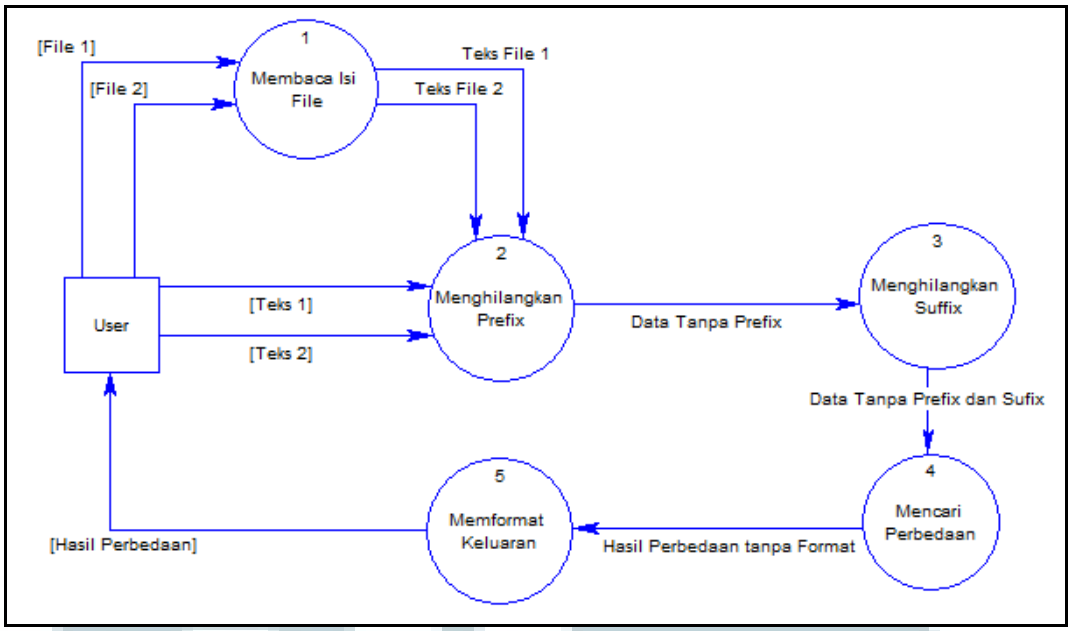
3.2 Perancangan Aplikasi

3.2.1 Data Flow Diagram



Gambar 3.1 Diagram Konteks Aplikasi

Gambar 3.1 merupakan *Data Flow Diagram* level 0 atau diagram konteks untuk aplikasi *Data Comparison Tools*. *User* dapat memilih dua buah cara untuk memeriksa perbedaan yang terjadi diantara dua buah data. Cara pertama adalah dengan memasukan data tersebut ke dalam aplikasi dalam bentuk teks, dengan cara menyalin dan menempelkannya ke dalam kolom input yang telah disediakan di dalam aplikasi. Cara kedua adalah dengan mengunggah data yang hendak dibandingkan dalam bentuk *file*, dengan aturan *file* yang diunggah haruslah sebuah *file* berbasis teks (*text based file*). Setelah *user* memasukan data baik dalam bentuk teks ataupun *file*, *user* dapat menekan tombol yang ada di dalam aplikasi untuk memulai proses perbandingan kedua data tersebut. Setelah proses perbandingan selesai, maka *user* akan disuguhkan hasil perbedaan kedua data yang dimasukan tadi.



Gambar 3.2 *Data Flow Diagram* Sub-proses dari Proses *Data Comparison Tools*

Gambar 3.2 merupakan *Data Flow Diagram* (DFD) level 1, yang sekaligus merupakan sub-proses dari aplikasi *Data Comparison Tools* yang berada pada level 0 atau diagram konteks. Terdapat lima sub-proses dari proses aplikasi utama yang dibangun.

Sub-proses yang pertama merupakan proses Membaca Isi *File*. Proses ini akan menerima input berupa dua buah *file* berbasiskan teks (*text-based file*), yang berasal dari *user*. Jika *user* memilih untuk memasukan data yang ingin dikomparasi dengan cara mengunggah dua buah *file*, maka kedua *file* tersebut akan diterima oleh proses ini dan akan dilakukan pembacaan isi dari *file* tersebut. Pembacaan isi *file* yang diberikan oleh *user* dilakukan satu-persatu terhadap masing-masing data secara terpisah. Sub-proses ini juga akan mendeteksi apakah *file* yang diberikan oleh *user* merupakan *file* berbasis teks atau bukan. Sub-proses ini menghasilkan data yang berada di dalam isi *file* dan akan melanjutkan data tersebut menuju sub-proses selanjutnya.

Sub-proses kedua memiliki nama Menghilangkan Prefix. Proses ini menerima input dua buah data dalam bentuk teks. Proses ini akan melakukan pencocokan kedua teks tersebut mulai dari depan, sampai menemukan karakter yang tidak lagi sesuai diantara kedua teks tersebut. Proses pencocokan tersebut dilakukan secara sekuensial mulai dari karakter pertama pada kedua buah data sampai menemukan perbedaan karakter atau sampai tidak ada lagi karakter yang dapat dicocokan. Proses ini akan menghasilkan data yang sudah tidak memiliki kesamaan data di awal teks (*prefix*) lagi dan akan melanjutkan data tanpa *prefix* tersebut ke sub-proses selanjutnya.

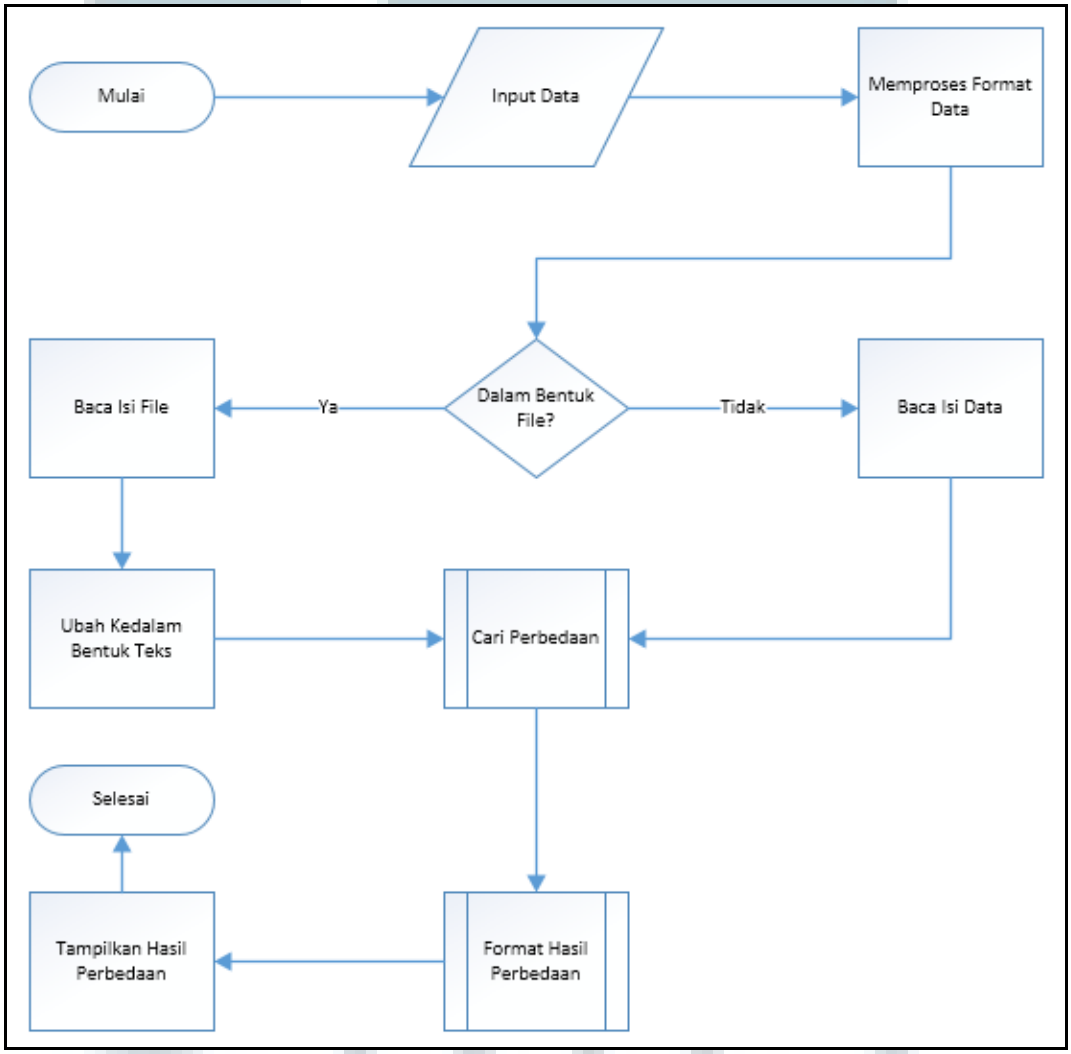
Setelah *prefix* dari kedua teks masukan sudah dieliminasi, maka sub-proses berikutnya bertanggung jawab untuk menghilangkan *suffix*, atau kesamaan data di akhir kedua teks. Sub-proses ketiga diberi nama Menghilangkan Suffix. Sesuai dengan namanya, proses ini bertujuan untuk menghilangkan *suffix* yang ada di dalam kedua buah teks masukan. Proses ini akan mencari kecocokan kedua teks mulai dari belakang, dan berhenti saat menemukan ketidak-cocokan karakter dari kedua teks. Proses pencocokan dilakukan dengan cara sekuensial mulai dari akhir karakter kedua buah data, sampai menemukan ketidak-cocokan karakter atau sampai tidak ada lagi karakter yang dapat dicocokan. Proses ini akan menghasilkan data yang sudah tidak memiliki kesamaan data di akhir teks (*suffix*) lagi, dan karena data masukan pada proses ini sudah tidak memiliki kesamaan *prefix* lagi, maka data keluaran dari proses ini adalah data tanpa *prefix* dan *suffix*.Tujuan dari penghilangkan *prefix* dan *suffix* ini adalah untuk mempercepat kerja aplikasi dengan mengurangi pekerjaan sia-sia yang mungkin dapat dilakukan pada sub-proses selanjutnya.

Sub-proses keempat adalah proses inti dari aplikasi *Data Comparison Tools* ini. Sub-proses yang diberinama Mencari Perbedaan ini bertujuan untuk mencari dan menandai perbedaan yang ada di dalam dua buah masukan (yang sudah tidak memiliki kesamaan *prefix* dan *suffix*) yang diberikan oleh *user*. Perbedaan yang dapat terjadi didefinisikan kedalam dua jenis. Jenis perbedaan pertama adalah penambahan. Penambahan adalah jenis perbedaan yang ditandai kepada karakter-karakter yang tidak ada di dalam teks pertama, namun ditemui pada teks kedua. Karakter-karakter tersebut berarti merupakan hasil penambahan dari teks pertama, sehingga dapat menjadi teks kedua. Jenis perbedaan kedua adalah pengurangan atau menghapusan. Kebalikan dari penambahan, pengurangan atau penghapusan merupakan jenis perbedaan yang diberikan kepada karakter-karakter yang dapat ditemui di dalam karakter pertama, namun dapat ditemui pada teks kedua. Karakter-karakter tersebut berarti merupakan hasil penghapusan dari karakter-karakter yang ada di dalam teks pertama sehingga dapat menjadi teks kedua. Hasil dari sub-proses keempat ini adalah hasil perbedaan dari kedua buah data (atau *file*) yang diberikan. Namun, hasil perbedaan ini merupakan hasil perbedaan yang belum memiliki format yang baik dan masih sangat sulit untuk dimengerti jika disuguhkan langsung kepada *user*.

Sub-proses kelima merupakan sub-proses terakhir dari keseluruhan rangkaian sub-proses yang membentuk proses utama *Data Comparison Tools*. Proses yang memiliki nama Memformat Keluaran ini bertanggung jawab penuh terhadap data hasil perbedaan tanpa format yang diberikan dari sub-proses sebelumnya, dan melakukan *formating* terhadap data tersebut agar memiliki hasil akhir yang indah dan mudah dimengerti. Proses ini menerima hasil perbedaan

yang belum memiliki format, kemudian melakukan iterasi terhadap data tersebut dari awal hingga akhir. Dalam setiap iterasi yang dilakukan, proses ini akan menerjemahkan data tersebut satu persatu dan menjadikannya sebuah data yang jika disatukan, akan menghasilkan sebuah informasi yang mudah dimengerti *user*. Hasil dari proses ini adalah hasil perbedaan, dan akan langsung diteruskan kepada *user* dan dapat langsung dilihat.

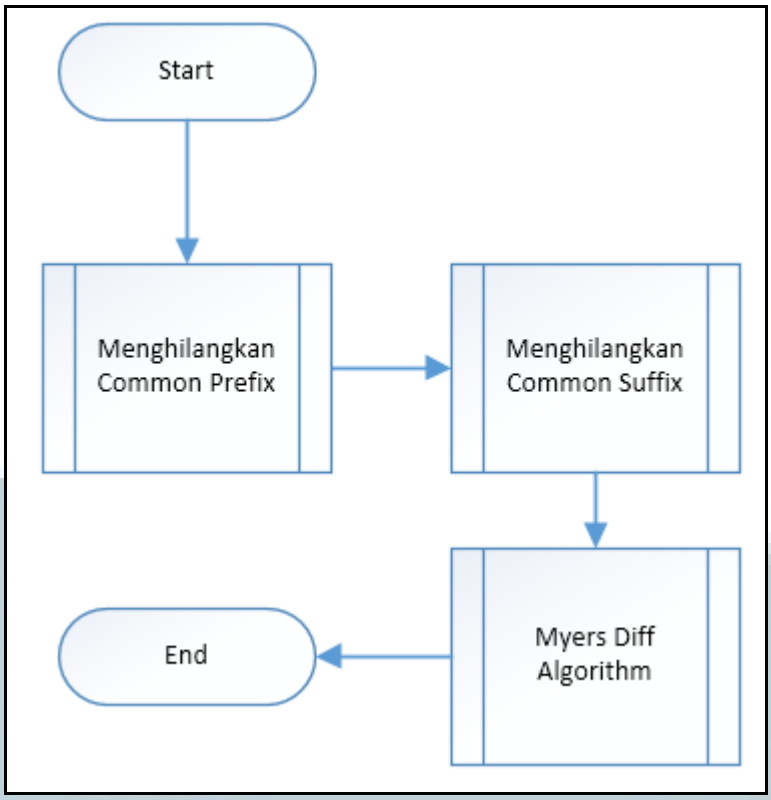
3.2.2 Flow Chart Aplikasi



Gambar 3.3 *Flow Chart* Aplikasi *Data Comparison Tools*

Gambar 3.3 merupakan *flow chart* umum untuk aplikasi *data comparison tools* yang dibangun. Proses dimulai dengan *user* memasukan dua buah data yang

ingin dibandingkan. Seperti yang sudah dijelaskan pada bagian 3.2.1, terdapat dua jenis format masukan dari data yang akan dibandingkan. Format masukan pertama adalah data dengan format teks, dan yang kedua adalah data dengan format *file* yang masih berbasis teks (*text-based file*). Setelah *user* memasukan data dan menekan tombol yang tersedia, maka aplikasi akan memeriksa format masukan yang dipilih oleh *user*, apakah format teks, atau *file*. Jika masukan berformat *file*, maka aplikasi akan membaca isi dari kedua *file* yang diberikan oleh *user*, dan memproses data yang ada didalamnya. Sedangkan, jika format masukan adalah teks, maka data dari kedua *text-box* yang ada akan langsung di proses dan dibandingkan. Proses selanjutnya adalah proses utama dari aplikasi yang dibangun. Proses ini bertujuan untuk mencari perbedaan dari kedua buah data yang diberikan dengan menggunakan algoritma Myers Diff. Proses ini akan mencari perbedaan seminimal mungkin antara kedua buah data yang diberikan. Untuk detail dari proses utama ini akan dijelaskan pada bagian selanjutnya. Setelah selesai dari proses Cari Perbedaan, maka akan masuk ke dalam proses selanjutnya yaitu melakukan format hasil perbedaan yang ditemukan. Masukan dari proses ini adalah hasil perbedaan dari proses sebelumnya. Hasil perbedaan yang diberikan kepada proses ini akan diformat sesuai dengan format yang telah ditentukan sebelumnya, kemudian hasil format keluaran tersebut akan dilanjutkan ke proses selanjutnya. Proses terakhir adalah proses untuk menampilkan hasil perbedaan yang sudah diformat pada proses sebelumnya. Proses ini akan menampilkan hasil perbedaan dalam bentuk yang mudah dimengerti oleh *user*. Selain itu, hasil keluaran yang diberikan akan memiliki tiga jenis format yang berbeda.

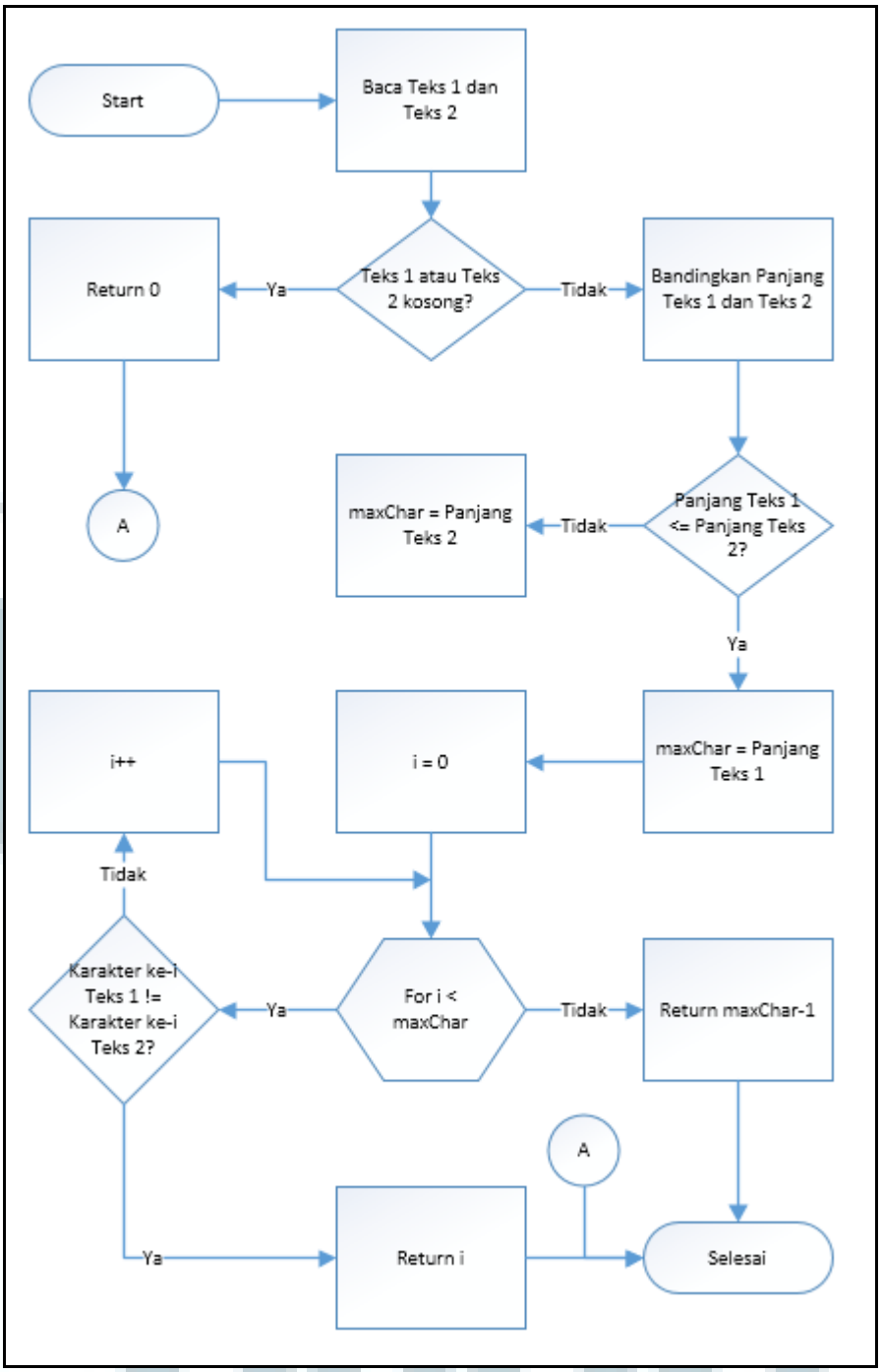


Gambar 3.4 *Flow Chart* dari Proses Cari Perbedaan

Gambar 3.4 merupakan detail *flow chart* untuk proses Cari Perbedaan yang ada pada gambar 3.3. Proses dimulai dengan melakukan eliminasi terhadap *common prefix*. *Common prefix* merupakan kesamaan *string* yang ada di awal teks 1 dan juga teks 2. Setelah menghilangkan perbedaan *string* yang ada di awal kedua teks, maka data yang dihasilkan tidak lagi akan memiliki kesamaan pada bagian awalnya. Setelah menghilangkan *common prefix*, maka proses dilanjutkan dengan menghilangkan *common suffix*. *Common suffix* merupakan kesamaan *string* yang ada di akhir kedua teks. Kedua proses awal ini (menghilangkan *common prefix* dan menghilangkan *common suffix*) merupakan langkah yang diambil untuk mempercepat kinerja aplikasi. Setelah data yang dihasilkan sudah lagi tidak memiliki *common prefix* dan juga *suffix*, maka data tersebut akan dilanjutkan ke dalam proses ketiga yaitu *Myers Diff Algorithm*. Proses ini

merupakan implementasi langsung algoritma Myers Diff. Dengan memasukan kedua data (teks) ke dalam proses ini, maka akan menghasilkan hasil perbedaan yang sudah dikalkulasi dengan menggunakan algoritma Myers Diff. Setelah mendapatkan hasil perbedaan, maka proses akan selesai dan berlanjut ke pada proses selanjutnya, yaitu melakukan format keluaran.

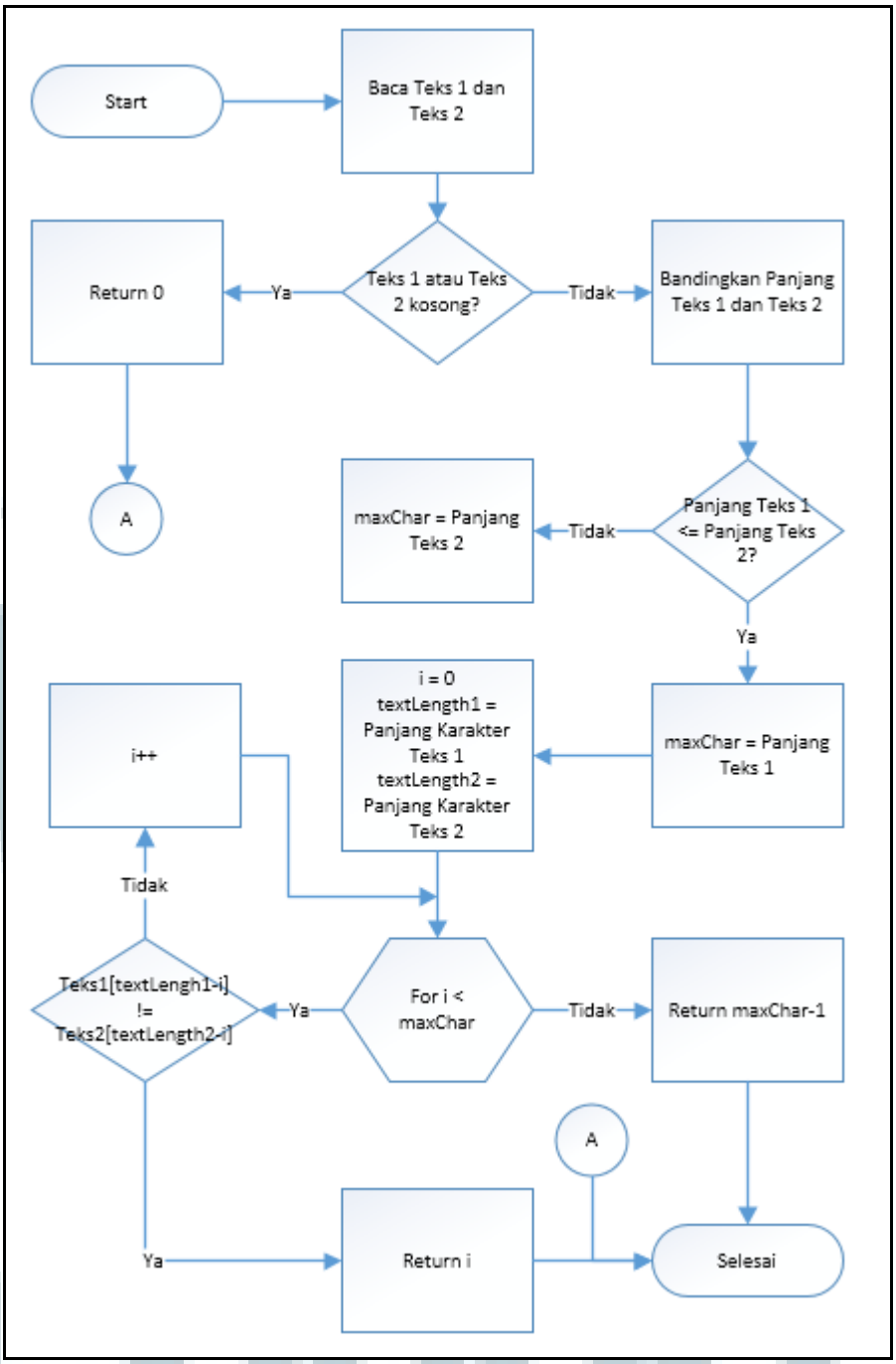
UMN



Gambar 3.5 *Flow Chart* Sub-proses Menghilangkan *Common Prefix*

Gambar 3.5 menunjukan *flow chart* untuk sub-proses Menghilangkan Common Prefix. Proses dimulai dengan mengecek apakah teks 1 atau teks 2 kosong, jika teks 1 dan/atau teks 2 kosong, maka proses akan langsung mengembalikan nilai 0, yang artinya tidak memiliki *common prefix* sama sekali.

Jika teks 1 dan teks 2 tidak kosong, maka proses akan berlanjut dengan menentukan maxChar. Proses akan mengecek apakah panjang karakter yang ada pada teks 1 lebih sedikit atau sama dengan panjang karakter yang ada pada teks 2, jika ya, maka maxChar akan diisi dengan panjang karakter teks 1, jika tidak, maka maxChar akan diisi dengan panjang karakter yang ada pada teks 2. Proses kemudian akan melakukan inisialisasi i dan diisi dengan nilai 0. Setelah itu, maka akan dilakukan pengulangan yang dimulai dengan nilai i sama dengan 0, sampai i lebih besar dari maxChar. Untuk setiap pengulangan, akan dilakukan pengecekan apakah karakter ke-i pada teks 1 dan teks 2 sama atau tidak, jika tidak sama, maka proses akan langsung mengembalikan nilai i dan proses akan selesai, sedangkan jika karakter pada teks 1 dan teks 2 sama, maka proses akan menambahkan nilai i sebesar 1, dan mengulang proses perulangan. Jika sampai proses perulangan selesai tidak ditemukan karakter yang berbeda, maka proses akan mengembalikan nilai maxChar-1, dan proses akan berakhir.

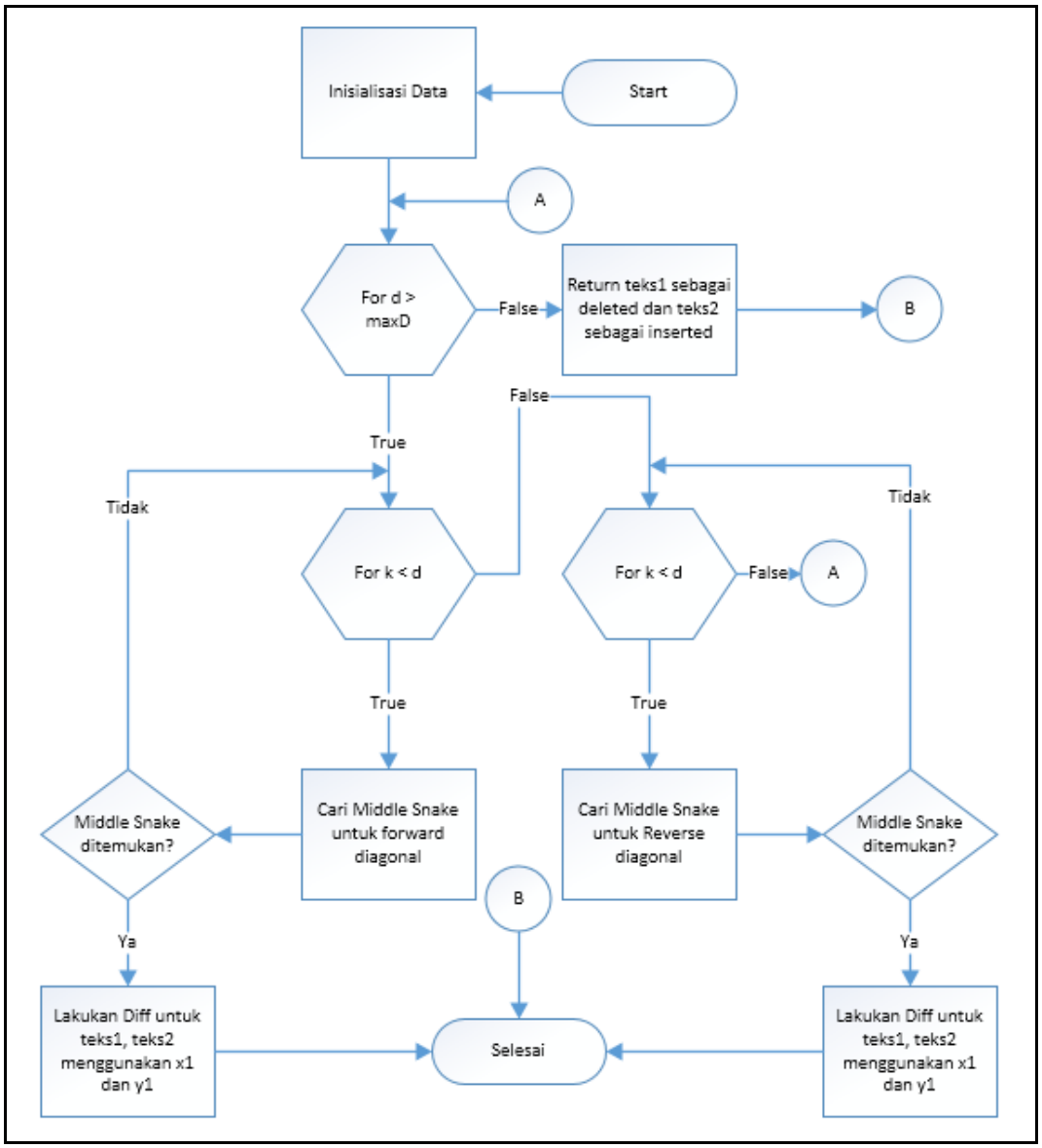


Gambar 3.6 *Flow Chart* Sub-proses Menghilangkan *Common Suffix*

Gambar 3.6 menunjukan *flow chart* untuk sub-proses Menghilangkan Common Suffix. Proses dimulai dengan mengecek apakah teks 1 atau teks 2 kosong, jika teks 1 dan/atau teks 2 kosong, maka proses akan langsung mengembalikan nilai 0, yang artinya tidak memiliki *common suffix* sama sekali.

Jika teks 1 dan teks 2 tidak kosong, maka proses akan berlanjut dengan menentukan maxChar. Proses akan mengecek apakah panjang karakter yang ada pada teks 1 lebih sedikit atau sama dengan panjang karakter yang ada pada teks 2, jika ya, maka maxChar akan diisi dengan panjang karakter teks 1, jika tidak, maka maxChar akan diisi dengan panjang karakter yang ada pada teks 2. Proses kemudian akan melakukan inisialisasi i dan diisi dengan nilai 0, textLength1 diisi dengan panjang karakter yang ada pada teks 1, dan textLength2 diisi dengan panjang karakter yang ada pada teks 2. Setelah itu, maka akan dilakukan pengulangan yang dimulai dengan nilai i sama dengan 0, sampai i lebih besar dari maxChar. Untuk setiap pengulangan, akan dilakukan pengecekan apakah karakter ke-(textLength1-i) pada teks 1 dan karakter ke-(textLength2-i) pada teks 2 sama atau tidak, jika tidak sama, maka proses akan langsung mengembalikan nilai i dan proses akan selesai, sedangkan jika karakter pada teks 1 dan teks 2 sama, maka proses akan menambahkan nilai i sebesar 1, dan mengulang proses perulangan. Jika sampai proses perulangan selesai tidak ditemukan karakter yang berbeda, maka proses akan mengembalikan nilai maxChar-1, dan proses akan berakhir.

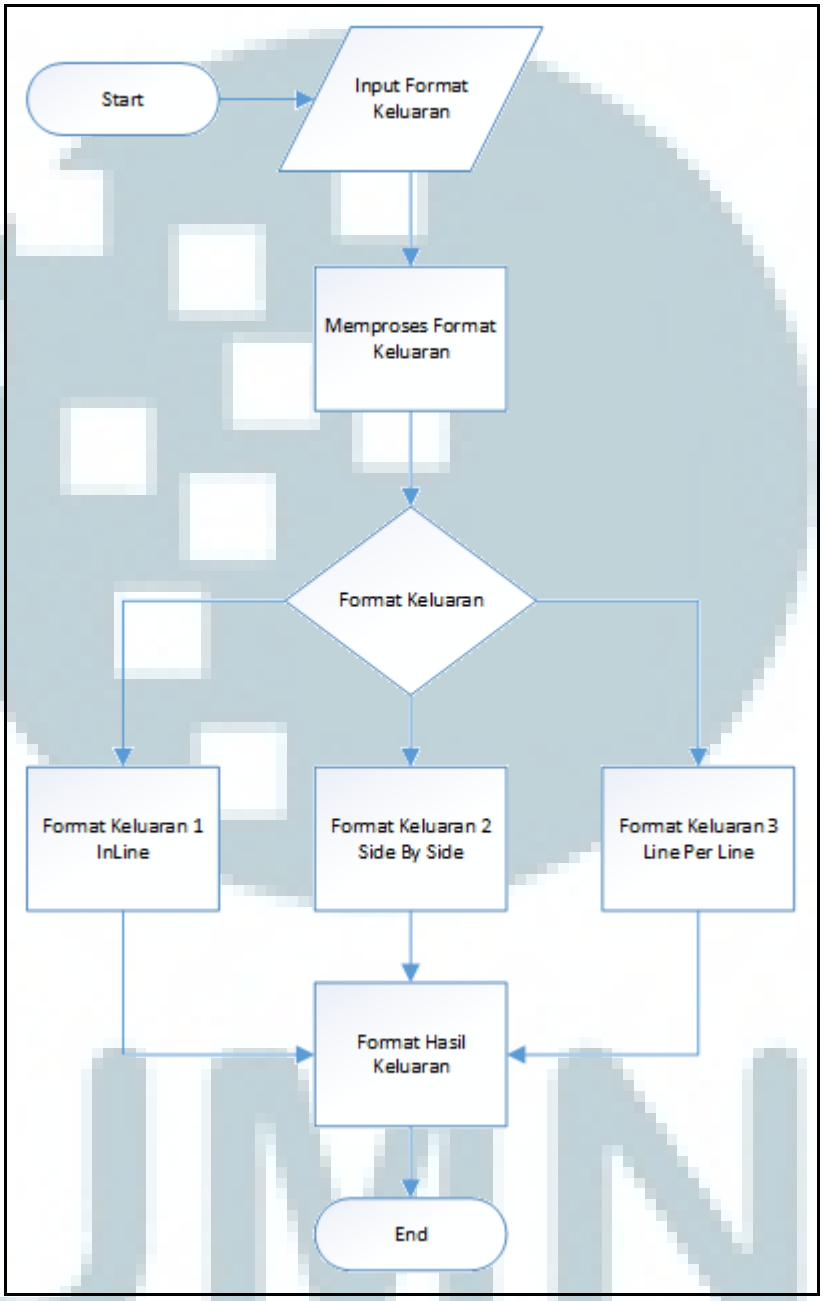
UMN



Gambar 3.7 *Flow Chart* Sub-proses Myers Diff Algorithm

Gambar 3.7 merupakan *flow chart* untuk sub-proses Myers Diff Algorithm. Proses dimulai dengan melakukan inisialisasi data atau variable yang diperlukan. Setelah itu, proses ini akan melakukan looping dimulai dari d bernilai 0, sampai dengan nilai maxD. Dalam setiap perulangan, akan dicari *middle snake* untuk *forward diagonal* dan juga dicari untuk *reverse diagonal*. *Forward diagonal* merupakan middle snake yang didapatkan dari awal koordinat (0,0)

sampai dengan akhir (M,N), sedangkan reverse diagonal sebaliknya, mulai dari titik akhir (M,N) sampai dengan titik awal (0,0).



Gambar 3.8 *Flow Chart* saat *user* memilih format keluaran yang tersedia

Gambar 3.8 merupakan alur saat seorang *user* ingin merubah format keluaran dari data hasil perbedaan yang sudah dibuat oleh aplikasi yang dibangun.

Proses ini merupakan detail dari proses Format Hasil Perbedaan yang ada di dalam gambar 3.3. Proses akan mengecek apakah *user* memilih untuk menampilkan perbedaan yang ditemukan ke dalam format keluaran yang pertama atau *inline*, format keluaran kedua atau *side by side* atau format keluaran ketiga *line per line*.

Proses memformat keluaran dilakukan dengan cara mengolah data hasil perbedaan yang diberikan dari proses sebelumnya yang belum memiliki format dan masih sulit dibaca. Proses format dilakukan secara terpisah untuk setiap format keluaran yang sudah ditentukan.

Format keluaran pertama atau *inline*, dilakukan iterasi terhadap data hasil keluaran yang belum diformat dan mengecek setiap elemen yang ada di dalam hasil keluaran yang belum diformat itu satu persatu. Jika jenis dari elemen tersebut merupakan penambahan, maka kata-kata yang ada di dalam elemen tersebut akan dibungkus dengan menggunakan tag <ins></ins>, jika elemen tersebut merupakan pengurangan, maka akan dibungkus dengan menggunakan tag <ins></ins> dan jika elemen tersebut adalah kata-kata yang sama, maka tidak akan dilakukan format apapun. Semua hasil perulangan tersebut akan disatukan menjadi sebuah kesatuan teks dan hasil teks tersebut diberikan kepada proses selanjutnya.

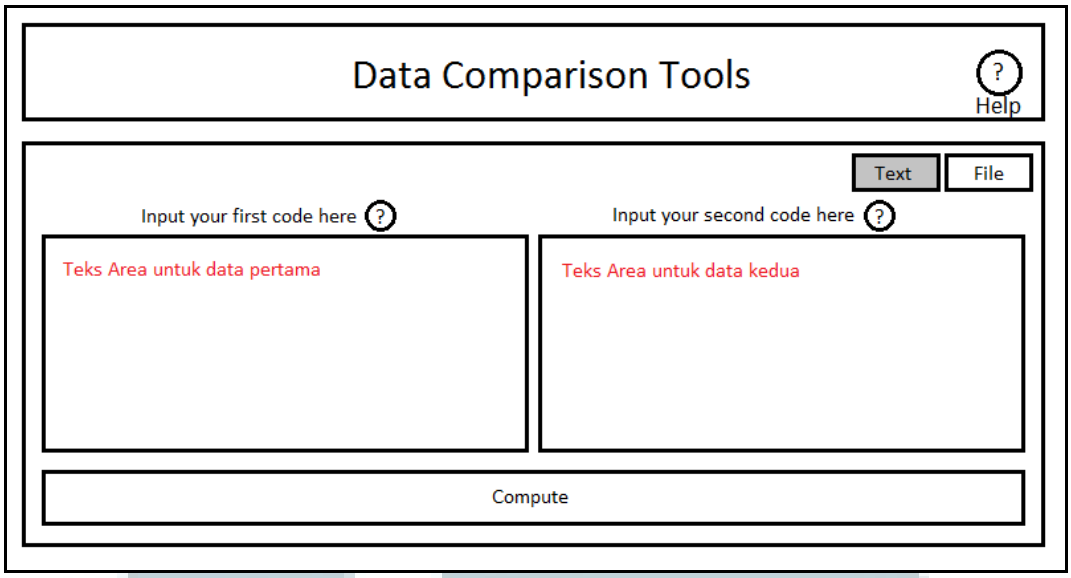
Format keluaran kedua atau *side by side*, disediakan dua buah teks yang berfungsi sebagai penampun untuk data 1 dan juga penampung untuk data 2. Dilakukan iterasi kepada semua elemen dalam hasil perbedaan yang belum diformat, kemudia dilakukan pengecekan untuk setiap elemen tersebut, jika elemen tersebut merupakan penambahan, maka kata-kata yang ada di dalam

elemen tersebut akan dibungkus dengan menggunakan tag <ins></ins> dan kata-kata yang sudah dibungkus itu ditambahkan ke dalam teks 2. Jika elemen yang dicek merupakan pengurangan, maka maka kata-kata yang ada di dalam elemen tersebut akan dibungkus dengan menggunakan tag <del></del> dan kata-kata yang sudah dibungkus itu ditambahkan ke dalam teks 1. Jika elemen yang dicek adalah elemen yang sama, maka kata-kata yang ada di dalam elemen tersebut akan ditambahkan ke dalam teks 1 dan juga teks 2.

Format keluatan ketiga atau *line per line*, mirip dengan proses yang dilakukan untuk melakukan format keluaran kedua. Hanya saja, untuk setiap elemen, akan dilakukan pengecekan kata-kata yang ada di dalam elemen tersebut. Jika terdapat karakter *new line* atau baris baru, maka akan ditambahkan juga karakter *new line* atau baris baru pada teks yang lainnya.

3.3 Perancangan Antar Muka Sistem

Berikut ini merupakan rancangan antar muka sistem untuk aplikasi *Data Comparison Tools* yang akan dibangun.

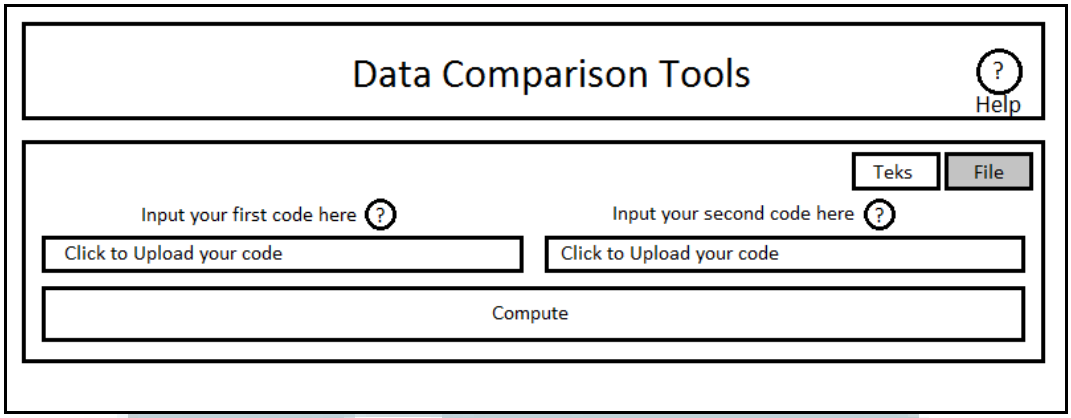


Gambar 3.9 Rancangan antar muka awal aplikasi

Gambar 3.9 merupakan rancangan antar muka awal saat seorang *user* baru pertama mengakses aplikasi *Data Comparison Tools*. Pada bagian awal ini, *user* akan disuguhkan dengan dua buah teks area tempat *user* memasukan data. Secara *default*, *user* akan diberikan pilihan untuk memasukan data dengan cara menempelkan kode sumber mereka, dalam bentuk teks kedalam teks area yang sudah diberikan. Setelah *user* selesai memasukan kedua data mereka ke dalam teks area yang diberikan, *user* dapat menekan tombol "*Compute*" yang ada di bawah teks area tersebut. Setelah *user* menekan tombol "*Compute*", maka aplikasi akan mencari perbedaan dari kedua data yang diberikan *user* dan lalu menampilkannya. Selain itu, *user* juga dapat menekan tombol "?" yang ada di pojok kanan atas halaman untuk mendapatkan bantuan yang dapat memandu *user* untuk memahami aplikasi *Data Comparison Tools* lebih dalam lagi.

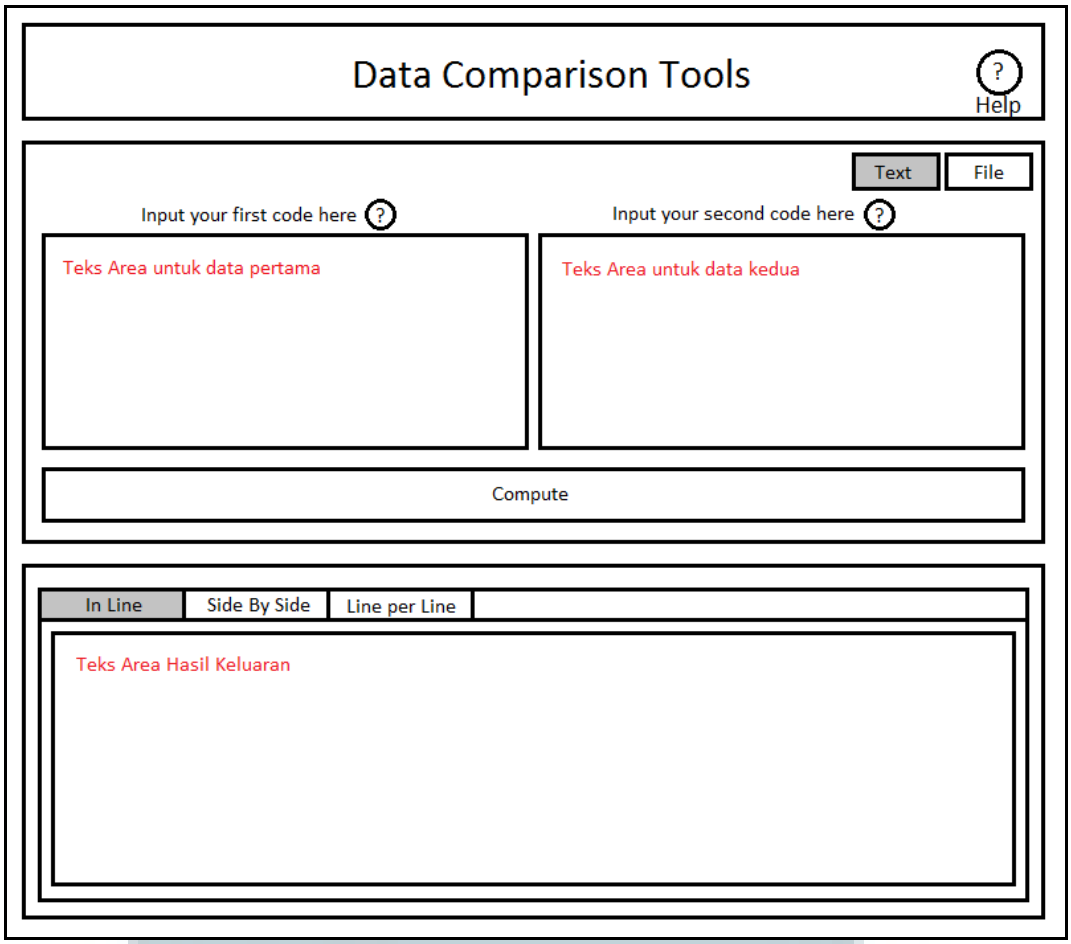
User juga dapat memilih cara memasukan data yang lain yaitu berupa *file*, dengan cara menekan tombol "*File*" yang ada di bawah tombol "?". Dengan

menekan tombol tersebut, maka *user* akan diberikan dua buah *field* untuk mengunggah dua buah *file* yang ingin dicari perbedaannya.



Gambar 3.10 Rancangan antar muka untuk meunggah *file* kode sumber

Gambar 3.10 merupakan rancangan antar muka yang ditampilkan untuk membantu *user* mengunggah kode sumber yang dimilikinya dalam bentuk *file*. Rancangan antar muka ini mirip dengan gambar 3.9, namun teks area untuk memasukan data diganti dengan *textfield* untuk mengunggah *file* kode sumber. Setelah *user* memilih kode sumber yang akan dicari perbedaannya, *user* dapat menekan tombol "*Compute*" dan aplikasi akan langsung memproses kedua *file* yang diberikan oleh *user*.



Gambar 3.11 Rancangan antar muka dengan hasil perbedaan

Gambar 3.11 merupakan rancangan antar muka setelah *user* selesai memasukan data dan menekan tombol "*Compute*" yang tersedia. Akan muncul sebuah bagian baru di bawah tombol "*Compute*" berada. Bagian ini akan menampilkan hasil perbedaan dari kedua buah data yang diberikan, baik data berupa teks maupun *file*. Bagian keluaran ini memiliki tiga buah tombol aktif yang dapat *user* pilih untuk mengganti format keluaran dari data hasil perbedaan yang telah dikomputasi oleh aplikasi *Data Comparison Tools*. Tombol-tombol ini akan merubah format keluaran dari data hasil perbedaan sesuai dengan keinginan *user*.

Tombol pertama adalah tombol “In Line”. Tombol ini akan menampilkan hasil perbedaan yang ditemukan kedalam sebuah kesatuan teks yang saling berkolaborasi, namun tetap menampilkan perbedaan (penambahan dan/atau pengurangan) dengan cara meng-*highlight*-nya dengan warna yang berbeda.

Tombol kedua adalah tombol “Side By Side”. Tombol ini akan menampilkan hasil perbedaan ke dalam dua buah teks area yang berbeda. Teks area pertama adalah teks area untuk menampilkan data pertama, beserta dengan pengurangan atau penghapusan yang terjadi terhadap data pertama tersebut, dan yang kedua merupakan teks area yang menampilkan data kedua, beserta dengan penambahan yang terjadi terhadap data kedua tersebut.

Tombol ketiga adalah tombol “Line Per Line”. Tombol ini akan menampilkan hasil perbedaan kedalam format yang mirip dengan format “Side By Side”, namun akan mengikuti jumlah baris yang ada diantara kedua data tersebut, sehingga akan lebih akurat untuk melacak perbedaan yang terjadi secara baris per baris.

UMN