WORDNET BROWSER UNTUK IPHONE / IPAD

Ir. Gunawan, M.Kom.
Arif Muliadi Chandra[1]

[1]de.zerocode@gmail.com

***Abstrak* - Pada tugas akhir ini dibuat sebuah ekstraktor Lexical Database untuk bahasa pemrograman Objective-C dan sebuah browser untuk menampilkan informasi hasil ekstraksi. Lexical database merupakan suatu sumber relasi semantik dalam suatu bahasa. Browser yang dibuat pada tugas akhir dikembangkan untuk iPhone dan iPad.**

Ekstraksi Lexical Database terdiri dari dua tahap. Tahap pertama adalah ekstraksi pada kelompok index files. Hasil ekstraksi pada tahap ini berupa synset offset dan relasi yang dimiliki oleh kata. Pada tahap ini synset offset pemilik relasi masih belum diketahui.Tahap kedua adalah ekstraksi pada kelompok data files. Hasil ekstraksi pada tahap ini berupa gloss, sinonim dan relasi yang dimiliki oleh synset. Pada tahap ini synset pemilik dari relasi telah diketahui dan apakah relasi tersebut adalah relasi semantik atau lexical.

Hasil dari ekstraksi akan ditampilkan melalui aplikasi WordNet Browser. Aplikasi tersebut akan menampilkan informasi mengenai kata yang dicari oleh pengguna. Apabila sebuah kata tidak terdapat pada Lexical Database maka aplikasi akan menampilkan kata-kata yang memiliki kemiripan penulisan dengan kata yang diinputkan oleh pengguna.

***Kata Kunci* – iOS, WordNet, WordNet Browser, Objective-C, Xcode, iPhone, iPad**

I. PENDAHULUAN

iPhone merupakan salah satu mobile device yang terkenal dan sukses di kalangan masyarakat.Untuk mendukung perkembangan aplikasi iPhone maka iPhone menyediakan SDK yang dapat digunakan oleh para programmer untuk membuat aplikasi pada iPhone atau iPad serta meletakkannya pada iTunes store supaya para pengguna iPhone atau iPad dapat mengunduhnya. Dengan adanya SDK ini maka munculah berbagai macam aplikasi pada iPhone dengan fungsi yang berbeda-beda. Misalnya permainan Grand Theft Auto, sarana pembelajaran bahasa inggris Miss Spell's Class, dan sebagainya.

Tetapi hingga saat ini masih belum banyak aplikasi kamus berbahasa Indonesia pada iTunes. Kamus-kamus yang tersediapun hanya menyediakan informasi yang sangat terbatas, yaitu hanya berupa definisi dan contoh penggunaan kata. Karena itu pada tugas akhir ini dibuat sebuah WordNet Browser yang menyediakan berbagai informasi yang dimiliki oleh sebuah kata, meliputi definisi kata, sinonim, hipernim, hiponim, holonim, meronim dan antonim.

WordNet yang digunakan pada tugas akhir ini adalah WordNet bahasa Indonesia yang telah dikembangkan oleh mahasiswa-mahasiswa yang lain. WordNet tersebut memiliki struktur standar WordNet yang diciptakan oleh Princeton University. Ekstraktor yang disediakan oleh Princeton University hanya dapat digunakan oleh bahasa

pemrograman .Net dan Java, karena itu untuk melakukan ekstraksi informasi dari WordNet maka dibuatlah sebuah ekstraktor Lexical Database dengan menggunakan bahasa pemrograman Objective-C. Aplikasi yang ditujukan untuk iPad dan iPhone haruslah dibuat dengan menggunakan suatu bahasa pemrograman khusus yaitu Objective-C sehingga ekstraktor yang dibuat pada tugas akhir ini akan dapat digunakan bagi pengembang yang memerlukan ekstraksi informasi pada WordNet.

Aplikasi yang dihasilkan pada tugas akhir ini dapat menghasilkan informasi yang dimiliki oleh kata yang diinputkan oleh pengguna. Selain itu disediakan juga fitur suggestion, apabila kata yang dicari tidak terdapat pada Lexical Database WordNet maka kata-kata yang memiliki kemiripan penulisan akan ditampilkan. Jika pengguna memiliki ketertarikan terhadap suatu kata maka kata tersebut akan dapat dicatat pada halaman bookmark. Dengan demikian aplikasi ini akan menjadi aplikasi yang sangat membantu dalam hal pencarian definisi kata

II. TEORI DASAR

Adapun teori-teori yang digunakan saat proses pembuatan adalah:

2.1 Xcode

Xcode adalah IDE (Integrated Development Environment) utama untuk Mac. Xcode tidak hanya dapat digunakan untuk mengembangkan aplikasi iPhone tetapi juga aplikasi untuk mac dan iPad. Didalam Xcode terdapat berbagai macam tool yang memiliki kegunaan berbeda-beda. iPhone SDK adalah suatu Software Development Kit yang dibuat oleh Apple untuk membuat aplikasi iPhone. iPhone SDK diperkenalkan oleh Apple ke publik pada tahun 2008. Agar memudahkan pengguna Mac, maka iPhone SDK ditanamkan secara langsung kedalam Xcode.

2.2 Objective-C

Objective-C adalah sebuah bahasa pemrograman yang dikembangkan dari bahasa pemrograman C. Pertama kali diciptakan pada tahun 1980 dan digunakan oleh perusahaan komputer yang dikembangkan oleh Steve Jobs yaitu NeXT. Beberapa tahun kemudian Apple mengakuisisi NeXT dan menggunakan Objective-C sebagai bahasa pemrograman utama untuk Macintosh. Pada tahun 2008 Objective-C juga digunakan sebagai bahasa pemrograman untuk iPhone.

Pada Objective-C sebuah class terbagi menjadi 2 buah file, yaitu file dengan ekstensi *.m dan *.h. File dengan ekstensi *.h dikenal dengan file Interface yang berisi deklarasi dari property dan procedure yang bersifat public.

2.3 WordNet

WordNet[8] adalah sebuah kamus yang didesain menyerupai sebuah thesaurus. Pada WordNet sebuah kata disimpan dalam bentuk sinonim set (synset). Synset adalah kata-kata yang memiliki bentuk yang berbeda tetapi memiliki arti kata yang sama. Tetapi berbeda dengan thesaurus, WordNet tidak hanya mengandung kumpulan kata-kata yang memiliki arti yang sama tetapi juga mengandung informasi lain berupa relasi dari kata-kata tersebut beserta gloss nya. Informasi tersebut tersimpan pada Lexical Database Files.

Lexical Database yang dimiliki oleh WordNet dituliskan dalam format ASCII. Hal ini dilakukan untuk memudahkan pengaksesan dalam berbagai bahasa pemrograman. File lexical database ini dapat dilihat dengan menggunakan text editor pada berbagai

OS, misalnya TextEdit pada Mac dan WordPad pada Windows. Kelemahan dari sistem penyimpanan dalam format ascii ini adalah file lexical database akan sangat mudah diubah oleh orang awam.

Lexical Database bahasa Indonesia yang digunakan pada tugas akhir ini mengandung 9 buah file dengan ukuran 11,4 MB. Berikut ini adalah tabel yang menunjukkan nama, fungsi serta ukuran dari masing-masing file:

Tabel 1
Daftar File Lexical Database

Nama File	Fungsi	Size (KB)
index.noun	Index file untuk kata noun	1,952
index.verb	Index file untuk kata verb	1,215
index.adj	Index file untuk kata adjective	54
index.adv	Index file untuk kata adverb	470
data.noun	Data file untuk kata noun	5,352
data.verb	Data file untuk kata verb	1,992
data.adj	Data file untuk kata adjective	646
data.adv	Data file untuk kata adverb	61
lexnames	Mencatat lexnames	1

Semua Lexical Database Files yang terdapat pada tabel 1 ditulis dengan menggunakan format ASCII. Untuk pemisah setiap kata pada index file maupun synset pada data file digunakan karakter enter (newline). Sedangkan untuk pemisan antar field yang ada digunakan karakter spasi.

Pencarian sebuah kata akan melalui dua tahap proses yaitu: pencarian kata pada setiap index file dan ekstraksi synset data pada data file. Pencarian kata pada setiap index file dilakukan untuk mengatasi kemungkinan sebuah kata dapat berada pada lebih dari 1 class kata.

2.3.1 Index Files

Index Files berisi daftar semua kata pada WordNet sesuai dengan class kata masing-masing (noun, verb, adj, adv). Kata-kata tersebut disusun urut sesuai dengan alphabet dan ditulis dengan huruf kecil. Setiap kata yang ada dipisahkan oleh karakter enter (newline) dan setiap field informasi dari sebuah kata dipisahkan oleh karakter spasi. Berikut adalah contoh kata pada index files:

abc n 3 2 #p %m 3 0 05321050 04785693 01743487

Setiap field pada kata mewakili sebuah informasi. Jenis field yang dimiliki oleh index files adalah sebagai berikut:

1: Lemma kata
2: Class kata
3: Jumlah sinonim yang dimiliki
4: Jumlah relasi yang dimiliki
5: Relasi-relasi yang dimiliki
6: Synset Offset sinonim

2.3.2 Data Files

Data Files berisi data-data yang dimiliki oleh sebuah synset. Dimulai dari sinonim, gloss, example dan relasi-relasi lexical yang dimiliki oleh synset tersebut. Sama seperti index files, setiap field informasi pada data filesakan dipisahkan oleh karakter spasi. Berikut adalah contoh kata pada data files:

00041707 03 n 01 lemusir 0 003 #p 04088869 n 0000 #p 04771649 n 0000 #p 05406744 n 0000 | bahu atau belikat dikatakan tentang daging kerbau, sapi, dan sebagainya

Setiap field pada kata mewakili sebuah informasi. Jenis field yang dimiliki oleh data files adalah sebagai berikut:

1: Synset offset
2: Lexnames
3: Class synset
4: Hexadecimal yang menunjukkan jumlah lemma
5: Lemma kata
6: Id Lemma
7: Bilangan integer yang menunjukkan jumlah relasi yang dimiliki
8: Jenis relasi
9: Synset offset pemilik relasi
10: Class synset offset pemilik relasi
11: Tipe relasi (semantik / leksikal)
12: Gloss synset

2.3.3 Lexnames

Lexnames merupakan hasil proses grinder terhadap lexicographer file. Lexnames mencatat informasi mengenai tipe class kata dan kelompok kata.

```
00      adj.all    3
01      adj.pert 3
02      adv. all 4
03      noun.Tops 1
04      noun.act   1
05      noun.animal      1
06      noun.artifact    1
07      noun.atribut     1
08      noun.body  1
09      noun.cognation   1
```

Gambar 1. 10 Baris Pertama Lexnames

Field-field pada lexnames juga dipisahkan oleh karakter spasi. Jenis field yang dimiliki oleh lexnames adalah sebagai berikut:

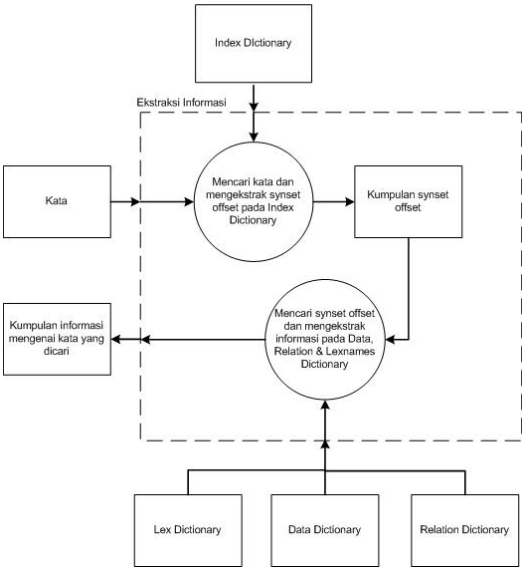
1: Nomor lexnames
2: Lexnames
3: Class kata

2.4 Levenshtein Distance

Levenshtein Distance adalah sebuah algoritma yang digunakan untuk menghitung perbedaan antara dua buah sequence. Pada Tugas Akhir ini levenshtein distance digunakan untuk menghitung biaya yang digunakan untuk merubah sebuah kata menjadi kata lain dengan cara menambahkan atau mengurangi huruf dan menukar posisi huruf yang ada pada sebuah kata.

III. EKSTRAKSI INFORMASI

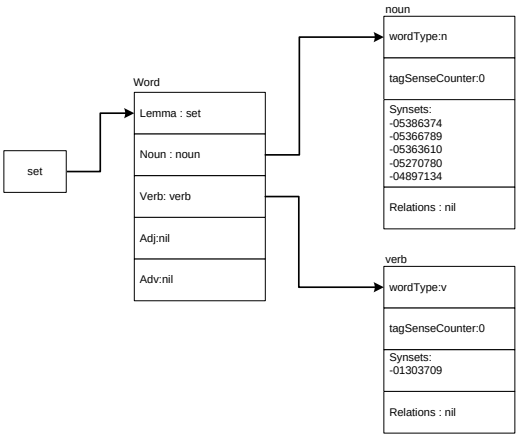
Kata yang dicari oleh penggunakan akan digunakan sebagai input untuk melakukan ekstraksi informasi pada Lexical Database. Sebelum proses ekstraksi informasi dilakukan index files, data files, dan lexnames akan terlebih dahulu diproses untuk disimpan kedalam kelompok dictionary. Ekstraksi akan dilakukan pada kelompok dictionary tersebut.



Gambar 2. Proses Ekstraksi Informasi

Pada gambar 2 ditunjukkan bahwa proses ekstraksi informasi mengalami dua tahap proses, tahap pertama adalah pencarian kata pada index dictionary

dan tahap kedua ekstraksi informasi pada data dictionary. Tahap pertama akan menghasilkan kumpulan synset yang dimiliki oleh dan dan pada tahap kedua akan dihasilkan informasi mengenai gloss, sinonim, dan relasi yang dimiliki oleh setiap synset tersebut.

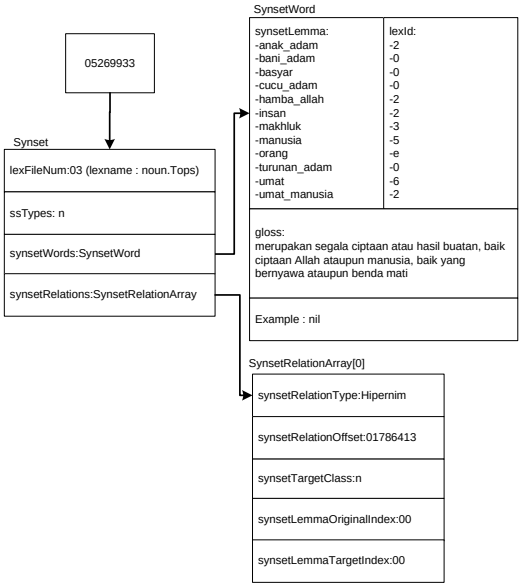


Gambar 3. Hasil Ekstraksi Tahap Pertama pada Kata Set

Tahap pertama proses ekstraksi akan melibatkan kelompok index dictionary dimana pencarian kata yang diinputkan oleh pengguna pada kelompok index dictionary dilkakukan. Pada tahap ini informasi yang didapatkan adalah lemma kata, serta synset offset yang dimiliki oleh kata tersebut pada setiap class kata, selain itu juga terdapat relasi yang dimiliki oleh kata tersebut. Relasi yang didapatkan masih belum mencakup synset offset pemilik relasi tersebut. Synset offset yang didapatkan akan diproses pada tahap selanjutnya bersama dengan ekstraksi relasi yang dimiliki.

Hasil ekstraksi ini akan disimpan dengan menggunakan sebuah struktur yang memungkinkan penyimpanan semua informasi hasil ekstraksi. Ilustrasi dari hasil ekstraksi ini ditunjukkan oleh gambar 3 dimana kata set akan digunakan sebagai input pencarian.

Tahap kedua proses ekstraksi akan melibatkan kelompok data dictionary, lex dictionary, dan relation dictionary dimana synset offset yang didapatkan dari proses tahap pertama akan diproses. Proses akan dimulai dari pencarian synset offset pada data dictionary, pada tahap ini akan didapatkan gloss dari synset, sinonim dan relasi yang dimiliki, serta nomor lexnames synset.



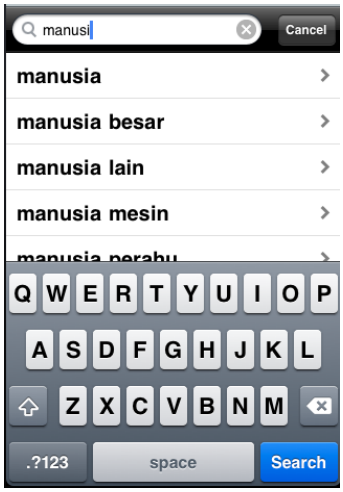
Gambar 4. Hasil Ekstraksi Tahap Kedua pada Synset 05269933

Relasi yang didapatkan pada tahap ini berupa simbol relasi, synset offset pemilik relasi dan tipe relasi (semantik / leksikal) agar dapat mengetahui jenis relasi synset maka simbol relasi yang didapatkan akan dicari pada relation dictionary. Begitu juga dengan lexnames, untuk mendapatkan lexnames synset maka nomor lexnames yang didapatkan akan dicari pada lex dictionary. Ilustrasi dari hasil ekstraksi ini ditunjukkan oleh gambar 4 dimana synset offset 05269933 akan digunakan sebagai input pencarian.

Hasil ekstraksi yang ditunjukkan pada gambar 4 menggunakan stuktur penyimpanan khusus untuk menampung hasil ekstraksi

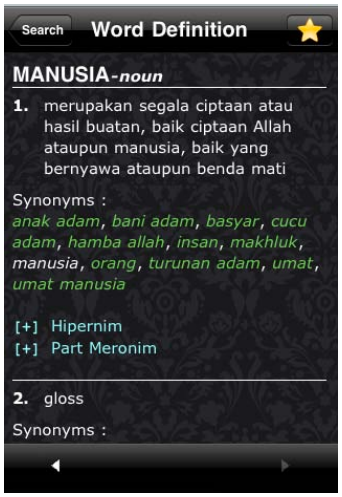
IV. APLIKASI YANG DIHASILKAN

Aplikasi yang dihasilkan pada tugas akhir ini akan meminta inputan dari pengguna berupa kata yang ingin dicari dan menampilkan informasi yang dimiliki oleh kata tersebut. Pada gambar 5 ditunjukkan tampilan aplikasi dimana pengguna menginputkan kata yang ingin dicari.Ketika pengguna mulai mengetikkan kata maka aplikasi akan menampilkan kata yang ingin mungkin ingin diketikkan oleh pengguna. Pencarian kata dapat dilakukan dengan melakukan penekanan tombol *search* yang berada pada *keyboard* atau dengan memilih kata yang ditampilkan oleh aplikasi.



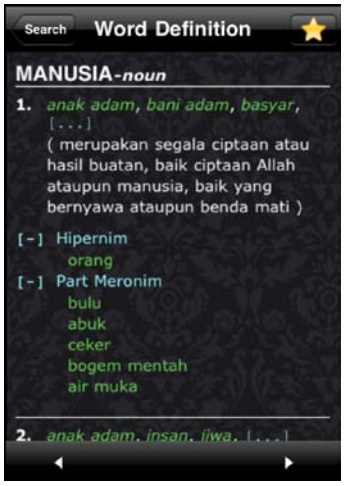
Gambar 5. Memasukkan Kata yang Ingin Dicari

Pemilihan kata dapat dilakukan dengan melakukan tap pada kata yang dicari. Apabila pengguna ingin menghilangkan keyboard dan melakukan *scroll* pada daftar kata maka pengguna dapat menekan tombol cancel yang muncul ketika pengguna mulai mengetikkan kata.Kata yang diinputkan oleh pengguna akan diproses oleh program, dimana ekstraksi informasi kata pada Lexical Database akan dilakukan. Setelah informasi kata didapatkan maka aplikasi akan menampilkan informasi tersebut kepada pengguna. Gambar 6 menunjukkan tampilan informasi yang dimiliki oleh kata manusia.



Gambar 6. Informasi Milik Kata Manusia

Sense yang dimiliki oleh kata akan dipisahkan dengan menggunakan sebuah garis putih. Dalam setiap sense akan terdapat nomor sense, gloss, sinonim kata dan relasi yang dimiliki oleh sense. Apabila sebuah kata terdapat pada lebih dari satu class kata maka informasi pada setiap class kata akan terletak pada sebuah layar.



Gambar 7. Relasi yang Terbuka

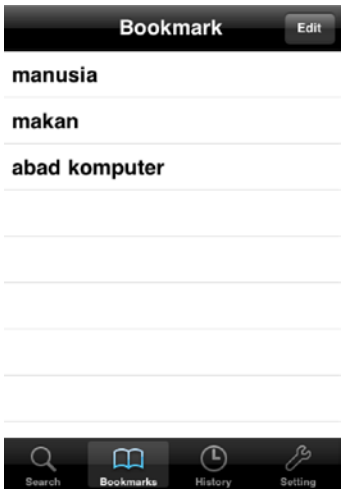
Pengguna dapat melihat informasi pada class kata lain dengan melakukan *swipe* pada layar. Apabila pengguna ingin melihat kata yang sebelumnya dicari maka terdapat tombol *back* pada tab bar, terdapat juga tombol *next* untuk melihat kata yang dicari yang lain. Kedua tombol tersebut tidak akan dapat ditekan ketika pengguna berada pada posisi kata pertama atau terakhir.

Pencarian kata dapat dilakukan dengan memasukkan kata pada halaman *search* atau dengan memilih kata berwarna hijau yang tampil. Ketika kata berwarna hijau dipilih maka secara otomatis pencarian terhadap kata tersebut akan dilakukan tanpa harus kembali ke halaman *search* dan mengetikkannya. Relasi dari kata akan berwarna biru, yang berarti apabila relasi dipilih maka kata pemilik relasi tersebut akan ditampilkan. Hal ini ditunjukkan oleh gambar 7 dimana relasi hipernim dan part meronim ditampilkan.

Pada gambar 7 ditunjukkan susunan informasi yang berbeda dengan gambar 6. Aplikasi menyediakan dua buah tampilan yang berbeda. Secara *default* susunan pada gambar 6 digunakan. Susunan ini mengikuti susunan kamus conventional dan ditujukan bagi pengguna yang tidak terbiasa menggunakan WordNet. Susunan kedua mengikuti susunan Princeton WordNet. Susunan ini ditujukan bagi para pengguna yang sudah terbiasa menggunakan WordNet. Apabila pengguna ingin melakukan *bookmark* pada kata maka pengguna dapat menekan tombol dengan simbol bintang yang terletak dikanan atas. Ketika tombol ini ditekan maka akan muncul konfirmasi apakah *bookmark* akan dilakukan. Pada gambar 8 ditunjukkan konfirmasi kepada pengguna.

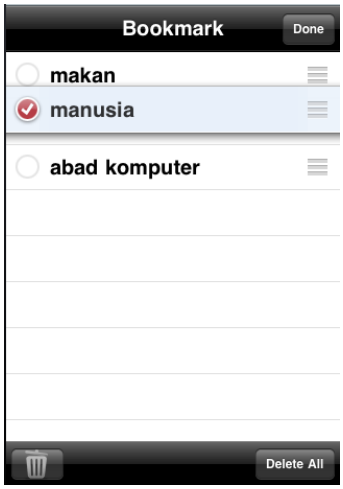
Gambar 8. Menambahkan Bookmark

Apabila sebuah kata sudah di*bookmark* maka penekanan tombol akan memunculkan konfirmasi apakah bookmark akan dihapus. Pengguna dapat kembali ke halaman *search* dengan menekan tombol *back* yang terletak di kiri atas. Apabila pengguna berasal dari halaman *search* maka ketika tombol ditekan pengguna akan dibawah ke halaman *search* begitu juga bila pengguna berasal dari halaman *bookmark* atau *history*. Text pada tombol tersebut akan berbeda-beda, menyesuaikan dengan halaman terakhir pengguna berada. Kata-kata yang telah di*bookmark* oleh pengguna dapat dilihat pada halaman *bookmark*.



Gambar 9. Halaman Bookmark

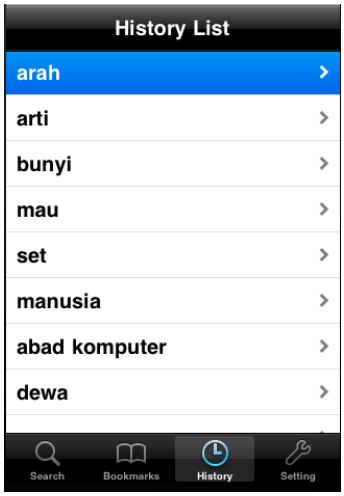
Gambar 9 menunjukkan tampilan dari halaman *bookmark* dimana terdapat tiga kata yang telah di*bookmark* oleh pengguna yaitu manusia, makan dan abad komputer.Pada halaman bookmark terdapat dua buah mode, mode pertama adalah mode *view* yang ditunjukkan oleh gambar 9. Pada mode *view* apabila sebuah kata dipilih oleh pengguna maka kata tersebut akan diproses dan pengguna akan dibawa menuju halaman result untuk melihat informasi milik kata tersebut.



Gambar 10. Mode Edit Halaman Bookmark

Mode kedua adalah mode edit dimana pengguna dapat memindahkan posisi dari kata yang terdapat pada halaman *bookmark* atau pun menghapus satu atau lebih kata. Untuk memasuki mode *edit* maka pengguna cukup menekan tombol *edit* yang terletak pada kanan atas. Tampilan dari mode edit ditunjukkan oleh gambar 10.

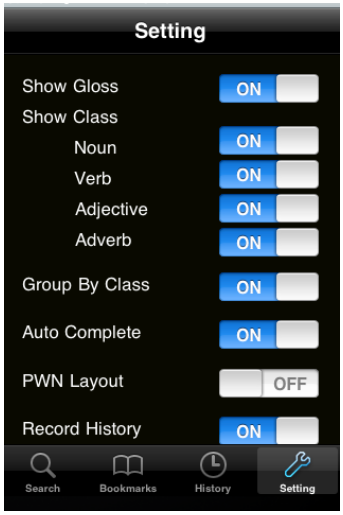
Pada gambar 10 ditunjukkan perpindahan kata abad komputer yang berasal pada posisi pertama menuju posisi tengah. Perpindahan ini dapat dilakukan dengan menahan *icon* perpindahan yang berada disebelah kanan kata dan memindahkannya pada posisi yang diinginkan. Apabila pengguna ingin menghapus satu atau lebih kata maka pengguna perlu memberikan tanda pada kata-kata tersebut terlebih dahulu. Pemberian tanda dilakukan dengan cara menekan *checkbox* yang berada disebelah kiri kata. Ketika semua kata yang diingin kan sudah ditandai maka penghapusan dilakukan dengan menekan tombol bergambar tong sampah yang terletak di kiri bawah. Apabila pengguna ingin menghapus semua kata maka pengguna tidak perlu member tanda pada semua kata, cukup dengan menekan tombol *delete all* yang terletak di kanan bawah. Untuk kembali ke mode *view* maka pengguna harus menekan tombol *done* yang terletak pada posisi kanan atas.



Gambar 11. Halaman History

Gambar 11 menunjukkan tampilan dari halaman *history*. Semua kata yang pernah dicari pengguna akan dicatat oleh aplikasi. Untuk melihat kata-kata tersebut maka pengguna bisa mengakses halaman *history*. Pada halaman ini semua kata yang pernah dicari pengguna akan ditampilkan, kata yang terakhir kali dicari akan berada pada posisi paling atas dan kata yang pertama kali dicari pengguna akan berada pada posisi paling bawah. Apabila pengguna melihat sebuah kata pada *history* maka kata tersebut akan memliki *background* biru yang

menandakan bahwa pengguna baru saja melihat kata tersebut. Jumlah kata yang dicatat oleh aplikasi dapat diatur melalui halaman setting. Secara *default* jumlah kata yang dicatat adalah 50.

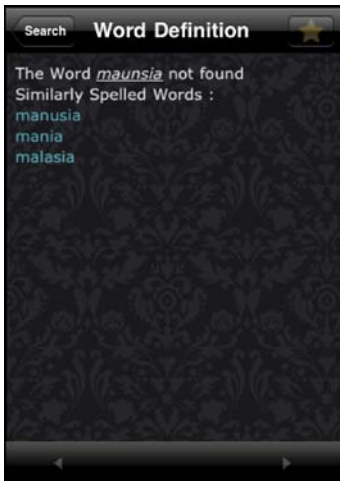


Gambar 12. Halaman Setting

Halaman *setting* merupakan halaman dimana pengguna dapat merubah pengaturan milik aplikasi. Hampir semua fitur milik aplikasi dapat diubah oleh pengguna, misalnya bila pengguna tidak ingin pencarian kata dicatat oleh aplikasi maka fitur *history* dapat dinonaktifkan atau bila pengguna ingin mengubah jumlah kata yang dicatat oleh aplikasi maka *slider* yang mewakili jumlah dapat diubah. Tampilan dari halaman setting ini ditunjukkan oleh gambar 12.

Selain fitur-fitur yang telah dijelaskan, masih terdapat sebuah fitur lagi yaitu fitur *suggestion*. Fitur ini akan menampilkan kata-kata yang memiliki kemiripan penulisan dengan kata yang diinputkan oleh pengguna. Fitur ini diaktifkan ketika kata yang diinputkan oleh pengguna tidak terdapat pada Lexical Database. Gambar 13 menunjukkan tampilan dari fitur *suggestion* ketika kata" *maunsia"* dicari

Fitur suggestion ini memanfaatkan levenshetein distance untuk mencari kata-kata yang memiliki kemiripan penulisan dengan kata yang dicari, Kata-kata yang ditampilkan dapat dipilih oleh user agar pencarian dapat langsung dilakukan. Pada saat fitur ini dijalankan tombol *bookmark*, *next*, dan *back* tidak akan dapat digunakan. Selain itu kata yang diinputkan juga tidak akan tercatat dalam catatan pencarian aplikasi.



Gambar 13. Fitur Suggestion

Halaman *search*, *bookmark*, *history* dan *setting* dapat berhubungan satu sama lain. Perpindahan dapat dilakukan dengan memilih *icon* halaman yang berada pada tab bat. Tetapi untuk halaman result hanya dapat diakses ketika pencarian kata dilakukan, begitu juga dengan fitur *suggestion*. Fitur tersebut hanya akan dijalankan ketika pencarian kata yang tidak terdapat pada Lexical Database dilakukan

V. KESIMPULAN

Dari proses pengerjaan Tugas Akhir ini diperoleh beberapa kesimpulan yang akan dijelaskan sebagai berikut:

1. WordNet telah memiliki sejarah yang lama dan dalam perkembangannya, tampilan yang digunakan untuk menyajikan informasi telah mengalami banyak perubahan agar menjadi lebih baik. Untuk menerapkan tampilan tersebut

ke dalam iPhone yang memiliki desain tampilan yang baru diperlukan berbagai percobaan untuk mendapatkan tampilan yang tepat dan dapat menggabungkan keduanya.

2. Dalam mendesain interface untuk iPhone dan iPad perlu diperhatikan mengenai kebiasaan dari pengguna iPhone dan iPad. Peletakan tombol pada posisi yang tidak biasa akan membingungkan pengguna.
3. Penggunaan bahasa pemrograman C++ akan sangat membantu dalam melakuka pemrosesan data dalam jumlah besar. Pada tugas akhir ini penggunaan bahasa pemrograman C++ sangat mempersingkat waktu yang diperlukan untuk mengolah Lexical Database Files.
4. Memory Management merupakan faktor yang perlu diperhatikan dalam pembuatan aplikasi iPhone dan iPhone. Apabila memory management tidak dilakukan maka aplikasi akan menggunakan banyak memory dari device sehingga aplikasi hanya dapat berjalan untuk waktu yang singkat sebelum aplikasi diberhentikan oleh device.
5. Desain Interface sangat penting dalam pembuatan sebuah WordNet Browser. Design yang dibuat haruslah menyajikan informasi secara lengkap kepada user tetapi tidak memberikan kesan sesak dan tidak membuat user untuk malas membaca.
6. Memory leak merupakan masalah yang harus diperhatikan. Apabila memory leak terjadi maka aplikasi akan berjalan lebih lambat selain itu ketika memory yang digunakan melebihi batas maka aplikasi akan secara otomatis ditutup oleh device.
7. Instruments merupakan aplikasi yang berguna untuk mengatasi memory leak. Dengan menggunakan tool instruments maka memory leak dalam aplikasi akan dapat dideteksi. Selain itu instruments juga akan menampilkan jenis variabel yang menyebabkan terjadinya memory leak.
8. WordNet Browser yang dibuat pada tugas akhir ini akan kompatibel dengan Lexical Database bahasa lain. Selama format yang digunakan pada Lexical Database tersebut merupakan format standar.

VI. SARAN

Tidak semua proses dalam pembuatan Tugas Akhir berjalan sesuai keinginan pembuat. Masih ada beberapa kekurangan yang terdapat dalam pembuatan Tugas Akhir ini. Berikut adalah saran yang dapat dipakai untuk pengembangan lebih lanjut.

1. Pertimbangkan pemanfaatan sqlite untuk menyimpan hasil ekstraksi informasi Lexical Database. Dengan menggunakan sqlite ekstraksi tidak perlu dilakukan berkali-kali sehingga dapat mengurangi waktu loading aplikasi.
2. Manfaatkan feedback yang disediakan oleh iTunes untuk mengembangkan WordNet browser yang lebih baik. Dengan feedback tersebut dapat diketahui kelemahan dari browser yang perlu diperbaiki.

REFERENSI

[1] Anonymous. *Cocoa Application Tutorial*, Apple Inc. 2009

[2] Dalrymple, Mark, Scott Knaster. *Learn Objective-C on the Mac.* Apress. 2009.

[3] Kochan, Stephen G. *Programming in Objective-C 2.0 (2nd Edition)*. Addison-Wesley Professional. 2009

[4] Mark, Dave, Jeff La Marche. *Beginning iPhone Development: Exploring The iPhone SDK*. Apress. 2009

[5] Sadun, Erica. *The iPhone Developer's Cookbook : Building Applications with the iPhone SDK*. Addison-Wesley. 2009
[6] Dan Pilone, Tracey Pilone, *Head First iPhone Development*. O'Reilly Media. 2009
[7] Wei-Meng Lee, *iOS 4 Application Development*. Wiley Publishing. 2010
[8] Pangestu. S., *Pembuatan Prototype Database Lexical untuk Bahasa Indonesia yang mengacu pada Wordnet*. 2007