**JURUSAN TEKNIK INFORMATIKA**

**FAKULTAS TEKNOLOGI INFORMASI**

**INSTITUT TEKNOLOGI SEPULUH NOPEMBER**

**USULAN TUGAS AKHIR**

1. **IDENTITAS PENGUSUL**

**Nama : Erfan Ahmadiyono**

**NRP : 5109 100 046**

**Dosen Wali : Waskitho Wibisono, S.Kom., M.Eng., Ph.D.**

**Dosen Pembimbing : 1. Dwi Sunaryono, S.Kom., M.Kom.**

**2. Adhatus Solichah A., S.Kom., M.Sc.**

# **JUDUL TUGAS AKHIR**

**Implementasi Pencocokan Perintah Suara Pada Aplikasi Facetube**

# **LATAR BELAKANG**

Facebook dan Youtube merupakan dua situs yang sangat populer digunakan oleh masyarakat dewasa ini, karena masing-masing kelebihannya yang semakin dibutuhkan. Maraknya pengunggahan video di internet menjadikan kedua situs tersebut semakin dimanfaatkan untuk perihal berbagi video. Facebook sendiri adalah sebuah layanan jejaring sosial dimana pengguna yang terdaftar dapat membuat profil. mengunggah foto dan video, mengirim pesan dan tetap berhubungan dengan teman-teman, keluarga dan koleganya. Sedangkan Youtube merupakan situs web berbagi video yang memungkinkan pengguna mengunggah, menonton, dan berbagi video.

Facebook yang lebih fokus pada fitur jejaring sosial mempunyai kekurangan dimana beberapa konten didalamnya tidak terpublikasi secara umum dan dapat dilihat secara bebas. Hal ini menjadi sebuah permasalahan dimana jika seseorang yang tidak memiliki akun Facebook hendak ingin melihat video yang tersedia didalamnya. Demikian juga Youtube, meskipun memiliki kelebihan dimana video dapat diakses dan dilihat secara umum tanpa terikat suatu akun, Youtube memiliki kekurangan dimana tidak terlalu menangani fitur-fitur jejaring sosial.

Berangkat dari permasalahan di atas, dalam Tugas Akhir ini dibuatlah sebuah aplikasi yang menggabungkan keunggulan Facebook dan Youtubeyang dinamakan Facetube. Facetube adalah aplikasi perekam video menggunakan perintah suara yang memiliki fitur utama yaitu otomatisasi unggah video ke situs Youtube dan kemudian mengirimkan sebuah status berisi link ke situs Facebook. Facetube menggunakan algoritma pencocokan *string* untuk mencocokkan *string* perintah suara dengan data yang ada di dalam *database* agar perintah suara pengguna dapat seoptimal mungkin menjalankan aplikasi. Facetube juga memiliki fitur lainnya seperti dapat memberikan *tag reference* dan merekam narasi cerita. Aplikasi ini ditujukan untuk pengguna alat komunikasi bergerak dengan basis sistem operasi Android, yang dewasa ini banyak digunakan oleh masyarakat.

# **RUMUSAN MASALAH**

Rumusan masalah pada Tugas Akhir ini adalah sebagai berikut :

1. Bagaimana cara membangun aplikasi gabungan keunggulan dari Facebook dan Youtube yang dapat melakukan otomatisasi.
2. Bagaimana menerapkan algoritma pencocokan *string* antara perintah suara pengguna dengan data yang ada dalam *database* sistem.
3. Bagaimana aplikasi yang dibangun dapat menyediakan fitur pemberian *tag reference* pada video yang akan diunggah.
4. Bagaimana aplikasi yang dibangun dapat merekam narasi cerita berupa suara pengguna.
5. Bagaimana aplikasi dapat mendeteksi kamera mana yang seharusnya digunakan dalam banyak kondisi.
6. Bagaimana aplikasi dapat mendeteksi perintah suara pengguna dengan baik.

# **BATASAN MASALAH**

Dalam pengerjaan Tugas Akhir ini masalah yang timbul diatas dibatasi oleh:

* + - 1. Aplikasi ini hanya dapat mengunggah video ke dalam situs Youtube saja, sedangkan Facebook hanya sebatas link referensi dari Youtube.
      2. Tag referensi pada video yang akan diunggah adalah berupa pemberian judul, lokasi dan waktu video tersebut diambil.
      3. Algoritma pencocokan *string* yang digunakan adalah algoritma *Levenshtein Distance* dan *Soundex*.
      4. Durasi waktu perekaman adalah selama 5 menit setelah perintah suara untuk merekam dari pengguna dideteksi oleh aplikasi.
      5. *Front end* aplikasi perangkat komunikasi bergerak ini dibangun dengan menggunakan bahasa *native* untuk sistem operasi Android.
      6. Aplikasi ini dapat dijalankan pada perangkat komunikasi bergerak dengan spesifikasi sistem operasi Android minimal 2.2 (Froyo).

# **TUJUAN TUGAS AKHIR**

Tujuan dari disusunnya Tugas Akhir ini dapat dituliskan sebagai berikut:

1. Membangun aplikasi yang dapat membantu mempermudah pengguna dalam proses perekaman video dan publikasinya ke dalam situs Youtube dan Facebook.
2. Mengimplementasikan algoritma *Soundex* pada aplikasi untuk pencocokan *string* antara perintah suara pengguna dengan data yang ada di dalam *database* sistem.
3. Membangun aplikasi yang multifungsi, dimana aplikasi juga memiliki fitur lain, seperti: dapat memberikan *tag reference* dan perekaman suatu narasi.
4. Membangun aplikasi perekaman video yang mudah dipakai dalam berbagai situasi dan kondisi.
5. Membuat aplikasi yang dapat berjalan dengan baik pada perangkat komunikasi bergerak berbasis Android.
6. **MANFAAT TUGAS AKHIR**

Manfaat dari disusunnya Tugas Akhir ini adalah untuk membantu mempercepat proses publikasi video ke Youtube serta Facebook, serta membantu mempermudah masyarakat dalam perekaman video di segala situasi dan kondisi dengan hanya menggunakan perintah suara.

# **TINJAUAN PUSTAKA**

Pada bagian ini akan dibahas mengenai tinjauan pustaka yang menjadi dasar dari pembuatan Tugas Akhir ini. Beberapa teori dan pustaka yang mendasari pengerjaan Tugas Akhir ini adalah definisi pencocokan *string,* algoritma *Soundex*, algoritma *Levenshtein Distance, Web Service*, dan *Application Programming Interface.* Berikut ini penjelasan masing-masing tinjauan pustaka.

## Pencocokan String

*String* adalah susunan dari karakter-karakter dan biasanya direpresentasikan sebagai struktur data *array*. *String* dapat berupa kata, frase, atau kalimat. Pencocokan *string* diartikan sebagai sebuah permasalahan untuk menemukan pola susunan karakter *string* di dalam *string* lain atau bagian dari isi teks [1].

## Algoritma Soundex

Algoritma *Soundex* pertama kali dipatenkan oleh Margaret O’Dell dan Robert C. Russell pada tahun 1918 [3]. Algoritma *Soundex* mengambil masukan berupa sebuah kata atau nama. Algoritma ini menghasilkan sebuah *string* yang mengidentifikasi apakah sepasang kata tersebut mirip secara fonetik. *String* ini disebut dengan kode fonetik.

Fonetik adalah ilmu yang menyelidiki bunyi bahasa tanpa melihat fungsi bunyi itu sebagai pembeda makna dalam suatu bahasa. Pencocokan *string* fonetik adalah suatu teknik pencocokan *string* yang membandingkan suatu *string* dengan *string* yang lain berdasarkan kode fonetik masing-masing [2]. Sebuah *string* yang berbeda namun mempunyai cara pengucapan yang sama, akan memiliki kode fonetik yang sama. Contohnya, “akhmad” dengan “achmad”.

Metode yang digunakan algoritma *Soundex* berdasarkan pada klasifikasi fonetik dari suara cara berbicara (*human speech sound*) yaitu *labial, dental, alveolar, palate-alveolar, palatal, velar, dan glottal* [3]. Klasifikasi ini berdasarkan pada dimana manusia meletakkan bibir dan lidahnya ketika melafalkan bunyi tertentu.

Aturan pemberian kode fonetik per huruf pada algoritma *Soundex* [4] dapat dilihat pada Tabel 1.

Tabel 1. Aturan pemberian kode fonetik pada algortima Soundex [4]

|  |  |  |
| --- | --- | --- |
| Huruf | Kode | Klasifikasi Fonetik |
| A,E,H,I,O,U,W,Y | 0 | Diperlakukan sebagai bunyi vokal |
| B,F,P,V | 1 | *Labial* dan *labio-dental* |
| C,G,J,K,Q,S,X,Z | 2 | *Glottal* |
| D,T | 3 | *Dental-mute* |
| L | 4 | *Palatal fricative* |
| M,N | 5 | *Labio-nasal* dan *dental* |
| R | 6 | *Dental fricative* |

Langkah-langkah algoritma *Soundex* dalam menghasilkan kode fonetik dari sebuah *string* masukan adalah sebagai berikut [3]:

1. Ubah sebuah huruf menjadi huruf kapital, dan hilangkan tanda baca.
2. Pertahankan huruf pertama pada kata tersebut.
3. Ubah huruf lainnya menjadi kode fonetik berdasarkan tabel.
4. Hapus semua pasangan dari kode fonetik yang berurutan.
5. Hapus semua kode fonetik 0.
6. Tulis empat posisi pertama yang mengikuti pola: *<uppercase letter><digit><digit><digit>*. Ini adalah kode fonetik sebagai keluaran. Jika kode fonetik tidak sampai empat karakter, maka ditambahkan digit 0 sampai menjadi empat karakter.

Misalkan ada sebuah *string* masukan “Upload”. Langkah-langkah pemberian kode fonetiknya adalah sebagai berikut. Ubah semua huruf menjadi huruf kapital (“UPLOAD”). Pertahankan huruf pertama, lalu ubah huruf lainnya menjadi kode fonetik berdasarkan tabel (U14003). Hapus semua pasangan dari kode fonetik yang berurutan (U1403). Hapus semua kode fonetik 0 (U143). Empat karakter pertama adalah kode fonetik sebagai keluaran (U143).

## Algoritma Levenshtein Distance

Algoritma *Levenshtein Distance* dinamakan berdasarkan penemunya, yaitu Dr. Vladimir Levenshtein. Dr. Vladimir Levenshtein dikenal sebagai bapak dari teori koding (*father of coding theory*) di Rusia [6]. *Levenshtein Distance* saat ini adalah dasar dari banyak aplikasi komputer di bidang pemeriksaan ejaan.

*Levenshtein Distance* adalah jumlah minimal operasi yang dibutuhkan untuk mengubah suatu *string* ke *string* yang lain. Dengan algoritma ini, dapat diketahui tingkat perbedaan dua buah *string* dalam representasi angka. Operasi yang dilakukan ada 3 macam yaitu [7]:

1. *Insertion*

*Insertion* adalah penyisipan sebuah karakter ke dalam sebuah *string* tertentu. Contohnya, penyisipan karakter ‘m’ ke dalam *string* “julah” pada posisi setelah karakter ‘u’ akan mengubah *string* “julah” menjadi “jumlah”.

1. *Deletion*

*Deletion* adalah penghapusan sebuah karakter dari sebuah *string* tertentu. Contohnya, penghapusan karakter ‘a’ pada posisi terakhir *string* “algoritma” akan mengubah *string* “algoritma” menjadi “algoritm”.

1. *Substitution*

*Substitution* adalah penggantian sebuah karakter dari sebuah *string* dengan karakter lain. Contohnya, penggantian karakter ‘l’ dengan karakter ‘b’ pada *string* “julah” akan mengubah *string* “julah” menjadi “jubah”.

Misalkan ada dua buah masukan *string* yaitu “kepala” dan “kapal”. Langkah untuk mengubah “kepala” menjadi “kapal” adalah sebagai berikut. Substitusi karakter ‘e’ dengan karakter ‘a’. Kemudian hapus karakter ‘a’ pada posisi terakhir. Jadi, jumlah operasi yang dilakukan adalah dua kali. Satu kali *substitution* dan satu kali *deletion*. *Levenshtein Distance* adalah jumlah dari banyaknya operasi yang dilakukan. Sehingga, *Levenshtein Distance* dari “kepala” dan “kapal” adalah 2.

*Levenshtein Distance* menggunakan konsep *dynamic programming* dalam penerapannya. Algoritma ini menggunakan sebuah matriks berukuran panjang dari *string*-*string* yang menjadi masukan. *Levenshtein Distance* yang dihasilkan adalah angka yang terletak di kotak paling bawah kanan dari matriks, sebagai jumlah minimum operasi yang dibutuhkan untuk mengubah *string* awal menjadi *string* target.

## Web Service

*Web service* adalah suatu sistem perangkat lunak yang dirancang untuk mendukung interoperabilitas dan interaksi antar sistem pada suatu jaringan. *Web service* digunakan sebagai suatu fasilitas yang disediakan oleh suatu situs web untuk menyediakan layanan (dalam bentuk informasi) kepada sistem lain, sehingga sistem lain dapat berinteraksi dengan sistem tersebut melalui layanan-layanan yang disediakan oleh suatu sistem yang menyediakan *web service*. *Web service* menyimpan data informasi dalam format XML, sehingga data ini dapat diakses oleh sistem lain walaupun berbeda *platform*, sistem operasi, maupun bahasa *compiler*.

Dasar *platform* dari *web service* adalah XML dan HTTP. XML menyediakan bahasa yang dapat digunakan antara *platform* dan bahasa pemrograman yang berbeda tetapi masih tetap mengandung pesan dan kegunaan yang sama. Sedangkan HTTP adalah protokol internet yang paling banyak digunakan. Elemen *platform* dari *web service* antara lain SOAP (*Simple Object Access Protocol*), UDDI (*Universal Description, Discovery, and Integration*), dan WSDL (*Web Services Description Language*) [7].

## Antarmuka Pemrograman Aplikasi

Antarmuka Pemrograman Aplikasi atau yang dalam bahasa Inggris disebut *Application Programming Interface* (API) adalah sekumpulan [perintah](http://id.wikipedia.org/w/index.php?title=Perintah&action=edit&redlink=1), [fungsi](http://id.wikipedia.org/wiki/Fungsi), dan [protokol](http://id.wikipedia.org/wiki/Protokol) yang dapat digunakan oleh [*programmer*](http://id.wikipedia.org/wiki/Programmer) saat membangun [perangkat lunak](http://id.wikipedia.org/wiki/Perangkat_lunak) untuk [sistem operasi](http://id.wikipedia.org/wiki/Sistem_operasi) tertentu. API memungkinkan *programmer* untuk menggunakan [fungsi standar](http://id.wikipedia.org/w/index.php?title=Fungsi_standar&action=edit&redlink=1) untuk berinteraksi dengan sistem operasi.

API dapat juga diartikan sebagai serangkaian instruksi dan standar pemrograman untuk mengakses aplikasi atau layanan berbasis web. Sebuah perusahaan *software* atau penyedia layanan berbasis web merilis API mereka kepada publik. Dengannya, pengembangan lain dapat mendesain aplikasi yang memanfaatkan layanan mereka.

Sebagai contoh, Amazon merilis API sehingga para pengembang web dapat lebih mudah mengakses informasi produk-produk Amazon dari *website* mereka. Menggunakan API dari Amazon, *website* pihak ketiga dapat mem-*posting* link langsung ke produk-produk Amazon dengan harga aktual dan opsi "*buy now*" [8]. Untuk aplikasi Facetube ini akan dipergunakan API dari Facebook dan Youtube.

# **METODOLOGI**

Pada bagian ini akan dijelaskan metodologi yang digunakan dalam proses pelaksanaan Tugas Akhir. Metodologi tersebut adalah sebagai berikut.

## Penyusunan Proposal

Facetube adalah aplikasi perekam video menggunakan perintah suara yang memiliki fitur utama yaitu otomatisasi unggah video ke situs Youtube dan kemudian mengirimkan sebuah status berisi link ke situs Facebook. Facetube menggunakan algoritma pencocokan *string* untuk mencocokkan *string* perintah suara dengan data yang ada di dalam *database* agar perintah suara pengguna dapat seoptimal mungkin menjalankan aplikasi.

## Studi Literatur

Pada tahap ini akan dilakukan pencarian, pembelajaran, penyaringan dan pemahaman konsep yang berkaitan dengan aplikasi yang akan dibuat. Tahap awal yang dilakukan adalah pengumpulan informasi terkait dengan aplikasi dan juga mencari infomasi mengenai kebutuhan pengguna.

Kemudian tahap selanjutnya adalah studi literatur tentang pemahaman konsep aplikasi berbasis kamera untuk melakukan perekaman video dan pemahaman konsep algoritma *Soundex* dan *Levenshtein Distance*. Selanjutnya dilakukan pembelajaran konsep *login* serta fitur-fitur lainnya menggunakan API Youtube dan Facebook, konsep pemberian narasi pada sebuah video, dan pembelajaran konsep *speech-to-text* pada Android.

Setelah itu tahap studi literatur tentang pemahaman teknologi yang akan digunakan untuk pengembangan aplikasi ini, seperti API apa saja yang hendak dipergunakan.

## Analisis dan desain perangkat lunak

Pada tahap analisis dilakukan pencarian, pemahaman, dan penyaringan data dan informasi tentang bagaimana Facetube akan dibangun, serta bagaimana proses bisnisnya. Aplikasi ini dibangun agar pengguna lebih mudah melakukan publikasi video ke situs Youtube dan Facebook dengan pemberian fitur-fitur tambahan. Aplikasi ini menggunakan Antarmuka Pemrograman Aplikasi atau API, baik dari API Youtube maupun API Facebook. Pada tahap desain, akan dibuat tipe arsitektur dari aplikasi yang menggambarkan garis besar alur kerja sistem dengan memakai *flowchart* serta gambaran arsitektur sistem.

Pada Gambar 1 dapat dilihat arsitektur sistem dari aplikasi yang akan dibangun. Pengguna akan diminta untuk *login* pada akun Youtube dan Facebook yang dimilikinya terlebih dahulu. Kemudian aplikasi akan menunggu perintah suara dari pengguna untuk melakukan perekaman video. Jika terdapat perintah suara yang terdeteksi maka aplikasi akan mengirimkan hasil perintah suara kepada *Web Service* yang selanjutnya akan dilakukan proses pencocokan *string* dengan data yang ada pada *Database Server* menggunakan algoritma pencocokan *string*. Jika perintah suara tersebut terdeteksi benar, maka aplikasi akan memulai perekaman video maupun suara. Aplikasi dapat mendeteksi kamera mana yang seharusnya dipakai (kamera depan atau kamera belakang) untuk merekam berdasarkan kondisi dimana kamera *smartphone* tersebut tertutup oleh objek atau tidak. Kemudian diberikan durasi selama beberapa menit sebelum akhirnya perekaman akan selesai secara otomatis dan video yang telah terekam akan disimpan pada memori *smartphone*. Setelah itu aplikasi akan menambahkan *tag reference* berupa lokasi dan waktu dimana video tersebut diambil. Jika proses tersebut telah selesai maka video akan diunggah ke Youtube dan selanjutnya akan mengirimkan link ke Facebook. Setelah proses telah selesai akan ada pemberitahuan bahwa video telah berhasil terunggah. Aplikasi Facetube ini akan terus menyala pada *background* sistem perangkat Android dan akan langsung berjalan ketika menerima perintah suara tertentu.



Gambar 1. Arsitektur Sistem

Adapun untuk kebutuhan pengguna didapatkan sebagaimana diacu pada Gambar 2.

Gambar 2. Diagram Kasus Penggunaan

Pada Gambar 2dapat dilihat diagram kasus penggunaan untuk aplikasi ini, yang memperlihatkan aktivitas-aktivitas yang dapat dilakukan oleh pengguna sekaligus adalah fitur-fitur dari aplikasi ini, yaitu sebagai berikut:

1. Melakukan login pada akun Facebook dan Youtube

Pengguna diwajibkan untuk *login* pada situs Facebook dan Youtube terlebih dahulu jika terdeteksi belum pernah melakukan *login* sama sekali pada aplikasi. Jika pengguna telah berhasil *login* pada kedua situs tersebut maka untuk penggunaan aplikasi selanjutnya pengguna tidak usah melakukan *login* kembali.

1. Melakukan perintah rekam video dan narasi

Pengguna harus memberikan perintah suara tertentu untuk dapat menjalankan aplikasi. Ketika aplikasi berhasil mendeteksi bahwa perintah suara tersebut cocok maka aplikasi akan melakukan perekaman video dan narasi secara otomatis. Aplikasi akan secara langsung mendeteksi kamera mana yang dapat digunakan kemudian setelah beberapa menit perekaman maka video akan otomatis disimpan di dalam memori *smartphone* dan selanjutnya melalui proses pemberian *tag reference* dan pengunggahan.

1. Melakukan pembatalan perekaman video

Fitur ini adalah fitur dimana pengguna dapat membatalkan perekaman video yang kemungkinan disebabkan oleh terjadinya kesalahan dari pengguna yang tidak sengaja melakukan perekaman atau tidak jadi melakukan perekaman.

Gambar 3 memperlihatkan garis besar aliran proses aplikasi Facetube, dimulai dari proses *login*, perekaman video serta narasi, pemberian *tag reference* oleh aplikasi, pembatalan perekaman video, dan pengunggahan video ke situs Youtube dan pengiriman link ke Facebook.



Gambar 3. Diagram Aliran Data Aplikasi Facetube

* 1. **Implementasi perangkat lunak**

Pada tahap ini hasil studi literatur yang telah dilakukan, diterapkan dalam implementasi rancangan yang terbuat oleh barisan kode program.

* 1. **Pengujian dan evaluasi**

Pada tahap ini dilakukan uji coba terhadap sistem yang telah dibuat, untuk mengamati kinerja sistem, serta mengidentifikasi kesalahan. Proses pengujian yang akan dilakukan adalah menguji fungsionalitas dari aplikasi, mengevaluasi jalannya program, mendeteksi kesalahan-kesalahan yang mungkin terjadi, dan melakukan perbaikan bila terdapat kekurangan untuk menyempurnakan hasil. Pengujian juga dilakukan untuk mengevaluasi apakah program yang dibuat akan menghasilkan solusi sesuai dengan tujuan dan manfaat dari Tugas Akhir ini.

* 1. **Penyusunan Buku Tugas Akhir**

Pada tahap ini dilakukan penyusunan laporan yang menjelaskan dasar teori dan metode yang digunakan dalam tugas akhir ini serta hasil dari implementasi pemodelan konten yang telah dibuat. Sistematika penulisan buku tugas akhir secara garis besar adalah sebagai berikut:

1. Pendahuluan
   1. Latar Belakang
   2. Rumusan Masalah
   3. Batasan Tugas Akhir
   4. Tujuan
   5. Metodologi
   6. Sistematika Penulisan
2. Tinjauan Pustaka
3. Desain dan Implementasi
4. Uji Coba dan Evaluasi
5. Kesimpulan dan Saran
6. Daftar Pustaka

# **JADWAL KEGIATAN**

Jadwal kegiatan tugas akhir ini ditunjukkan oleh Tabel 3.

Tabel 3. Jadwal Kegiatan Pengerjaan Tugas Akhir

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **No.** | **Kegiatan** | **Bulan** | | | | | | | | | | | | | | | | | | | |
| **Maret 2014** | | | | **April 2014** | | | | **Mei 2014** | | | | **Juni 2014** | | | | **Juli 2014** | | | |
| 1. | Penyusunan Proposal Tugas Akhir |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 2. | Studi Literatur |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 3. | Analisis dan Perancangan |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 4. | Implementasi |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 5. | Pengujian dan Evaluasi |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 6. | Penyusunan Buku Tugas Akhir |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

# **DAFTAR PUSTAKA**

|  |  |
| --- | --- |
| [1] | Syaroni, Mokhamad dan Munir, Rinaldi. 2004. *Pencocokan String Berdasarkan Kemiripan Ucapan* (*Phonetic String Matching*) *dalam Bahasa Inggris*. Departemen Teknik Informatika, Fakultas Teknologi Industri, Institut Teknologi Bandung. |
| [2] | Purnamasari, Tryas Ayu dan Luthfi, Emha Taufiq. 2012. *Membangun Aplikasi Pencocokan String Berdasarkan Penulisan dan Kemiripan Pengucapan*. Jurusan Teknik Informatika, STMIK AMIKOM Yogyakarta. |
| [3] | Creativyst, Inc.”Understanding Classic Soundex Algorithms” [Online]. Available: <http://www.creativyst.com/Doc/Articles/SoundEx1/SoundEx1.htm>. [Diakses 31 Maret 2014]. |
| [4] | Soundex Coding Rules. [Online]. Available: <http://www.genealogyintime.com/GenealogyResources/Articles/what_is_soundex_and_>  how\_does\_soundex\_work\_page2.html. [Diakses 31 Maret 2014]. |
| [5] | Karhendana, Arie, Wizanajani, Dicky, dan Yuliawan, Fajar. 2004. *Normalisasi String untuk Optimasi Phonetic String Matching dalam Bahasa Indonesia.* Sekolah Teknik Elektro dan Informatika, Institut Teknologi Bandung. |
| [6] | Vladimir I. Levenshtein biography. [Online]. Available: http://www.ieeeghn.org/wiki/index.php/Vladimir\_L\_Levenshtein. [Diakses 31 Maret 2014]. |
| [7] | Pinzon, Yoan. 2006. *Algorithms for Approximate String Matching*. Universidad Nacional de Colombia. |
| [8] | “W3Schools.com,” [Online]. Available: http://www.w3schools.com/webservices/ws\_intro.asp. [Diakses 21 Oktober 2013]. |
| [9] | Botoykoma, "http://botoykoma.blogspot.com/2013/01/api-aplikasi-programing-interface.html," [Online]. [Diakses 21 Oktober 2013]. |