# APLIKASI PENGENALAN KARAKTER ALFANUMERIK MENGGUNAKAN ALGORITMA HAMMING DISTANCE

**Matheus Supriyanto Rumetna[1*], Marla Pieter, Monica Manurung[1]**
[1]Fakultas Ilmu Komputer dan Manajemen, Universitas Sains dan Teknologi Jayapura
Jl. Raya Sentani - Padang Bulan, Abepura, Vim, Abepura, Kota Jayapura, Papua 99224
[*1]Email: matheus.rumetna@gmail.com

**Abstrak**

*Karakter Alfanumerik merupakan perpaduan antara karakter huruf dan angka, karakter alfanumerik ini pun sudah dikenal dan diketahui oleh semua orang. Tetapi masalah yang dihadapi dalam kondisi ini adalah bagaimana melakukan pengenalan karakter alfanumerik dengan cepat dan tepat. Permasalahan pengenalan karakter alfanumerik dapat dicari penyelesaiannya dengan menggunakan konsep Jaringan Syaraf Tiruan (JST) dan melakukan perhitungan menggunakan Algoritma Hamming Distance. Tujuan dari perancangan perangkat lunak ini adalah untuk mengenali pola karakter dan nilai persentase kemiripan karakter dari hasil pengenalan dengan menggunakan Algoritma Hamming Distance. Hasil yang didapatkan adalah perancangan aplikasi pengenalan karakter alfanumerik dengan menggunakan Algoritma Hamming Distance. Dan telah terbukti bahwa Algoritma Hamming Distance lebih cepat dan tepat dalam melakukan pengenalan karakter alfanumerik.*

***Kata kunci**: Jaringan Syaraf Tiruan, Hamming Distance, Segmentasi Citra*

## 1. PENDAHULUAN

Seiring dengan pesatnya kemajuan teknologi di segala bidang, tidak terlepas pada bidang komputerisasi. Komputer saat ini telah menjadi alat bantu utama bagi manusia, segala aktifitas manusia dapat di kerjakan dengan menggunakan komputer. Bukan hanya untuk menyelesaikan permasalahan di tempat kerja atau bermain *game*, namun dapat juga digunakan untuk mendapat informasi yang dibutuhkan secara cepat, tepat dan akurat. Pengenalan pola merupakan salah satu hal yang paling sering terjadi dalam lingkungan kerja, hal ini mendorong manusia untuk membuat aplikasi yang dapat mengenali pola (Putra D, 2009).

Pengenalan pola yang cukup kompleks adalah pengenalan karakter alfanumerik (‘A’ … ‘Z’ dan ‘0’ … ‘9’). Masalah yang dihadapi adalah tidak tersedianya fungsi matematika yang jelas yang bisa menghasilkan translasi yang diinginkan sehingga penerjemahan karakter alfanumerik menjadi cukup memusingkan karena untuk menghasilkan translasi, salah satu cara yang dapat dilakukan adalah dengan melakukan korelasi piksel demi piksel dan membutuhkan waktu yang relatif lama. Selain itu, tidak tertutup kemungkinan pengenalan karakter alfanumerik ini menghadapi masalah citra yang mengandung *noise* atau kurang lengkap. Untuk itulah memerlukan perkembangan teknologi yang lebih efektif sehingga pengenalan pola ini dapat dilakukan dengan lebih cepat dan tepat (Desiani dan Arhami, 2006).

Banyak cara yang dapat dilakukan dalam upaya mengenali pola karakter alfanumerik salah satunya adalah dengan menggunakan algoritma *Hamming Distance* (Freeman and Skapura, 2009). Untuk itu, dalam penelitian ini akan dibahas tentang cara mengimplementasi algoritma *Hamming Distance* dalam melakukan pengenalan pola karakter alfanumerik. Tujuannya untuk mengenali pola karakter dan nilai persentase kemiripan karakter dari hasil pengenalan dengan menggunakan Algoritma *Hamming Distance*. Aplikasi ini dibangun menggunakan *software Microsoft Visual Basic.Net 2012.*

## 2. METODOLOGI

Untuk memperoleh data serta informasi yang dibutuhkan dalam penelitian ini, maka metode pengumpulan data dilakukan dengan studi literatur. Metode ini bertujuan untuk meneliti dan mengumpulkan data melalui riset dan menggunakan referensi buku serta jurnal yang berhubungan dengan kecerdasan buatan (*Artificial Intelligence*) dan mempelajari algoritma yang akan digunakan (Kusumadewi, 2006). Setelah memperoleh data serta informasi, kemudian melakukan eksperimen dengan menggunakan algoritma *Hamming Distance* untuk mengenali pola karakter dan nilai

persentase kemiripan karakter dari hasil pengenalan (Freeman and Skapura, 2009). Proses pengenalan pola dilakukan dengan persamaan sebagai berikut:

$$d_{PQ} = q + r \quad (1)$$

Dimana:

$q$ = jumlah dari variabel dengan nilai 1 untuk citra i dan 0 untuk citra j, dan
$r$ = jumlah dari variabel dengan nilai 0 untuk citra i dan 1 untuk citra j

Kemudian untuk nilai persentase kemiripan karakter dari hasil pengenalan dihitung dengan menggunakan persamaan sebagai berikut:

$$D_O = \frac{\sum_{i=1}^{N}\sum_{j=1}^{N} P(i,j) \otimes Q(i,j)}{N^2} \quad (2)$$

Dimana:

$D_O$ = nilai *Hamming distance* ternormalisasi
$P$ = vektor-1
$Q$ = vektor-2
$\otimes$ = operator *exclusive* OR (XOR), yang akan menghasilkan nol, jika dan hanya jika bit dari *P(i, j)* sama dengan *Q(i, j)*.
$i$ = posisi kolom pada matriks (secara horizontal / $x$)
$j$ = posisi baris pada matriks (secara vertikal / $y$)
$N$ = jumlah baris atau jumlah kolom vektor. Ukuran matriks vektor dinyatakan oleh $N \times N$.

## 3. HASIL DAN PEMBAHASAN

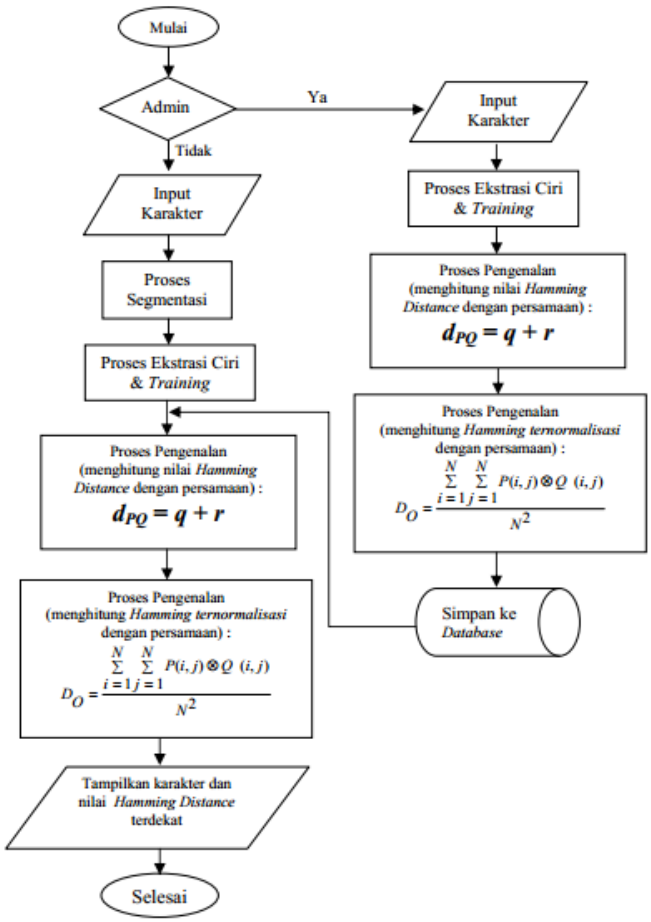
Sistem kerja yang dibuat dalam penelitian ini yaitu sistem pengenalan karakter alfanumerik menggunakan algoritma *Hamming Distance*, dimana hasilnya akan lebih tepat dari pada melakukan korelasi piksel demi piksel dibutuhkan waktu yang relatif lama.

### 3.1. Langkah Kerja

Tahap awal dilakukan analisa kebutuhan sistem yang merupakan proses identifikasi dan evaluasi permasalahan yang ada, sehingga sistem yang dibangun sesuai dengan kriteria yang diharapkan. Berdasarkan analisis, aplikasi hasil rancangan harus memenuhi ketentuan sebagai berikut:

(1) Sistem harus menyediakan area penggambaran, sehingga pengguna dapat menuliskan karakter yang diinginkan, dengan menggunakan *mouse*.
(2) Aplikasi dapat melakukan proses segmentasi terhadap karakter-karakter yang tidak bersambung satu sama lain dengan algoritma 8-*connected*.
(3) Aplikasi dapat melakukan proses ekstraksi ciri terhadap setiap karakter, dengan ukuran ekstraksi ciri sebesar 10 x 10 kotak, atau 100 *bit biner*.
(4) Aplikasi dapat melakukan proses *training* (pelatihan/pembelajaran) terhadap karakter baru.
(5) Aplikasi dapat menampilkan semua karakter yang sudah pernah dilatih dan tersimpan di dalam *database*.
(6) Aplikasi dapat melakukan proses pengenalan terhadap karakter yang dituliskan oleh pengguna, serta menampilkan langkah-langkah perhitungan algoritma *Hamming Distance*.

Sesuai dengan hasil analisa terhadap kebutuhan sistem berupa aplikasi pengenalan karakter alfanumerik, maka dapat dijabarkan alur program melalui *flowchart*. Berikut ini *flowchart* dari aplikasi pengenalan karakter alfanumerik.



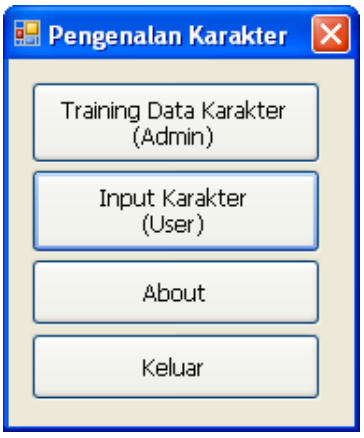

**Gambar 1. *Flowchart* sistem**

### 3.2. Tampilan Aplikasi

Aplikasi pengenalan karakter alfanumerik ini memiliki sembilan buah *form*, yaitu: *Form* Utama, *Form Input Password*, *Form* Ubah *Password*, *Form* Data Karakter, *Form Input* karakter, *Form Training*, *Form* Pengenalan, *Form Database* dan *Form About*.

#### 3.2.1 *Form* Utama

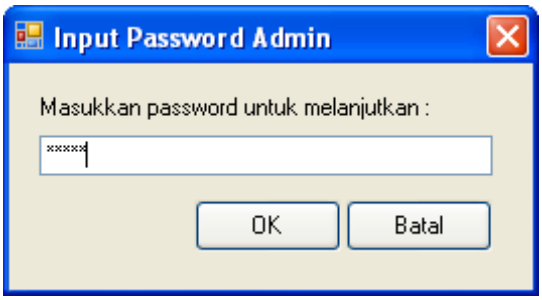
Pada *form* Utama, pengguna dapat meng*input* karakter alfanumerik yang akan dipelajari. *Form* ini berisi pilihan untuk menampilkan *form* Data karakter, form *Input* Karakter atau *form About*.




**Gambar 2. *Form* utama**

### 3.2.2 *Form Input Password*

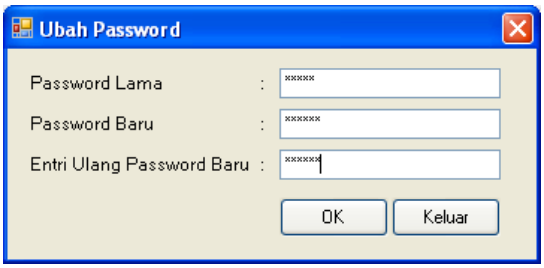
*Form* ini berfungsi untuk memasukkan *password admin*. Apabila *input password* benar, maka *form* Data Karakter seperti pada Gambar 5. *Form* ini akan muncul jika *admin* ingin memasukkan karakter baru yang belum dipelajari ke dalam *database*.




**Gambar 3. *Form input password***

### 3.2.3 *Form* Ubah *Password*

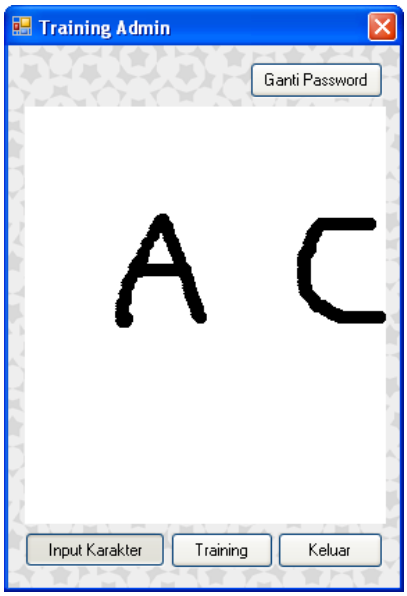
*Form* ini berfungsi untuk mengganti *password Admin*, tekan tombol "Ganti *Password*" pada form Data Karakter dan *form* Ganti *Password* akan muncul.




**Gambar 4. *Form* ubah *password***

### 3.2.4 *Form* Data Karakter

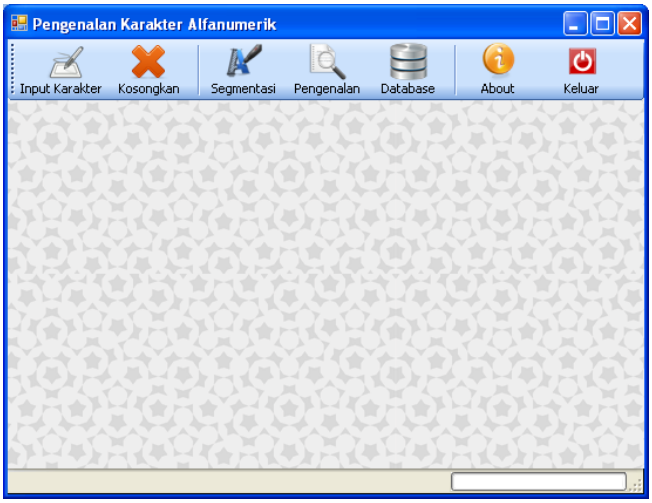
*Form* ini berfungsi untuk memasukkan (*training*) data karakter yang baru ke dalam *database*. *Form* ini hanya dapat diakses oleh *admin*.




**Gambar 5. *Form* data karakter**

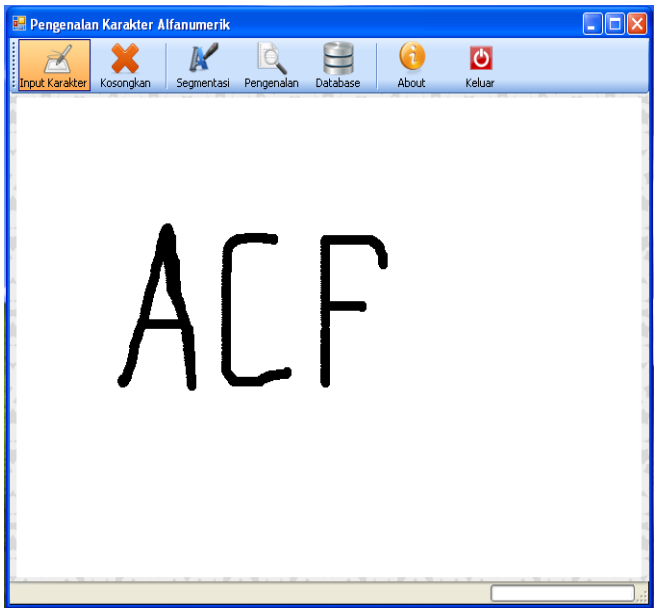
### 3.2.5 *Form Input* Karakter

Pada *form input* karakter, pengguna dapat menuliskan karakter alfanumerik, melakukan segmentasi, proses *training* dan pengenalan karakter, serta melihat karakter yang sudah tersimpan di dalam *database*. *Form* ini berisi tombol-tombol yang memiliki sejumlah fungsi utama, seperti *input* karakter, kosongkan, segmentasi, pengenalan, *database*, *about* dan keluar.




**Gambar 6. *Form input* karakter**

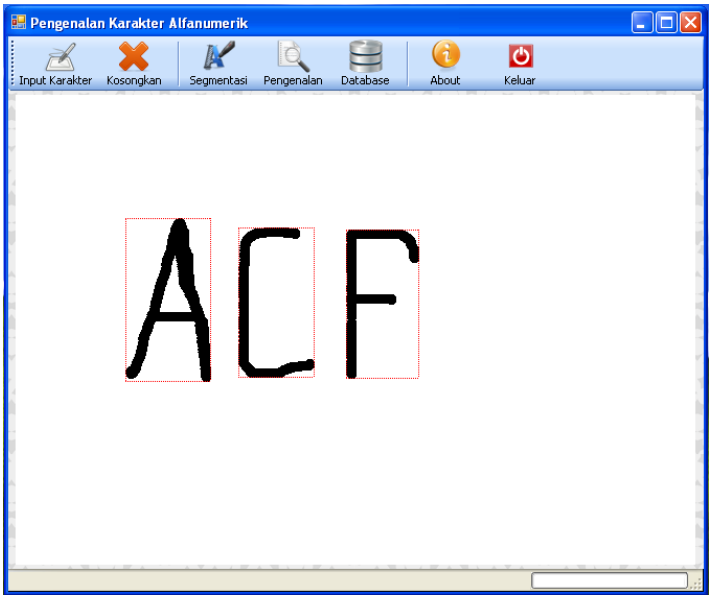
Tekan tombol "*Input* Karakter" dan area coretan akan muncul. *User* dapat mencoret beberapa karakter sekaligus pada area coretan, seperti terlihat pada gambar 7 berikut:




**Gambar 7. Tampilan coretan karakter pada *form* utama**

Untuk mengosongkan coretan, tekan tombol "Kosongkan" pada *toolbar*. Untuk melakukan segmentasi atau pemisahan karakter dengan menggunakan algoritma 8-*connected*, tekan tombol 'Segmentasi' dan proses segmentasi akan berjalan.

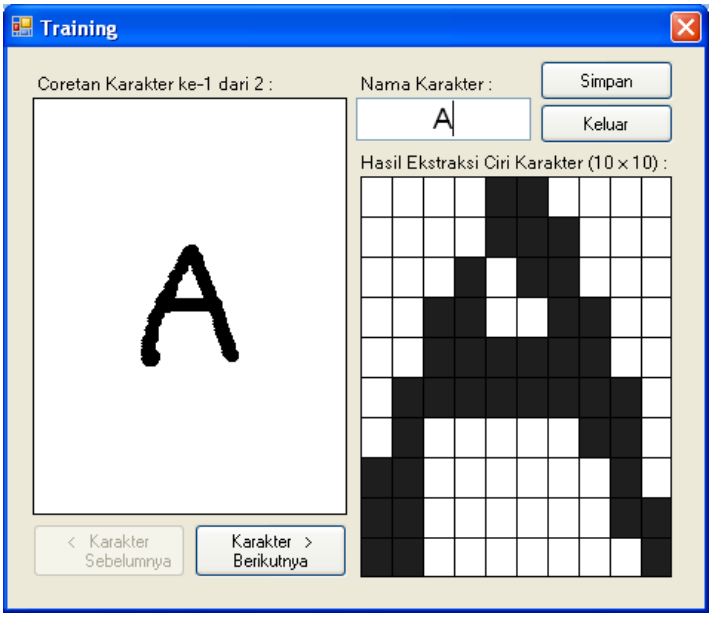
Pada akhir proses, karakter akan tersegmentasi seperti terlihat pada gambar 8 berikut:




**Gambar 8. Tampilan hasil segmentasi**

### 3.2.6 *Form Training*

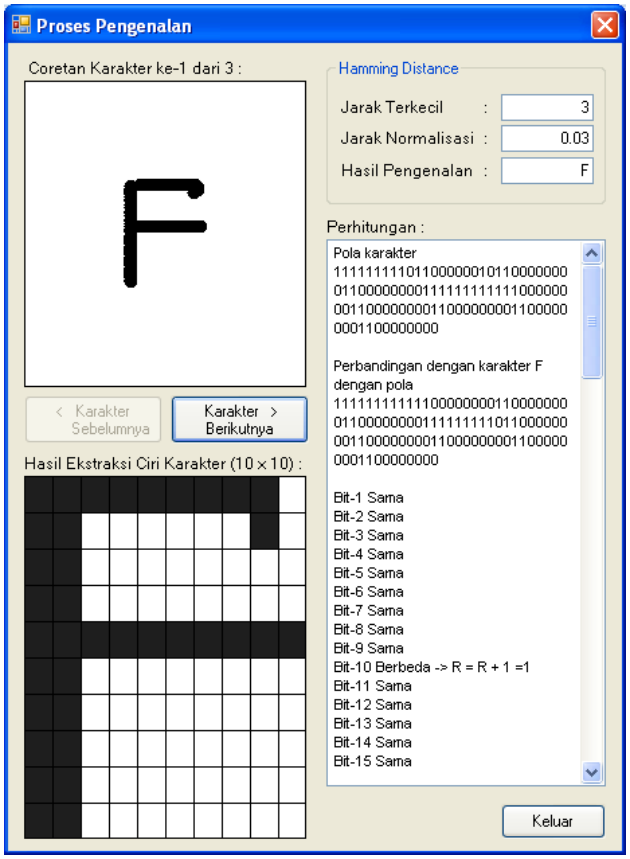
*Form* ini berfungsi untuk menyimpan *bit-bit* ekstraksi ciri dari karakter ke *database*, sehingga aplikasi dapat mengenali pola karakter dimaksud. Karakter yang belum pernah dilatih atau di-*training*, tidak akan dapat dikenali. *Admin* dapat melakukan *training* terhadap pola karakter dengan mencoret karakter dan menekan tombol ”Simpan”.




**Gambar 9. *Form training***

### 3.2.7 *Form* Pengenalan

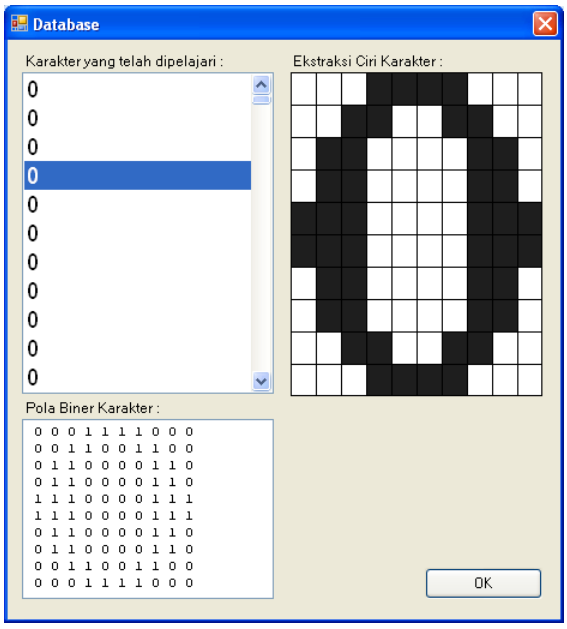
Pada *form* ini, aplikasi melakukan proses pengenalan terhadap karakter, dengan menghitung nilai *Hamming Distance* antara bit ekstraksi ciri dari karakter yang ingin dikenali dengan *bit* ekstraksi ciri dari karakter-karakter yang terdapat di dalam *database*. Langkah-langkah perhitungan nilai *Hamming Distance*, juga akan ditampilkan pada *form* ini, sehingga pengguna dapat melihat cara perhitungan *Hamming Distance* yang terjadi di dalam aplikasi.




**Gambar 10. *Form* pengenalan**

### 3.2.8 *Form Database*

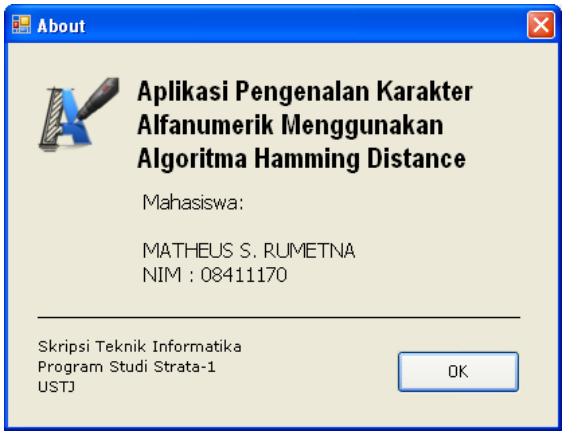
Semua nama karakter yang sudah pernah dilatih, beserta dengan hasil ekstraksi cirinya akan ditampilkan pada *form Database*.




**Gambar 11. *Form database***

### 3.2.9 *Form About*

*Form* ini berisi nama dan logo aplikasi serta identitas mahasiswa.




**Gambar 12. *Form about***

## 4. KESIMPULAN

Setelah menyelesaikan perancangan aplikasi pengenalan karakter alfanumerik menggunakan algoritma *Hamming Distance*, maka dapat ditarik beberapa kesimpulan sebagai berikut:

(1) Algoritma *Hamming Distance* dapat digunakan untuk melakukan pengenalan terhadap pola karakter alfanumerik.
(2) Proses pengenalan disertai dengan langkah-langkah proses perhitungan, sehingga dapat memberi penjelasan mengenai cara perhitungan jarak *Hamming* di dalam aplikasi.
(3) Proses pelatihan dapat dilakukan beberapa kali untuk satu karakter, untuk meningkatkan akurasi pengenalan terhadap karakter tersebut. Bila hasil pengenalan terhadap suatu pola salah, maka pola tersebut perlu dilatih ulang.

### 4.1. Saran

Adapun saran-saran yang dapat diberikan untuk pengembangan aplikasi lebih lanjut adalah sebagai berikut:

(1) Dipertimbangkan untuk mengembangkan aplikasi sehingga dapat melakukan pengenalan terhadap kalimat di dalam *file* gambar.
(2) Algoritma *Hamming Distance* dapat dijadikan dasar bagi aplikasi pengenalan pola lainnya, seperti: aplikasi pengenalan wajah, aplikasi pengenalan tanda tangan dan aplikasi pengenalan sidik jari.

## DAFTAR PUSTAKA

Desiani, A. dan Arhami, M., (2006), Konsep Kecerdasan Buatan, Penerbit Andi, Yogyakarta.
Freeman, J.A., and Skapura, D.M, (2009), Neural Networks: Algorithms, Applications, and Programming Techniques, Addison-Wesley, Reading, MA.
Kusumadewi, S., (2006), Artificial Intelligence (Teknik dan Aplikasinya), Penerbit Graha Ilmu, Yogyakarta.
Kuswadi, S., (2006), Kendali Cerdas (Teori dan Aplikasi Praktisnya), Penerbit Andi, Yogyakarta.
Puspitaningrum, D., (2006), Pengantar Jaringan Syaraf Tiruan, Penerbit Andi, Yogyakarta.
Putra, D., (2008), Sistem Biometrika, Penerbit Andi, Yogyakarta.
Putra, D., (2009), Pengolahan Citra Digital, Penerbit Andi, Yogyakarta.
Siang, JJ., (2004), Jaringan Syaraf Tiruan dan Pemogramannya Menggunakan Matlab, Penerbit Andi, Yogyakarta.