PERANCANGAN DETEKTOR PLAGIARISME *SOURCE CODE* PADA BAHASA PEMROGRAMAN JAVA

Lutfiyah Dwi Setia[1], Tri Lestariningsih[2]
Politeknik Negeri Madiun
Lutfiyah17@pnm.ac.id, trilestariningsih@pnm.ac.id

ABSTRAK

Perkembangan internet semakin pesat hal ini berimbas kepada semakin banyaknya informasi yang tersedia. Hal ini memudahkan seseorang dalam melakukan penjiplakan suatu karya termasuk source code. Penjiplakan menurut Kamus Besar Bahasa Indonesia (KBBI) berarti menggambar atau menulis garis-garis gambaran atau tulisan yang telah tersedia (dengan menempelkan kertas kosong pada gambar atau tulisan yang akan ditiru), mencontoh atau meniru tulisan atau pekerjaan orang lain, mencuri karangan orang lain dan mengakui sebagai karangan sendiri, mengutip karangan orang lain tanpa seizin penulisnya. Penelitian ini bertujuan untuk mendesain aplikasi pendeteksian plagiat source code pada bahasa pemrograman java menggunakan algoritma Rabin-Karpp. Metode penelitian yang digunakan adalah R&D dan dalam pembangunan sistem menggunakan metode prototipe. Aplikasi pendeteksian plagiat yang dihasilkan berbasis desktop sedangkan untuk sistem pelaporan berbasis web. Aplikasi yang dibangun berhasil mendeteksi source code yang sama antara berkas yang diuji dengan berkas yang ada pada repository. Source code yang sama antara berkas yang diuji dengan berkas yang ada pada repository ditandai dengan warna merah, sementara source code yang tidak identik ditandai dengan warna hitam.

Kata Kunci : Plagiat, algoritma, source code

PENDAHULUAN

Maraknya kasus penjiplakan oleh golongan intelektual menjadi suatu tragedi dalam dunia pendidikan Indonesia seperti kasus profesor termuda bidang hubungan internasional yang diberhentikan secara tidak hormat di tahun 2010. Akibat banyaknya informasi tersedia secara *online* maka kebiasaan *copy–paste* tanpa menyebutkan referensi menjadi mudah dilakukan. Sehingga karya ilmiah yang dibuat menjadi hasil plagiat dari karya ilmiah lain. Namun dikarenakan sebagian besar karya ilmiah belum dilindungi Undang-Undang Hak atas Kekayaan Intelektual (HaKI) maka plagiarisme digolongkan sebagai kejahatan akademik yang termasuk sebagai pelanggaran etika dan sulit untuk dipidanakan. Sebagai langkah awal untuk mencegah kasus serupa diperlukan cara mendeteksi kemungkinan terjadinya penjiplakan seperti di lingkungan perguruan tinggi yaitu utamanya pada hasil tugas akhir calon Ahli Madya D3, sarjana S1 maupun tesis calon sarjana S2 dan disertasi calon sarjana S3 yang rawan penjiplakan.

Praktik plagiat tidaklah menjadi hal asing lagi, apalagi di kalangan mahasiswa yang hampir setiap hari mengerjakan tugas yang diberikan oleh dosen. Tak terkecuali pula, mahasiswa sering mendapat tugas untuk membuat program aplikasi dengan menggunakan bahasa pemrograman tertentu Dalam pengerjaan tugas tersebut, praktik plagiat tak terelakkan lagi untuk dilakukan mengingat waktu pengerjaan tugas yang terbatas dan tidak adanya motivasi untuk berusaha menyelesaikan tugas dengan kemampuan sendiri. Praktik plagiat dilakukan dengan cara tukar-menukar kode program (*source code*) yang telah berhasil. Mahasiswa yang memplagiat dapat dengan mudah menyalin atau mengganti kode program yang telah didapatkan secara cepat dengan menggunakan fitur-fitur yang disediakan oleh komputer. Untuk mengatasi praktik plagiat, tidaklah cukup hanya mengingatkan kepada mahasiswa bahwa tindakan plagiat tidak baik dilakukan. Pendeteksian praktik plagiat merupakan solusi yang sebaiknya dilakukan sehingga tindakan curang tersebut dapat diminimalisasi.

Berdasarkan uraian di atas, Penelitian ini di fokuskan untuk membuat aplikasi yang mampu mendeteksi plagiat, yang permasalahannya diperoleh sebagai berkut:

(1) Bagaimana mendesain dan mengembangkan aplikasi pendeteksian plagiarisme?
(2) Bagaimana menentukan sebuah *source code*, teks, kalimat, karya ilmiah atau jurnal termasuk dalam plagiat atau bukan?
(3) Bagaimana menerapkan algoritma Rabin-Karpp yang akan digunakan dalam pengembangan aplikasi pendeteksian plagiat?

KAJIAN TEORI

Plagiat adalah teknik penyalinan atau meniru karya orang lain yang diklaim menjadi hasil karya sendiri. Tidak adanya motivasi ataupun kemudahan dalam proses penyalinan dengan harapan tidak diketahui orang lain menjadi alasan utama terjadinya praktik plagiat. Beberapa jenis plagiat yang dikenal selama ini, yaitu:

a. *Word-for-word plagiarism* : menyalin setiap kata secara langsung tanpa diubah sedikitpun
b. *Plagiarism of the form of a source* : menyalin dan atau menulis ulang kode-kode program tanpa mengubah struktur dan jalannya program
c. *Plagiarism of authorship*: mengakui hasil karya orang lain sebagai hasil karya sendiri dengan cara mencantumkan nama sendiri menggantikan nama pengarang sebenarnya

Beberapa contoh praktik plagiat pada sebuah program, yaitu:

a) Leksikal: perubahan pada kode (*source code*) program, misalnya:
    1. Komentar diubah (ditambah, dikurangi, atau diganti)
    2. Format penulisan diubah
    3. Nama variabel diubah
b) Struktural : perubahan struktur program
    1. Perubahan urutan algoritma yang tidak mengubah jalannya program
    2. Prosedur diubah menjadi fungsi atau sebaliknya
    3. Pemanggilan prosedur diganti dengan isi prosedur itu sendiri.

Algoritma Rabin-Karpp

Sebuah algoritma pencarian *string* sederhana yaitu *brute-force algorithm* yang kemudian berkembang menjadi beberapa algoritma diantaranya adalah algoritma Knuth-Morris Pratt (KMP), algoritma Boyer-Moore (BM) dan Algoritma Rabin dan Karpp. Menurut (Abdeen, 2011) pada prinsipnya algoritma Rabin-Karp menghitung sebuah fungsi *hash* untuk mencari pola didalam sebuah teks yang diberikan. Setiap karakter M *subsequence* dari pada teks akan dikomparasi, jika nilai *hash* tidak sama algoritma akan menghitung nilai *hash* untuk karakter M *subsequence* berikutnya. Dan jika nilai *hash* sama maka algoritma akan melakukan perbandingan secara *brute-force* antara pola dan karakter M *subsequence,* dengan cara ini hanya akan ada satu perbandingan per teks *subsequence* dan *brute-force* hanya dibutuhkan jika nilai *hash* cocok atau sama. (Jain, et., al, 2012). Menurut Arliandinda dalam (Mutiara, et al, 2008) terdapat empat kategori proses perbandingan yaitu : a)Dari kanan ke kiri b)Dari kiri ke kanan c)Dalam order spesifik d) Dalam order apapun.

Berdasarkan keempat kategori tersebut algoritma Rabin-Karp termasuk dalam kategori dari kiri ke kanan. Algoritma Rabin-Karp menerapkan fungsi *hash* yang menyediakan metode sederhana untuk mencegah kompleksitas waktu $O(m^2)$. Fungsi *hash* setidaknya harus menyediakan empat properti yaitu :

1. Mampu melakukan komputasi secara efisien
2. Diskriminasi string yang tinggi
3. Fungsi *hash* (y[j+1 ... j+m]) harus mudah mengkomputasi dari :
    - *Hash* (y[j...j+m-1])
    - *Hash* (y[j+m])

Algoritma Rabin-Karp menandai langkah-langkah berikut ini :

a) Menerapkan fungsi *hash*
b) Fase preproses dalam kompleksitas waktu O(m) dan waktu yang konstan
c) Fase pencarian dalam kompleksitas waktu O(m)
d) O(n+m) mengestimasi waktu aktif

Kerangka Berpikir

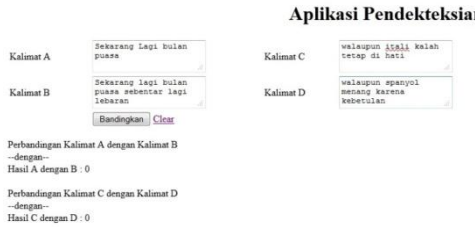
Plagiat adalah masalah yang sering terjadi pada lingkungan pendidikan, hal ini difasilitasi dengan semakin mudahnya proses pengiriman informasi melalui berbagai media. Beberapa pendekatan telah digunakan untuk mendeteksi plagiat seperti menggunakan fingerprinting dari dokumen (Schelimer, et al, 2003) atau yang lebih lanjut lagi menggunakan conceptual footprint yang menggunakan thesaurus dari suatu ide untuk mendeteksi plagiat (Dreher, 2007), atau yang menggunakan database daftar sinonim sendiri (Anzelmi, et al, 2011), juga pendeteksian berbasis citasi (Gipp, et al, 2011). Beberapa sistem plagiat sudah berhasil dibuat, misalnya sistem untuk

pendeteksian plagiarism pada tugas pemrograman (Maharani, et al, 2011) aplikasi yang berbasis web seperti Moss dan JPlag (Lukashenko, 2007), juga aplikasi yang berbasis desktop seperti WCopyFind.

Sistem yang diusulkan akan berusaha mencoba memecahkan masalah plagiarisme pada lingkungan fakultas teknik dengan cara menggunakan suatu metode pendeteksian yang menggunakan algoritma pencocokan string (string matching) yaitu Rabin-Karpp, selain itu sistem yang akan dibangun menyediakan fungsi-fungsi lain yang dibutuhkan sesuai dengan hasil tahapan analisis kebutuhan.

METODE PENELITIAN

Proses pembuatan perangkat lunak aplikasi pendeteksian plagiat menggunakan model *prototyping*, dimana tahapan-tahapan pada model *prototyping* yaitu *Planning*, *Analysis*, *Design*, *Implementation*, *System Prototype*, *Implementation* dan *System*. Alasan digunakannya model *prototyping* karena dapat meminimalisir implementasi sistem yang belum sempurna atau belum stabil serta lebih cepat merespon kebutuhan pengguna. *System prototype* yang sudah dicoba dari sistem pendeteksian plagiat, dimana pada *system prototype* ini yang dibandingkan adalah dua buah kalimat source code. Semakin kecil nilai yang dikeluarkan oleh model matematis pada *system prototype* ini maka bisa dikatakan kalimat tersebut identik atau plagiat akan tetapi sebaliknya apabila angka yang dikeluarkan oleh *system prototype* tersebut besar maka kedua kalimat tersebut tidak identik atau bukan plagiat. Berikut adalah *system prototype* yang sangat sederhana tentang penggambaran aplikasi pendeteksi plagiat :



Gambar 1 *System prototype* Input

Pada Gambar 1 pengguna diminta memasukan empat buah kalimat, kalimat a akan dibandingkan kalimat b dan kalimat c akan dibandingkan dengan kalimat d, hasilnya seperti pada Gambar.2 berikut :

Gambar 2 *System prototype* Output

Sistem akan mengeluarkan angka dari perbandingan keduanya, apabila angka yang dihasilkan kecil atau mendekati nol maka bisa dikatakan kalimat tersebut terindikasi plagiat tapi tidak sebaliknya.

Berdasarkan *system prototype* tersebut, maka sistem pendeteksian bukan hanya terbatas pada kalimat melainkan file, dimana file-file yang akan dibandingkan akan *dimuat* kedalam sistem dan sistem tersebut akan mendeteksi apakah file-file tersebut plagiat atau tidak. Nantinya di dalam penelitian ini akan dikembangkan lebih lanjut sistem yang lebih lengkap dan lebih banyak memiliki fungsi-fungsi untuk mendeteksi plagiat.

Adapun langkah-langkah penelitian yang akan dilakukan yaitu :

1) Studi kepustakaan, dimana untuk menemukan filosofis dan teori-teori baik mengenai algoritma Rabin-Karpp maupun penelitian-penelitian yang terkait dengan aplikasi pendeteksian plagiat, termasuk didalamnya mengidentifikasi masalah-masalah yang sering muncul dan cara penyelesaiannya.

2) Perancangan sistem dan arsitektur, langkah ini akan mencakup terkait dengan penerapan model *prototyping* melalui pendekatan *object oriented design*. Selain itu disertai dengan perancangan arsitektur sistem yang akan diterapkan
3) Pembuatan *system prototype* dan *system*, dimana untuk mengembangkan lebih lanjut *system prototype* awal yang telah dibuat selagi diimplementasikan secara parsial sehinggal nantinya tercipta *system* yang utuh dan stabil.
4) Implementasi, perawatan dan sosialisasi sistem, pada tahapan akhir dilakukan implementasi yang sesungguhnya dalam arti diterapkan untuk menguji source code, tugas, makalah, karya ilmiah, jurnal dan lainnya disertai dengan perawatan sistem serta sosialisasi penggunaan sistem tersebut di lingkungan Jurusan pada khususnya atau bahkan pada tingkatan Direktorat.

Metode Penelitian

Model rancangan eksperimen untuk menguji produk rancangan

Pengujian yang akan dilakukan terhadap sistem sebagian besar melalui pengujian dengan metode BlackBox, yaitu dengan membuat suatu input tertentu dan memperhatikan output yang dihasilkan, apabila output yang dihasilkan sesuai dengan output yang seharusnya (menurut spesifikasi requirement sistem) maka sistem berhasil lulus pada item pengujian yang diujikan. Rencana pengujian pengujian yang akan dilakukan :

Tabel 3.1 Pengujian Sistem

No	Item Pengujian	Jenis Pengujian
1	Pengujian Penginputan dan Penghapusan Berkas di Database	BlackBox
2	Pengujian pendeteksian plagiat pada berkas yang memiliki nilai kesamaan 0%, 25% 50%, 75% dan 100%.	BlackBox
3	Pengujian metode pengkonversian representasi data	BlackBox
4	System test menggunakan sample data real yaitu menggunakan data tugas akhir jurusan komputer akuntansi konsentrasi aplikasi.	BlackBox

Sampel data yang akan digunakan adalah berupa data tugas akhir dari alumni jurusan komputer akuntansi konsentrasi aplikasi Politeknik Negeri Madiun. Sedangkan Prosedur pengumpulan data adalah dengan meminta kerja sama dengan tenaga administrasi jurusan untuk menyediakan dan untuk mengumpulkan data tugas akhir alumni konsentrasi aplikasi

HASIL DAN PEMBAHASAN

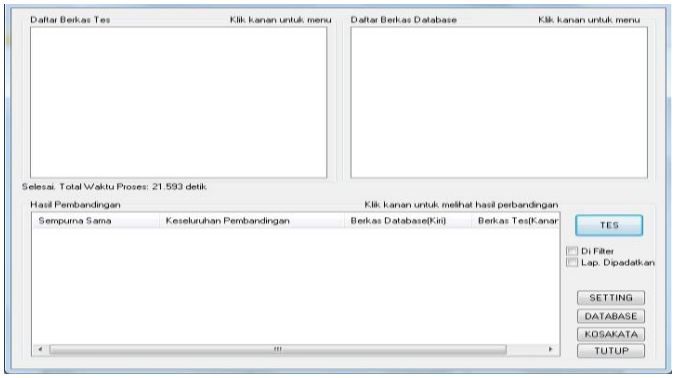
Perancangan Arsitektur Sistem

Arsitektur sistem yang digunakan adalah client-server yaitu sistem akan menggunakan server basis data yang akan menjadi repository dari dokumen- dokumen yang akan dijadikan bahan pembanding untuk pendeteksian plagiarism. Pada arsitektur ini, pada saat akan mendeteksi plagiat, aplikasi plagiat detector pada client akan mengirimkan permintaan dokumen pada server basis data. Setelah aplikasi mendapatkan koleksi dokumen, aplikasi akan melakukan proses pendeteksian dan membuat report plagiat. Report plagiat ini kemudian akan dikirimkan ke web browser untuk ditampilkan.

Model Pendeteksian

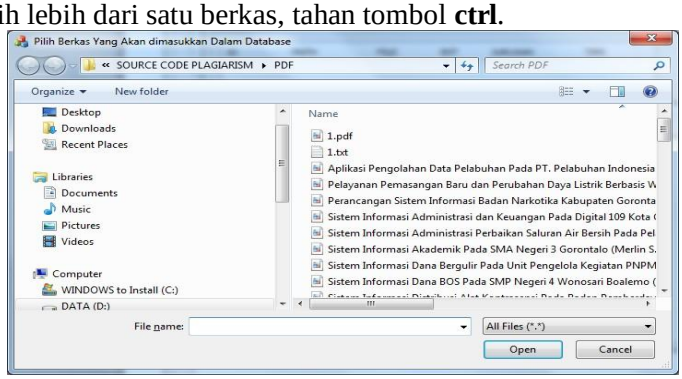
Model pendeteksian yang akan digunakan pada sistem ini adalah menggunakan algoritma pencocokan string Rabin-Karp yang dimodifikasi. Proses pendeteksian plagiarism masih mengandalkan tenaga manusia, sehingga sangat terbatas pada kemampuan memperhatikan dan mengingat untuk membandingkan dokumen. Beberapa alat bantu yang digunakan misalnya seperti aplikasi pembandingan secara visual (gambar dibawah). Pada alat tersebut dapat mendeteksi plagiat dengan membandingkan dua buah dokumen (terlihat pada jendela kiri dan kanan), baris berlatar putih termasuk plagiat, baris yang berwarna adalah baris yang tidak termasuk plagiat. Tool tersebut membantu dalam memperhatikan perbedaan dan persamaan pada suatu dokumen, tetapi belum cukup mengingat banyaknya tugas kuliah, tugas akhir/skripsi yang harus diperiksa..

Saat aplikasi pendeteksian plagiat dijalankan maka aplikasi akan menampilkan form utama.



Gambar 3. Form Utama Aplikasi Pendeteksian Plagiat

Pada proses awalnya Aplikasi belum meng-create database, untuk meng-create database user harus masuk pada form database dengan menekan tombol **database**. Untuk memasukkan berkas ke dalam database, pengguna dapat menekan tombol **tambah**, sehingga akan muncul form pilih berkas. Pada form ini pengguna dapat memilih satu atau lebih berkas yang akan dimuat ke dalam database, untuk memilih lebih dari satu berkas, tahan tombol **ctrl**.



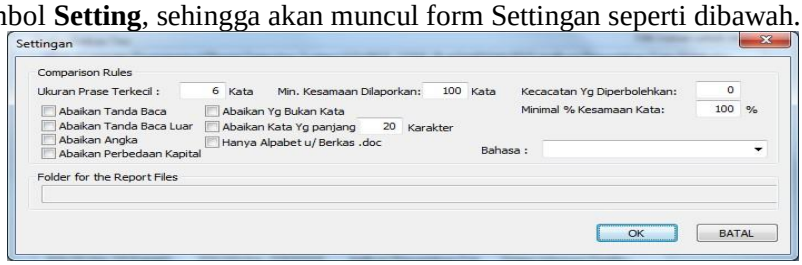
Gambar 4. Form pilih berkas

Setelah database dibuat, pengguna dapat kembali ke form utama, dan dapat memuat berkas dari database yang telah dibuat sebelumnya dengan melakukan klik kanan mouse pada bagian "Daftar Berkas Database" kemudian memilih **Load dari Database**. Apabila database telah dimuat, maka berkas yang ada dalam database tersebut akan ditampilkan.

Setelah database selesai dimuat, selanjutnya adalah memuat berkas yang akan diuji, caranya adalah dengan melakukan click kanan pada bagian "Daftar Berkas Tes" dan memilih **cari berkas**.

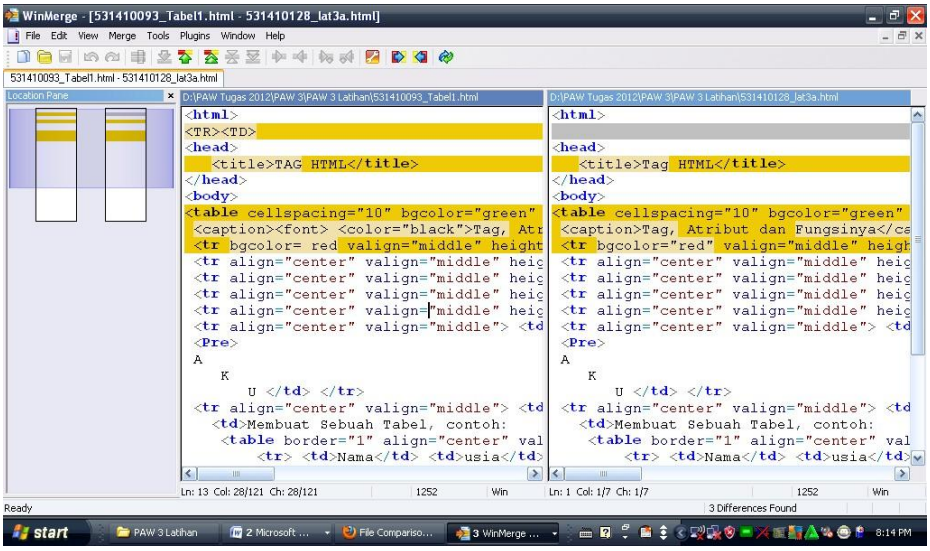
Apabila berkas yang akan diuji selesai dimuat maka berkas akan ditampilkan pada bagian "Daftar Berkas Tes".

Sebelum melakukan pendeteksian, pengguna dapat melakukan pengaturan proses pendeteksian, yaitu dengan menekan tombol **Setting**, sehingga akan muncul form Settingan seperti dibawah.



Gambar 5. Form Settingan

Setelah selesai melakukan pengaturan, pengguna dapat menekan tombol **OK** untuk kembali ke form utama. Untuk memulai proses pendeteksian pengguna dapat menekan tombol **TES**, sehingga aplikasi akan memulai pendeteksian dan menampilkan hasilnya pada bagian "Hasil Pembandingan".



Gambar 4.1 Aplikasi pembantu pendeteksian secara visual.

KESIMPULAN

1. Aplikasi pendeteksian plagiat didesain berbasis *desktop* sedangkan untuk sistem pelaporan berbasis web. Pada proses pengembangannya menggunakan model prototipe.
2. Aplikasi yang dibangun berhasil mendeteksi kalimat yang sama antara berkas yang diuji dengan berkas yang ada pada *repository*. Kalimat yang sama antara berkas yang diuji dengan berkas yang ada pada *repository* ditandai dengan warna merah, sementara kalimat yang tidak identik ditandai dengan warna hitam.
3. Peneliti melakukan modifikasi pada algoritma rabin-karpp khususnya pada teknik *hashing*, hal ini dilakukan agar proses pencarian kalimat menjadi lebih cepat.

DAFTAR PUSTAKA

[1] Abdeen, Ali.,Rawan, 2011, *An Algorithm for String Searching Based on Brute-Force Algorithm,* Intenational Journal of Computer Science and Network Security, Vol.11 No.7.

[2] Anzelmi, Daniele., et. Al, 2011, Plagiarism Detection Based SCAM Algorithm, Proceedings of the International MultiConference of Engineers and Computer Scientist 2011, Vol.1.

[3] Dreher,. Heinz, 2007, Automatic Conceptual Analysis for Plagiarism Detection, Issues in Informing Science and Information Technology Volume 4,

[4] Firdaus, Bagus, 2008, *Deteksi Plagiat Dokumen Menggunakan Algoritma Rabin- Karp,* Makalah If2251 Strategi Algoritmik Tahun 2008

[5] Gipp, Bela, et. Al, 2011, Citation Pattern Matching Algorithms for Citation-based Plagiarism Detection: Greedy Citation Tiling, Citation Chunking and Longest Common Citation Sequence, scholar.google.com, diakses 14 November 2012.

[6] Jain, Shivani., Rao, Nersimha, A.L., Agarwal, Pankaj, *A Relative Study of PatternMatching Algorithms*, Journal of Computing Technologies, Vol.2 Issue 1.

[7] Lukashenko, Romans., et. al., 2007, Computer-Based Plagiarism Detection Methods and Tools: An Overview, International Conference on Computer Systems and Technologies - CompSysTech'07.

[8] Mutiara, Benny. A, Agustina., Sinta, 2008, *Anti Plagiarism Application With Algorithm Karp-Rabin At Thesis in Gunadharma University*, Gunadharma University, Jakarta.

[9] Maharani., Puanta Della, et. al., 2012, Penilai Otomatis Praktikum Pemrograman dengan Pendeteksi Plagiarisme untuk Praktikum Pengenalan Teknologi Informasi (PTI) B, Jurnal Sarjana Institut Teknologi Bandung Bidang Teknik Elektro dan Informatika Volume 1, Number 2, Juli 2012.

[10] Schelimer., Saul, et. Al, 2003 Winnowing: Local Algorithms for Document Fingerprinting, http://dl.acm.org/citation.cfm?id=872770, diakses 13 November 2012