STMIK Amik Riau

SATIN – Sains dan Teknologi Informasi

journal homepage : http://jurnal.stmik-amik-riau.ac.id

Sistem Pendeteksi Tingkat Kesamaan Teks pada Pengusulan Proposal Penelitian Internal Menggunakan Algoritma Rabin-Karp

Rahmaddeni
Teknik Informatika
STMIK Amik Riau
rahmaddeni@stmik-amik-riau.ac.id

Didik Sazali
Teknik Informatika
STMIK Amik Riau
didiksazali@gmail.com

Agustin
Teknik Informatika
STMIK Amik Riau
agustin@stmik-amik-riau.ac.id

STMIK Amik Riau yang merupakan perguruan tinggi komputer di Provinsi Riau memiliki suatu unit lembaga yang mengelola penelitian yang bernama LPPM STMIK Amik Riau. Untuk mewujudkan visi LPPM STMIK Amik Riau, lembaga ini memulainya dari hal internal terlebih dahulu berupa adanya pengelolaan manajemen penelitian yang baik dalam sebuah website yang dapat diakses secara online. Permasalahan yang terjadi selama ini dalam sebuah website LPPM STMIK Amik Riau adalah dalam hal pengajuan proposal penelitian internal. Dalam pengajuan proposal internal yang dilakukan oleh dosen STMIK Amik Riau melalui sistem yang ada, belum bisa mendeteksi adanya kesamaan teks yang diajukan melalui paparan abstrak yang diberikan. Sistem yang ada menerima semua usulan yang diberikan dalam hal abstrak tanpa memberikan rekomendasi ke reviewer akan tingkat kesamaan usulan penelitian yang diajukan dengan penelitian yang ada. Menanggapi permasalahan yang ada dan untuk mewujudkan visi LPPM STMIK Amik Riau, maka perlu adanya sistem yang mampu memberikan rekomendasi bagi reviewer akan tingkat kesamaan teks melalui usulan abstrak penelitian yang diajukan oleh dosen STMIK Amik Riau. Tujuan dari penelitian ini adalah menerapkan algoritma Rabin-Karp pada website LPPM STMIK Amik Riau untuk mendeteksi tingkat kesamaan abstrak penelitian internal STMIK Amik Riau agar meminimalisir terjadinya penelitian yang plagiat. Selain berbasis website, sistem juga dapat diakses melalui perangkat mobile android.

Kata kunci : Kesamaan Teks, Proposal Internal, Algoritma Rabin-Karp, Website

1. Pendahuluan

STMIK Amik Riau yang merupakan perguruan tinggi komputer di Provinsi Riau memiliki suatu unit lembaga yang mengelola penelitian yang bernama LPPM STMIK Amik Riau. Lembaga ini memiliki visi untuk menjadi lembaga penelitian dan pengabdian masyarakat yang memiliki produktifitas dan kualitas tinggi serta mampu berperan aktif dalam pengembangan pendidikan dan pengetahuan (RIP STMIK Amik Riau, 2016). Untuk mewujudkan visi tersebut LPPM STMIK Amik Riau memulainya dari hal internal terlebih dahulu berupa adanya pengelolaan manajemen penelitian yang baik dalam sebuah website yang dapat diakses secara online.

Permasalahan yang terjadi selama ini dalam sebuah website LPPM STMIK Amik Riau adalah dalam hal pengajuan proposal penelitian internal. Dalam pengajuan proposal internal yang dilakukan oleh dosen STMIK Amik Riau melalui sistem yang ada, belum bisa mendeteksi adanya kesamaan teks yang diajukan melalui paparan abstrak yang diberikan. Sistem yang ada menerima semua usulan yang diberikan dalam hal abstrak tanpa memberikan rekomendasi akan tingkat kesamaan usulan penelitian yang diajukan dengan penelitian yang ada. Rekomendasi yang diberikan dari sistem merupakan tolak ukur bagi LPPM sendiri maupun reviewer yang akan mereview usulan proposal penelitian internal.

Menanggapi permasalahan yang ada dan untuk mewujudkan visi LPPM STMIK Amik Riau 2016-2020, maka perlu adanya sistem yang mampu memberikan rekomendasi bagi reviewer akan tingkat kesamaan teks melalui usulan abstrak penelitian yang diajukan oleh dosen STMIK Amik Riau. Tingkat kesamaan teks melalui abstrak penelitian yang diusulkan dapat diketahui dengan menerapkan

algoritma Rabin-Karp dalam website LPPM STMIK Amik Riau yang ada.

Algoritma *Rabin-Karp* adalah algoritma pencocokan *string* yang menggunakan fungsi *hash* sebagai pembanding antara *string* yang dicari (m) dengan *substring* pada teks (n) (Doddi Aria Putra et al, 2015).

Penelitian dengan menerapkan algoritma Rabin-Karp telah banyak dibahas oleh beberapa peneliti. Penelitian tentang penggunaan algoritma Rabin Karp pada MPI oleh Nupur Kohli dari San Jose State University dalam penelitiannya yang berjudul "Implementation of Rabin Karp String Matching Algorithm Using MPI". MPI (Message Passing Interface) adalah spesifikasi API (Application Programming Interface) yang memungkinkan terjadinya komunikasi antar komputer pada network dalam usaha untuk menyelesaikan suatu tugas. Dalam penelitiannya, penulis ingin mencapai kinerja maksimum pada algoritma string matching untuk memproses file berukuran besar.

Sonawane Kiran Shivaji dari Ahmednagar, Maharashtra dalam penelitiannya yang berjudul "Plagiarism Detection by using Karp-Rabin and String Matching Algorithm Together" dan diterbitkan oleh International Journal of Computer Applications (0975 – 8887) Volume 116 – No. 23, April 2015 membahas tentang penggunaan algoritma Rabin Karp dalam mengetahui penjiplakan pada program dengan menerapkan fungsi hash. Penulis memberikan batas presentase plagiat sebesar 10%. Penelitian selaras dengan penggunaan algoritma tersebut juga diteliti oleh Hari Bagus Firdaus dari Institut Teknologi Bandung dalam penelitiannya yang berjudul "Deteksi Plagiat Dokumen Menggunakan Algoritma Rabin Karp". Peneliti menyimpulkan kemiripan pola antar dua buah dokumen dapat dicari dengan menerapkan prinsip algoritma pencarian string Rabin-Karp. Algoritma Rabin-Karp menghasilkan efisiensi waktu yang baik dalam mendeteksi string yang memiliki lebih dari satu pola. Hal ini membuat algoritma Rabin-Karp dimanfaatkan dalam melakukan pendeteksian terhadap tindak plagiat dokumen.

Berdasarkan permasalahan yang ada dan beberapa penelitian yang membahas penerapan algoritma Rabin-Karp, maka peniliti bermaksud untuk melakukan penelitian dalam membangun sebuah sistem pendeteksian tingkat kesamaan teks pada pengusulan proposal penelitian internal dengan menggunakan algoritma Winnowing dalam website LPPM STMIK Amik Riau berbasis mobile.

2. Algoritma Rabin-Karp

Algoritma *Rabin-Karp* adalah algoritma pencocokan *string* yang menggunakan fungsi *hash* sebagai pembanding antara *string* yang dicari (m) dengan *substring* pada teks (n). Apabila *hash value* keduanya sama maka akan dilakukan perbandingan sekali lagi terhadap karakter-karakternya. Apabila hasil keduanya tidak sama, maka *substring* akan bergeser ke kanan. Pergeseran dilakukan sebanyak (n-m) kali (Doddi Aria Putra et al, 2015).

Langkah-langkah yang dapat dilakukan dalam penerapan Algoritma Rabin-Karp (Handrie Noprison et al, 2013) yaitu :

1. *Preprocessing*

Tahap preprocessing harus dilalui untuk menentukan keyword pada kedua dokumen yang akan dilakukan pengujian yaitu dokumen asli dan dokumen uji. Pada tahap preprocessing ada beberapa tahap yang harus dilalui oleh sistem antara lain :

1.1 Case Folding

Case folding adalah cara mengubah semua huruf kapital yang ada pada kalimat menjdai huruf kecil, dan karakter selain huruf dihilangkan.

Sintax yang digunakan didalam program pada penelitian ini pada tahapan case folding yaitu :

```
//CaseFolding
$this->k1   =   strtolower($this->word1);
$this->k2   =   strtolower($this->word2);
```

1.2 Tokenizing

Tokenizing adalah proses menghilangkan tanda baca pada kalimat seperti titik (.), koma (,), plus (+) dan lain-lain.

Sintax yang digunakan didalam program pada penelitian ini pada tahapan case tokenizing yaitu :

```
//Tokenizing
$this->kal1     =     str_replace('','',preg_replace("/[^a-zA-Z0-9\s-]/",     "",     $this->k1));
$this->kal2     =     str_replace('','',preg_replace("/[^a-zA-Z0-9\s-]/",     "",     $this->k2));
```

1.3 Filtering

Filtering adalah proses mengambil kata-kata penting seperti pada, untuk, ada, dari, akan dan lain-lain.

Sintax yang digunakan didalam program pada penelitian ini pada tahapan case filtering yaitu :

```
//Filtering
$filter=array("untuk","sebuah","yang","dapat","tidak","bisa","dengan","dan","jika","maka","pada","itu","ada",

"dari","akan","karena","oleh","dimana","anda","lain","telah");

$this->kalimat1                    =
str_replace($filter,"",$this-
```

```
>kal1);
$this->kalimat2                    =
str_replace($filter,"",$this-
>kal2);
```

2. *Parsing K-gram*

Parsing K-gram adalah proses membentuk pola kata pada teks dengan memecah kata menjadi potongan-potongan dimana setiap potongan mengandung karakter sejumlah k dari sebuah kata yag secara kontinuitas dibaca dari awal hingga akhir kalimat. Sintax yang digunakan didalam program pada penelitian ini pada tahapan parsing k-gram yaitu :

```
//Parsing                          K-gram
$this->arr_n_gram1       =       $this-
>n_gram($this->kalimat1,         $this-
>n_gram_value);
$this->arr_n_gram2       =       $this-
>n_gram($this->kalimat2,         $this-
>n_gram_value);
```

3. *Hashing*

Hasing merupakan proses untuk mentransformasi string menjadi suatu nilai yang unik (*hash value*) dengan panjang tertentu (*fixed length*) yang berfungsi sebgai penanda string tersebut. Pada sistem ini proses hashing memanfaatkan tabel ASCII dengan rumus hash :

$$H = C1*a^{(k-1)}+C2*a^{(k-2)}+.....+C5*a^{0}$$

Keterangan :

H = Nilai Hash
C = Nilai ASCII suatu karakter
a = Basis (tidak boleh 1 dan 0)
k = Banyaknya karakter

Sintax yang digunakan didalam program pada penelitian ini pada tahapan hasing yaitu :

```
//Hashing
$this->arr_rolling_hash1    =    $this-
>rolling_hash($this->arr_n_gram1);
$this->arr_rolling_hash2    =    $this-
>rolling_hash($this->arr_n_gram2);
```

4. Similarity

Inti dari k-grams dibagi mejadi 2 tahap. Pertama , membagi kata menjadi k-grams. Kedua, mengelompokkan hasil terms dari k-grams yang sama, kemudian untuk menghitung similarity dari kumpulan kata tersebut maka digunakan rumus pengukuran nilai similaritas untuk pasangan kata yang digunakan

$$S = ((2*Nt) / (Nx + Ny) * 100$$

Keterangan :

S = Similaritas
Nt = Jumlah hashing yang sama
Nx = Total substring asli (k-gram asli)
Ny = Total substring usul (k-gram usul)

Sintax yang digunakan didalam program pada penelitian ini pada tahapan similarity yaitu :

```
//Similarity
$this->similarity       =       $this-
>similarity($this->arr_rolling_hash1,
$this->arr_rolling_hash2);
```

3. Persentase Nilai Similaritas

Untuk menentukan jenis dokumen yang diuji ada 5 jenis penilaian persentase similaritas (Handrie Noprison et al, 2013) :

1. 0% : Dokumen tersebut benar-benar berbeda baik dari segi isi dan kalimat secara keseluruhan.
2. < 15% : Dokumen tersebut hanya mempunyai sedikit kesamaan.
3. 15-50% : Dokumen tersebut termasuk plagiat tingkat sedang.
4. 50-80% : Dokumen tersebut mendekati plagiarisme.
5. 80-100% : Dokumen tersebut adalah plagiat.

4. Penerapan Algoritma Rabin-Karp

Secara garis besar, langkah-langkah dalam penerapan algoritma *Rabin-Karp* adalah sebagai berikut:

1. Preprocessing yang dilalui dalam 3 tahapan yaitu Case Folding, Tokenizing dan Filtering.
2. Pembentukan parsing *gram* dengan ukuran k.
3. Penghitungan nilai *hash*.
4. Perolehan nilai simalarity

Berikut adalah contoh kalimat yang akan dideteksi menggunakan Algoritma Rabin-Karp:

Potongan Abstrak 1 (Usulan) :

"Rancang Bangun Aplikasi Pendeteksi Penjiplakan Dokumen"

Potongan Abstrak 2 (Yang telah ada di sistem) :

"Rancang Bangun Aplikasi Pendeteksi Penjiplakan pada File Teks"

Langkah-langkah penerapan algoritmanya adalah :

1. *Preprocesing* yang dilakukan dalam tiga tahapan (*Case Folding*, *Tokenizing* dan *Filtering*)

Potongan Abstrak 1 (Usulan) :

Rancang Bangun Aplikasi Pendeteksi Penjiplakan Dokumen
Menjadi :
Rancangbangunaplikasipendeteksipenjiplakandokumen

Potongan Abstrak 2 (Yang telah ada di sistem) :
Rancang Bangun Aplikasi Pendeteksi Penjiplakan pada File Teks
Menjadi :
rancangbangunaplikasipendeteksipenjiplakanpadafileteks

2. Membentuk rangkaian *k-grams*, misalnya dengan ukuran 7.

   Untuk Potongan Abstrak 1 (Usulan), terbentuklah rangkaian *k-gram* sebagai berikut :
   rancang ancangb ncangba cangban angbang ngbangu gbangun banguna angunap ngunapl gunapli unaplik naplika aplikas plikasi likasip ikasipe kasipen asipend sipende ipendet pendete endetek ndeteks deteksi eteksip teksipe eksipen ksipenj sipenji ipenjip penjipl enjipla njiplak jiplaka iplakan plakand lakando akandok kandoku andokum ndokume dokumen

   Sedangkan untuk Potongan Abstrak 2 (Yang telah ada di sistem), terbentuk rangkaian *k-grams* sebagai berikut :
   rancang ancangb ncangba cangban angbang ngbangu gbangun banguna angunap ngunapl gunapli unaplik naplika aplikas plikasi likasip ikasipe kasipen asipend sipende ipendet pendete endetek ndeteks deteksi eteksip teksipe eksipen ksipenj sipenji ipenjip penjipl enjipla njiplak jiplaka iplakan plakanp lakanpa akanpad kanpada anpadaf npadafi padafil adafile dafilet afilete filetek ileteks

3. Perhitungan nilai *hash* menggunakan persamaan 2.1, dimana b=2 dan k=7.

   Perhitungan untuk Potongan Abstrak 1 (Usulan) :
   $H_{(rancang)}$ = ascii(r) * $2^{(6)}$ + ascii(a) * $2^{(5)}$ + ascii(n) * $2^{(4)}$ + ascii(c) * $2^{(3)}$ + ascii(a) * $2^{(2)}$ + ascii(n) * $2^{(1)}$ + ascii(g) * $2^{(0)}$
   = (114 * 64) + (97 * 32) + (110 * 16) + (99 * 8) + (97 * 4) + (110 * 2) + (103 * 1)
   = 7296 + 3104 + 1760 + 792 + 388 + 220 + 103
   = 13663
   $H_{(ancangb)}$ = (13663 – ascii(r) * $2^{(6)}$) * 2 + ascii(b) * $2^{(0)}$
   = (13663 – (114 * 64)) * 2 + (98 * 1)
   = (13663 – 7296) * 2 + 98
   = (6367 * 2) + 98
   = 12743 + 98
   = 12832
   $H_{(ncangba)}$ = (12832 – ascii(a) * $2^{(6)}$) * 2 + ascii(a) * $2^{(0)}$
   = (12832 – (97 * 64)) * 2 + (97 * 1)
   = (12832 – 6208) * 2 + 97
   = (6624 * 2) + 97
   = 13248 + 97
   = 13345

   Maka diperoleh hasil perhitungannya sebagai berikut :
   13663 12832 13345 12720 12871 13443 12916 12745 13058 13808 13641 14205 13531 13097 13883 13542 13361 13392 13188 14061 13518 13697 13165 13517 13059 13430 14033 13328 13834 14077 13546 13760 13281 13741 13499 13540 13740 13255 12793 13287 12987 13659 13348

   Perhitungan untuk Potongan Abstrak 2 (Yang telah ada di sistem) :
   $H_{(rancang)}$ = ascii(r) * $2^{(6)}$ + ascii(a) * $2^{(5)}$ + ascii(n) * $2^{(4)}$ + ascii(c) * $2^{(3)}$ + ascii(a) * $2^{(2)}$ + ascii(n) * $2^{(1)}$ + ascii(g) * $2^{(0)}$
   = (114 * 64) + (97 * 32) + (110 * 16) + (99 * 8) + (97 * 4) + (110 * 2) + (103 * 1)
   = 7296 + 3104 + 1760 + 792 + 388 + 220 + 103
   = 13663
   $H_{(ancangb)}$ = (13663 – ascii(r) * $2^{(6)}$) * 2 + ascii(b) * $2^{(0)}$
   = (13663 – (114 * 64)) * 2 + (98 * 1)
   = (13663 – 7296) * 2 + 98
   = (6367 * 2) + 98
   = 12743 + 98
   = 12832
   $H_{(ncangba)}$ = (12832 – ascii(a) * $2^{(6)}$) * 2 + ascii(a) * $2^{(0)}$
   = (12832 – (97 * 64)) * 2 + (97 * 1)
   = (12832 – 6208) * 2 + 97
   = (6624 * 2) + 97
   = 13248 + 97
   = 13345

   Maka didapatkan hasil sebagai berikut :
   13663 12832 13345 12720 12871 13443 12916 12745 13058 13808 13641 14205 13531 13097 13883 13542 13361 13392 13188 14061 13518 13697 13165 13517 13059 13430 14033 13328 13834 14077 13546 13760 13281 13741 13499 13540 14152 14065 14406 16493 19392 26473 38974 63713 115126 217553 422797 832653

   Potongan Abstrak 1 (Usulan) :
   [13663 12832 13345 **12720**] [12832 13345 12720 12871] [13345 12720 12871 13443] [12720 12871 13443 12916] [12871 13443 12916 **12745**] [13443 12916 12745 13058] [12916 12745 13058 13808]

[12745 13058 13808 13641] [**13058** 13808 13641 14205] [13808 13641 14205 **13531**] [13641 14205 13531 **13097**] [14205 13531 13097 13883] [13531 13097 13883 13542] [13097 13883 13542 13361] [13883 13542 **13361** 13392] [13542 13361 13392 **13188**] [13361 13392 13188 14061] [13392 13188 14061 13518] [13188 14061 13518 13697] [14061 13518 13697 **13165**] [13518 13697 13165 13517] [13697 13165 13517 **13059**] [13165 13517 13059 13430] [13517 13059 13430 14033] [13059 13430 14033 13328] [13430 14033 **13328** 13834] [14033 13328 13834 14077] [13328 13834 14077 13546] [13834 14077 **13546** 13760] [14077 13546 13760 **13281**] [13546 13760 13281 13741] [13760 13281 13741 13499] [13281 13741 13499 13540] [13741 **13499** 13540 13740] [13499 13540 13740 **13255**] [13540 13740 13255 **12793**] [13740 13255 12793 13287] [13255 12793 13287 12987] [12793 13287 12987 13659] [13287 **12987** 13659 13348]

Potongan Abstrak 2 (Yang telah ada di sistem) :
[13663 12832 13345 **12720**] [12832 13345 12720 12871] [13345 12720 12871 13443] [12720 12871 13443 12916] [12871 13443 12916 **12745**] [13443 12916 12745 13058] [12916 12745 13058 13808] [12745 13058 13808 13641] [**13058** 13808 13641 14205] [13808 13641 14205 **13531**] [13641 14205 13531 **13097**] [14205 13531 13097 13883] [13531 13097 13883 13542] [13097 13883 13542 13361] [13883 13542 **13361** 13392] [13542 13361 13392 **13188**] [13361 13392 13188 14061] [13392 13188 14061 13518] [13188 14061 13518 13697] [14061 13518 13697 **13165**] [13518 13697 13165 13517] [13697 13165 13517 **13059**] [13165 13517 13059 13430] [13517 13059 13430 14033] [13059 13430 14033 13328] [13430 14033 **13328** 13834] [14033 13328 13834 14077] [13328 13834 14077 13546] [13834 14077 **13546** 13760] [14077 13546 13760 **13281**] [13546 13760 13281 13741] [13760 13281 13741 13499] [13281 13741 13499 13 40] [13741 **13499** 13540 14152] [13499 13540 14152 14065] [**13540** 14152 14065 14406] [14152 **14065** 14406 16493] [14065 14406 16493 19392] [**14406** 16493 19392 26473] [**16493** 19392 26473 38974] [**19392** 26473 38974 63713] [**26473** 38974 63713 115126] [**38974** 63713 115126 217553] [**63713** 115126 217553 422797] [**115126** 217553 422797 832653]

4. Berikut nilai *hash* yang dihasilkan berdasarkan indeks :

   Untuk Potongan Abstrak 1 (Usulan) :
   [12720,3] [12745,7] [13058,8] [13531,12] [13097,13] [13361,16] [13188,18] [13165,22] [13059,24] [13328,27] [13546,30] [13281,32] [13499,34] [13255,37] [12793,38] [12987,40]
   Sehingga *grams* yang digunakan :
   cangban banguna angunap naplika aplikas ikasipe asipend endetek deteksi eksipen ipenjip enjipla jiplaka lakando akandok andokum

   13663 12832 13345 **12720** 12871 13443 12916 **12745 13058** 13808 **13641** 14205 13531 **13097** 13883 13542 **13361** 13392 **13188** 14061 13518 13697 **13165** 13517 **13059** 13430 14033 **13328** 13834 14077 **13546** 13760 **13281** 13741 **13499** 13540 13740 **13255 12793** 13287 **12987** 13659 13348

   Untuk Potongan Abstrak 2 (Yang telah ada di sistem) :
   [12720,3] [12745,7] [13058,8] [13531,12] [13097,13] [13361,16] [13188,18] [13165,22] [13059,24] [13328,27] [13546,30] [13281,32] [13499,34] [13540,35] [14065,37] [14406,38] [16493,39] [19392,40] [26473,41] [38974,42] [63713,43] [115126, 44]
   Sehingga *grams* yang digunakan:
   cangban banguna angunap naplika aplikas ikasipe asipend endetek deteksi eksipen ipenjip enjipla jiplaka iplakan lakanpa akanpad kanpada anpadaf npadafi padafil adafile dafilet

   13663 12832 13345 **12720** 12871 13443 12916 **12745 13058** 13808 13641 14205 **13531 13097** 13883 13542 **13361** 13392 **13188** 14061 13518 13697 **13165** 13517 **13059** 13430 14033 **13328** 13834 14077 **13546** 13760 **13281** 13741 **13499 13540** 14152 **14065 14406 16493 19392 26473 38974 63713 115126** 217553 422797 832653

5. Similarity

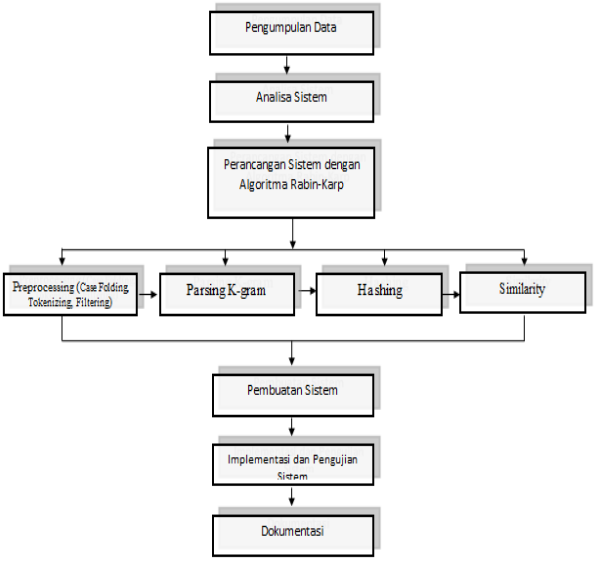
   D1 = [12720] [12745] [13058] [13531] [13097] [13361] [13188] [13165] [13059] [13328] [13546] [13281] [13499] [13255] [12793] [12987]
   D2 = [12720] [12745] [13058] [13531] [13097] [13361] [13188] [13165] [13059] [13328] [13546] [13281] [13499] [13540] [14065] [14406] [16493] [19392] [26473] [38974] [63713] [115126]

   S = ((2*Nt) / (Nx + Ny) * 100
   S = ((2*13) / (16 + 22) * 100
   S = ((26/38) * 100
   S = 68,42 (Kalimat Mendekati Plagiarisme)

Berdasarkan hasil persentase, maka kedua dokumen tersebut bisa dikatakan mendekati plagiarisme.

5. Metodologi Penelitian

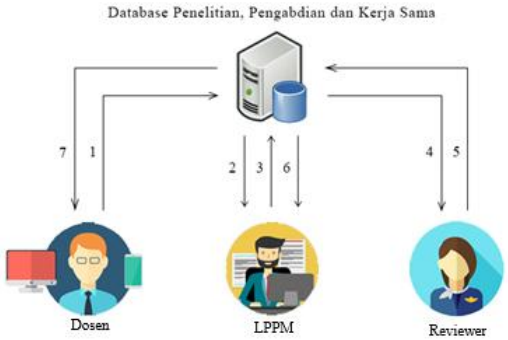
Metodologi penelitian merupakan urutan-urutan yang dilakukan dalam sebuah penelitian. Kerangka kerja (*frame work*) dalam penelitian ini digambarkan pada gambar berikut ini:



Gambar 1. *Flowchart* kerangka kerja penelitian

6. Analisa Sistem

Analisa sistem yang dimaksud meliputi analisa yang ingin diusulkan untuk memperbaiki kekurangan dari sistem yang sedang berjalan serta adanya penambahan modul kerja sama ke dalam sistem. Hal ini akan lebih menghemat *resource* karena pengelolaan penelitian dosen, pengabdian masyarakat dan kerja sama berada dalam sebuah sistem. Untuk gambaran sistem yang diusulkan dapat dilihat pada gambar 2 dibawah ini.



Gambar 2. Sistem yang diusulkan

Adapun keterangan dari gambar 2 diatas dapat dijelaskan sebagai berikut:

1. Dosen mengunggah proposal ke dalam sistem. Kemudian sistem akan mendeteksi kesamaan abstrak yang ada di sistem dengan abstrak usulan peneliti.
2. Admin LPPM melihat proposal yang baru masuk dan belum direview.
3. Admin LPPM menunjuk *reviewer* untuk menilai proposal.
4. Reviewer menerima data proposal untuk dinilai.
5. Reviewer menilai proposal yang masuk.
6. Admin melihat proposal yang sudah dinilai.
7. Dosen mengetahui apakah proposalnya lulus atau tidak.

6.1 Kebutuhan Hardware dan Software

Adapun kebutuhan hardware dan software yang digunakan yaitu:

1. Perangkat keras (*hardware*)

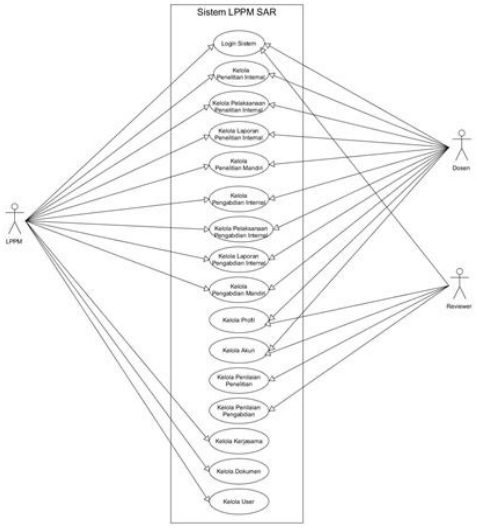
Perangkat keras yang digunakan dalam membangun sistem ini adalah laptop dengan spesifikasi *processor core* i3, *Random Access Memory* (RAM) dengan kapasitas 4 GB serta *harddisk* dengan kapasitas 500 GB.

2. Perangkat lunak (*software*)

Perangkat lunak yang digunakan untuk membangun sistem ini yaitu, sistem operasi Windows 10, XAMPP sebagai paket aplikasi dengan fitur PHP sebagai bahasa pemrograman dan MySQL sebagai database serta PHP Storm sebagai *text editor* dan *browser* Google Chrome.

6.2 *Use Case Diagram*

Adapun *use case diagram* dari sistem yang diusulkan ini dapat dilihat pada gambar 3 berikut.



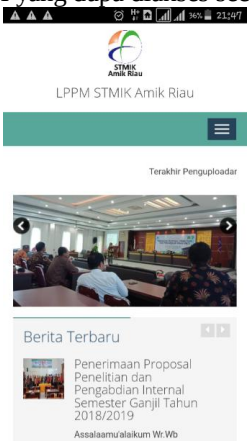
Gambar 3. *Use case diagram* sistem yang diusulkan

7. Implementasi

Adapun hasil dari implementasi perangkat lunak dapat ditampilkan berupa beberapa halaman pada sistem yang sudah dibangun.

1. Halaman Utama

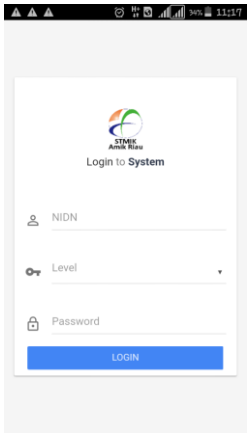
Pada halaman ini berisi informasi umum mengenai LPPM yang dapa diakses secara publik.



Gambar 4. Halaman Utama Sistem LPPM

2. Halaman Login

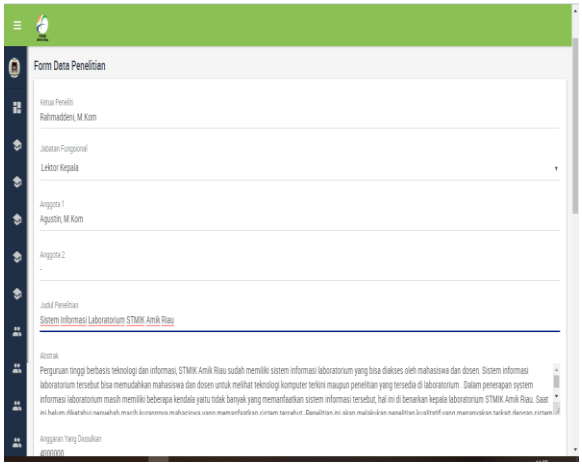
Halaman *login* digunakan oleh semua *user* untuk masuk ke sistem sesuai dengan levelnya masing-masing.



Gambar 5. Halaman Login Sistem LPPM

3. Halaman Input Proposal Penelitian

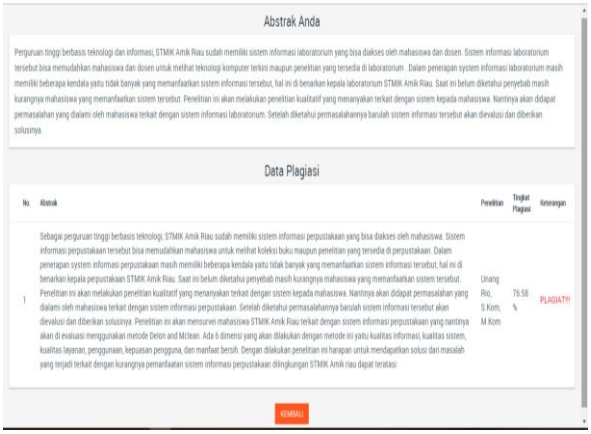
Halaman ini digunakan oleh dosen untuk memasukkan proposal penelitian.



Gambar 6. Halaman Input Proposal Penelitian

4. Tampilan Proposal Terdeteksi Plagiat

Adapun tampilan ketika sistem mendeteksi ada abstrak dari proposal yang sama dengan penelitian lain dapat dilihat seperti berikut.



Gambar 7. Tampilan Sistem Mendeteksi Plagiat

Teknik pengujian yang digunakan dalam sistem yang telah dibangun ini menggunakan metode Black Box, yaitu metode yang menguji kinerja dari fungsionalitas sistem apakah sudah berjalan dengan semestinya atau masih memiliki kesalahan.

Kesimpulan dari hasil pengujian sistem yang sudah dibangun ini adalah sistem dapat berjalan sebagaimana mestinya sesuai tahap perancangan sistem.

8. Simpulan

Berdasarkan penelitian yang telah dilakukan, maka dapat penulis ambil kesimpulan yaitu:

1. Tercapainya pengembangan sistem LPPM dapat diakses melalui web browser maupun menggunakan perangkat mobile android.
2. Aplikasi ini dapat mendeteksi persentase kesamaan antar dokumen menggunakan Algoritma Rabin-Karp.
3. Menghasilkan suatu sistem website LPPM yang lengkap sekaligus dapat memberikan rekomendasi bagi reviewer terhadap usulan dari sebuah penelitian internal dosen.

9. Referensi

Firdaus, Hari Bagus. (2008), Deteksi Plagiat Dokumen Menggunakan Algoritma Rabin-Karp, Institut Teknologi Bandung, Bandung.

Kiran Shivaji, Sonawane (2015) "Plagiarism Detection by using Karp-Rabin and String Matching Algorithm Together" International Journal of Computer Applications (0975 – 8887) Volume 116 – No. 23.

Kohli, Nupur. Rushikesh Joshi., (2014),

"Implementation of Rabin Karp String Matching Algorithm Using MPI ".

Noprison, Handrie. Susilo, Boko. Ernawati., (2013), Implementasi Algoritma Rabin-Karp untuk Menentukan Keterkaitan Antar Publikasi Penelitian Dosen Tahun 2013 (Studi Kasus : Website Lembaga Penelitian Universitas Bengkulu), Volume 9 Nomor 2.

Nugroho, Eko. Perancangan Sistem Deteksi Plagiarisme Dokumen Teks Dengan Menggunakan Algoritma Rabin-Karp. Malang : Program Studi Ilmu Komputer, Jurusan Matematika, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Brawijaya.

Putra, Doddi Aria. Sujaini, Herry. Pratiwi, Helen Sasty., (2015), Implementasi Algoritma Rabin-Karp untuk Membantu Pendeteksian Plagiat Pada Karya Ilmiah, Jurnal JUSTIN Vol.1 No.1.

Steven. Perancangan Program Aplikasi Pendeteksian Plagiarisme Dokumen Berbasis Teks Menggunakan Algoritma Rabin-Karp. Jakarta : Program Ganda Teknik Informatika dan - Matematika, Universitas Bina Nusantara. 2009.