4. IMPLEMENTASI SISTEM

Bab ini akan menjelaskan implementasi dari dasar teori dan desain sistem yang dijelaskan pada bab sebelumnya hingga menjadi suatu aplikasi yang siap digunakan berupa penggalan *source code* program dalam bahasa pemrograman PHP.

Penjelasan implementasi sistem akan dibagi menjadi 5 (lima) proses yang merupakan proses-proses utama dari sistem yaitu proses *crawling*, proses pencarian kata kunci dari halaman *web*, proses *porter stemmer*, proses perhitungan TF-IDF dan *similarity*, dan proses pencarian *similarity*.

4.1. Proses *Crawling*

Fungsi ini digunakan saat *administrator* menekan tombol *crawl* pada bagian *setting* di halaman admin. Fungsi ini akan mencari *link-link* yang ada pada sebuah halaman *web* dan menambahkannya pada *database*. *administrator* dapat melihat url hasil proses ini pada bagian *view url* yang terdapat pada halaman *administrator*. Berikut ini adalah *segmen* program (*flowchart* dari *segmen* program dapat lihat pada Gambar 3.2 dan Gambar 3.3) dari proses tersebut :

```
function crawling($link_scan)
{
    $query = "SELECT * FROM crawl_list WHERE next_crawl='N' ";
   $res_main1 = mysql_query($query) or die(mysql_error());
   $count=mysql_num_rows($res_main1);
   if ($count==0)
   {
        $query_domain = "SELECT * FROM domain_list WHERE next_crawl='N' limit 0,1";
        $res_domain = mysql_query($query_domain) or die(mysql_error());
        $count_domain=mysql_num_rows($res_domain);
        if ($count_domain==0)
        {
          $url = "http://en.wikipedia.org/wiki/Learned_Hand" ;
   $domain_crawl=get_domain($url);
          $query = "SELECT * FROM domain_list WHERE url='$domain_crawl'";
          $res = mysql_query($query) or die(mysql_error());
          $count=mysql_num_rows($res);
```

```
        if($count==0)
        {
            $query = "INSERT INTO domain_list VALUES('','$domain_crawl')";
            $res = mysql_query($query) or die(mysql_error());
            $query = "SELECT * FROM domain_list WHERE url='$domain_crawl'";
            $res = mysql_query($query) or die(mysql_error());
        }
        $row = mysql_fetch_row($res);
        $id_domain= $row[0];
        $query = "SELECT * FROM crawl_list WHERE url='$url'";
        $res_main = mysql_query($query) or die(mysql_error());
        $count=mysql_num_rows($res_main);
        if ($count==0)
        {
            $query = "INSERT INTO crawl_list VALUES ('','$url','$id_domain','N')";
            $res_main = mysql_query($query) or die(mysql_error());
        }
        $query = "SELECT * FROM crawl_list WHERE next_crawl='N'";
        $res_main1 = mysql_query($query) or die(mysql_error());
    }
    else
    {
        $res_main1=$res_domain;
    }
}

while($row1 = mysql_fetch_row($res_main1))
{
    $url=$row1[1];
    if($count_domain==0)
    {
        $domain_crawl=get_domain($url);
        $id=$row1[0];
        $query = "UPDATE crawl_list set next_crawl='Y' WHERE id='$id'";
        $res_main = mysql_query($query) or die(mysql_error());
        $query = "SELECT * FROM domain_list WHERE url='$domain_crawl'";
        $res = mysql_query($query) or die(mysql_error());
        $row = mysql_fetch_row($res);
        $id_domain= $row[0];
    }
    else
    {
        $id_domain=$row1[0];
        $query = "UPDATE domain_list set next_crawl='Y' WHERE id_domain='$id_domain'";
        $res_main = mysql_query($query) or die(mysql_error());
    }
```

```
        $page = scanPage($url);
    preg_match_all('/<a(?:[^>]*)href="(.*?)"(?:[^>]*)>(.*?)<\/a>/is',$page,$output,PREG_SE
T_ORDER);
        $q=0;
        foreach($output as $item)
        {
            $q++;
            if($q==$link_scan)
            break;
            $link=get_link_only($item[0]);
            $check_ext=check_extension($item[1]);
            if ((substr($item[1], $pos1, 4) <> "http")&&($check_ext==0))
            {
                $join_url=$domain_crawl.$item[1];
                $head = scanHead($join_url);
                $en=get_lang($head);
                $code=get_code($head);
                $join_url_page="";
                if($en!=0)
                {
                    $join_url_page=scanPage($join_url);
                    $encoding = mb_detect_encoding($join_url_page, "auto");
                    $text = iconv( $encoding, "utf-8", $join_url_page );
                    $eng_word=check_english($text);
                    $min_eng=get_setting("min_eng");
                    if($eng_word>$min_eng)
                    $en=2;
                }
    if(($en==2)&&(!ereg("#",$item[1]))&&($code==1))
                {
                    $query = "SELECT * FROM domain_list WHERE url='$domain_crawl'";
                    $res = mysql_query($query) or die(mysql_error());
                    while($row = mysql_fetch_row($res))
                    {
                        $id_domain= $row[0];
                    }
                    $query = "SELECT * FROM crawl_list WHERE url='$join_url'";
                    $res = mysql_query($query) or die(mysql_error());
                    $count=mysql_num_rows($res);
                    if ($count==0)
                    {
                        $query = "INSERT INTO crawl_list VALUES ('','$join_url','$id_domain','N')";
                        $res = mysql_query($query) or die(mysql_error());
                        if($join_url_page=="")
                        {
```

```
                    $join_url_page=scanPage($join_url);
        $encoding = mb_detect_encoding($join_url_page, "auto");
                    $text = iconv( $encoding, "utf-8", $join_url_page );
                }
                $query = "SELECT * FROM crawl_list WHERE url='$join_url'";
                $res = mysql_query($query) or die(mysql_error());
                $row = mysql_fetch_row($res);
                save_page($row[0],$join_url_page);
            }
        }
    }
    else if (($check_ext==0)||($check_ext==""))
    {
        $link=get_link_only($item[0]);
        $new_domain=get_domain($link);
        $link=str_replace('"',"",$link);
        $head = scanHead($link);
        $en=get_lang($head);
        $code=get_code($head);
        if($en!=2)
        {
            $link_page=scanPage($link);
            $encoding = mb_detect_encoding($link_page, "auto");
            $text = iconv( $encoding, "utf-8", $link_page );
            $eng_word=check_english($text);
            $min_eng=get_setting("min_eng");
            if($eng_word>$min_eng)
            $en=2;
        }
        if(($en==2)&&(!ereg("#",$item[1]))&&($code==1))
        {
            if ($count_domain==0)
            {
                $query = "SELECT * FROM domain_list WHERE url='$new_domain'";
                $res = mysql_query($query) or die(mysql_error());
                $count=mysql_num_rows($res);
                    if ($count==0)
                    {
                        $new_domain=str_replace('"','',$new_domain);
                        $query = "INSERT INTO domain_list VALUES('','$new_domain')";
                        $res = mysql_query($query) or die(mysql_error());
                    }
                $query = "SELECT * FROM domain_list WHERE url='$new_domain'";
                $res = mysql_query($query) or die(mysql_error());
                $row = mysql_fetch_row($res);
                $id_domain= $row[0];
```

```
                    }
                    $query = "SELECT * FROM crawl_list WHERE url='$link'";
                    $res = mysql_query($query) or die(mysql_error());
                    $count=mysql_num_rows($res);
                    if ($count==0)
                    {
                        $link=str_replace("'",'',$link);
                        $query = "INSERT INTO crawl_list VALUES ('','$link','$id_domain','N')";
                        $res = mysql_query($query) or die(mysql_error());
                    }
                }
            }
        }
    }
}
```

Segmen Program 4.1. Fungsi *Crawling*

Pada mulanya ketika fungsi ini dipanggil fungsi akan memeriksa dalam tabel *crawl_list* pada *database* ada tidaknya url yang memiliki status *crawl_next*=0. Apabila tidak ada fungsi akan memeriksa dengan cara yang sama adakah *domain* yang memiliki status *next_crawl*=0 pada tabel *domain_list* di *database*, apabila tidak ada maka pecarian akan dimulai dari http://en.wikipedia.org terlebih dahulu. Setelah mendapatkan url yang akan dibaca maka proses dilanjutkan ke pemeriksaan url-url baru yang didapatkan apakah dapat dimasukkan ke dalam *database*. Apabila didapatkan *link* baru yang memenuhi syarat untuk masuk ke *database*, maka *file* dari halaman tersebut akan disimpan secara otomatis. Data *file* yang didapat akan disimpan pada *folder* yang sama dengan sistem. Untuk melihat *link-link* yang disudah ada *admin* dapat melihat pada bagian *view_crawl* yang terdapat pada halaman *admin*.

Berikut ini adalah *segmen* program untuk menyimpan *file* :

```
function save_page($fname,$text)
{
    $fh = fopen($fname, 'w')or die("can't open file");
    fwrite($fh, $text);
    fclose($fh);
}
```

Segmen Program 4.2. Fungsi menyimpan halaman

Berikut ini adalah *segmen* program untuk memanggil fungsi *get_link_only*, dimana proses melakukan pemotongan dari data yang diperoleh dari fungsi *scan_page* sehingga yang didapatkan adalah url-nya saja.

```
function get_link_only($url)
{
    $pos=0;
    $pos = strpos($url,'"');
    $pos=$pos+1;
    $pos_end = strpos($url, '"',$pos+3);
    $result=substr($url,$pos,$pos_end-($pos-1));
    return $result;
}
```

Segmen Program 4.3. Fungsi pengambilan *link*

Berikut ini adalah fungsi *scan_page*, dimana proses ini dilakukan untuk mendapatkan sebuah halaman *web* secara utuh.

```
function scanPage($url) {
    //$proxy="proxy.petra.ac.id:8080";
    $ch = curl_init();
    curl_setopt($ch, CURLOPT_URL,$url);
    //curl_setopt($ch, CURLOPT_PROXY, $proxy);
    //curl_setopt($curl, CURLOPT_USERPWD, "m26405014:123456");
    curl_setopt($ch, CURLOPT_TIMEOUT, 120); //timeout after 120 seconds
    curl_setopt($ch, CURLOPT_RETURNTRANSFER,1);
    $result=curl_exec($ch);
    curl_close ($ch);
    return $result ;
}
```

Segmen Program 4.4. Fungsi membaca halaman *web*

Berikut ini adalah fungsi *scan_head*, dimana proses ini dilakukan untuk mendapatkan informasi *header* saja dari sebuah halaman *web*.

```
function scanHead($url)
{
    //$proxy="proxy.petra.ac.id:8080";
    $ch = curl_init();
    curl_setopt($ch, CURLOPT_URL,$url);
    //curl_setopt($ch, CURLOPT_PROXY, $proxy);
    //curl_setopt($curl, CURLOPT_USERPWD, "m26405014:123456");

    curl_setopt($ch, CURLOPT_TIMEOUT, 120);
    curl_setopt($ch, CURLOPT_RETURNTRANSFER,1);
```

```
    curl_setopt($ch, CURLOPT_HEADER, true);
    curl_setopt($ch, CURLOPT_NOBODY, true);
    $result=curl_exec($ch);
    curl_close ($ch);
    return $result ;
}
```

Segmen Program 4.5. Fungsi untuk membaca *Header* halaman *web*

Berikut ini adalah fungsi *get_domain*, dimana proses ini dilakukan untuk mendapatkan *domain* dari sebuah url.

```
function get_domain($url)
{
    $pos=0;
    $url.="/";
     $pos = strpos($url,'/', 7);
    if ($pos==0)
    {
        $result=$url;
    }else
    {
        $result=substr($url,0, $pos);
    }
    return $result;
}
```

Segmen Program 4.6. Fungsi untuk mendapatkan *domain*

Berikut ini adalah *segmen* program untuk memeriksa sebuah halaman *web* merupakan halaman yang berbahasa inggris atau tidak berdasarkan hasil data dari proses *scan_head*.

```
function get_lang($page)
{
    $pos=1;
    $pos = strpos($page,'Content-Language');
    if ($pos==0)
    {
        $en=1;
       return $en;
     }
    $pos1=strpos($page," ",$pos);
    $pos1=$pos1+1;
    $pos2=strpos($page," ",$pos1);
    $en=1;
```

```
    $lang=substr($page,$pos1,$pos2-$pos1);
    if(ereg("en",$lang))
        $en=2;
    return $en;
}
```

Segmen Program 4.7. Fungsi mencari *language* dari sebuah halaman *web*

Berikut ini adalah *segmen* program untuk memeriksa sebuah halaman *web* merupakan halaman yang berbahasa inggris atau tidak berdasarkan persentase *basic english word* yang ada pada kata tersebut. Daftar kata yang termasuk dalam *basic english word* terdapat dalam *file english word*.

```
function check_english($text)
 {
    $text = strip_html_tags( $text );
    $text = html_entity_decode( $text, ENT_QUOTES, "UTF-8" );
    $text = strip_symbols( $text );
    $text = strip_numbers( $text );
    $text = mb_strtolower( $text, "utf-8" );
    $punc=array('[',']','(',')','"','-');
    $text=str_replace($punc,' ',$text);
    $text = strip_punctuation( $text );
    mb_regex_encoding( "utf-8" );
    $text = preg_replace('/\s+/', ' ', $text);
    $words = mb_split( ' +', $text);
    $all_word_count=count($words);
    $data=open_file("../english_word");
    $engWords = mb_split( '\s+',$data );
    $result = array_intersect($words, $engWords);
    $result_count = count($result);
    $eng_word_percentage=$result_count/$all_word_count*100;
    return $eng_word_percentage;
 }
```

Segmen Program 4.8. Fungsi Pengecekan Bahasa Inggris

4.2. Proses pencarian kata kunci

Fungsi ini digunakan program untuk mencari kata yang yang dianggap penting untuk disimpan. Proses ini dilakukan ketika ingin mencari kemiripan dari sebuah halaman *web*. Berikut ini adalah *segmen* program (*flowchart* dari *segmen* program dapat dilihat pada Gambar 3.4 dan Gambar 3.15) dari proses tersebut :

```
function get_keyword($id_url,$utf8_text)
{
$encoding = mb_detect_encoding($utf8_text, "auto");
$utf8_text = iconv( $encoding, "utf-8", $utf8_text );
$utf8_text = strip_html_tags( $utf8_text );
/* Decode HTML entities */
$utf8_text = html_entity_decode( $utf8_text, ENT_QUOTES, "UTF-8" );
$punc=array('[',']','(',')','"',':',';');
$utf8_text=str_replace($punc,' ',$utf8_text);
$utf8_text=str_replace(' +',' ',$utf8_text);
$utf8_text = strip_symbols( $utf8_text );
$utf8_text = strip_numbers( $utf8_text );
$utf8_text = mb_strtolower( $utf8_text, "utf-8" );
$utf8_text_clear = trim(preg_replace('#[^\p{L}\p{N}]+#u', ' ', $utf8_text));
$utf8_text_clear = strip_punctuation( $utf8_text );
$punc=array('.',',',';','/','-','"');
$utf8_text_clear=str_replace($punc,' ',$utf8_text_clear);
mb_regex_encoding( "utf-8" );
$utf8_text_clear = preg_replace('/\s+/',' ',$utf8_text_clear);
$words = mb_split( ' +', $utf8_text_clear );
$query = "SELECT * FROM stopword_list";
$res = mysql_query($query) or die(mysql_error());
$i=0;
while($row = mysql_fetch_row($res))
{
    $stopW[$i]= $row[0];
    $i++;
}
$words=array_diff($words,$stopW);

foreach ( $words as $no => $val )
{
    $words[$no]=trim(stem(str_replace("'","",$val)));
    if(strlen(trim($val))==1)
        $words[$no]=$val.".";
}
$words=array_diff($words,$stopW);
$word_count=array_count_values($words);
$words = array_unique($words);
$i=0;
$list_2words = array();
$list_3words = array();
unset($words[0]);
unset ($word_count[' ']);
unset ($word_count['']);
$q=array();
```

```
$array_pos=array();
foreach ( $word_count as $word => $val )
{
    if(substr($word,-1)=="i")
        $word=substr($word, 0, -1);
    $nrMatches = substr_count($utf8_text, $word);
    $pos=0;
    unset ($word_pos);
    for($i=0;$i<$nrMatches;$i++)
    {
        $pos1=strpos($utf8_text,$word,$pos);
        $pos2=strpos($utf8_text," ",$pos1+1);
        $char_before=substr($utf8_text,$pos1-1,1);
        if (!ereg ("[A-Za-z0-9]",$char_before))
        {
            $pos2=$pos2+1;
            $word1=substr($utf8_text,$pos1,$pos2-$pos1);
            $word1_stem=stem(trim(str_replace($punc,"",$word1)));
            $pos3=strpos($utf8_text," ",$pos2);
            $check_punc1=substr($utf8_text,$pos2-2,1);
            $check_punc2=substr($utf8_text,$pos3-1,1);
            $word2=substr($utf8_text,$pos2,$pos3-$pos2);
            $len=strlen($word2);
            $poscheck=($pos3-$len)-$pos2;
            $word2_stem=stem(trim(str_replace($punc,"",$word2)));

    {
    if((array_key_exists($word2_stem,$word_count))&&(array_key_exists($word1_stem,$word_c
ount))&&(strcmp($punc,$check_punc1)!=1)&&($poscheck<3))
        {
            $pos3=$pos3+1;
            $pos4=strpos($utf8_text," ",$pos3);
            $word3=substr($utf8_text,$pos3,$pos4-$pos3);
            $len2=strlen($word2);
            $poscheck2=($pos4-$len2)-$pos3;
            $word3_stem=stem(trim(str_replace($punc,"",$word3)));
    if((in_array($word3_stem,$words))&&(strcmp($punc,$check_punc2)!=1)&&($poscheck2<3))
    {
    if((!(array_key_exists($pos3,$array_pos)))||(!(array_key_exists($pos2,$array_pos)))||(!(arra
y_key_exists($pos1,$array_pos))))
    {
            array_push($list_3words,trim($word1_stem)."                    ".trim($word2_stem)."
".trim($word3_stem));
            if(!(array_key_exists($pos3,$array_pos)))
            {
                $array_pos[$pos3]=1;
```

```
            $word_count[$word3_stem]=$word_count[$word3_stem]-1;
            }
        }
    }
    Else
if((in_array($word3_stem,$words))&&($check_punc2==".")&&(strlen($word2_stem)==2)&&($po
scheck2<3))
    {
    if((!(array_key_exists($pos3,$array_pos)))or(!(array_key_exists($pos2,$array_pos)))or(!(arr
ay_key_exists($pos1,$array_pos))))
        {
            array_push($list_3words,trim($word1_stem)."                    ".trim($word2_stem)."
".trim($word3_stem));
            if(!array_key_exists($pos3,$array_pos))
            {
                $array_pos[$pos3]=1;
                $word_count[$word3_stem]=$word_count[$word3_stem]-1;
            }
        }
    }
    Else
    {
    if((!(array_key_exists($pos1,$array_pos)))or(!(array_key_exists($pos2,$array_pos))))
        array_push($list_2words,trim($word1_stem)." ".trim($word2_stem));
    }
    if(!array_key_exists($pos1,$array_pos))
    {
        $word_count[$word1_stem]=$word_count[$word1_stem]-1;
        $array_pos[$pos1]=1;
    }
    if(!array_key_exists($pos2,$array_pos))
    {
        $word_count[$word2_stem]=$word_count[$word2_stem]-1;
        $array_pos[$pos2]=1;
    }
}
}
    $pos=$pos2;
}
}
$total=strlen($utf8_text);
$total=($total*30)/100;
arsort($word_count);
$i=0;
```

```
foreach ( $word_count as $word => $val )
{
    $temp=trim($word);
    if(substr($temp,-1)=="i")
        $temp=substr($temp, 0, -1);

    $temp=" ".$temp;
    $first_show=0;
    $pos=strpos($utf8_text,$temp);
    if (($pos<$total)&&($val>0)&&($pos!=""))
    {
        $first_show=1;
    }

    if ($val>0)
    {
        $query = "SELECT * FROM keyword_list WHERE keyword='$temp'";
        $res = mysql_query($query) or die(mysql_error());
        $count=mysql_num_rows($res);
        if($count==0)
        {
            $query = "INSERT INTO keyword_list VALUES('','$temp')";
            $res = mysql_query($query) or die(mysql_error());
            $query = "SELECT * FROM keyword_list WHERE keyword='$temp'";
            $res = mysql_query($query) or die(mysql_error());
        }
        $row = mysql_fetch_row($res);
        $id_key= $row[0];
        $query = "INSERT INTO TF_list VALUES('$id_url','$id_key','$val',0)";
        $res = mysql_query($query) or die(mysql_error());
        if($first_show==1)
        {
            $query = "INSERT INTO TF2_list VALUES('$id_url','$id_key')";
            $res = mysql_query($query) or die(mysql_error());
        }
    }
}
$word2_count=array_count_values($list_2words);
arsort($word2_count);

$list_2words = array_unique($list_2words);
foreach ( $word2_count as $word => $val )
{

    $temp=trim($word);
    if(substr($temp,-1)=="i")
```

```
        $temp=substr($temp, 0, -1);

    $temp=" ".$temp;
    $first_show=0;
    $pos=strpos($utf8_text,$temp);
    if (($pos<$total)&&($val>0)&&($pos!=""))
    {
        $first_show=1;
    }

    if ($val>0)
    {
        $query = "SELECT * FROM keyword_list WHERE keyword='$temp'";
        $res = mysql_query($query) or die(mysql_error());
        $count=mysql_num_rows($res);
        if($count==0)
        {
                $query = "INSERT INTO keyword_list VALUES('','$temp')";
                $res = mysql_query($query) or die(mysql_error());
                $query = "SELECT * FROM keyword_list WHERE keyword='$temp'";
                $res = mysql_query($query) or die(mysql_error());
        }
        $row = mysql_fetch_row($res);
        $id_key= $row[0];
        $query = "INSERT INTO TF_list VALUES('$id_url','$id_key','$val',0)";
        $res = mysql_query($query) or die(mysql_error());

        if($first_show==1)
        {
            $query = "INSERT INTO TF2_list VALUES('$id_url','$id_key')";
            $res = mysql_query($query) or die(mysql_error());
        }
    }
}
$word3_count=array_count_values($list_3words);
arsort($word3_count);
foreach ( $word3_count as $word => $val )
{
    $temp=trim($word);
    $first_show=0;
    $pos=strpos($utf8_text,$temp);
    if (($pos<$total)&&($val>0)&&($pos!=""))
    {
        $first_show=1;
    }
    if ($val>0)
```

```
    {
        $temp=trim($word);
        $query = "SELECT * FROM keyword_list WHERE keyword='$temp'";
        $res = mysql_query($query) or die(mysql_error());
        $count=mysql_num_rows($res);
        if($count==0)
        {
                $query = "INSERT INTO keyword_list VALUES('','$temp')";
                $res = mysql_query($query) or die(mysql_error());
                $query = "SELECT * FROM keyword_list WHERE keyword='$temp'";
                $res = mysql_query($query) or die(mysql_error());
        }
        $row = mysql_fetch_row($res);
        $id_key= $row[0];
        $query = "INSERT INTO TF_list VALUES('$id_url','$id_key','$val',0)";
        $res = mysql_query($query) or die(mysql_error());
        if($first_show==1)
        {
            $query = "INSERT INTO TF2_list VALUES('$id_url','$id_key')";
            $res = mysql_query($query) or die(mysql_error());
        }
    }
}
```

Segmen Program 4.9. Fungsi Pencarian Kata Kunci

Berikut ini adalah *segmen* dari program untuk menghilangkan tag-tag HTML.

```
function strip_html_tags( $text )
{
    // PHP's strip_tags() function will remove tags, but it
    // doesn't remove scripts, styles, and other unwanted
    // invisible text between tags.  Also, as a prelude to
    // tokenizing the text, we need to insure that when
    // block-level tags (such as <p> or <div>) are removed,
    // neighboring words aren't joined.

    $text = preg_replace(
        array(
            // Remove invisible content
            '@<head[^>]*?>.*?</head>@siu',
            '@<style[^>]*?>.*?</style>@siu',
            '@<script[^>]*?.*?</script>@siu',
            '@<object[^>]*?.*?</object>@siu',
```

```
            '@<embed[^>]*?.*?</embed>@siu',
            '@<applet[^>]*?.*?</applet>@siu',
            '@<noframes[^>]*?.*?</noframes>@siu',
            '@<noscript[^>]*?.*?</noscript>@siu',
            '@<noembed[^>]*?.*?</noembed>@siu',

            // Add line breaks before & after blocks
            '@<((br)|(hr))@iu',
            '@</?((address)|(blockquote)|(center)|(del))@iu',
            '@</?((div)|(h[1-9])|(ins)|(isindex)|(p)|(pre))@iu',
            '@</?((dir)|(dl)|(dt)|(dd)|(li)|(menu)|(ol)|(ul))@iu',
            '@</?((table)|(th)|(td)|(caption))@iu',
            '@</?((form)|(button)|(fieldset)|(legend)|(input))@iu',
            '@</?((label)|(select)|(optgroup)|(option)|(textarea))@iu',
            '@</?((frameset)|(frame)|(iframe))@iu',
        ),
        array(
            ' ', ' ', ' ', ' ', ' ', ' ', ' ', ' ', ' ', ' ',
            "\n\$0", "\n\$0", "\n\$0", "\n\$0", "\n\$0", "\n\$0",
            "\n\$0", "\n\$0",
        ),
        $text );

    // Remove all remaining tags and comments and return.
    return strip_tags( $text );
}
```

Segmen Program 4.10. Fungsi Penghilangan *tag-tag* HTML

Fungsi *strip_html_tags* merupakan tambahan dari fungsi *strip_tags* yang merupakan bawaan dari PHP. Fungsi *strip_tags* hanya menghilangkan tag-tag standar dari HTML. Apabila terdapat *script*, *style*, dan kata-kata yang tidak diinginkan diantara tag tidak dapat terdeteksi. Berikut ini adalah *segmen* dari program untuk menghilangkan tanda baca.

```
function strip_punctuation( $text )
{
   $urlbrackets    = '\[\]\(\)';
   $urlspacebefore = ':;\'_\*%@&?!' . $urlbrackets;
   $urlspaceafter  = '\.,:;\'\-_\*@&\/\\\\\?!#' . $urlbrackets;
   $urlall         = '\.,:;\'\-_\*%@&\/\\\\\?!#' . $urlbrackets;
   $specialquotes  = '\'"\*<>';

   $fullstop      = '\x{002E}\x{FE52}\x{FF0E}';
   $comma         = '\x{002C}\x{FE50}\x{FF0C}';
```

```
    $arabsep       = '\x{066B}\x{066C}';
    $numseparators  = $fullstop . $comma . $arabsep;
 // echo $numseparators."<br>";
    $numbersign     = '\x{0023}\x{FE5F}\x{FF03}';
    $percent       = '\x{066A}\x{0025}\x{066A}\x{FE6A}\x{FF05}\x{2030}\x{2031}';
    $prime         = '\x{2032}\x{2033}\x{2034}\x{2057}';
    $nummodifiers   = $numbersign . $percent . $prime;
  //echo $nummodifiers."<br>";
    return preg_replace(
        array(
        // Remove separator, control, formatting, surrogate,
        // open/close quotes.
            '/[\p{Z}\p{Cc}\p{Cf}\p{Cs}\p{Pi}\p{Pf}]/u',
        // Remove other punctuation except special cases
            '/\p{Po}(?<![' . $specialquotes .
                $numseparators . $urlall . $nummodifiers . '])/u',
        // Remove non-URL open/close brackets, except URL brackets.
            '/[\p{Ps}\p{Pe}](?<![' . $urlbrackets . '])/u',
        // Remove special quotes, dashes, connectors, number
        // separators, and URL characters followed by a space
            '/[' . $specialquotes . $numseparators . $urlspaceafter .
                '\p{Pd}\p{Pc}]+((?= )|$)/u',
        // Remove special quotes, connectors, and URL characters
        // preceded by a space
            '/((?<= )|^)[' . $specialquotes . $urlspacebefore . '\p{Pc}]+/u',
        // Remove dashes preceded by a space, but not followed by a number
            '/((?<= )|^)\p{Pd}+(?![\p{N}\p{Sc}])/u'//,
        // Remove consecutive spaces
          // '/ +/',
        ),
        ' ',
        $text );
}
```

Segmen Program 4.11. Fungsi Penghilangan Tanda Baca

Berikut ini adalah segmen program untuk menghilangkan angka-angka yang tidak diperlukan.

```
function strip_numbers( $text )
{
    $urlchars      = '\.,:;\'=+\-_\*%@&\/\\\\?!#~\[\]\(\)';
    $notdelim      = '\p{L}\p{M}\p{N}\p{Pc}\p{Pd}' . $urlchars;
    $predelim      = '((?<=[^' . $notdelim . '])|^)';
    $postdelim     = '((?=[^'  . $notdelim . '])|$)';
```

```
    $fullstop     = '\x{002E}\x{FE52}\x{FF0E}';
    $comma        = '\x{002C}\x{FE50}\x{FF0C}';
    $arabsep      = '\x{066B}\x{066C}';
    $numseparators = $fullstop . $comma . $arabsep;
    $plus         = '\+\x{FE62}\x{FF0B}\x{208A}\x{207A}';
    $minus        = '\x{2212}\x{208B}\x{207B}\p{Pd}';
    $slash        = '[\/\x{2044}]';
    $colon        = ':\x{FE55}\x{FF1A}\x{2236}';
    $units        = '%\x{FF05}\x{FE64}\x{2030}\x{2031}';
    $units       .= '\x{00B0}\x{2103}\x{2109}\x{23CD}';
    $units       .= '\x{32CC}-\x{32CE}';
    $units       .= '\x{3300}-\x{3357}';
    $units       .= '\x{3371}-\x{33DF}';
    $units       .= '\x{33FF}';
    $percents     = '%\x{FE64}\x{FF05}\x{2030}\x{2031}';
    $ampm         = '([aApP][mM])';

    $digits       = '[\p{N}' . $numseparators . ']+';
    $sign         = '[' . $plus . $minus . ']?';
    $exponent     = '([eE]' . $sign . $digits . ')?';
    $prenum       = $sign . '[\p{Sc}#]?' . $sign;
    $postnum      = '([\p{Sc}' . $units . $percents . ']|' . $ampm . ')?';
    $number       = $prenum . $digits . $exponent . $postnum;
    $fraction     = $number . '(' . $slash . $number . ')?';
    $numpair      = $fraction . '([' . $minus . $colon . $fullstop . ']' .
       $fraction . ')*';

    return preg_replace(
       array(
       // Match delimited numbers
          '/' . $predelim . $numpair . $postdelim . '/u',
       // Match consecutive white space
          '/ +/u',
       ),
       ' ',
       $text );
}
```

Segmen Program 4.12. Fungsi Penghilangan Angka

Berikut ini adalah segmen program untuk menghilangkan simbol-simbol yang tidak diperlukan.

```
function strip_symbols( $text )
{
  // $brackets    = '\[\]\(\)';
```

```
    $plus   = '\+\x{FE62}\x{FF0B}\x{208A}\x{207A}';
   $minus  = '\x{2012}\x{208B}\x{207B}';

   $units  = '\\x{00B0}\x{2103}\x{2109}\\x{23CD}';
   $units .= '\\x{32CC}-\\x{32CE}';
   $units .= '\\x{3300}-\\x{3357}';
   $units .= '\\x{3371}-\\x{33DF}';
   $units .= '\\x{33FF}';

   $ideo   = '\\x{2E80}-\\x{2EF3}';
   $ideo  .= '\\x{2F00}-\\x{2FD5}';
   $ideo  .= '\\x{2FF0}-\\x{2FFB}';
   $ideo  .= '\\x{3037}-\\x{303F}';
   $ideo  .= '\\x{3190}-\\x{319F}';
   $ideo  .= '\\x{31C0}-\\x{31CF}';
   $ideo  .= '\\x{32C0}-\\x{32CB}';
   $ideo  .= '\\x{3358}-\\x{3370}';
   $ideo  .= '\\x{33E0}-\\x{33FE}';
   $ideo  .= '\\x{A490}-\\x{A4C6}';

   return preg_replace(
      array(
      // Remove modifier and private use symbols.
         '/[\p{Sk}\p{Co}]/u',
      // Remove mathematics symbols except + - = ~ and fraction slash
         '/\p{Sm}(?<![' . $plus . $minus . '=~\x{2044}])/u',
      // Remove + - if space before, no number or currency after
         '/((?<= )|^)[' . $plus . $minus . ']+((?![\p{N}\p{Sc}])|$)/u',
      // Remove = if space before
         '/((?<= )|^)=+/u',
      // Remove + - = ~ if space after
         '/[' . $plus . $minus . '=~]+((?= )|$)/u',
      // Remove other symbols except units and ideograph parts
         '/\p{So}(?<![' . $units . $ideo . '])/u',
      // Remove consecutive white space
         '/ +/'
       //   '/'.$brackets.'([1-9A-Za-z]*)'.$brackets.'/'
      ),
      ' ',
      $text );
}
```

Segmen Program 4.13. Fungsi Penghilangan Simbol

4.3. Proses *Porter Stemmer*

Proses ini diperlukan untuk melakukan perubahan bentuk suatu kata menjadi bentuk kata dasarnya dengan menghilangkan akhiran yang terkandung didalamnya. Berikut ini adalah *segmen* program (*flowchart* dari *segmen* program dapat lihat pada Gambar 3.5-Gambar 3.14) dari proses tersebut :

```
function stem($word)
{
$step2list=array( 'ational'=>'ate', 'tional'=>'tion', 'enci'=>'ence', 'anci'=>'ance', 'izer'=>'ize',
'iser'=>'ise', 'bli'=>'ble', 'alli'=>'al', 'entli'=>'ent', 'eli'=>'e', 'ousli'=>'ous', 'ization'=>'ize',
'isation'=>'ise', 'ation'=>'ate', 'ator'=>'ate', 'alism'=>'al', 'iveness'=>'ive', 'fulness'=>'ful',
'ousness'=>'ous', 'aliti'=>'al', 'iviti'=>'ive', 'biliti'=>'ble', 'logi'=>'log' );

$step3list=array('icate'=>'ic', 'ative'=>'', 'alize'=>'al', 'alise'=>'al', 'iciti'=>'ic', 'ical'=>'ic',
'ful'=>'', 'ness'=>'' );

$c = "[^aeiou]"; # consonant
$v = "[aeiouy]"; # vowel
$C = "${c}[^aeiouy]*"; # consonant sequence
$V = "${v}[aeiou]*"; # vowel sequence

$mgr0 = "^(${C})?${V}${C}"; # [C]VC... is m>0
$meq1 = "^(${C})?${V}${C}(${V})?" . '$'; # [C]VC[V] is m=1
$mgr1 = "^(${C})?${V}${C}${V}${C}"; # [C]VCVC... is m>1
$_v = "^(${C})?${v}"; # vowel in stem

if (strlen($word)<3) return $word;

      $word=preg_replace("/^y/", "Y", $word);

#Step 1a
        if (preg_match("/(ss|i)es$/", $word)) {
           $word=substr($word, 0, -2);
        }
         if (preg_match("/[^s]s$/", $word)) {
           $word=substr($word, 0, -1);
        }

     // $word=preg_replace("/(ss|i)es$/", "\\1", $word);       # sses-> ss, ies->es
      //$word=preg_replace("/([^s])s$/", "\\1", $word);       #       ss->ss but s->null

#Step 1b
      if (preg_match("/eed$/", $word)) {
            $stem=preg_replace("/eed$/", "", $word);
```

```
            if (ereg("$mgr0", $stem)) {
                $word=preg_replace("/.$/", "", $word);
            }
        }
        elseif (preg_match("/(ed|ing)$/", $word)) {
            $stem=preg_replace("/(ed|ing)$/", "", $word);
            if (preg_match("/$_v/", $stem)) {
                $word=$stem;

                if (preg_match("/(at|bl|iz|is)$/", $word)) {
                    $word=preg_replace("/(at|bl|iz|is)$/", "\\1e", $word);
                }

                elseif (preg_match("/([^aeiouylsz])\\1$/", $word)) {
                    $word=preg_replace("/.$/", "", $word);
                }

                elseif (preg_match("/^${C}${v}[^aeiouwxy]$/", $word)) {
                    $word.="e";
                }
            }
        }

#Step 1c (weird rule)
        if (preg_match("/y$/", $word)) {
            $stem=preg_replace("/y$/", "", $word);
            if (preg_match("/$_v/", $stem))
                $word=$stem."i";
        }

#Step 2
        If
(preg_match("/(ational|tional|enci|anci|izer|iser|bli|alli|entli|eli|ousli|ization|isation|ation|ator|ali
sm|iveness|fulness|ousness|aliti|iviti|biliti|logi)$/",
$word, $matches)) {

$stem=preg_replace("/(ational|tional|enci|anci|izer|iser|bli|alli|entli|eli|ousli|ization|isation|ation
|ator|alism|iveness|fulness|ousness|aliti|iviti|biliti|logi)$/",
"", $word);
            $suffix=$matches[1];
            if (preg_match("/$mgr0/", $stem)) {
                $word=$stem.$step2list[$suffix];
            }
        }

#Step 3
```

```
      if (preg_match("/(icate|ative|alize|alise|iciti|ical|ful|ness)$/", $word, $matches)) {
            $stem=preg_replace("/(icate|ative|alize|alise|iciti|ical|ful|ness)$/", "", $word);
            $suffix=$matches[1];
            if (preg_match("/$mgr0/", $stem)) {
                  $word=$stem.$step3list[$suffix];
            }
      }

#Step 4
      if
(preg_match("/(al|ance|ence|er|ic|able|ible|ant|ement|ment|ent|ou|ism|ate|iti|ous|ive|ize|ise)$
/",
$word, $matches)) {

   $stem=preg_replace("/(al|ance|ence|er|ic|able|ible|ant|ement|ment|ent|ou|ism|ate|iti|ous|i
ve|ize|ise)$/", "", $word);
            $suffix=$matches[1];
            if (preg_match("/$mgr1/", $stem)) {
                  $word=$stem;
            }
      }
      elseif (preg_match("/(s|t)ion$/", $word)) {
            $stem=preg_replace("/(s|t)ion$/", "\\1", $word);
            if (preg_match("/$mgr1/", $stem)) $word=$stem;
      }

#Step 5
      if (preg_match("/e$/", $word, $matches)) {
            $stem=preg_replace("/e$/", "", $word);
            if      (preg_match("/$mgr1/",$stem)      |      (preg_match("/$meq1/",      $stem)&
~preg_match("/^${C}${v}[^aeiouwxy]$/", $stem)))
{
                  $word=$stem;
            }
      }
      if(preg_match("/ll$/",$word) & preg_match("/$mgr1/", $word))
        $word=preg_replace("/.$/", "", $word);

# and turn initial Y back to y
      preg_replace("/^Y/", "y", $word);
      return $word;
}
```

Segmen Program 4.14. Fungsi Porter Stemmer

4.4. Proses Perhitungan TF-IDF dan *similarity*

Proses ini menggabungkan proses untuk menghitung IDF (*Inverse Document Frequency*) dan juga proses untuk menghitung *similarity* antar dokumen. Proses ini merupakan kelanjutan dari proses pencarian kata kunci. Pada proses pencarian kata kunci sudah sekaligus didapatkan TF (*Term Frequency*) dari setiap *keyword*. Berikut ini adalah *segmen* program (*flowchart* dari *segmen* program dapat lihat pada Gambar 3.16-Gambar 3.18) dari proses tersebut :

```
function calc_similarity()
{
    $query = "TRUNCATE TABLE jaccard_list";
    $res = mysql_query($query) or die(mysql_error());

    $query = "SELECT * FROM crawl_list";
    $res = mysql_query($query) or die(mysql_error());
    $all_url=mysql_num_rows($res);

    $query = "SELECT id_keyword,count(id_crawl) as idf FROM tf_list GROUP BY id_keyword";
    $res = mysql_query($query) or die(mysql_error());
    $all_key=mysql_num_rows($res);
    while($row = mysql_fetch_row($res))
    {
        $ins="UPDATE tf_list SET idf='$row[1]' WHERE id_keyword='$row[0]'";
        $res1 = mysql_query($ins) or die(mysql_error());
    }

    $query = "SELECT * FROM tf_list";
    $count_tf=array();
    $res = mysql_query($query) or die(mysql_error());
    while($row = mysql_fetch_row($res))
    {
        $temp=$row[1];
        $temp_idf=$row[3];
        $count_tf["$temp"]=$temp_idf;
        $temp_idcrawl=$row[0];
        $temp_tf=$row[2];
        $list_tf[$temp_idcrawl][$temp]=$temp_tf;
    }
    $i=0;
    $j=0;
    $num=1;
    foreach($list_tf as $i => $q)
    {
        $id_crawl[$num]=$i;
```

```
        foreach($q as $j =>$w)
        {
            if(($list_tf[$i][$j]==""))
            {
                $tf_idf[$i][$j]=0;
            }
            else
            {
                If((isset($front_show[$i][$j]))&&($front==1))
                {
                    $list_tf[$i][$j]=$list_tf[$i][$j]*2;
                }
                $tf_idf[$i][$j]=$list_tf[$i][$j]*(((log($all_url/$count_tf[$j]))/log(10))+1);
             }

        }
        $num++;
    }
    $query = "TRUNCATE TABLE jaccard_list";
    $res = mysql_query($query) or die(mysql_error());
    unset($count_tf);
    unset($list_tf);
    for($i=1;$i<$all_url;$i++)
    {
        for($j=$i+1;$j<=$all_url;$j++)
        {
            $sum1=0;
            $sum2=0;
            $sum3=0;
            for($k=1;$k<=$all_key;$k++)
            {
                $temp1=$id_crawl[$i];
                $temp2=$id_crawl[$j];
                $sum1+=$tf_idf[$temp1][$k]*$tf_idf[$temp2][$k];
                $sum2+=pow($tf_idf[$temp1][$k],2);
                $sum3+=pow($tf_idf[$temp2][$k],2);
            }
            $jacaard[$temp1][$temp2]=$sum1/($sum2+$sum3-$sum1);
            $temp=$jacaard[$temp1][$temp2]*100;
                        $query = "INSERT INTO jaccard_list VALUES('$temp1','$temp2','$temp')";
                        $res = mysql_query($query) or die(mysql_error());
        }
    }
}
```

Segmen Program 4.15. Fungsi Perhitungan *Similarity*

Proses pada *Segmen* Program 4.9 dilakukan setelah sistem selesai melakukan proses pencarian kata kunci, karena proses ini memerlukan informasi setiap kata dalam artikel yang telah disimpan pada tabel *TF_list*.

4.5. Proses Pencarian *Similarity*

Proses ini digunakan untuk mendapatkan daftar url yang memiliki kemiripan kata kunci(*keyword)* dari *table jaccard_list* dimana perhitungan sudah dilakukan pada proses perhitungan TF-IDF dan *similarity*. Berikut ini adalah *segmen* program (*flowchart* dari *segmen* program dapat lihat pada Gambar 3.16 – 3.18) dari proses pencarian *similarity* :

```
$conn = mysql_connect($dbserver,$dbuser,$dbpass)or die(mysql_error());
mysql_select_db($dbname,$conn);
$query_checkid = "SELECT id FROM crawl_list WHERE url='$url_find'";
$res_checkid = mysql_query($query_checkid) or die(mysql_error());
$count_checkid=mysql_num_rows($res_checkid);
if ($count_checkid==0)
{
    $new_domain=get_domain($url_find);
    $link=str_replace("'","",$url_find);
    $head = scanHead($url_find);
    $en=get_lang($head);
    $code=get_code($head);
    if($en!=2)
    {
        $link_page=scanPage($url_find);
        $encoding = mb_detect_encoding($link_page, "auto");
        $text = iconv( $encoding, "utf-8", $link_page );
        $eng_word=check_english($text);
        if($eng_word>30)
        $en=2;
    }
    if(($en==2)&&(!ereg("#",$item[1]))&&($code==1))
    {
        $query = "SELECT * FROM domain_list WHERE url='$new_domain'";
        $res = mysql_query($query) or die(mysql_error());
        $count=mysql_num_rows($res);
        if ($count==0)
        {
            $new_domain=str_replace("'",'',$new_domain);
```

```
            $query = "INSERT INTO domain_list VALUES('','$new_domain','N')";
            $res = mysql_query($query) or die(mysql_error());
        }
        $query = "SELECT * FROM domain_list WHERE url='$new_domain'";
        $res = mysql_query($query) or die(mysql_error());
        $row = mysql_fetch_row($res);
        $id_domain= $row[0];
        $query = "INSERT INTO crawl_list VALUES ('','$url_find','$id_domain','N')";
        $res = mysql_query($query) or die(mysql_error());
        $query = "SELECT id FROM crawl_list WHERE url='$url_find'";
        $res = mysql_query($query) or die(mysql_error());
        $row_checkid = mysql_fetch_row($res);
        $temp_id=$row_checkid[0];
        save_page($temp_id,$text);
        get_keyword($row_checkid[0],$text);
        calc_similarity();
        }else if ($en!=2)
        {
            echo "url not an English site";
        }else if ($code!=1)
        {
            echo "url not giving an ok response";
        }else
        {
            echo "url contain # a dynamic page, remove it";
        }
    }
    else
    {
        $row_checkid = mysql_fetch_row($res_checkid);
    }
        $checkid=$row_checkid[0];
        $min_sim=get_setting("min_sim");
        $query = "SELECT id_url1,id_url2,similarity FROM jaccard_list WHERE(id_url1='$checkid'
or id_url2='$checkid') and (similarity)>$min_sim order by similarity desc limit 0,5";
        $res = mysql_query($query) or die(mysql_error());
        $count_sim=mysql_num_rows($res);
        if($count_sim!=0)
        {
            while($row = mysql_fetch_row($res))
            {
                if($checkid==$row[0])
                    $url_find=$row[1];
                else
                    $url_find=$row[0];
```

```
            $query_geturl = "SELECT url FROM crawl_list WHERE id='$url_find'";
            $res_geturl = mysql_query($query_geturl) or die(mysql_error());
            $row_geturl = mysql_fetch_row($res_geturl);
            $geturl=$row_geturl[0];
            $similarity=$row[2];
            $text=open_file("../$url_find");
            $title=get_title(text);
            echo "<tr><td><a href='$geturl'>$title</a> - $similarity</td></tr>
                  <tr> <td> $geturl</td> </tr>
                  <tr><td> </td></tr>";
        }
    }
    else {echo "No result Found";}
```

Segmen Program 4.16. Proses Pencarian *Similarity*

Proses ini dilakukan ketika *user* menekan tombol *search* pada halaman utama setelah terlebih dahulu memasukkan url yang ingin dicari kesamaannya pada *text field* yang sudah disediakan. Pada proses ini sistem akan memeriksa apakah url yang ingin dicari kesamaannya sudah pernah disimpan atau tidak. Apabila belum pernah disimpan di *database,* maka sistem akan melakukan proses *scan_page,* pencarian kata kunci dan proses perhitungan TF-IDF dan *similarity*. Setelah perhitungan ulang, maka hasil pencarian ditampilkan pada *user*.