BAB V
IMPLEMENTASI

Pada bab ini akan dibahas implementasi berdasarkan metodologi dan perancangan yang telah dibahas pada bab sebelumnya.

5.1 Lingkungan Implementasi

Lingkungan implementasi yang akan dijelaskan pada sub bab ini adalah lingkungan perangkat keras dan lingkungan perangkat lunak yang digunakan untuk pengembangan perangkat lunak pengkategorian pesan singkat pada jejaring sosial Twitter Berbahasa Indonesia.

5.1.1 Lingkungan Perangkat Keras

Perangkat keras yang digunakan dalam pembuatan sistem aplikasi pengkategorian pesan singkat pada jejaring sosial Twitter berbahasa Indonesia adalah sebagai berikut :

- Prosesor Intel Core 2 Duo 2.26 GHz
- Memori RAM 2 GB 1067 MHz DDR3
- Hardisk 160.4 GB
- Monitor 13"
- Keyboard
- Mouse

5.1.2 Lingkungan Perangkat Lunak

Perangkat lunak yang digunakan dalam pembuatan aplikasi adalah sebagai berikut :

- Apache Web Server
- PHP
- MySQL

- FileZilla
- Aptana Studio
- Squel Pro
- Sistem Operasi Mac OSX Snow Leopard

5.2 Implementasi Perangkat Lunak

Pada implementasi perangkat lunak ini akan dijelaskan bagaimana tahapan-tahapan aplikasi pengkategorian pesan singkat pada jejaring sosial Twitter Berbahasa Indonesia, dimana tahapan-tahapan aplikasi pengkategorian akan dijelaskan per tahap sesuai dengan alur kerja sistem yang diterangkan pada bab perancangan. Penjelasan tahapan akan melibatkan beberapa potongan fungsi yang dipanggil oleh program untuk memproses mulai dari proses mendapatkan data latih, pengolahan data latih, melakukan hubungan dengan API Twitter, dan proses perhitungan pengkategorian untuk mendapatkan kategori.

5.2.1 Pemrosesan data latih

Pada bagian ini akan dilakukan proses pembentukan tabel frekuensi yang digunakan dalam perhitungan algoritma yang digunakan sebagai pada data latih. Data yang diolah berasal dari dokumen RSS yang diperloleh dari beberapa situs berita yang sudah memiliki kategori pada tabel dibawah ini :

Tabel 5.1 Daftar Alamat URL Sumber RSS

No	URL
1	http://rss.detik.com/index.php/finance
2	http://rss.detik.com/index.php/detiknews
3	http://rss.detik.com/index.php/hot
4	http://detikinet.com/index.php/detik.feed
5	http://rss.detik.com/index.php/otomotif
6	http://rss.detik.com/index.php/sport
7	http://rss.vivanews.com/get/teknologi
8	http://rss.vivanews.com/get/bola

9	http://rss.vivanews.com/get/sport
10	http://rss.vivanews.com/get/politik
11	http://rss.vivanews.com/get/nasional
12	http://rss.vivanews.com/get/dunia
13	http://rss.vivanews.com/get/metro
14	http://rss.vivanews.com/get/bisnis
15	http://rss.vivanews.com/get/showbiz
16	http://rss.vivanews.com/get/otomotif
17	http://rss.vivanews.com/get/korupsi

Proses untuk memperoleh data latih diawali dengan proses pengambilan konten RSS secara online dengan menggunakan fungsi pada bahasa PHP yang selanjutnya juga dilakukan proses penyimpanan. Fungsi untuk melakukan pengambilan konten membutuhkan bantuan sebuah *class library* RSS PHP.

Pada saat kita akan melakukan pengambilan konten RSS yang kita inginkan, maka kita harus mengetahui alamat URL website yang akan kita ambil konten RSS nya. Hal yang harus dipastikan dalam pengambilan konten RSS adalah apakah website yang ada tersebut merupakan website memiliki konten RSS atau tidak. Implementasi dari penjelasan pada paragraf ini adalah pada potongan *source code* fungsi yang dipanggil untuk mengambil konten RSS :

```
1  public function load($url=false, $unblock=true)
2  {
3          if($url)
4          {
5                  if($unblock)
6                  {
7                          $this->loadParser(file_get_contents($url, false, $this-
   >randomContext()));
8                  }
9                  else
10                 {
11                         $this->loadParser(file_get_contents($url));
12                 }
13         }
14 }
```

Gambar 5.1 *Source Code* Pengambilan RSS

Proses selanjutnya adalah melakukan pengambilan konten yang diperlukan pada hasil kembalian RSS. Tidak semua yang ada dikembalikan pada dokumen RSS

digunakan, yang digunakan adalah *title* dan *description* dari berita RSS. Komponen *title* merupakan judul dari sebuah elemen dari data latih, dan *description* adalah isi dari RSS yang nantinya akan diolah menjadi tabel data latih. Potongan fungsi untuk melakukan pembersihan elemen yang tidak digunakan adalah sebagai berikut :

```
unset($hasil['rss']['channel']['title']);
unset($hasil['rss']['channel']['link']);
unset($hasil['rss']['channel']['description']);
unset($hasil['rss']['channel']['language']);
unset($hasil['rss']['channel']['pubDate']);
unset($hasil['rss']['channel']['lastBuildDate']);
unset($hasil['rss']['channel']['docs']);
unset($hasil['rss']['channel']['managingEditor']);
unset($hasil['rss']['channel']['webMaster']);
unset($hasil['rss']['channel']['copyright']);
unset($hasil['rss']['channel']['category']);
unset($hasil['rss']['channel']['ttl']);
unset($hasil['rss']['channel']['image']);
$jumlah        =       count($hasil['rss']['channel']);
for($i=0;$i<$jumlah;$i++)
{
$hasil['rss']['channel']['item:'.$i]['description']=strip_tags($hasil['rss']['chann
el']['item:'.$i]['description']);
        $title          = ($hasil['rss']['channel']['item:'.$i]['title']);
        $description    = ($hasil['rss']['channel']['item:'.$i]['description']);
        $title          =  preg_replace('/[^a-z0-9]+/i', ' ', $title);
        $description    =  preg_replace('/[^a-z0-9]+/i', ' ', $description);
        $this->simpan_db($title, $description,$this->kategori, $this->sumber);
}
```

Gambar 5.2 *Source Code* Pembersihan RSS

Seteleh konten RSS sudah didapatkan langkah selanjutnya adalah penyimpanan data-data latih tersebut pada database. Saat disimpan sebuah data latih sudah memiliki label kategori tertentu berdasarkan sumber RSS. Pada saat melakukan proses penyimpanan pertama kali dilakukan pengecekan apakah data RSS tersebut sudah ada atau belum, apabila data sudah ada maka tidak akan dilakukan proses penyimpanan. Berikut adalah implementasi penyimpanan teks RSS yang digunakan sebagai data latih :

```
if($title=="")
{
 exit;
}
$link=mysql_connect("127.0.0.1", "root", "") or die(mysql_error());
mysql_select_db("tweecat_devel") or die(mysql_error());
$result=mysql_query("SELECT * FROM konten_klasifikasi where judul='".$title."'");
$num_rows = mysql_num_rows($result);
if($num_rows==0)
{
mysql_query("INSERT INTO konten_klasifikasi(kategori,judul,konten,waktu,sumber)
VALUES
('".$kategori."','".$title."','".$description."',CURRENT_TIMESTAMP(),'".$sumber."'
```

```
)") or die(mysql_error());
}
mysql_close($link);
```

Gambar 5.3 *Source Code* Penyimpanan RSS pada Database

Proses terakhir untuk membentuk data latih adalah dengan membentuk tabel frekuensi yang nantinya digunakan olah sistem yang melakukan perhitungan algoritma untuk menentukan kategori apa yang dimiliki oleh dokumen Tweet. Proses yang dilakukan apabila terdapat kata yang belum ada adalah memasukan ke dalam tabel frekuensi, apabila terdapat kata yang sudah ada di dalam tabel frekuensi maka akan dilakukan proses *update* jumlah atau frekuensi yang ada pada tabel frekuensi.

```
for($j=0;$j<count($arr_kata[$i]['kata']);$j++)
{
        $result         =$this->db->query("SELECT word FROM table_index WHERE
word='".$arr_kata[$i]['kata'][$j]."'");
        $this->db->close();

        if($result->num_rows()=='')
        {
                if($arr_kata[$i]['kategori']==1)
                {
                        $this->db->query("INSERT INTO table_index
(word,is_olahraga) values('".$arr_kata[$i]['kata'][$j]."',1)");
                        $this->db->close();
                }
        }
        else
        {
                if($arr_kata[$i]['kategori']==1)
                {
                        $this->db->query("UPDATE table_index SET
is_olahraga=is_olahraga+1 WHERE word='".$arr_kata[$i]['kata'][$j]."'");
                        $this->db->close();
                }
        }
}
```

Gambar 5.4 *Source Code* Pembentukan Tabel Frekuensi

5.2.2 Sistem Pengkategorian

Tahapan pengkategorian pada aplikasi ini merupakan bagian utama karena merupakan inti yang melakukan perhitungan klasifikasi *naive bayes*. Pada tahapan ini terbagi menjadi beberapa tahapan sebelum dilakukan perhitungan dalam pengkategorian, diantaranya adalah preproses terhadap data uji dan transformasi menjadi bentuk yang dikehendaki. Tahapan preproses dibagi menjadi dua bagian tahap yaitu *case folding* dan *parsing*. *Case folding* merupakan tahap awal yang

dilakukan sebelum melakukan pengklasifikasian yaitu proses untuk merubah seluruh teks yang akan diolah kedalam huruf kecil, hal ini untuk memastikan ketika diolah kata-kata yang ada memiliki jenis huruf yang sama. Implementasi dari penjelasan proses *case folding* adalah pada bagian dibawah ini :

```
1 private function case_folding($content)
2 {
3     return strtolower($content);
4 }
```

Gambar 5.5 *Source Code* untuk *Case Folding*

Tahapan selanjutnya untuk preproses teks setelah dilakukan *case folding* adalah melakukan *parsing* yang merupakan proses untuk memisahkan atau memecah kalimat-kalimat menjadi satuan kata yang dipisahkan oleh spasi. Proses ini dilakukan untuk memastikan bahwa proses dilakukan pada setiap satuan kata. Proses *parsing* diimplementasikan menggunakan fungsi *explode* yang disediakan oleh PHP. Berikut adalah potongan kode untuk melakukan proses pemisahan setiap kata pada kode program :

```
1 $this->load->model('data_training_process');
2 $kata=explode(' ', trim($content));
3 $id_stop="";
```

Gambar 5.6 *Source Code* Fungsi *Explode*

Tahapan selanjutnya setelah preproses selesai dilakukan adalah proses transformasi pada tingkat kata. Hal-hal yang dilakukan pada tahap transformasi adalah penghapusan *stopword* yang merupakan proses untuk menghapus kata-kata yang dianggap tidak mempengaruhi maksud dari isi atau makna dari keseluruhan kalimat. Kata-kata yang setelah di *parsing* untuk setiap kata akan melakukan pengecekan ke dalam daftar kata *stopword.* Apabila kata tersebut terdapat di dalam daftar *stopword* maka akan dilakukan penghapusan pada kata yang bersangkutan. Selain pada kata-kata yang ada di daftar, proses ini juga akan menghapus seluruh tanda baca, angka, dan link URL yang dianggap tidak mempengaruhi isi atau makna keseluruhan kalimat. Implementasi dari penjelasan terlihat pada potongan kode program :

```
$small  =    preg_replace('/\b(https?|ftp|file):\/\/[-A-Z0-9+&@#\/%?=~_|$!:,.;]*[A-Z0-9+&@#\/%=~_|$]/i', '', $small);
$small  = preg_replace("/[0-9]/","", $small);
$small  = preg_replace('/\s+/', ' ',$small);
$small  = preg_replace('/[^a-z0-9]+/i', ' ', $small);
for($i=0;$i<count($kata);$i++)
{
        $is_exsist=$this->data_training_process->stopword_remove($kata[$i]);
        if($is_exsist==1)
        {
                $id_stop.=$i." ";
        }
}
```

Gambar 5.7 *Source Code* Fungsi Penghapusan Karakter

Selanjutnya proses yang dilakukan pada tahap transformasi adalah proses *stemming* pada kata. *Stemming* merupakan proses untuk mendapatkan kata dasar dari setiap kata yang ada pada dokumen. Proses ini bertujuan untuk mendapatkan kata penting pada kalimat. Proses *stemming* melakukan proses perubahan dari kata-kata yang ada menjadi kata dasar yang nantinya digunakan dalam pembobotan kalimat. Banyak metode yang bisa diterapkan dalam melakukan proses *stemming* bahasa Indonesia, namun dalam penelitian ini peneliti mengimplementasikan *stemming* dengan metode yang digagas oleh Arifin dan Setiono dimana konsep detail sudah dijelaskan pada bab 2. Implementasi kode dari proses *stemming* akan ditampilkan pada potongan kode dibawah ini :

```
$awalan                = $this->stemming_arifin->potong_awalan($content);
$panjang_2_awalan      = strlen($awalan[0]) + strlen($awalan[1]);
$akhiran               = $this->stemming_arifin->potong_akhiran($content);
$panjang_3_akhiran     =     strlen($akhiran[0])    +    strlen($akhiran[1])    +
strlen($akhiran[2]);
$kataDasar             =    substr($content,$panjang_2_awalan,strlen($content)    -
($panjang_3_akhiran +$panjang_2_awalan));
$_2hurufAwalAwalan = substr($awalan[0],0,2);
$_2hurufAkhirAwalan= substr($awalan[0],2,2);
$_1hurufAkhirAwalan= substr($awalan[0],2,1);
$tempKataDasar;
if($_2hurufAkhirAwalan == "ng")
{
   //untuk $kata seperti kontak, kantuk akan dilebur menjadi mengantuk
   //tambahkan dengan huruf k
   $tempKataDasar = "k".$kataDasar;
   if(in_array($tempKataDasar, $this->stemming_arifin->get_kata_kamus()))
   {
        return $tempKataDasar;
   }
}
```

Gambar 5.8 *Source Code* Fungsi *Stemming*

Pada bagian awal dari proses *stemming* terlihat seperti pada potongan kode program diatas, hal yang dilakukan adalah melakukan pendeteksian apakah kata tersebut merupakan sudah dalam bentuk kata dasar atau belum. Apabila kata tersebut sudah dalam bentuk kata dasar, maka kata tersebut yang nantinya akan diproses untuk perhitungan, apabila belum maka akan dilakukan pemotongan imbuhan pada proses *stemming*. Proses *stemming* dilakukan dengan menggunakan metode yang diperkenalkan oleh Arifin-Setiono merupakan metode *stemming* khusus untuk Bahasa Indonesia. Seluruh proses *stemming* dilakukan dengan bersumber pada morfologi Bahasa Indonesia. Pada kode program terdapat aturan-aturan yang akan diterapkan, apabila sampai pemotongan akhiran tidak ditemukan kata dasar dari suatu kata. Apabila kata tersebut sudah menggunakan aturan ini tapi masih belum juga didapatkan kata dasarnya maka kata akan dikembalikan ke bentuk semula dan dianggap kata baru. Proses transformasi selesai ketika sudah melewati tahap *stemming*. Seluruh kata sudah siap untuk dihitung bobot masing-masing kata menurut tabel frekuensi untuk mendapatkan kategori yang dimiliki oleh dokumen pada pesan singkat Twitter.

Bagian utama dari proses pengkategorian adalah pada proses *pattern discovery*. Proses ini melakukan perhitungan terhadap pesan teks Twitter yang akan dilakukan pengkategorian. Implementasi kode program dilakukan didasarkan pada algoritma naive bayes yang melakukan pengklasifikasian berdasarkan data latih yang sudah disiapkan sebelumnya. Setiap kata yang sudah melewati proses preproses teks dan proses transformasi akan dihitung probabilitas $P(w_k \mid v_j)$ dan $P(v_j)$. Berikut adalah proses perhitungan tersebut.

```
1  $jumlah_kata_kategori  = $this->db->query("select
2  sum(is_olahraga) as kata_olahraga,
3  sum(is_teknologi) as kata_teknologi,
4  sum(is_hiburan) as kata_hiburan,
5  sum(is_keuangan) as kata_keuangan,
6  sum(is_berita) as kata_berita,
7  sum(is_otomotif) as kata_otomotif
8  from table_index;");
9  foreach($jumlah_kata_kategori->result() as $row)
10 {
11         $result['kata_olahraga']=$row->kata_olahraga;
12         $result['kata_teknologi']=$row->kata_teknologi;
13         $result['kata_hiburan']=$row->kata_hiburan;
14         $result['kata_keuangan']=$row->kata_keuangan;
```

```
15        $result['kata_otomotif']=$row->kata_otomotif;
16        $result['kata_berita']=$row->kata_berita;
17 }
18
19 $jumlah_dokumen = $this->db->query("select * from
   (select count(*) as dokumen_olahraga from konten_klasifikasi where kategori=1 AND
   is_index=1)a,
   (select count(*) as dokumen_teknologi from konten_klasifikasi where kategori=6 AND
   is_index=1)b,
   (select count(*) as dokumen_hiburan from konten_klasifikasi where kategori=3 AND
   is_index=1)c,
   (select count(*) as dokumen_keuangan from konten_klasifikasi where kategori=2 AND
   is_index=1)d,
   (select count(*) as dokumen_berita from konten_klasifikasi where kategori=4 AND
   is_index=1)e,

   (select count(*) as dokumen_otomotif from konten_klasifikasi where kategori=5 AND
   is_index=1)f; ");
20 $this->db->close();
21 foreach($jumlah_dokumen->result() as $row)
22 {
23        $result['dokumen_olahraga']=$row->dokumen_olahraga;
24        $result['dokumen_teknologi']=$row->dokumen_teknologi;
25        $result['dokumen_hiburan']=$row->dokumen_hiburan;
26        $result['dokumen_keuangan']=$row->dokumen_keuangan;
27        $result['dokumen_otomotif']=$row->dokumen_otomotif;
28        $result['dokumen_berita']=$row->dokumen_berita;
29 }
30 $result['jumlah_dokumen']=    $result['dokumen_berita']+
                               $result['dokumen_otomotif']+
                               $result['dokumen_keuangan']+
                               $result['dokumen_hiburan']+
                               $result['dokumen_teknologi']+
                               $result['dokumen_olahraga'];
31
32 $result['p_dokumen_berita']= $result['dokumen_berita']/$result['jumlah_dokumen'];
33 $result['p_dokumen_otomotif']=$result['dokumen_otomotif']/$result['jumlah_dokumen'
   ];
34 $result['p_dokumen_keuangan']=$result['dokumen_keuangan']/$result['jumlah_dokumen'
   ];
35 $result['p_dokumen_hiburan']=$result['dokumen_hiburan']/$result['jumlah_dokumen'];
36 $result['p_dokumen_teknologi']=$result['dokumen_teknologi']/$result['jumlah_dokume
   n'];
37 $result['p_dokumen_olahraga']=$result['dokumen_olahraga']/$result['jumlah_dokumen'
   ];
38 return $result;
```

Gambar 5.9 *Source Code* Perhitungan $P(w_k \mid v_j)$ dan $P(v_j)$

Pada bagian ini akan ditampilkan implementasi perhitungan dari perhitungan $Vnb = \text{argmax } P(v_j) \Pi \ P(a_i \mid v_j)$ dengan potongan kode program di bawah ini :

```
1 $property=$this->twitter_model->statistic_property();
2 $peluang_olahraga      =1;
3 $peluang_hiburan       =1;
4 $peluang_berita        =1;
5 $peluang_otomotif      =1;
6 $peluang_keuangan      =1;
7 $peluang_teknologi     =1;
8
9 for($i=0;$i<count($statistic);$i++)
10 {
11 $peluang_olahraga      =
```

```
12 (($statistic[$i]['is_olahraga']+1)/($property['jumlah_kata']+$property['kata_olahr
   aga']))*$peluang_olahraga;
13
14 $peluang_hiburan      =
   (($statistic[$i]['is_hiburan']+1)/($property['jumlah_kata']+$property['kata_hibura
   n']))*$peluang_hiburan;
15
16 $peluang_berita            =
   (($statistic[$i]['is_berita']+1)/($property['jumlah_kata']+$property['kata_berita'
   ]))*$peluang_berita;
17
18 $peluang_otomotif      =
   (($statistic[$i]['is_otomotif']+1)/($property['jumlah_kata']+$property['kata_otomo
   tif']))*$peluang_otomotif;
19
20 $peluang_keuangan      =
   (($statistic[$i]['is_keuangan']+1)/($property['jumlah_kata']+$property['kata_keuan
   gan']))*$peluang_keuangan;
21 $peluang_teknologi     =
   (($statistic[$i]['is_teknologi']+1)/($property['jumlah_kata']+$property['kata_tekn
   ologi']))*$peluang_teknologi;
22                     }
23 $peluang_olahraga = $peluang_olahraga  *$property['p_dokumen_olahraga'];
24 $peluang_hiburan  = $peluang_hiburan   * $property['p_dokumen_hiburan'];
25 $peluang_berita   = $peluang_berita   * $property['p_dokumen_berita'];
26 $peluang_otomotif = $peluang_otomotif *$property['p_dokumen_otomotif'];
27 $peluang_keuangan = $peluang_keuangan *$property['p_dokumen_keuangan'];
28 $peluang_teknologi= $peluang_teknologi* $property['p_dokumen_teknologi'];
29
30 $aggregate_peluang['olahraga']       = ($peluang_olahraga);
31 $aggregate_peluang['hiburan']        = ($peluang_hiburan);
32 $aggregate_peluang['berita']         = ($peluang_berita);
33 $aggregate_peluang['otomotif']       = ($peluang_otomotif);
34 $aggregate_peluang['keuangan']       = ($peluang_keuangan);
35 $aggregate_peluang['teknologi']      = ($peluang_teknologi);
36 $nilai_max=max($aggregate_peluang);
37
38 $kategori=array_search($nilai_max, $aggregate_peluang);
39 switch($kategori)
40 {
41        case 'olahraga':{$icons="olahraga";break;}
42        case 'hiburan':{$icons="hiburan";break;}
43        case 'berita':{$icons="berita";break;}
44        case 'otomotif':{$icons="otomotif";break;}
45        case 'keuangan':{$icons="keuangan";break;}
46        case 'teknologi':{$icons="teknologi";break;}
47 }
```

Gambar 5.10 *Source Code* Perhitungan Vnb = argmax $P(v_j)$ Π $P(a_i \mid v_j)$

5.2.3 Twitter API

Proses yang dilakukan untuk mendapatkan akses data langsung pada Twitter adalah melalui API (*Application Programming Interface*) yang disediakan oleh Twitter. Untuk mendapatkan akses langsung yang harus dilakukan adalah mendaftarkan aplikasi kita pada Twitter dan kita akan mendapatkan akses pada API. Pada saat melakukan proses pendaftaran pada API hal yang kita lakukan adalah memberikan deskripsi tentang aplikasi yang akan kita bangun. Konfigurasi yang

didapatkan dan akan kita konfigurasi pada aplikasi terlihat pada potongan kode program di bawah ini.

```
1 define('CONSUMER_KEY','IhuYvQMEmCmQqSDXqBy1A');
2 define('CONSUMER_SECRET','C5S2m1PM1fGTeAfNg9UPvUUqFfl32UDp52aNKTQpso');
3 define('OAUTH_CALLBACK','http://127.0.0.1/tweecat/service_twitter_api/callback.php
4 ');
5 define('SERVICE_HOST','http://127.0.0.1/tweecat/service_algorithm/index.php/');
```

Gambar 5.11 *Source Code* Pendefinisian *Key*

Pada potongan kode program di atas terlihat `'CONSUMER_KEY'` dan `'CONSUMER_SECRET'` yang merupakan kode yang didapat dari API Twitter sebagai kunci untuk dapat melakukan koneksi pada Twitter. Sedangkan `'OAUTH_CALLBACK'` merupakan URL yang kita isikan sendiri dimana URL tersebut akan diakses ketika proses *login* pada *service* Twitter berhasil dilakukan. Pada definisi `'SERVICE_HOST'` merupakan tambahan dari implementasi dimana proses pengkategorian akan berlangsung melalui alamat *webservice* yang akan diakses pada proses klasifikasi.

Hal yang dilakukan oleh seorang pengguna aplikasi adalah dengan melakukan proses *login* dengan menggunakan *username* dan *password* yang sudah terdaftar pada Twitter sebelumnya. Twitter menerapkan metode OAuth untuk melakukan proses otentikasi agar pengguna dapat masuk ke dalam sistem. OAuth merupakan protokol yang disediakan oleh pemilik layanan (Twitter) untuk dapat dimanfaatkan pengguna agar bisa mengakses data yang ada pada Twitter. Pada OAuth pengguna tidak akan memasukkan *username* dan *password* ke dalam aplikasi pihak ketiga sehingga *password* tidak akan dapat dilihat oleh pengembang aplikasi. Pada implementasi kelas *library* yang digunakan adalah seperti potongan kode berikut ini.

```
1 class OAuthConsumer {
2   public $key;
3   public $secret;
4   function __construct($key, $secret, $callback_url=NULL) {
5     $this->key = $key;
6     $this->secret = $secret;
7     $this->callback_url = $callback_url;
8   }
9   function __toString() {
10    return "OAuthConsumer[key=$this->key,secret=$this->secret]";
11  }
12 }
```

Gambar 5.12 *Source Code* Kelas *Library* OAuth

Kelas OAuth akan dibutuhkan oleh *library* yang mengakses langsung pada API Twitter. *Library* memiliki nama TwitterOAuth yang disediakan sebelumnya. Pada TwitterOAuth disediakan proses mulai dari pembuatan koneksi sampai proses otentikasi dan dapat melakukan koneksi sehingga dapat mengakses data langsung pada layanan API Twitter. Potongan kode program adalah sebagai berikut.

```
/**
 * Set API URLS
 */
function accessTokenURL(){ return 'https://api.twitter.com/oauth/access_token'; }
function authenticateURL(){ return 'https://api.twitter.com/oauth/authenticate'; }
function authorizeURL()  { return 'https://api.twitter.com/oauth/authorize'; }
function requestTokenURL(){return 'https://api.twitter.com/oauth/request_token'; }
/**
  * Debug helpers
  */
function lastStatusCode() { return $this->http_status; }
function lastAPICall() { return $this->last_api_call; }
/**
  * construct TwitterOAuth object
  */
function __construct($consumer_key, $consumer_secret, $oauth_token = NULL,
$oauth_token_secret = NULL) {
    $this->sha1_method = new OAuthSignatureMethod_HMAC_SHA1();
    $this->consumer = new OAuthConsumer($consumer_key, $consumer_secret);
    if (!empty($oauth_token) && !empty($oauth_token_secret)) {
      $this->token = new OAuthConsumer($oauth_token, $oauth_token_secret);
    } else {
      $this->token = NULL;
    }
  }
```

Gambar 5.13 *Source Code Library* Twitter API

Setelah koneksi dan otentikasi pada API Twitter berhasil dilakukan, maka langkah selanjutnya adalah melakukan pengambilan data yaitu data pengguna dan teks pesan singkat yang ada pada *home timeline* untuk diproses sehingga mendapatkan label atau kategori. Pada bagian ini akan dijelaskan kelas yaitu akses_pengguna yang berisi kode program yang digunakan secara langsung untuk membuat aplikasi pengkategorian.

```
public function get_session_aktif(){return $this->akses_token;}
public function get_identitas(){return $this->screen_name;}
public function get_profil_image(){return $this->profile_image;}
```

Gambar 5.14 *Source Code* Fungsi-fungsi Akses Pengguna

Penjelasan potongan kode program diatas untuk *function get_session_aktif()* digunakan untuk mendapatkan status apakan pengguna sudah melakukan proses *login* atau belum, *function get_identitas()* digunakan untuk mengembalikan nama pengguna yang sudah melakukan *login* pada aplikasi, dan *function get_profile_image()* digunakan untuk mengembalikan alamat URL dari gambar foto profil dari pengguna yang sudah *login.*

```
1 $connection=newTwitterOAuth(CONSUMER_KEY,CONSUMER_SECRET,
  $access_token['oauth_token'],$access_token['oauth_token_secret']);
2 $content       = $connection->get('account/verify_credentials');
3 if(empty($_SESSION['id_str']))
4 {
5 $time_line=$connection-> get('statuses/home_timeline',array('count'=>'5',
  'include_entities'=>true,'include_rts'=>true));
6 }
```

Gambar 5.15 *Source Code* Pembuatan Koneksi Pada Twitter

Pada potongan kode program diatas baris 1 melakukan proses pembuatan koneksi pada Twitter API, setelah proses berhasil adalah melakukan pemanggilan pada baris 5 yang melakukan pemanggilan URL api yang disediakan Twitter untuk mendapatkan data berupa *statuses/home_timeline* yang memiliki parameter-paremeter tertentu. Proses selanjutnya untuk menghubungkan antara aplikasi dengan kode program yang melakukan pemrosesan pengkategorian. Hubungan yang dilakukan adalah dengan menggunakan mekanisme *webservice* dimana teks yang didapatkan akan di kirim melalui metode *post* melalui URL. Implementasi dari proses tersebut akan diperlihatkan pada potongan kode program dibawah ini.

```
1 $ch = curl_init();
2 curl_setopt($ch, CURLOPT_URL,SERVICE_HOST.'twitter_category_service/do_category');
3 curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
4 curl_setopt($ch, CURLOPT_POST, 1);
5 curl_setopt($ch, CURLOPT_CUSTOMREQUEST, 'POST');
6 $wsdata = http_build_query(array("tweet" =>$time_line[$i]['text']));
7 curl_setopt($ch, CURLOPT_POSTFIELDS, $wsdata);
8 $res = curl_exec($ch);
9 curl_close($ch);
```

Gambar 5.16 *Source Code* Penggunaan *webservice*

5.2.4 Tampilan Aplikasi

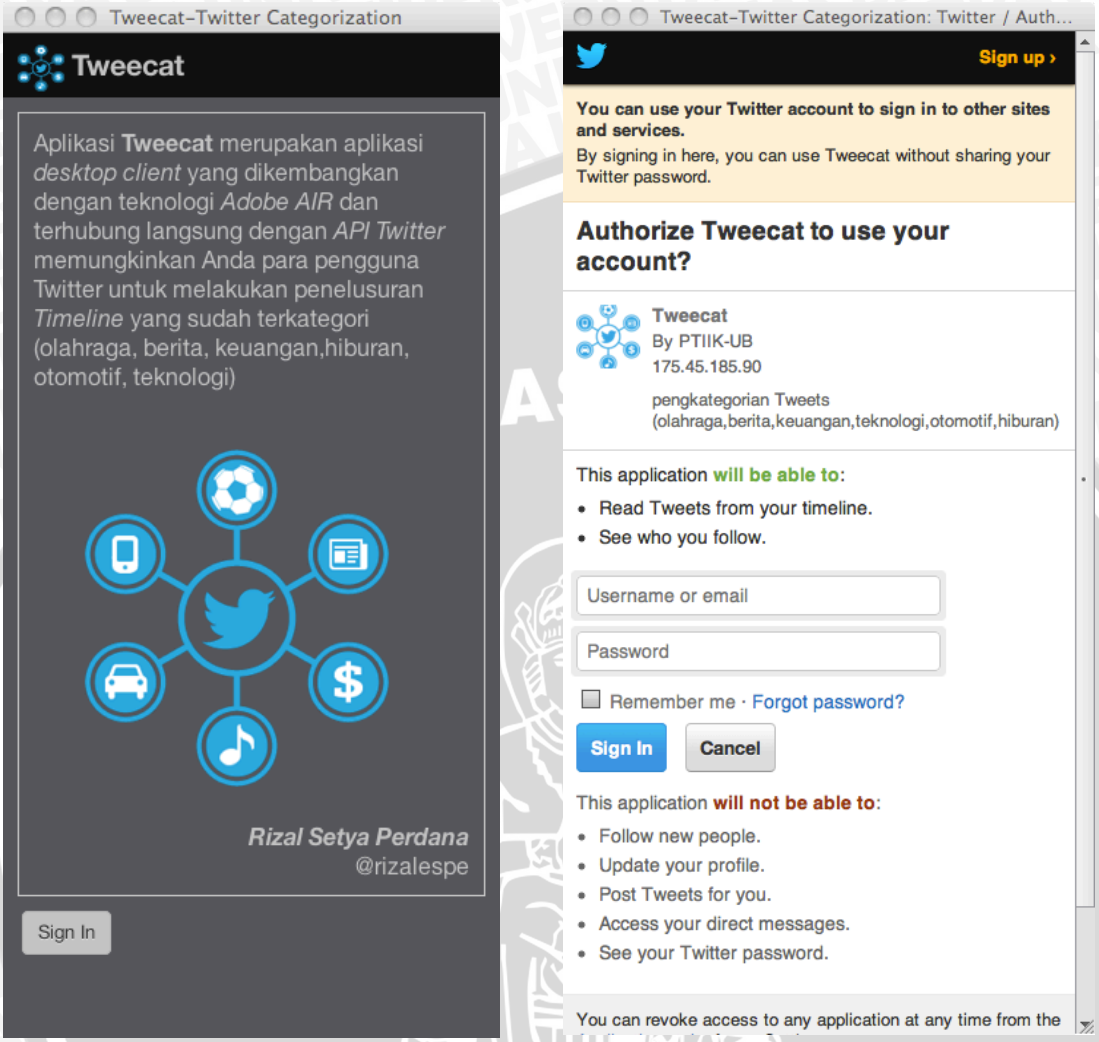
Aplikasi yang digunakan oleh pengguna secara langsung diimplementasikan menggunakan AIR (*Adobe Integrated Runtime*). Aplikasi berjalan berbasis desktop yang *multiplatform* dapat berjalan pada semua sistem operasi. Pada lingkungan AIR yang digunakan dalam pengembangan adalah HTML, XML, dan Javascript. Penggunaan AIR lebih dipilih karena aplikasi ini mudah untuk digunakan dan juga akses data akan lebih ringan karena tidak melakukan pengambilan secara berulang *assets* atau properti yang digunakan karena akan disimpan pada *client.* Potongan kode program yang digunakan dalam mengimplementasikan tampilan aplikasi adalah sebagai berikut :

```
function get_identitas()
{
        $.ajax({
                type: 'POST',
                url: this.host+'get_identitas.php',
                success:function(data)
                {
                        $('#screen_name').html(data);
                }
                });
}
function get_profile_image()
{
        $.ajax({
        type: 'POST',
        url: this.host+'get_profile_image.php',
        success:function(data)
        {
                $('#profile_image').html(data);
        }
                });
}
function gantiKategori(kode)
{
        api_get_detail({ callback: function(){InitScrooll()}});
}
```

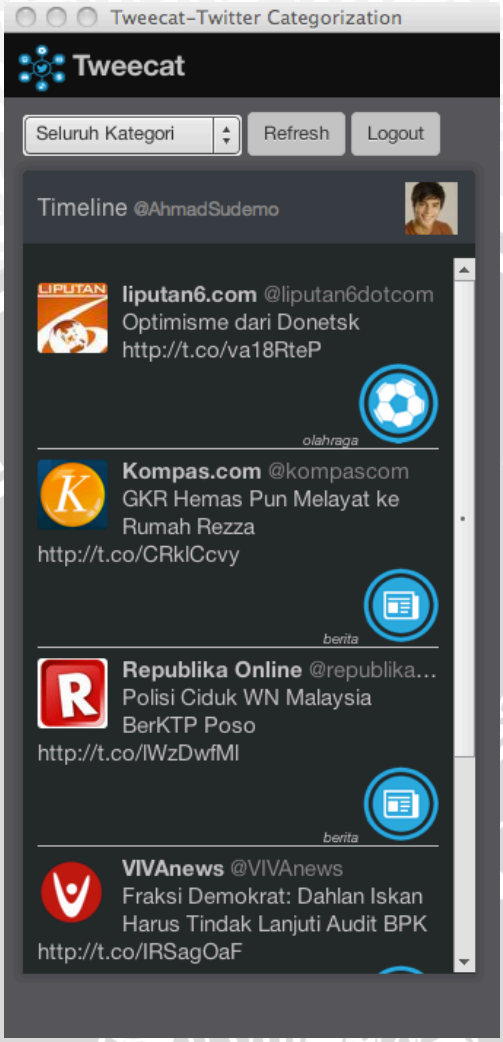
Gambar 5.17 *Source Code* Kode Javascript GUI

Pada potongan kode program di atas merupakan potongan kode program yang terdapat pada aplikasi *client* menggunakan bahasa Javascript untuk melakukan pengambilan properti yang dimiliki oleh pengguna. *Function get_identitas()* merupakan fungsi yang melakukan *request* menggunakan AJAX *post* pada URL yang

ditentukan yang mengembalikan URL foto profil yang dimiliki pengguna yang sudah login. Pada aplikasi *client* proses *request* dilakukan dengan menggunakan AJAX *post* yang secara tidak nampak akan melakukan *request* pada suatu alamat URL yang sudah ditentukan. Berikut adalah tampilan GUI yang merupakan implementasi dari aplikasi *client* dalam mengkategorikan pesan singkat pada Twitter.



Gambar 5.18 Tampilan Aplikasi *Client* Saat Dibuka Pertama Kali dan Halaman Login



Gambar 5.19 Tampilan Aplikasi *Client* pada saat Mengkategorikan