BAB IV
IMPLEMENTASI

Bab ini membahas implementasi perangkat lunak berdasarkan hasil yang telah diperoleh dari analisis kebutuhan dan proses perancangan perangkat lunak yang dibuat pada bab III. Bab ini terdiri dari penjelasan tentang spesifikasi sistem, batasan-batasan dalam implementasi, implementasi algoritma pada program, implementasi antarmuka, dan implementasi metode.

4.1 Spesifikasi SIstem

Hasil analisis kebutuhan dan perancangan perangkat lunak yang telah diuraikan pada bab III menjadi acuan untuk melakukan implementasi menjadi aplikasi yang dapat berfungsi sesuai dengan kebutuhan. Spesifikasi aplikasi diimplementasikan pada spesifikasi perangkat keras dan perangkat lunak.

4.1.1 Spesifikasi Perangkat Keras

Pengembangan aplikasi *Sentiment analysis* pada *review* barang berbahasa Indonesia dengan metode K-Nearest Neighbor ini menggunakan sebuah komputer dengan spesifikasi perangkat keras yang dijelaskan pada Tabel 4.1.

Tabel 4. 1 Spesifikasi Perangkat Keras Komputer

Nama Komponen	Spesifikasi
Prosesor	Intel(R) Core i3 Duo 2350M
Memori(RAM)	4 GB
Harddisk	640 GB
VGA	Nvidia GeForce 610M 2GB

Sumber: Implementasi

4.1.2 Spesifikasi Perangkat Lunak

Pengembangan aplikasi *Sentiment analysis* pada *review* barang berbahasa Indonesia dengan metode K-Nearest Neighbor ini menggunakan perangkat lunak dengan spesifikasi yang dijelaskan pada Tabel 4.2.

Tabel 4. 2 Spesifikasi Perangkat Lunak Komputer

Nama	Spesifikasi
Sistem Operasi	Windows 7 Ultimate 32-bit (6.1 build 7601)S
Bahasa Pemrograman	HTML dan PHP 5
Tools pemrograman	Aptana Studio 3, build: 3.0.6.201110251455
Server localhost	XAMPP versi 3.1.0
DBMS	MySQL versi 5.5.27
Web Browser	Google Chrome versi 26.0.1410.65

Sumber: Implementasi

4.2 Batasan-Batasan Implementasi

Batasan implementasi adalah batasan proses yang dapat dilakukan sistem sesuai dengan perancangan awal sistem. Batasan implementasi ditampilkan agar skripsi ini memiliki ruang lingkup yang jelas dalam mengimplementasikan sistem. Beberapa batasan dalam mengimplementasikan aplikasi *Sentiment analysis* pada *review* barang berbahasa Indonesia dengan metode K-Nearest Neighbor ini adalah sebagai berikut:

- Aplikasi *Sentiment analysis* pada *review* barang berbahasa Indonesia dengan metode K-Nearest Neighbor dirancang dan dijalankan menggunakan *webapplication.*
- Metode penyelesaian masalah yang digunakan adalah *k-Nearest Neighbor*.
- Dokumen yang digunakan sebagai data latih dan data uji adalah berasal dari *website* bhinneka.com pada jenis kamera digital.
- Dokumen yang digunakan merupakan dokumen berbahasa Indonesia.
- Output yang dikeluarkan berupa hasil analisis sentimen.

- Penentuan sentimen *review* berdasarkan pada frekuensi kemunculan kata, bukan pada struktur *semantic*.

4.3 Implementasi Algoritma

Aplikasi *Sentiment analysis* pada *review* barang berbahasa Indonesia dengan metode K-Nearest Neighbor ini mempunyai beberapa proses utama, yaitu proses mengecek kemiripan dokumen antara dokumen uji dengan semua dokumen latih, dan proses penentuan sentimen dari dokumen *review*.

4.3.1 Implementasi Algoritma Proses *Input* Dokumen

Proses *input*dokumen dilakukan dengan memasukkan *input* berupa daftar dokumen yang berisi kata-kata komentar / *review*. Data yang dimasukkan akan diolah dengan *text processing*, menghitung bobot dokumen menggunakan TF-IDF, dan melakukan pengklasifikasian.Tabel 4.3 adalah tabel daftar fungsi sistem penentuan kemiripan dokumen skripsi.

Tabel 4. 3 Daftar Fungsi Pada Sistem

No.	Proses	Fungsi	Keterangan
1.	Text Mining / Prepoccesing	- function Cleaning()	Fungsi untuk menghilangkan tanda baca seperti koma, titik, petik dua, dan lain-lain.
		- Function Casefolding()	Fungsi untuk mengecilkan huruf kapital.
		- function parsing()	Fungsi untuk memecah kalimat menjadi kata yang berdiri sendiri.
		- function stopword_removal()	Fungsi untuk menghilangkan kata yang tidak penting seperti ‘yang’,’adalah’,’itu’, dan lain – lain.
		- function stemming()	Fungsi untuk memecah kata menjadi kata dasar

No.	Proses	Fungsi	Keterangan
2.	Pembobotan	- function tf-idf.function()	Fungsi untuk menghitung

			nilai bobot berupa nilai DF, IDF, TF-IDF sampai nilai Cossine Similarity (nilai similaritas antar dokumen).

No.	Proses	Fungsi	Keterangan
3.	Pengklasifikasian	- function knn.function()	Fungsi untuk mengklasifikasikan sesuai dengan knn decision rules dan menghasilkan masing-masing kelas

Sumber: Implementasi

4.3.1.1 Proses Text Mining

Proses ini tahap pertama dalam menggali informasi suatu dokumen. Tahap ini terdiri dari *Cleaning*, *Casefolding, parsing, stopword removal dan stemming*. *Casefolding* berfungsi untuk mengecilkan huruf capital menjadi standar. *cleasing* berguna untuk membersihkan dokumen dari tanda baca, angka dan simbol-simbol yang merusak dokumen . *Parsing* untuk memecah dokumen menjadi kata-kata. Untuk *Stopword removal* berguna untuk menghilangkan kata-kata yang kurang berguna ada dokumen. Dan proses terakhir adalah *stemming*, *stemming* adalah proses memecah kata-kata berimbuhan menjadi kata dasar. *Stemming* yang digunakan adalah stemming Arifin-Setiono. *Sourcecode stemming* arifin-setiono dapat dilihat pada lampiran. Pemanggilan fungsi *Cleaning* secara keseluruhan ditunjukkan pada *Sourcecode* 4.1.

```
1  function Cleaning($data){
2      $list = array('.',',','[',']','(',')','{','}','⟨',':',';','、','–','!','«',
3      '»','?','‘','’','“','”','"',';',':','/','⁄','&','*','@','•','^','†','‡','°',
4      '〃','¡','¿','#','÷','ª','%','‰','㎜','¶','′','§','~','_','¦','|',
5      '©','®','℠','℗','™','⁂','☞','⁉','◊','※','ˆ','Ⱥ','฿','₡',
6      '¢','₡','₲','₤','$','₫','ъ','Ɒ','€','ƒ','₣','₲','₴','₭','□','₥',
7      '₥','₦','₧','₱','ﷺ','£','Rs','₪','₮','₩','¥','،','1','2','3','4','5','6','7','8','9','0');
8
```

```
9         return str_replace($list, " ", $data);
10    }
```

Sourcecode 4. 1 Implementasi Algoritma *Cleaning* Pada Dokumen

Sumber: Implementasi

Fungsi Cleaning membersihkan dokumen dari symbol-simbol yang kurang berguna untuk sistem. Berikut fungsi Casefolding yang ditunjukkan pada S*ourcecode* 4.2.

```
1   function Casefolding($data){
2         $data = strtolower($data); //mengubah huruf besar menjadi kecil
3         return $data;
4   }
```

Sourcecode 4. 2 Implementasi Algoritma *Casefolding*

Sumber: Implementasi

Fungsi Casefolding yaitu merubah semua huruf menjadi huruf kecil dengan menggunakan strtolower. Untuk fungsi parsing akan ditunjukan pada *Sourcecode* 4.3.

```
1   function parsing ($data){
2         $hasil = explode(" ", $data); //memecah dokumen (kalimat) menjadi kata-kata
3         return $hasil;
4   }
```

Sourcecode 4. 3 Implementasi Algoritma *Parsing*

Sumber: Implementasi

Fungsi parsing yaitu memecah kalimat menjadi berupa kata-kata. Untuk fungsi stopword removal akan ditunjukan pada *Sourcecode* 4.4,

```
1   Function stopword_removal($array){
2         $directory = array(".. ada pada lampiran..");
3         foreach ($array as $key => $value) {
4               if(in_array($value, $directory)){
5                     unset($array[$key]);}}
```

Sourcecode 4. 4 Implementasi Algoritma *Stopword Removal*

Sumber: Implementasi

Fungsi stopword removal yaitu menghilangkan kata-kata yang kurang berguna untuk sistem dan dihilangkan dengan cara membandingkan data terhadap kumpulan kata-kata yang kurang berguna. Untuk fungsi terakhir pada proses preprosessing data adalah tahap stemming, untuk *sourcecode* stemming ditunjukkan pada *Sourcecode* 4.5.

```
1  function stemming($array){
2          $newList = array();
3          $kamus_kata = array( ....kumpulan kata dasar selengkapnya pada lampiran..... );
4          foreach ($array as $key => $value) {
5                  $pieces = stemmingArifin($value,$kamus_kata);
6                  //if (in_array ($pieces, $kamus_kata)) //misal kata baru ditambahkan ke
7  kamus menjadi kata dasar baru
8                  array_push($newList,$pieces);}
9
```

Sourcecode 4. 5 Implementasi Algoritma *Stemming*

Sumber: Implementasi

Tahap *stemming* pada sistem ini ditunjukkan dengan fungsi *stemming*.Tahap pertama proses *stemming* adalah mengambil tiap kata pada dokumen kemudian dibandingkan dengan kamus kata dasar, jika ditemukan kata tidak dasar maka akan dijadikan kata dasar itu sendiri dengan memecahnya, sehingga nantinya akan menghasilkan kata dasar. *Stemming* yang digunakan dalam skripsi ini adalah stemming Arifin-Setiono.*Stemming* ini berdasarkan penelitian yang dilakukan oleh Arifin-Setiono. *Sourcecode stemming* Arifin-Setiono dapat dilihat pada lampiran.

4.3.1.2 Proses Pembobotan TF-IDF

Proses ini pembobotan TF-IDF berfungsi untuk mengetahui bobot tiap dokumen. Hasil dari pembobotan TF-IDF nantinya akan digunakan sebagai nilai dari masing-masing dokumen untuk menentukan kedekatan antar dokumen pada proses klasifikasi knn. *Sourcecode* 4.6 menunjukan proses pembobotan dokumen.

```
1  function tfidf($data_preprocessing, $label){
2          $DF = array();
```

```
3           $IDF = array();
4           for ($i=0; $i < count($data_preprocessing); $i++) {
5                   $DF[$i] = 0;
6                   $count = 0;
7                   for ($j=0; $j < count($data_preprocessing[$i])-1; $j++) {
8                           if($data_preprocessing[$i][$j] >= 1) $count++;
9                   }
10                  $DF[$i] = $count;
11          }
12          for ($i=0; $i <count($data_preprocessing); $i++) {
13                  $IDF[$i] = log10(count($data_preprocessing[$i])-1/$DF[$i]);
14          }
15          $TF_IDF=array();
16          for ($i=0; $i <count($data_preprocessing); $i++) {
17                  for ($j=0; $j <count($data_preprocessing[$i]); $j++) {
18                          $TF_IDF[$i][$j] = $IDF[$i] * $data_preprocessing[$i][$j];
19                  }}
```

Sourcecode 4. 6 Implementasi Algoritma TF-IDF, Nilai TF Dan DF Pada Dokumen

Sumber: Implementasi

Fungsi tfidf merupakan fungsi untuk menghitung nilai pembobotan menggunakan algoritma tf-idf untuk dokumen-dokumen yang digunakan. Setelah di dapat nilai tf-idf dari dokumen, maka langkah selanjutnya adalah untuk menghitung nilai similaritas antar dokumen yang nanti selanjutnya akan diperingkatkan.

```
1   $similarity = array();
2           for ($i=0; $i <count($TF_IDF); $i++) {
3                   $last_index = count($TF_IDF[$i]);
4                   for ($j=0; $j<$last_index-1; $j++){
5                           $similarity[$i][$j] = $TF_IDF[$i][$j] *
6   $TF_IDF[$i][$last_index-1];
7                   }
8                   }
9           $sum_similarity = array();
10          $coloumn = count($similarity[0]);
11          $row     = count($similarity);
```

```
12
13          for ($i=0; $i < $coloumn; $i++) {
14                  $sum_similarity[$i] = 0;
15                  for ($j=0; $j < $row ; $j++) {
16                          $sum_similarity[$i] += $similarity[$j][$i];
17          }}
18          $sqrt_similarity = array();
19          $coloumn = count($TF_IDF[0]);
20          $row     = count($TF_IDF);
21          for($i=0; $i < $coloumn; $i++){
22                  $sqrt_similarity[$i] = 0;
22                  for ($j=0; $j < $row; $j++){
23                          $sqrt_similarity[$i] += $TF_IDF[$j][$i];
24                  }
25                  $sqrt_similarity[$i] = sqrt($sqrt_similarity[$i]);
26      }
27          $cossim = array();
28          for ($i=0; $i <count($sum_similarity); $i++) {
29                  $cossim[$i] =
30  $sum_similarity[$i]/($sqrt_similarity[$i]*$sqrt_similarity[count($sqrt_similarity)-1]);
31          }return $cossim;
```

Sourcecode 4. 7 Implementasi Algoritma Pembobotan TF-IDF Pada Dokumen

Sumber: Implementasi

Fungsi bobotTfIdf merupakan proses penghitungan bobot kata pada tiap – tiap dokumen. Bobot ini nanti akan dinormalisasi menjadi bentuk yang lebih sederhana. Hasil normalisasi akan digunakan untuk penghitungan jarak *euclidean* dan menghitung kemiripan dokumen menggunakan *cosine similarity*.Penghitungan kemiripan dokumen dengan *cosine similarity* ditunjukkan pada *Sourcecode* 4.8.

```
1   function knn($cossim, $label, $k){
2           //          $k = 2;//$_POST["k"]; //nilai k
3           $all = array();
4           for ($i=0;$i<count($cossim);$i++){
5                   $all[] = array(
6                           array($cossim[$i], $label[$i])
7                   );
8           };
9           rsort($all);
10          $tmp = array();
11          for ($i=0;$i<$k;$i++){
12                  $tmp[] = $all[$i];
13          }
14          //echo "<br>potong $k : ";
15          //print_r($tmp);
16          $class1 = 0;
17          $class2 = 0;
18          $class3 = 0;
19          for ($i=0; $i <count($tmp) ; $i++) {
20                  if(strtolower($tmp[$i][0][1]) == "positif"){ // label 1 positif
21                          $class1  += $tmp[$i][0][0];
22                  }//else if(strtolower($tmp[$i][0][1]) == "netral"){ // label 2 netral
23                          //$class2 += $tmp[$i][0][0];  }
24                          else if(strtolower($tmp[$i][0][1]) == "negatif"){ // label 3
25  negatif
26                          $class3  += $tmp[$i][0][0];
27                  }}
28          return array(
29                  "positif" =>$class1,
30                  //"netral" =>$class2,
31                  "negatif" =>$class3 );}
```

Sourcecode 4. 8 Implementasi Algoritma Kemiripan Dokumen Dengan *Cosine Similarity*

Sumber: Implementasi

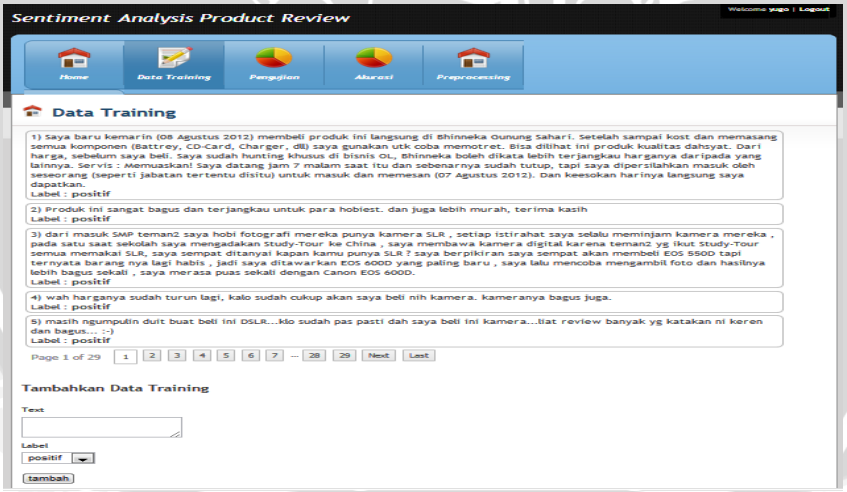
Fungsi knn merupakan proses perhitungan jarak *Euclidian* dan setelahnya dilakukan perhitungan kemiripan antar dokumen dengan menggunakan algoritma *cosine similarity*. Pada proses inilah dapat diketahui bahwa dokumen yang diujikan akan masuk kedalam kelas positif atau negatif, tentunya user akan menentukan inputan nilai k untuk proses tersebut.

4.4 Implementasi Antar Muka

Antarmuka aplikasi *Sentiment analysis* pada *review* barang berbahasa Indonesia dengan metode K-Nearest Neighbor digunakan oleh pengguna untuk berinteraksi dengan aplikasi. Antarmuka *sentiment analysis* ini dibagi menjadi antarmuka halaman cek kemiripan dokumen, halaman tambah dokumen, dan halaman daftar dokumen.

4.4.1 Tampilan Halaman Data Training

Tampilan halaman Data Training akan menampilkan *option* pilihan *input* dokumen baru atau memilih dokumen dari *database*. Gambar tampilan halaman data training ditunjukkan pada gambar 4.1.

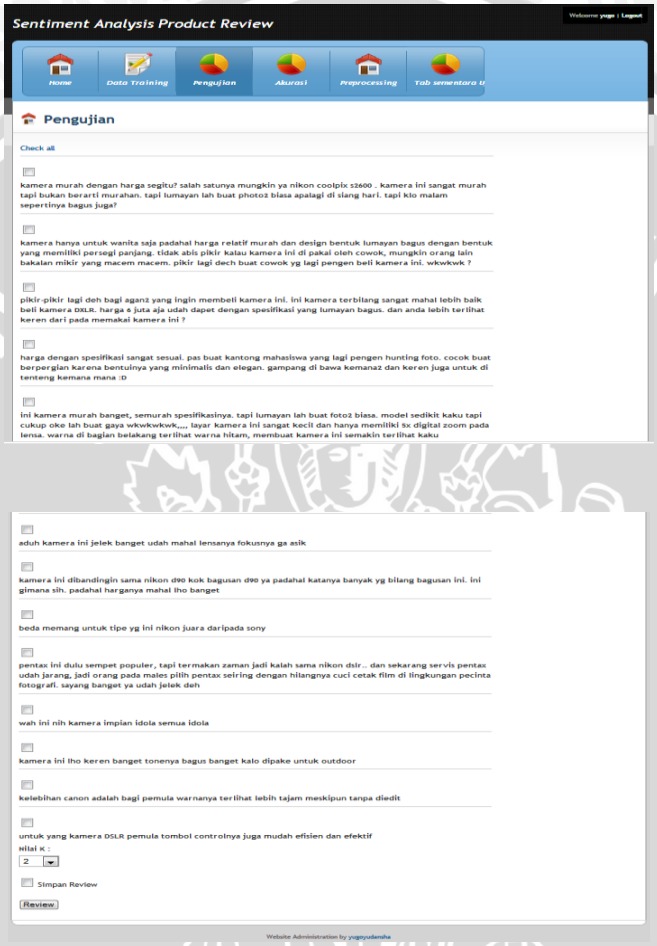


Gambar 4. 1 Tampilan Halaman dan *Form* Tambah Data Training

Sumber: Implementasi

4.4.2 Tampilan Halaman Pengujian

Tampilan halaman pengujian akan menampilkan *form* pilihan input dokumen yang akan diklasifikasikan menurut kelas yang tersedia dan penentuan nilai k yang diinginkan user untuk *input*. Gambar *form* pada halaman pengujian ditunjukkan pada Gambar 4.2.

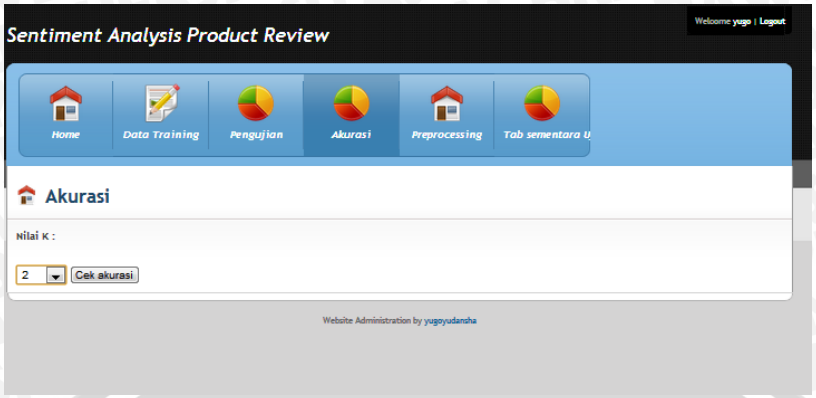


Gambar 4. 2 *Form* Pengujian

Sumber: Implementasi

4.4.3 Tampilan Halaman Akurasi

Tampilan halaman akurasi akan menampilkan hasil klasifikasi dokumen yang diujikan dan tersimpan pada *database* selanjutnya dihitung dengan menggunakan metode akurasi. Gambar *form* halaman akurasi ditunjukkan pada Gambar 4.3.



Gambar 4. 3 *Form* Halaman Akurasi

Sumber: Implementasi

4.4.4 Tampilan Halaman *Preprocessing*

Tampilan halaman *preprocessing* akan menampilkan hasil dari pemrosesan pada *preprocessing* pada masing-masing proses seperti *Cleaning*, *parsing*, *Casefolding*, *stopword removal* sampai proses *stemming* . Gambar *form* halaman *preprocessing* ditunjukkan pada Gambar 4.4.

Gambar 4. 4 *Form* Halaman *Preprocessing*

Sumber: Implementasi