**BAB V**

**HASIL PENELITIAN DAN PEMBAHASAN**

1. **Hasil Penelitian**
2. **Pengujian *White Box***

*White box testing* adalah metode desain *test case* yang menggnakan struktur kontrol desain prosedural untuk mendapatkan *test case*. Dalam pelaksanaannya, teknik pengujian *white box* ini mempunyai empat langkah, yaitu sebagai berikut:

1. Menggambarkan *flowgraph* (aliran kontrol) yang ditransfer dari *flowchart*
2. Menghitung *cyclomatic complexsity* (CC) untuk *flowgraph* yang telah dibuat
3. Menentukan jalur pengujian dari *floegraph* berjumlah sesuai dengan *cyclomatic complexsity* yang telah ditentukan
4. *Bases path testing*, yaitu teknik yang memungkinkan perancangan *test case* mengukur kompleksitas logis dari desain prosedural dan menggunakannya sebagai pedoman untuk menetapkannya basis set dari jalur eksekusi.

Hasil rancangan dengan menggunakan *white box testing* pada alur program, struktur logika program atau prosedur programnya dengan cara pemetaan *flowchart* ke dalam *flowgraph* kemudian menghitung besarnya jumlah *edge* dan *node* dimana jumlah *edge* dan *node* ini akan menentukan besarnya *cyclomatic complexsity* (CC). Perhitungan CC untuk melihat kesamaan nilai antara *white box testing*, jika nilai V(G)= CC pada *white box testing* dengan *bases path testing* maka pengujian telah berhasil.

Beberapa istilah saat pembuatan *flowgraph*:

1. *Node*, yaitu lingkaran pada *flowgraph* yang menggambarkan satu atau lebih perintah prosedural
2. *Edge*, yaitu tanda panah yang menggambarkan aliran kontrol dan setiap *node* harus mempunyai tujuan *node*
3. *Region*, yaitu daerah yang dibatasi oleh *node* dan *edge* dan untuk menghitung *region* daerah di luar *flowgraph* juga harus dihitung
4. *Predicate Node*, yaitu kondisi yang terdapat pada node dan mempunyai karakteristik dua atau lebih *edge* lainnya.
5. **Prosedur untuk proses enkripsi basis data**

<?php

error\_reporting(0);

$eks = array('.sql'); 3

$ekst = strrchr($\_FILES['teks']['name'],'.'); 1

$kunci = $\_POST['kcenkripsi']; 2

if(!in\_array($ekst,$eks)) {

echo "<script>alert('Masukkan File .sql!');location.replace('json-enkripsi-$result.mu');</script>";

}

4 if ($\_FILES['teks']['error'] == UPLOAD\_ERR\_OK && is\_uploaded\_file($\_FILES['teks']['tmp\_name'])) {

$kalimat = file\_get\_contents($\_FILES['teks']['tmp\_name']);

}

?>

<form role='form' method='post' action='json-simpann-<?php echo $result ?>.mu'>

<div class='form-group input-control file'>

<label><b>Plain Text</b></label>

<textarea class='form-control input-lg' name='plain'><?php echo $kalimat ?></textarea>

</div> 9

<div class='form-group'>

<label><b>Key</b></label>

<input class='form-control input-lg' type='text' name='kcenkripsi' value='<?php echo $kunci ?>' />

</div>

<?php

$time = round(microtime(true) \* 1000);

$time = explode(' ', $time); 5

$time = $time[1] + $time[0];

$start = $time;

?>

<?php

function setupkey(){

for($i=0;$i<strlen($kunci);$i++){

$key[$i]=ord($kunci[$i]); }

global $m;

$m=array();

for($i=0;$i<256;$i++){

$m[$i] = $i; }

$j = $k =0;

for($i=0;$i<256;$i++){

$a = $m[$i];

$j = ($j + $m[$i] + $key[$k]) % 256; 6

$m[$i] = $m[$j];

$m[$j] = $a;

$k++;

if($k>15){

$k=0;

}

}

}

function crypt2($inp){

global $m;

$x=0;$y=0;

$bb='';

$x = ($x+1) % 256; 6

$a = $m[$x];

$y = ($y+$a) % 256;

$m[$x] = $b = $m[$y];

$m[$y] = $a;

$bb = ($inp^$m[($a+$b) % 256]) % 256; 7

return $bb;

}

setupkey();

for($i=0;$i<strlen($kalimat);$i++){

$kode[$i]=ord($kalimat[$i]); 8

$b[$i]=crypt2($kode[$i]);

$c[$i]=chr($b[$i]);

}

?>

<?php

$time = round(microtime(true) \* 1000);

$time = explode(' ', $time); 5

$time = $time[1] + $time[0];

$finish = $time;

$total\_time = round(($finish - $start));

?>

<div class='form-group'>

<label><b>Cipher Text:</b></label>

<textarea class='form-control' style='height:500px;' name='cipher'>

<?php

for($i=0;$i<strlen($kalimat);$i++){

echo $c[$i]; }

?> 9

</textarea>

</div>

<div class='form-group'>

<label><b>Waktu Kecepatan Encrypsi data RC4</b></label>

<input class='form-control input-lg' type='text' name='kcenkripsi' value='<?php echo $qq ?> / Detik(s)'/>

</div>

<div class='form-group'>

<label><b>Nama File</b></label>

<input class='form-control input-lg' type='text' name='nama' required='required'/> 10

</div>

<div class='form-group'>

<input type='submit' value='Simpan' class='btn btn-primary' /> 11

</div>

</form>

1. **Flowgraph untuk pengamanan basis data**



**Gambar 5.1** *Flowgraph* proses pengamanan basis data

Dari *flowgraph* diatas, maka didapatkan:

Region (R) = 2

Node (N) = 11

Edge (E) = 11

Predicate Node (P) = 1

1. **Menghitung Nilai *Cyclomatic Complexity* (CC)**

*Cyclomatic Complexity* digunakan untuk mencari jumlah path dalam satu *flowgraph. Cyclomatic Complexity* V(G) untuk grafik alir dihitung dengan rumus:

V(G) = E – N + 2

= 11 – 11 + 2

= 0 + 2

= 2

Atau, V(G) = P + 1

= 1 + 1

V(G) = 2

1. **Menentuukan Basis Path**

**Tabel 5.1** Pengujian Basis Path

|  |  |  |
| --- | --- | --- |
| **Jalur** | **Path** | **Cek** |
| 1 | 1-2-3-5-6-7-8-9-10-11 | OK |
| 2 | 1-2-3-4-1-2-3-5-6-7-8-9-10-11 | OK |

Ketika aplikasi dijalankan, maka terlihat bahwa semua basis set yang dihasilkan oleh simpil telah dieksekusi satu kali. Berdasarkan ketentuan tersebut dari segi kelayakan *software*, sistem ini telah memenuhi syarat.

1. **Pengujian *Black Box***

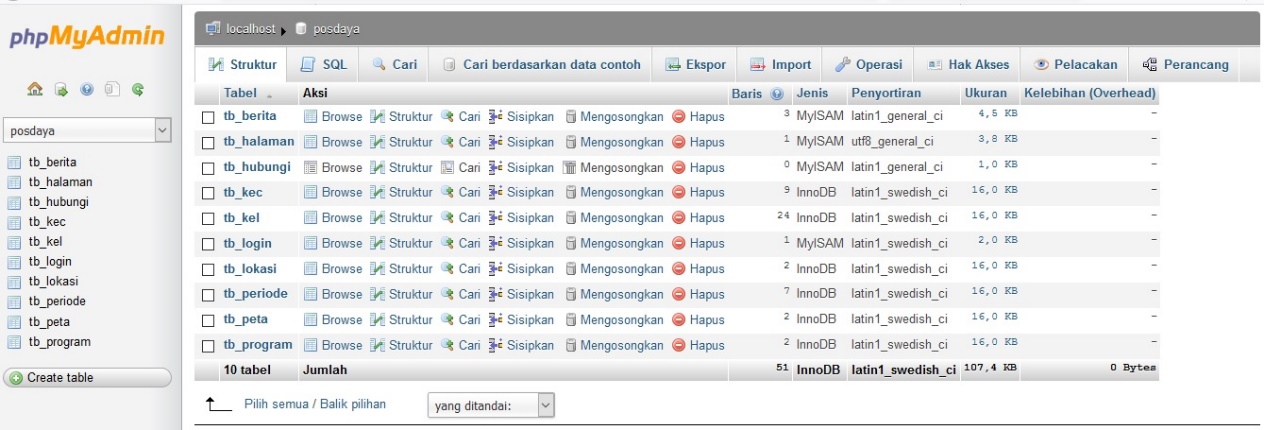
**Tabel 5.2** Pengujian *Black Box*

|  |  |  |  |
| --- | --- | --- | --- |
| **Input/Event** | **Fungsi** | **Hasil** | **Hasil Uji** |
| Klik menu enkripsi | Menampilkan menu enkripsi | Form enkripsi ditampilkan | Sesuai |
| Klik tombol browse | Memilih file plainteks basis data (.sql) | File plainteks basis data terpilih | Sesuai |
| Klik tombol enkripsi | Mengenkripsi file plainteks basis data dan menampilkan form otuput enkripsi | Basis data terenkripsi dan form output enkripsi ditampilkan | Sesuai |
| Klik tombol simpan | Menyimpan hasil enkripsi basis data | Cipherteks basis data tersimpan | Sesuai |
| Klik menu Dekripsi | Menampilkan menu dekripsi | Form dekripsi ditampilkan | Sesuai |
| Klik tombol browse | Memilih file basis data (.sql) | File cipherteks basis data terpilih | Sesuai |
| Klik tombol dekrripsi | Mengenkripsi file cipherteks basis data dan menampilkan form otuput dekripsi | Basis data terdekripsi dan form output dekripsi ditampilkan | Sesuai |
| Klik tombol simpan | Menyimpan hasil dekripsi basis data | Plainteks basis data tersimpan | Sesuai |
| Klik menu utama | Menampilkan menu utama | Menu utama ditampilkan | Sesuai |
| Klik Menu About | Menampilkan menu about | Menu about ditampilkan | Sesuai |

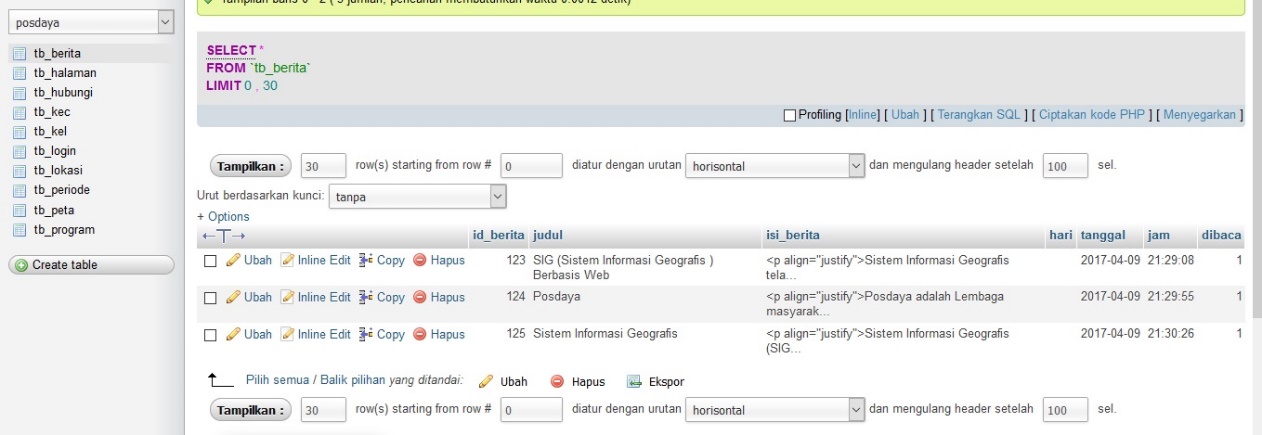
Ketika aplikasi dijalankan,maka terlihat semua pengujian *black box* yang dihasilkan telah dieksekusi satu kali. Berdasarkan ketentuan tersebut dari segi kelayakan *software*, sistem ini telah memenuhi syarat.

1. **Pembahasan**
2. **Pengujian enkripsi dan dekripsi basis data**
3. **Pengujian enkripsi basis data**

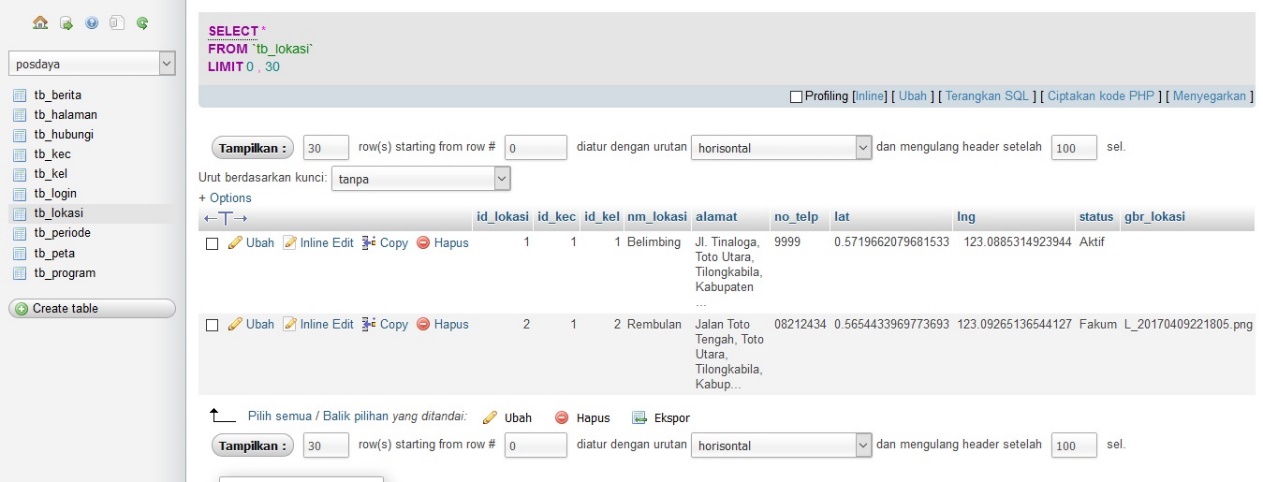
Pengujian akan dilakukan pada basis data sistem informasi geografis pemetaan posdaya Kota Gorontalo. Dibawah ini beberapa tampilan basis data sebelum di enkripsi.



**Gambar 5.2** Tampilan basis data sebelum di enkripsi

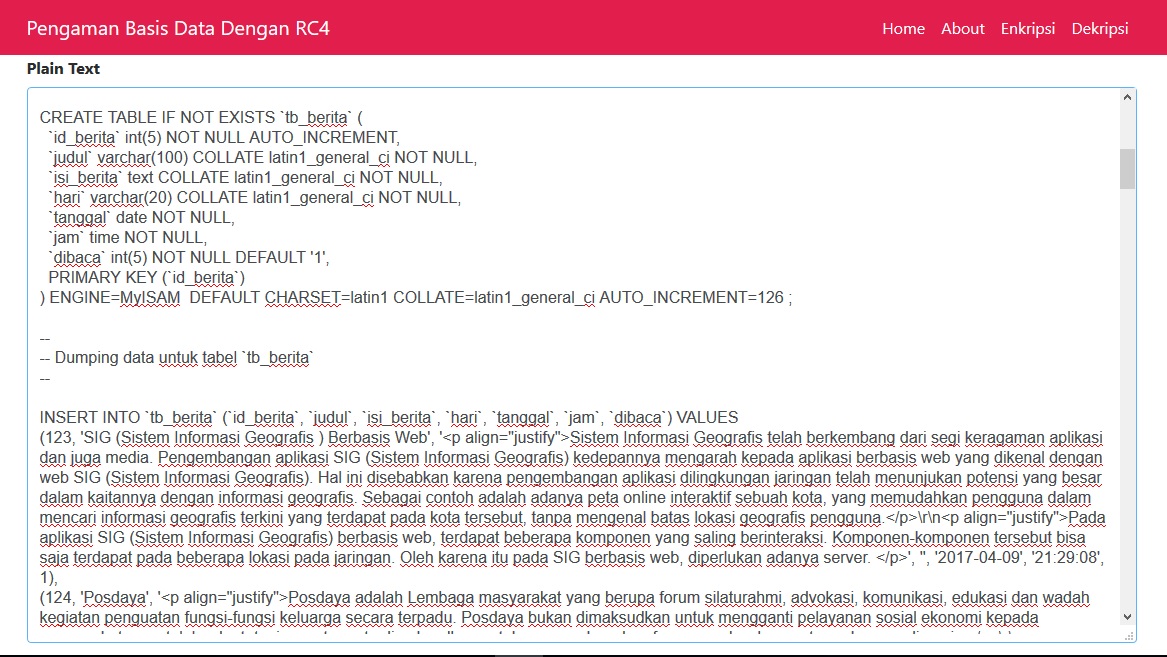


**Gambar 5.3** Tabel berita sebelum di enkripsi



**Gambar 5.4** Tabel lokasi sebelum di enkripsi

Berikut merupakan hasil dari pengenkripsian yang dilakukan pada basis data sistem informasi geografis pemetaan posdaya Kota Gorontalo.



**Gambar 5.5** Plainteks sebelum dienkripsi

Dari gambar diatas dapat dilihat bahwa plainteks sebelum melalui proses enkripsi dengan RC4 masih mempunyai makna yang dapat dimengerti.



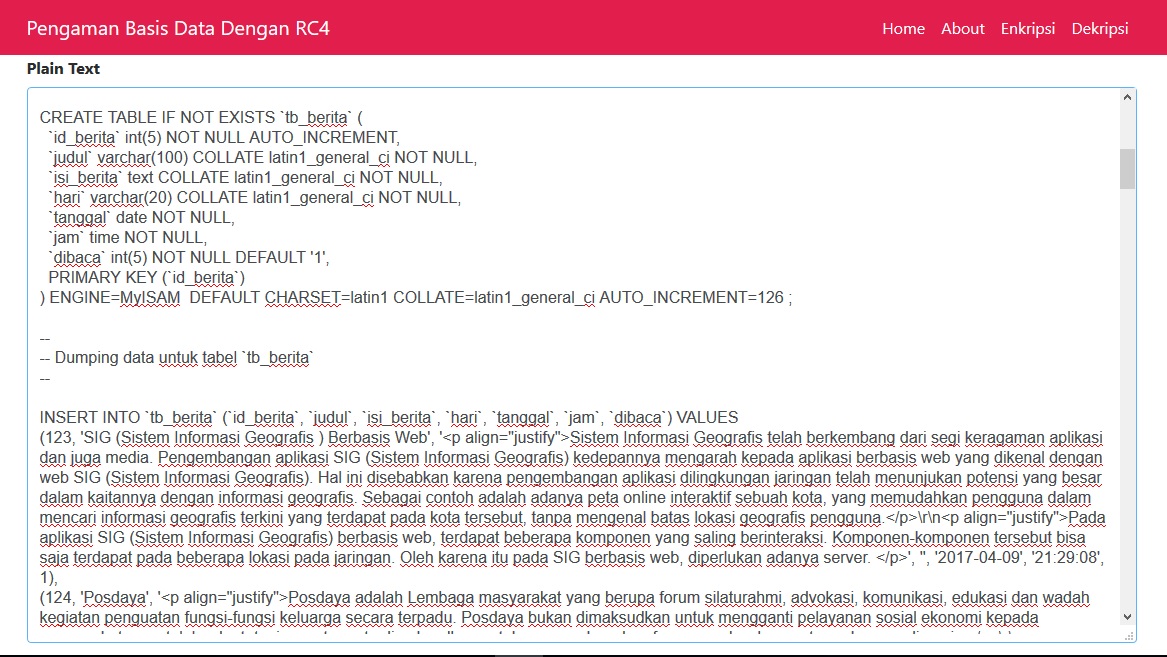
**Gambar 5.6** Plainteks setelah dienkripsi menjadi cipherteks

Setelah melalui proses pengenkripsian dengan algoritma RC4 plainteks yang sebelumnya masih mempunyai makna dan dapat di mengerti, menjadi tidak mempunyai makna lagi dan tidak dapat di mengerti (Cipherteks).

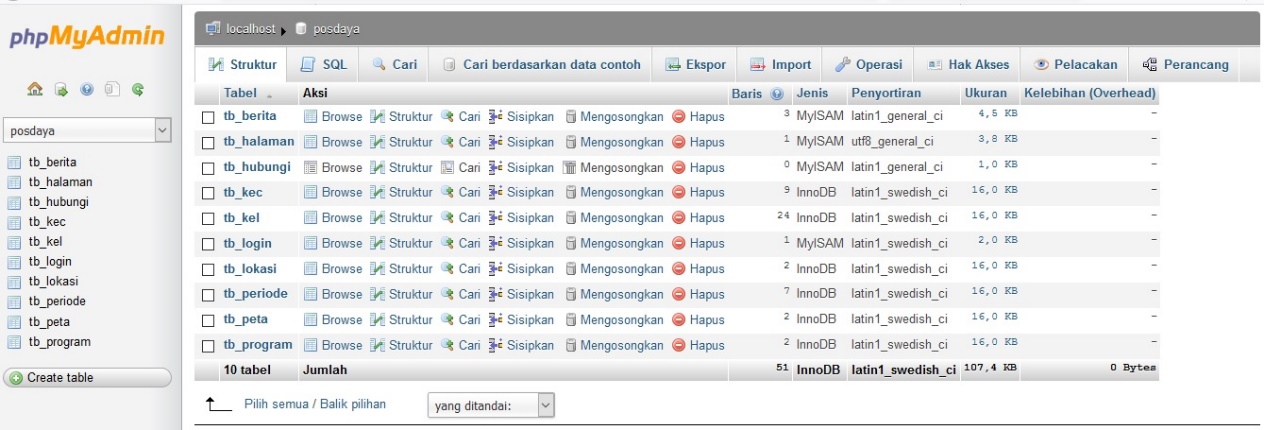
1. **Pengujian dekripsi basis data**



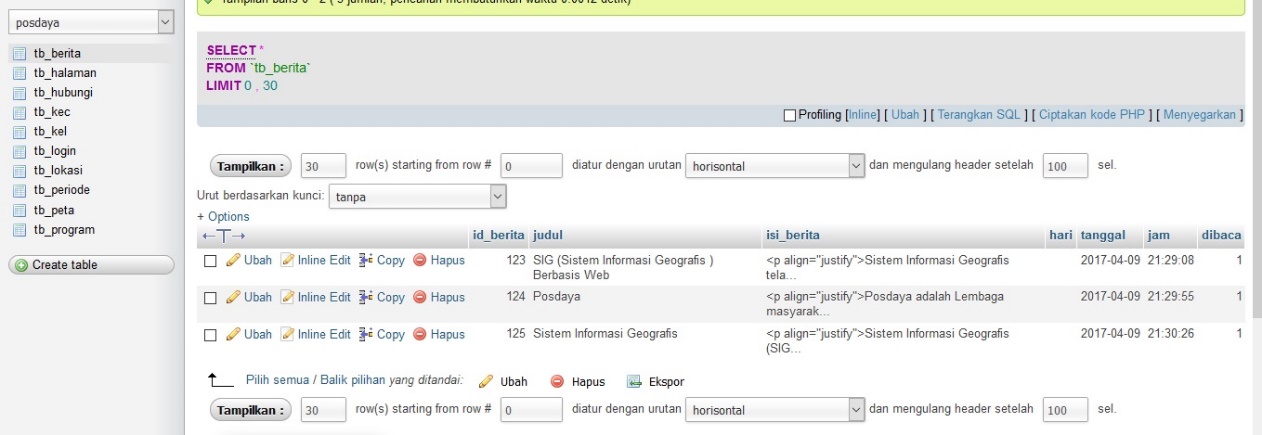
**Gambar 5.7** Cipherteks sebelum didekripsi



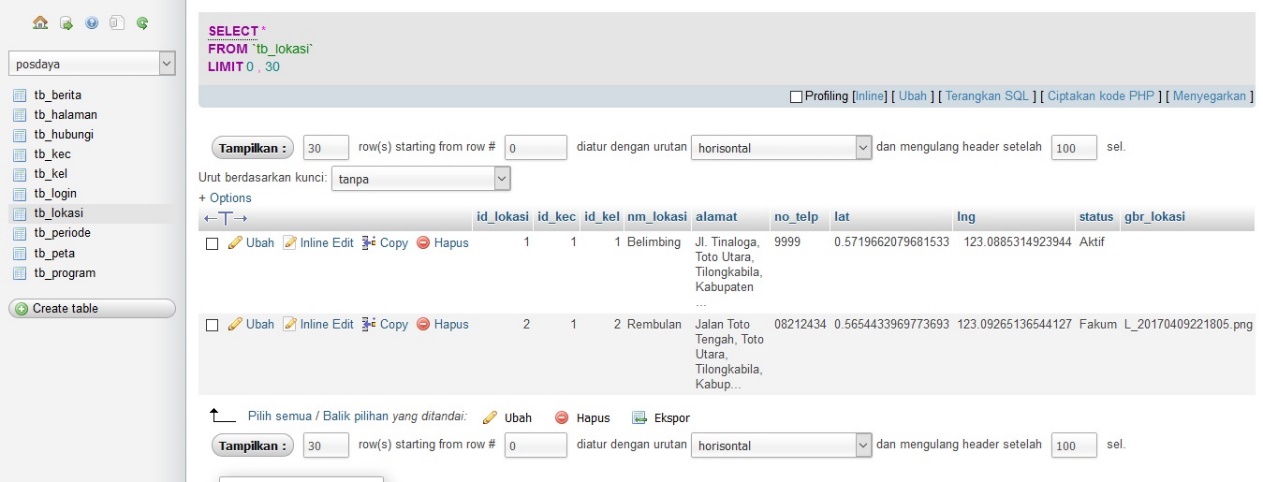
**Gambar 5.8** Cipherteks setelah didekripsi



**Gambar 5.9** Tampilan basis data setelah di dekripsi



**Gambar 5.10** Tabel berita setelah di dekripsi



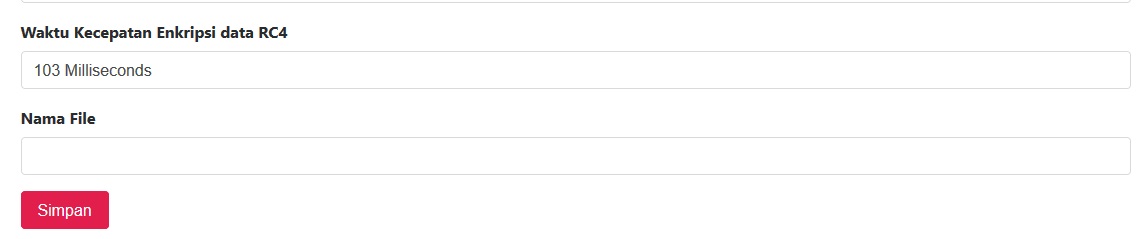
**Gambar 5.11** Tabel lokasi setelah di dekripsi

Dari gambar-gambar diatas dapat dilihat bahwa setelah melalui proses enkripsi dan kemudian di dekripsi kembali basis data tetap dapat digunakan kembali. Tidak ada perubahan isi pada basis data, dan juga tidak terdapat kerusakan data.

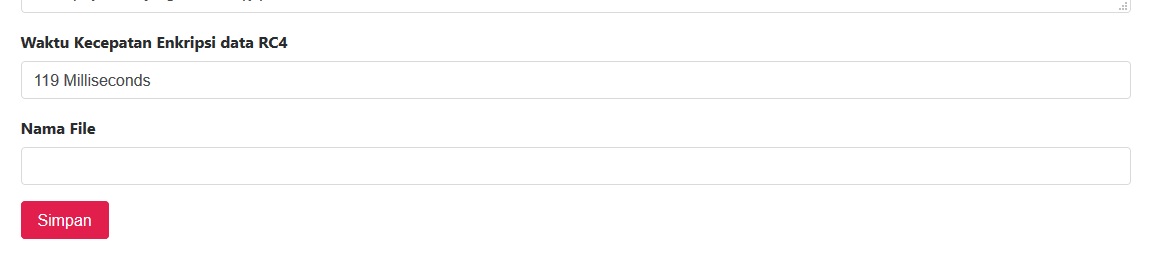
Dari hasil pengujian ini dapat disimpulkan bahwa sistem ini berhasil mengenkripsi dan dekripsi basis data.

1. **Pengujian Performance**

Pada tahap ini pengujian dilakukan untuk mengetahui penggunaan resource yang digunakan untuk menjalankan sistem pengamanan basis data. Akan dibandingkan antara RC4 128 *byte* dan RC4 256 *byte* dengan cara mencari selisih waktu antara awal proses enkripsi dan akhir proses enkripsi. Berikut merupakan hasil pengukuran pada sistem pengamanan basis data.



**Gambar 5.12** Performance RC4 128 *byte*



**Gambar 5.13** Performance RC4 256 *byte*

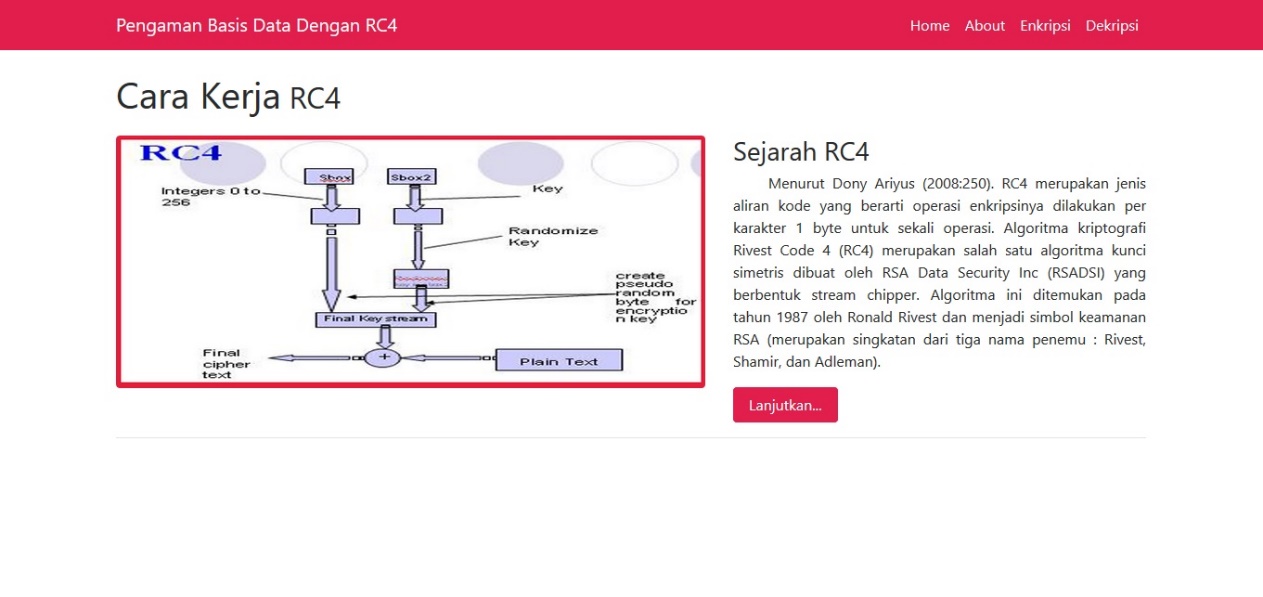
Berdasarkan hasil pengujian performance yang telah dilakukan seperti pada gambar 5.3, untuk melakukan proses enkripsi RC4 128 *byte* memakan waktu sekitar 103 *milliseconds*. Sedangkan hasil pengujian performa yang dilakukan seperti pada gambar 5.4, untuk melakukan proses enkripsi RC4 256 *byte* memakan waktu sekitar 119 *milliseconds*.

**Gambar 5.14** Grafik perbandingan RC4 128 *byte* dan RC4 256 *byte*

Dari hasil ini dapat disimpulkan bahwa pengaruh panjang kunci yang digunakan saat enkripsi algoritma RC4 berpengaruh pada waktu enkripsi, namun tidak begitu signifikan. Dengan panjang kunci 128 *byte* ataupun 256 *byte* algoritma RC4 masih tergolong cepat dalam mengeksekusi data.

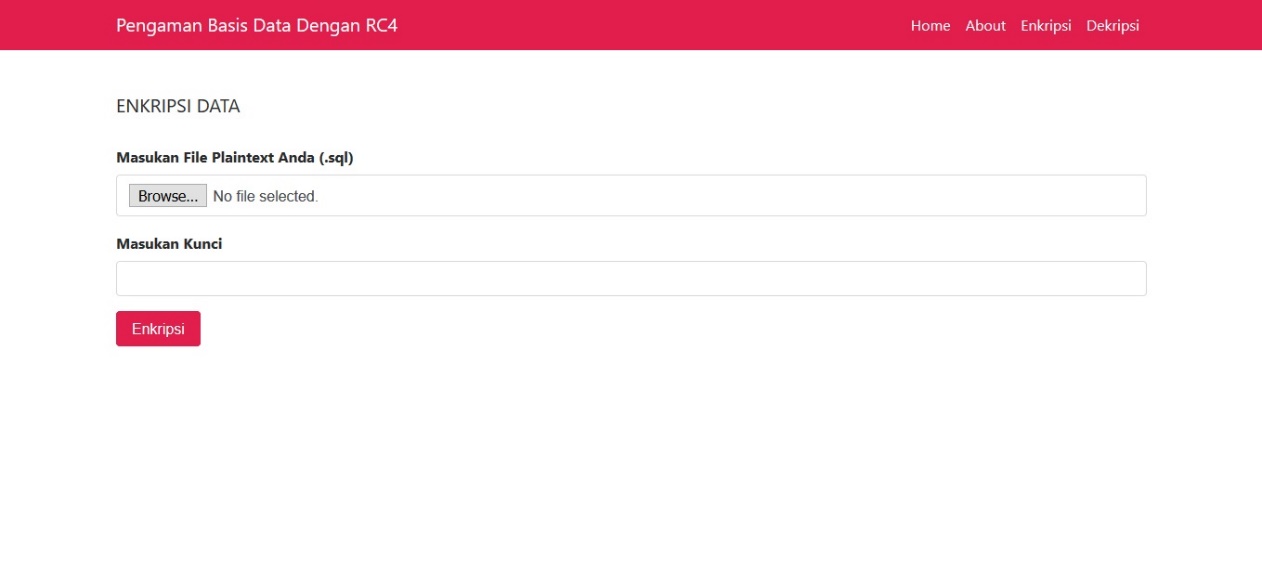
1. **Langkah-Langkah Penggunaan Sistem**

Untuk menjalankan program cukup mengetikkan alamat website alamat website pada tab *address.*



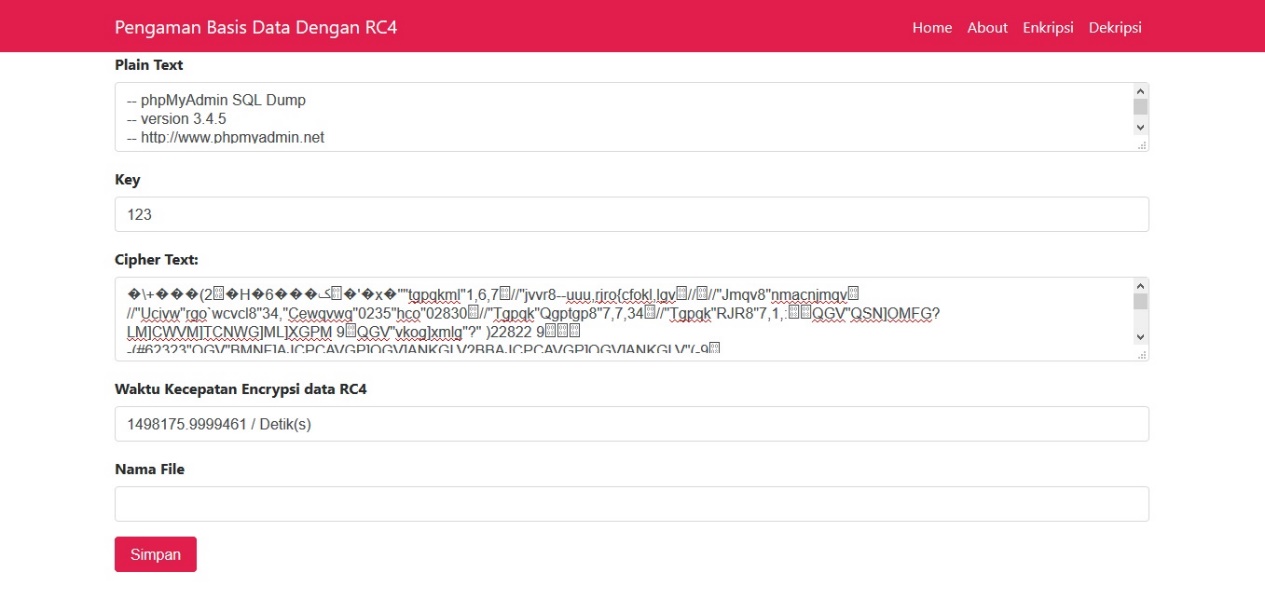
**Gambar 5.15** Tampilan Halaman Menu Utama

Halaman Menu Utama pertama kali tampil setelah memasukan alamat website. Halaman utama ini terdiri atas menu Home, About, Enkripsi dan Dekripsi.



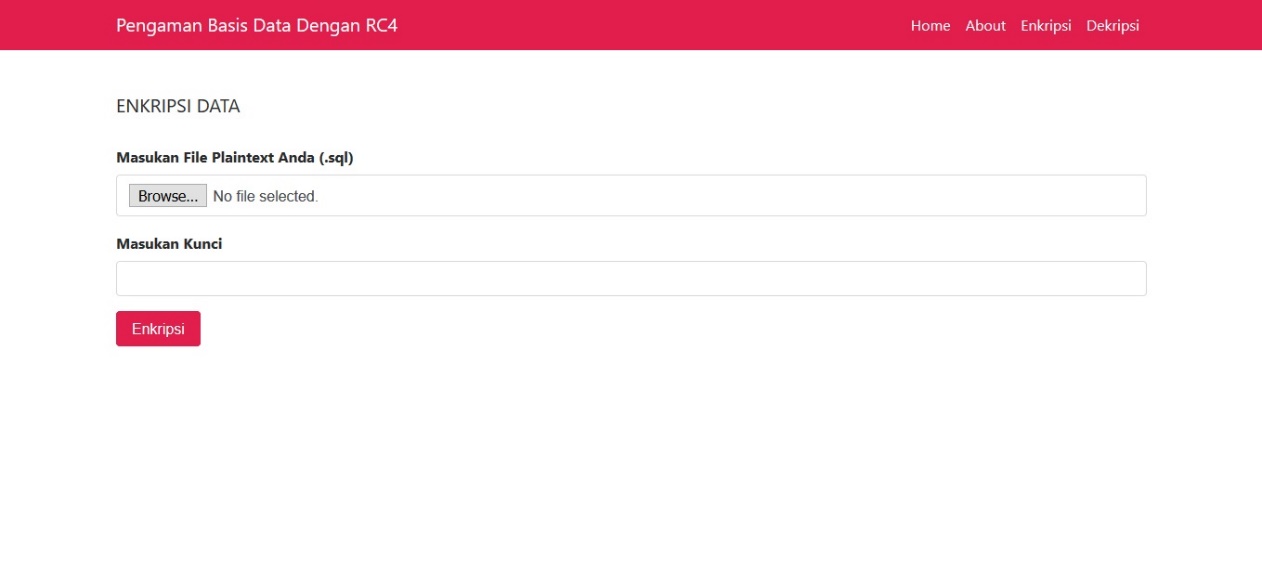
**Gambar 5.16** Tampilan Halaman Input Enkripsi

Halaman input enkripsi akan ditampilkan jika user memilih menu enkripsi yang terdapat pada menu utama. Di halaman ini user memilih plainteks basis data (.sql) dan memasukan kunci enkripsi.



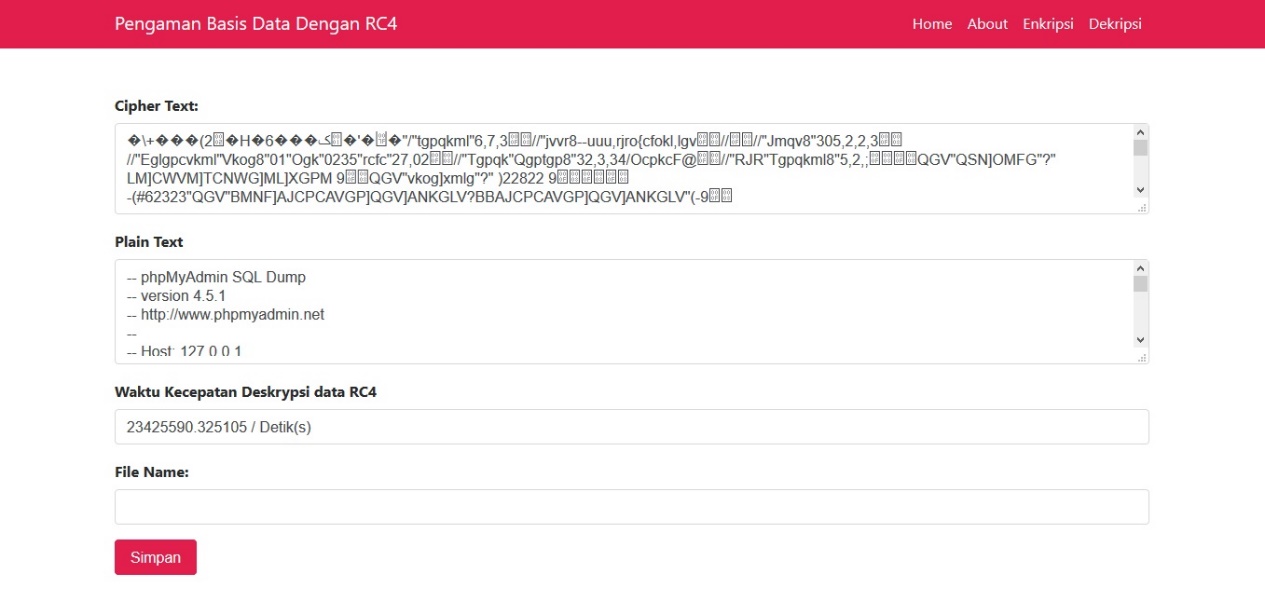
**Gambar 5.17** Tampilan Halaman Output Enkripsi

Halaman output enkripsi akan di tampilkan jika user memilih tombol enkripsi pada halaman input enkripsi. Halaman ini akan menampilkan output dari proses enkripsi basis data yaitu plainteks, cipherteks, kunci enkripsi dan waktu enkripsi. Terdapat pula teks box untuk mengisi nama file yang akan disimpan dan tombol simpan untuk menyimpan hasil enkripsi basis data.



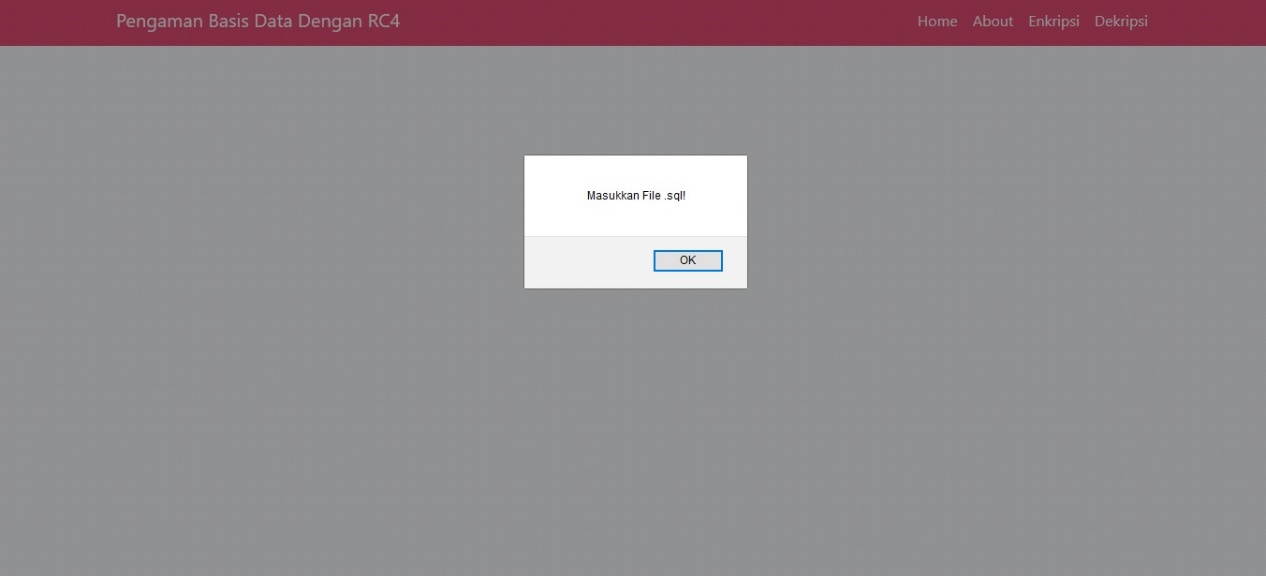
**Gambar 5.18** Tampilan Halaman Input Dekripsi

Halaman input dekripsi akan ditampilkan jika user memilih menu dekripsi yang terdapat pada menu utama. Di halaman ini user memilih cipherteks basis data (.sql) dan memasukan kunci dekripsi untuk selanjutnya di dekripsi.



**Gambar 5.19** Tampilan Halaman Output Dekripsi

Halaman output dekripsi akan di tampilkan jika user memilih tombol dekripsi pada halaman input dekripsi. Halaman ini akan menampilkan output dari proses dekripsi basis data yaitu cipherteks, plainteks, kunci dekripsi dan waktu dekripsi. Terdapat pula teks box untuk mengisi nama file yang akan disimpan dan tombol simpan untuk menyimpan hasil enkripsi basis data.



**Gambar 5.20** Tampilan Pesan Error File

Pesan ini akan muncul hanya jika user menginput file selain yang berekstensi (.sql)