BAB IV

IMPLEMENTASI DAN EVALUASI

4.1 Implementasi Sistem

Implementasi merupakan realisasi dari sebuah aplikasi. Pada tahap ini akan dijelaskan mengenai cari penggunaan aplikasi dan bagaimana aplikasi dijalankan.

4.1.1 *Splash Screen*

Gambar 4.1 Tampilan *Splash Screen*

Splash Screen merupakan tampilan pembuka ketika aplikasi pertama kali dijalankan. Tampilan *Splash Screen* dapat dilihat pada Gambar 4.1. Tampilan ini

hanya tampil sekitar tiga detik, pada tampilan ini terdapat logo dan nama dari aplikasi.

4.1.2 Halaman Awal

Halaman ini merupakan tampilan awal dari aplikasi. Pada tampilan ini terdapat ayat yang ditampilkan secara random dan menu navigasi pada bagian bawah. Menu navigasi terdiri dari halaman home, halaman pencarian, halaman petunjuk, dan halaman info. Tampilan halaman awal dapat dilihat pada Gambar 4.2 dan menu navigasi pada tabel 4.1.

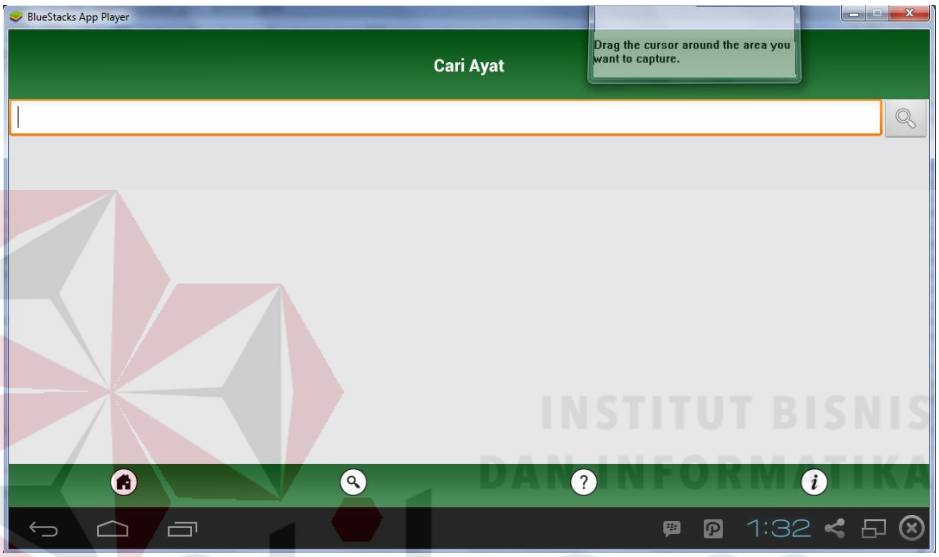
Gambar 4.2 Tampilan Halaman Awal

Tabel 4.1 Daftar Menu Navigasi

No	Menu	Icon	No	Menu	Icon
1	Home.		**3**	Help	
2	Pencarian.		**4**	Info	

4.1.3 Halaman Pencarian Ayat

Halaman ini merupakan halaman pencarian ayat al-Qur'an dimana *user* dapat menginputkan kata kunci tentang ayat yang ingin dicari. Pada halaman ini terdapat satu *textbox*, *button*,dan *list* untuk menampilkan hasil pencarian ayat al-Qur'an. Tampilan halaman pencarian dapat dilihat pada Gambar 4.3.

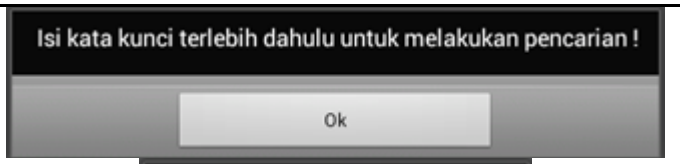
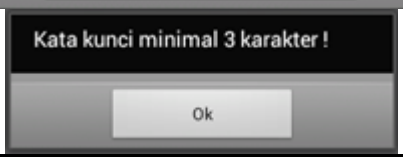


Gambar 4.3 Tampilan Halaman Pencarian

Mula-mula user akan menginputkan kata kunci tentang ayat yang ingin dicari, kunci berupa kata dalam bahasa Indonesia, lalu menekan *button* cari kemudian sistem akan melakukan pencarian dengan menggunakan algoritma Knuth-Morris-Pratt dan menampilkan hasil pencarian ayat berupa *list*. Pada proses pencarian ini aplikasi dapat melakukan pencarian dengan kata sinonim, contoh "shalat", "solat".

Ada beberapa pesan kesalahan pada halaman ini jika beberapa kondisi terpenuhi, yaitu jika textbox kosong dan jika karakter inputan kurang dari tiga karakter. Untuk lebih jelas dapat dilihat pada tabel 4.2

Tabel 4.2 Pesan Kesalahan Pada Pencarian

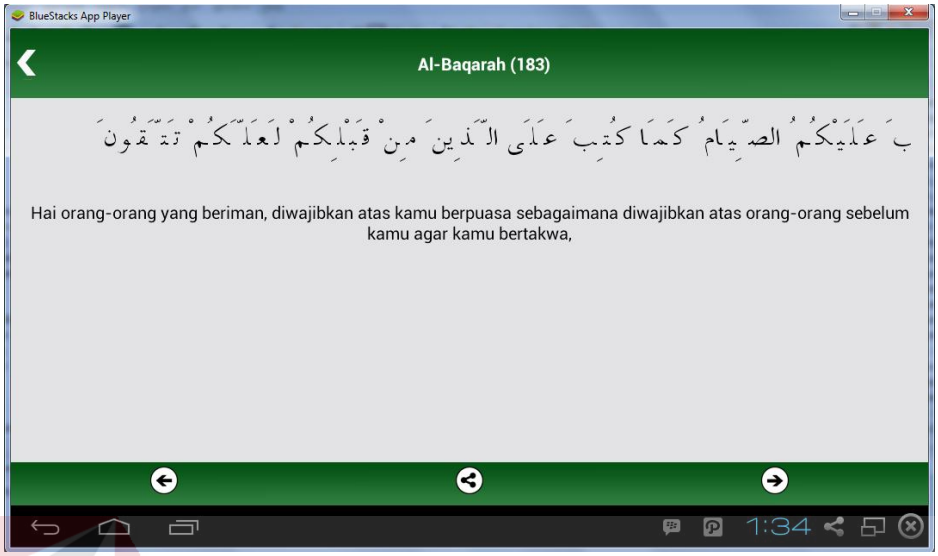
No	Kondisi	Pesan Kesalahan
1	Textboxt kosong.	
2	Karakter kurang dari tiga.	

Sebagai contoh, pada Gambar 4.4 kata kunci yang digunakan yaitu "puasa", ketika menekan *button* cari maka sistem akan melakukan pencarian ayat Al-Qur'an yang berkaitan tentang puasa dan menampilkannya dalam *list*.

Gambar 4.4 Tampilan Hasil Pencarian

Pada Gambar 4.4 merupakan hasil pencarian dari kata kunci "*puasa*", hasil pencarian ini ditampilkan adalah nama surat dan ayat dari surat tersebut dan ditampilkan dalam bentuk *list* yang dapat dipilih untuk melihat detail dari ayat tersebut.

4.1.4 Halaman Detail Ayat



Gambar 4.5 Tampilan Halaman Detail Ayat

Halaman ini berisi detail ayat dari hasil pencarian pada halaman sebelumnya. Pada halaman ini akan ditampilkan ayat Al-Qur'an dan terjemahan dari hasil pencarian yang telah dilakukan. Pada halaman ini terdapat navigasi *next* dan *prev* yang berfungsi untuk melihat ayat sebelum dan sesudah sesuai dengan urutan list hasil pencarian. Dan pada halaman ini juga fitur *share* untuk mambagi ayat ke berbagai media sosial. Tampilan halaman detail ayat dapat dilihat pada Gambar 4.5 dan navigasi pada tabel 4.3.

Tabel 4.3 Daftar Navigasi Ayat

No	Menu	Icon	No	Menu	Icon
1	Kembali ke hasil pencarian.		**3**	Melihat ayat selanjutnya.	
2	Melihat ayat sebelumnya.		**4**	Share.	

4.2 Pengujian Sistem

Setelah melakukan impelementasi sistem, tahap selanjutnya adalah melakukan uji coba terhadap sistem. Pengujian bertujuan untuk mengetahui apakah aplikasi yang dibangun telah benar dan sesuai dengan tujuan yang diharapkan. Pengujian dilakukan pada semua desain pengujian yang telah dilakukan sebelumnya.

4.2.1 Unit Testing

A. White Box Testing

Tujuan dari pengujian ini yaitu untuk mengetahui secara internal cara kerja suatu perangkat lunak. Pada pengujian ini fungsi yang diuji yaitu proses mencari padanan kata, proses menghitung *border function* dan *matching* pada algoritma Knuth-Morris-Pratt dengan cara menghitung *Cyclomatic Complexity* dari ketiga proses.

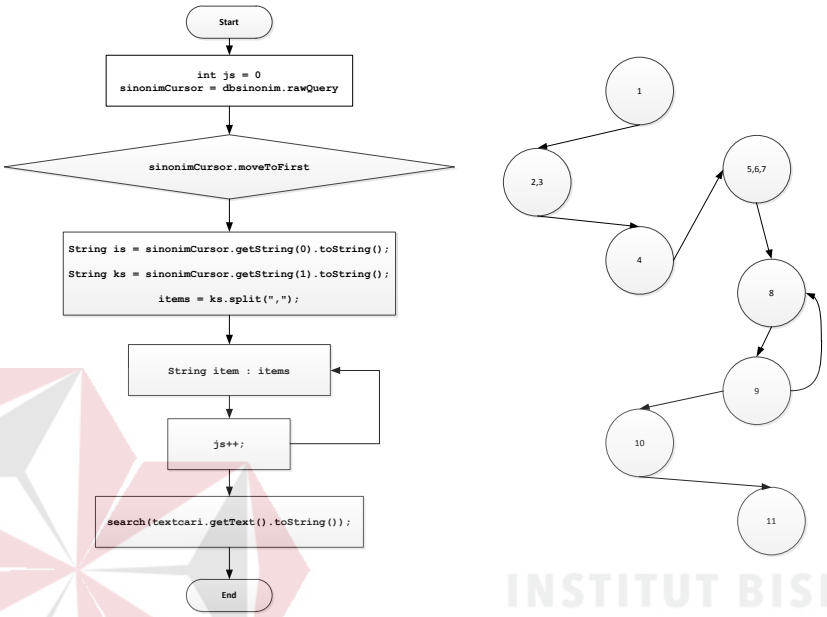
1. Proses Mencari Padanan Kata

Proses ini merupakan proses pertama dalam melakukan pencarian yang berfungsi untuk mencari kata yang memiliki arti yang sama dengan kata kunci. Kata kunci yang diinputkan oleh user akan dicari padanan kata yang ada pada database. Berikut adalah *source code* dari proses mencari padanan kata.

```
1       begin
2       int js = 0;
3       sinonimCursor = dbsinonim.rawQuery("SELECT id_sinonim,
        kata_sinonim FROM sinonim where  kata_sinonim like
        ('%"+textcari.getText().toString()+"%') limit 1", null);
4       if (sinonimCursor.moveToFirst()) {
5       String is = sinonimCursor.getString(0).toString();
6       String ks = sinonimCursor.getString(1).toString();
7       items = ks.split(",");
8       for (String item : items){
9       js++; }
10      search(textcari.getText().toString());
11      end;
```

Gambar 4.6 *Source Code* Mencari Padanan Kata

Dari *source code* pada Gambar 4.6 akan dibuatkan *flowchart* dan *flowgraph* untuk menentukan *cyclomatic complexity*. Berikut adalah *flowchart* dan *flowgraph* pada proses mencari padanan kata.



Gambar 4.7 *Flowchart* dan *Flowgraph* Mencari Padanan Kata

Berdasarkan *flowgraph* pada gambar 4.7 tahap selanjutnya yaitu menghitung *cyclomatic complexity* untuk menentukan jalur independen pada proses mencari padanan kata. *Cyclomatic complexity* pada proses mencari padanan kata adalah sebagai berikut :

V(G) = E (edge) – N (node) + 2

V(G) = 8 – 8 + 2 = 2

V(G) = P (predicted node) + 1

V(G) = 1 + 1 = 2

Dari hasil *cyclomatic complexity* diatas maka akan diketahui jumlah jalur independen pada proses mencari padanan kata adalah 2. Berdasarkan alurnya, maka didapatkan jalur independen sebagai berikut :

a. Jalur 1 : 1-2-3-4-5-6-7-8-9-10-11

b. Jalur 2 : 1-2-3-4-5-6-7-8-9-8-9-10-11

Gambar 4.8 merupakan *Graph matrix* yang menggambarkan jalur independen dari proses mencari padanan kata.

	1	2,3	4	5,6,7	8	9	10	11	
1		1							1-1=0
2,3			1						1-1=0
4				1					1-1=0
5,6,7					1				1-1=0
8						1			1-1=0
9					1		1		2-1=1
10								1	1-1=0
11									

Cyclomatic Complexity = 1+1 = 2

Gambar 4.8 *Graph Matrix* Mencari Padanan Kata

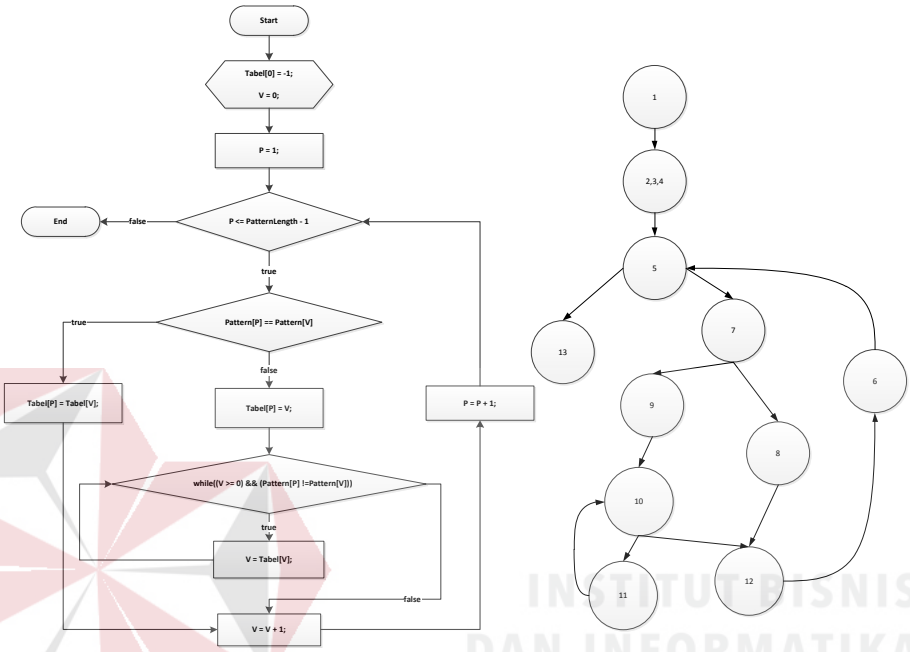
2. Proses Menghitung *Border Function*

Proses merupakan *preprocession* dari algoritma Knuth-Morris-Pratt. Proses ini berfungsi untuk menghitung nilai pinggiran dari kata kunci yang diinputkan oleh user. Berikut adalah *source code* dari proses menghitung *border function*.

```
1       begin
2       failure[0] = -1;
3       int i = 0;
4       for (int j = 1;
5       j <= pattern.length() - 1;
6       j++) {
7       if (pattern.charAt(j)== pattern.charAt(i))
8       failure[j] = failure[i];
9       else failure[j] = i;
10      while ((i >= 0) && (pattern.charAt(j)!= pattern.charAt(i)))
11      i = failure[i];
12      i++;
13      end;
```

Gambar 4.9 *Source Code* Menghitung *Border Function*

Dari *source code* pada Gambar 4.9 akan dibuatkan *flowchart* dan *flowgraph* untuk menentukan *cyclomatic complexity*. Berikut adalah *flowchart* dan *flowgraph* pada proses menghitung *border function*.



Gambar 4.10 *Flowchart* dan *Flowgraph* Menghitung *Border Function*

Berdasarkan *flowgraph* pada gambar 4.10 *Cyclomatic complexity* pada proses menghitung *border function* adalah sebagai berikut :

V(G) = E (edge) – N (node) + 2

V(G) = 13 – 11 + 2 = 4

V(G) = P (predicted node) + 1

V(G) = 3 + 1 = 4

Dari hasil *cyclomatic complexity* diatas maka akan diketahui jumlah jalur independen pada proses menghitung *border function* adalah 4. Berdasarkan alurnya, maka didapatkan jalur independen sebagai berikut :

a. Jalur 1 : 1-2-3-4-5-7-8-12-6-5-13

b. Jalur 2 : 1-2-3-4-5-7-9-10-12-6-5-13

c. Jalur 3 : 1-2-3-4-5-7-9-10-11-10-12-6-5-13

d. Jalur 4 : 1-2-3-4-5-13

Gambar 4.11 merupakan *Graph matrix* yang menggambarkan jalur independen dari proses mencari padanan kata.

	1	2,3,4	5	6	7	8	9	10	11	12	13	
1		1										1-1=0
2,3,4			1									1-1=0
5					1						1	2-1=1
6			1									1-1=0
7						1	1					2-1=1
8										1		1-1=0
9								1				1-1=0
10									1	1		2-1=1
11								1				1-1=0
12				1								1-1=0
13												

Cyclomatic Complexity = 3+1 = 4

Gambar 4.11 *Graph Matrix* Menghitung *Border Function*

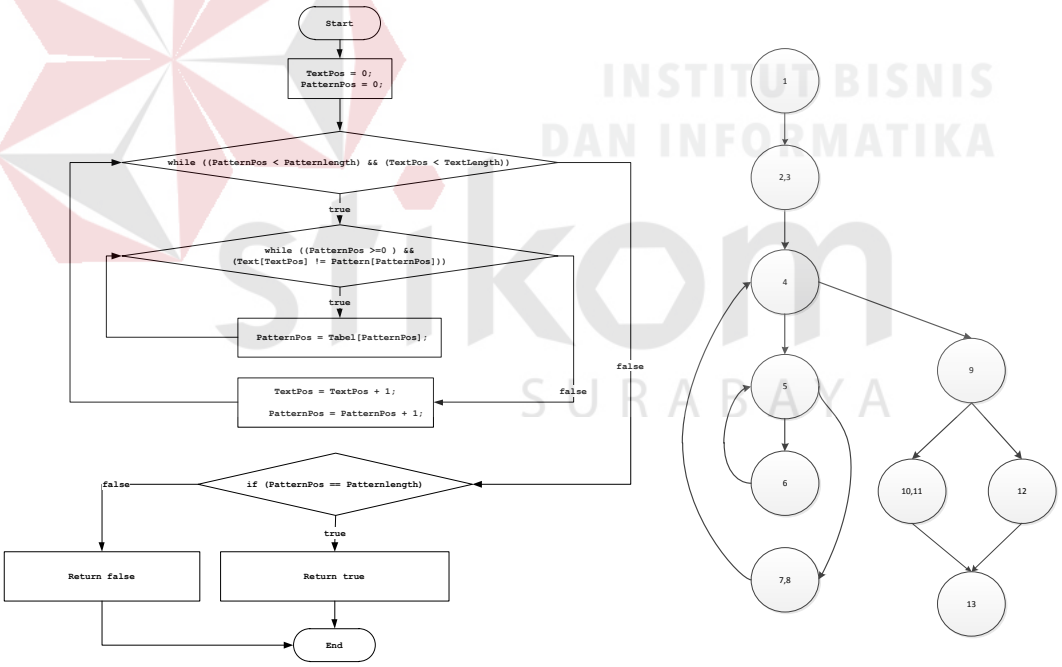
3. Proses *Matching*

Proses ini merupakan proses utama dari algoritma Knuth-Morris-Pratt. Proses akan melakukan pencocokan kata kunci atau pattern yang diinputkan oleh user dengan terjemahan Al-Qur'an yang ada pada *database*. Berikut adalah *source code* dari proses *matching* ditunjukan pada Gambar 4.12.

```
1       begin
2       int j = 0;
3       int i = 0;
4       while ((j < pattern.length()) && (i < string.length())) {
5       while ((j >=0 ) && (string.charAt(i)!= pattern.charAt(j)))
6       j = failure[j];
7       i++;
8       j++; }
9       if (j == pattern.length()) {
10      matchPoint = i - pattern.length();
11      return true; }
12      return false;
13      end;
```

Gambar 4.12 *Source Code Matching*

Dari *source code* pada Gambar 4.12 akan dibuatkan *flowchart* dan *flowgraph* untuk menentukan *cyclomatic complexity*. Berikut adalah *flowchart* dan *flowgraph* pada proses *matching*.



Gambar 4.13 *Flowchart* dan *Flowgraph Matching*

Berdasarkan *flowgraph* pada gambar 4.13 *Cyclomatic complexity* pada proses *matching* adalah sebagai berikut :

V(G) = E (edge) – N (node) + 2

V(G) = 12 – 10 + 2 = 4

V(G) = P (predicted node) + 1

V(G) = 3 + 1 = 4

Dari hasil *cyclomatic complexity* diatas maka akan diketahui jumlah jalur independen pada proses *matching* adalah 4. Berdasarkan alurnya, maka didapatkan jalur independen sebagai berikut :

a. Jalur 1 : 1-2-3-4-5-7-8-4-9-10-11-13

b. Jalur 2 : 1-2-3-4-5-7-8-4-9-12-13

c. Jalur 3 : 1-2-3-4-5-6-5-7-8-4-9-10-11-13

d. Jalur 4 : 1-2-3-4-5-6-5-7-8-4-9-12-13

Gambar 4.14 merupakan *Graph matrix* yang menggambarkan jalur independen dari proses *matching*.

	1	2,3	4	5	6	7,8	9	10,11	12	13	
1		1									1-1=0
2,3			1								1-1=0
4				1			1				2-1=1
5					1	1					2-1=1
6				1							1-1=0
7,8			1								1-1=0
9								1	1		2-1=1
10,11										1	1-1=0
12										1	1-1=0
13											

Cyclomatic Complexity = 3+1 = 4

Gambar 4.14 *Graph Matrix Matching*

Dari pengujian *white box* yang telah dilakukan pada tiga proses diatas, dapat disimpulkan bahwa tidak terdapat kesalahan pada logika dari ketiga proses tersebut. Selain itu nilai *cyclomatic complexity* dari ketiga proses tersebut kurang dari 10, dengan kata lain kinerja sistem tidak akan menurun.

B. Black Box Testing

Pada tahap ini bertujuan untuk mengetahui apakah fungsi utama dari aplikasi pencarian ayat Al-Qur'an ini menghasilkan output yang diharapkan. Proses utama dari aplikasi ini yaitu adalah proses pencarian ayat Al-Qur'an menggunakan metode algoritma *string matching* Knuth-Morris-Pratt.

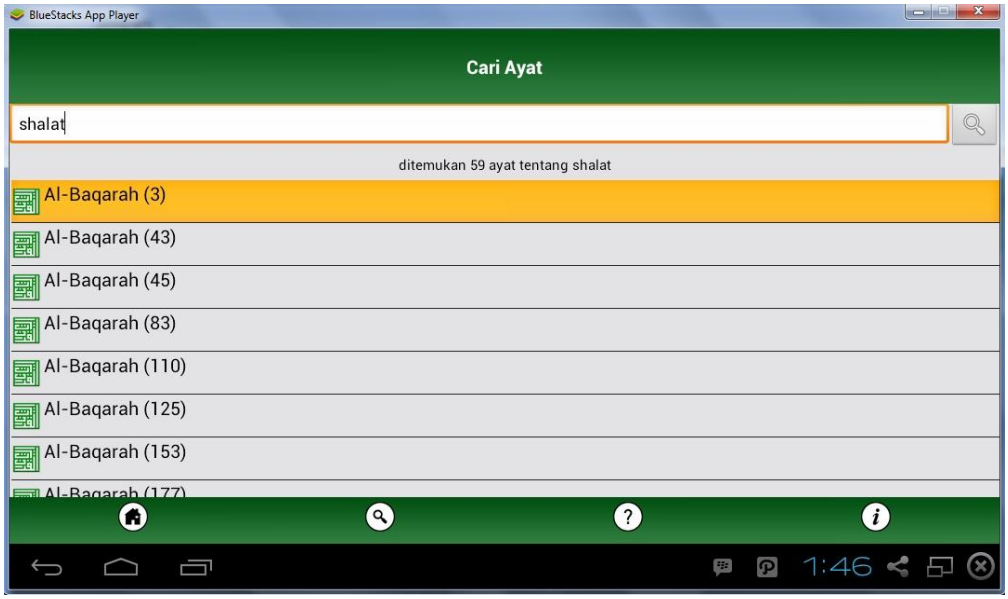
1. Uji Coba Pencarian

Proses pencarian dilakukan dengan cara menginputkan kata kunci atau *keyword* tentang ayat yang ingin dicari kemudian sistem akan menampilkan ayat yang sesuai dengan kata kunci. Hasil pengujian disajikan dalam bentuk tabel dan gambar hasil keluaran sistem untuk masing-masing *test case*.

Tabel 4.4 Hasil Uji Coba Test Case 1

No *Test Case*	**Tujuan**	**Input**	**Output Yang Diharapkan**	**Hasil Keluaran Sistem**
1	Pencarian ayat al-Qur'an yang memiliki hasil.	Memasukan kata kunci "shalat".	Hasil pencarian yang sesuai dengan kata kunci.	Sesuai

Pada Tabel 4.4 merupakan hasil uji coba *test case* 1, dimana hasil keluaran sistem sesuai dengan output yang diharapkan. Pembuktian test case 1 dapat dilihat pada Gambar 4.15.

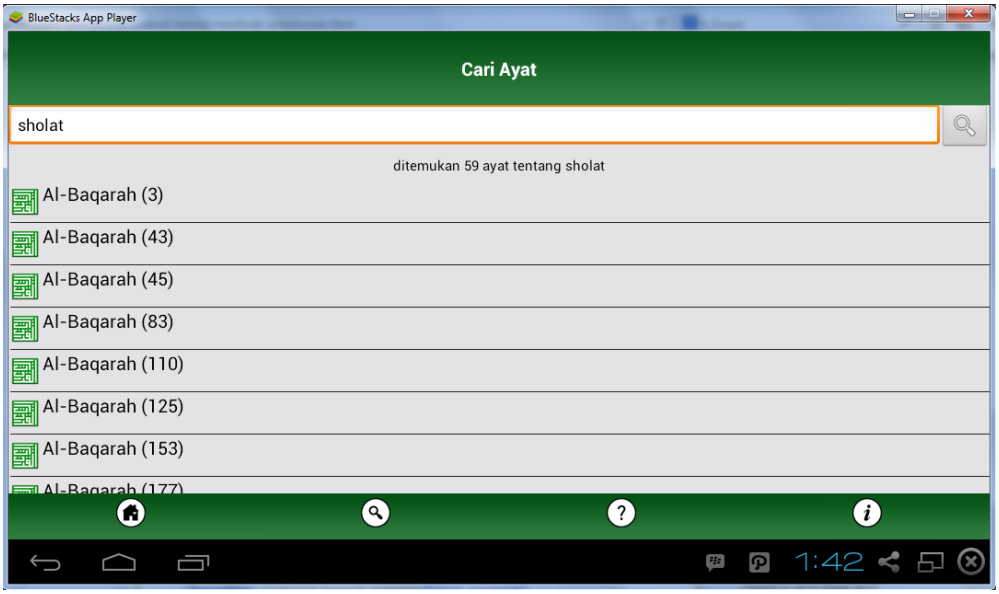


Gambar 4.15 Pembuktian *Test Case* 1

Tabel 4.5 Hasil Uji Coba *Test Case* 2

No *Test Case*	Tujuan	Input	Output Yang Diharapkan	Hasil Keluaran Sistem
2	Pencarian ayat al-Qur'an dengan kata yang mirip dan memiliki hasil.	Memasukan kata kunci "sholat" dan "salat"	Hasil pencarian yang sesuai dengan kata kunci.	Sesuai

Pada Tabel 4.5 merupakan hasil uji coba *test case* 2, dimana hasil keluaran sistem sesuai dengan output yang diharapkan. Kata sholat dan salat sama-sama memiliki keluaran 59 ayat. Pembuktian *test case* 2 dapat dilihat pada Gambar 4.16 dan Gambar 4.17.



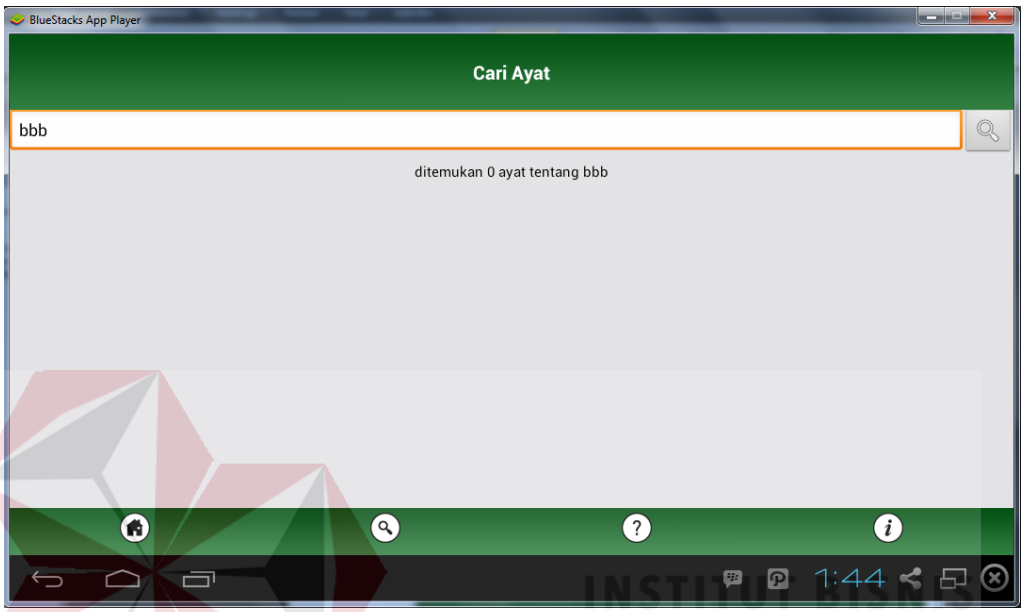
Gambar 4.16 Pembuktian *Test Case* 2 Dengan Kata Kunci Sholat

Gambar 4.17 Pembuktian *Test Case* 2 Dengan Kata Kunci Salat

Tabel 4.6 Hasil Uji Coba *Test Case* 3

No *Test Case*	**Tujuan**	**Input**	**Output Yang Diharapkan**	**Hasil Keluaran Sistem**
3	Pencarian ayat al-Qur'an yang tidak memiliki hasil.	Memasukan kata kunci "bbb".	Pencarian tidak menemukan hasil.	Sesuai

Pada Tabel 4.6 merupakan hasil uji coba *test case* 3, dimana hasil keluaran sistem sesuai dengan output yang diharapkan. Pembuktian test case 2 dapat dilihat pada Gambar 4.18.

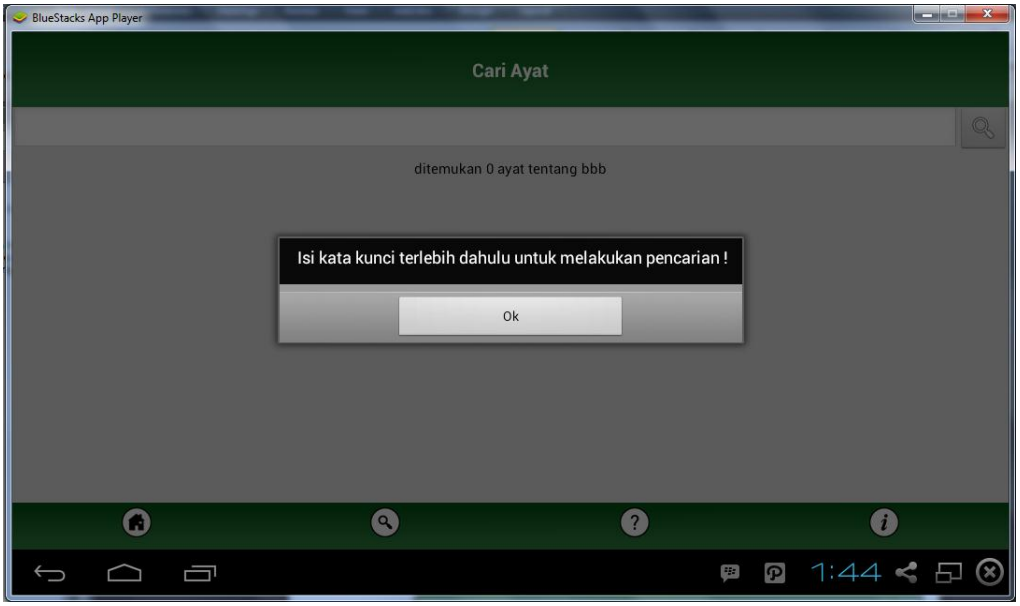


Gambar 4.18 Pembuktian *Test Case* 3

Tabel 4.7 Hasil Uji Coba *Test Case* 4

No *Test Case*	Tujuan	Input	Output Yang Diharapkan	Hasil Keluaran Sistem
4	Menampilkan pesan jika *textbox* kosong.	Menekan tombol cari tanpa mengisi *textbox*.	Aplikasi menampilkan pesan *textbox* kosong	Sesuai

Pada Tabel 4.7 merupakan hasil uji coba *test case* 4, dimana hasil keluaran sistem sesuai dengan output yang diharapkan. Pembuktian *test case* 4 dapat dilihat pada Gambar 4.19.

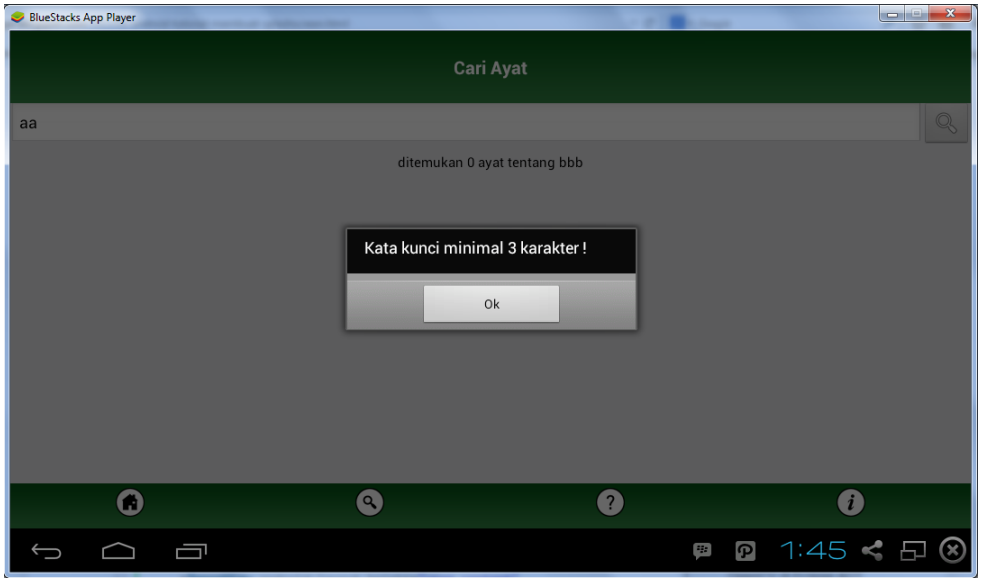


Gambar 4.19 Pembuktian *Test Case* 4

Tabel 4.8 Hasil Uji Coba *Test Case* 5

No *Test Case*	Tujuan	Input	Output Yang Diharapkan	Hasil Keluaran Sistem
5	Menampilkan pesan jika karakter yang diinput kurang dari tiga.	Memasukan kata kunci “aa”.	Aplikasi menampilkan pesan jumlah karakter yang diinput minimal tiga.	Sesuai

Pada Tabel 4.8 merupakan hasil uji coba *test case* 5, dimana hasil keluaran sistem sesuai dengan output yang diharapkan. Pembuktian *test case* 5 dapat dilihat pada Gambar 4.20.



Gambar 4.20 Pembuktian *Test Case* 5

2. Uji Coba Detail Ayat

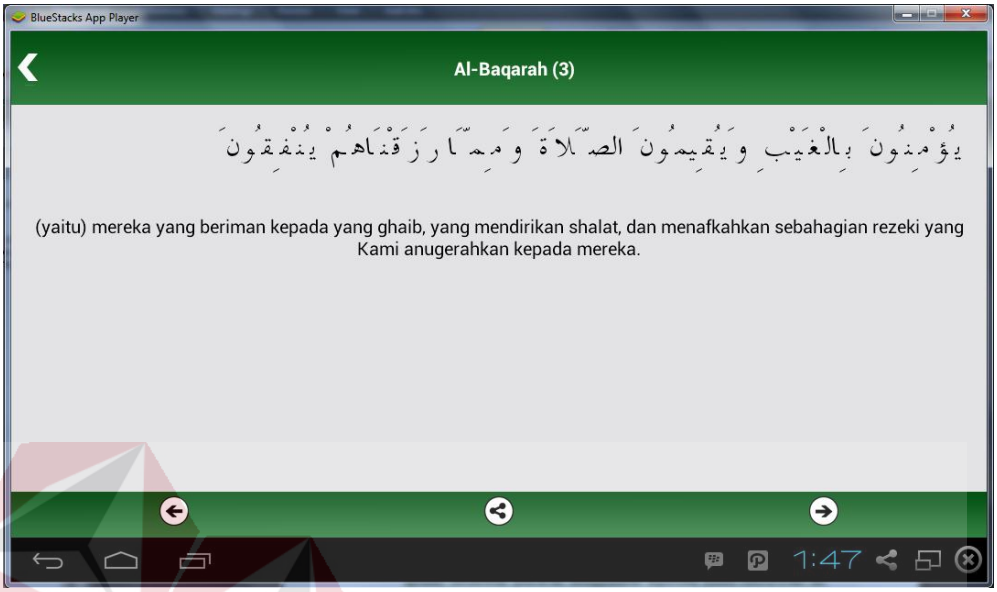
Halaman ini berfungsi untuk menampilkan detail ayat berupa ayat dan terjemahan. Pada proses ini *user* memilih salah satu ayat dari hasil pencarian dan sistem akan menampilkan ayat dan terjemahan yang sesuai dengan ayat yang dipilih oleh *user*. Hasil pengujian disajikan dalam bentuk tabel dan gambar hasil keluaran sistem untuk masing-masing *test case*.

Tabel 4.9 Hasil Uji Coba *Test Case* 6

No *Test Case*	**Tujuan**	**Input**	**Output Yang Diharapkan**	**Hasil Keluaran Sistem**
6	Menampilkan ayat dan terjemahan yang sesuai dengan pilihan *user*.	Pilih salah satu ayat dari hasil pencarian.	Aplikasi menampilkan ayat dan terjemahan yang sesuai dengan pilihan *user*.	Sesuai

Pada Tabel 4.9 merupakan hasil uji coba *test case* 6, dimana hasil keluaran sistem sesuai dengan output yang diharapkan. Pencarian menggunakan kata kunci

shalat kemudian memilih surat al-baqarah ayat 3. Pembuktian *test case* 6 dapat dilihat pada Gambar 4.21.

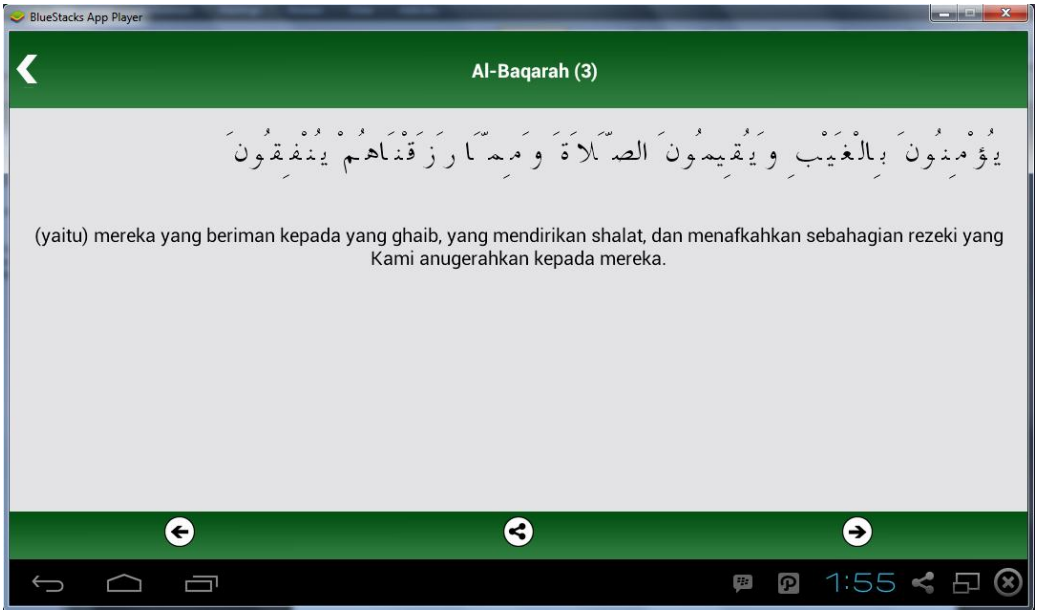


Gambar 4.21 Pembuktian *Test Case* 6

Tabel 4.10 Hasil Uji Coba *Test Case* 7

No *Tes Case*	Tujuan	Input	Output Yang Diharapkan	Hasil Keluaran Sistem
7	Menampilkan ayat dan terjemahan sebelumnya sesuai *list* hasil pencarian.	Pilih *button prev*	Aplikasi menampilkan ayat sebelumnya seseuai *list* hasil pencarian.	Sesuai

Pada Tabel 4.10 merupakan hasil uji coba *test case* 7, dimana hasil keluaran sistem sesuai dengan output yang diharapkan. Pencarian menggunakan kata kunci shalat kemudian memilih surat al-baqarah ayat 43 sesuai hasil pencarian pada Gambar 4.15, ayat sebelumnya yaitu surat al-baqarah ayat 3. Pembuktian *test case* 7 dapat dilihat pada Gambar 4.22.

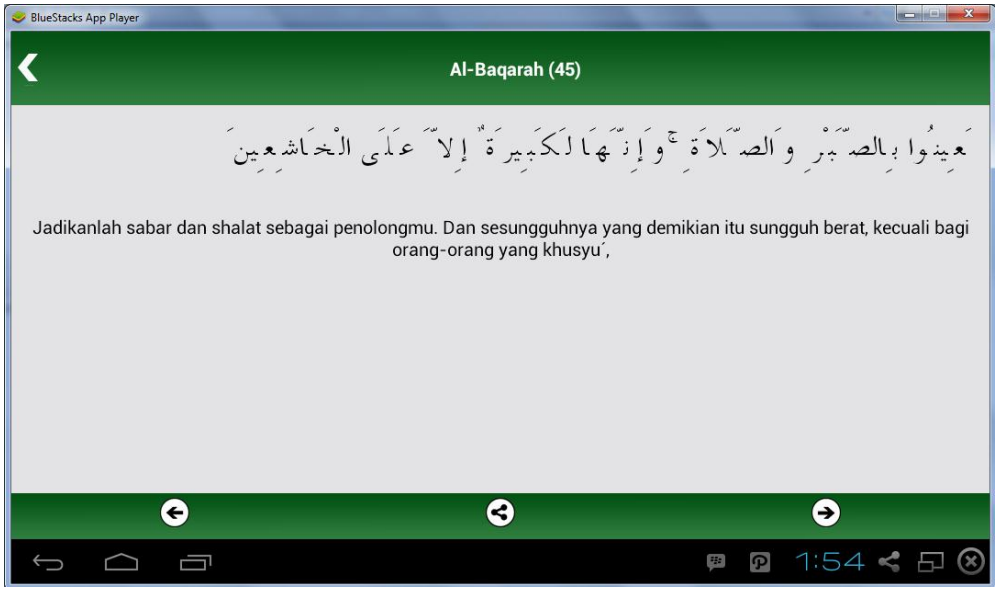


Gambar 4.22 Pembuktian *Test Case* 7

Tabel 4.11 Hasil Uji Coba *Test Case* 8

No *Tes Case*	Tujuan	Input	Output Yang Diharapkan	Hasil Keluaran Sistem
8	Menampilkan ayat dan terjemahan selanjutnya sesuai *list* hasil pencarian.	Pilih *button next*	Aplikasi menampilkan ayat selanjutnya seseuai *list* hasil pencarian.	Sesuai

Pada Tabel 4.11 merupakan hasil uji coba *test case* 8, dimana hasil keluaran sistem sesuai dengan output yang diharapkan. Pencarian menggunakan kata kunci shalat kemudian memilih surat al-baqarah ayat 43 sesuai hasil pencarian pada Gambar 4.22, ayat selanjutnya yaitu surat al-baqarah ayat 45. Pembuktian *test case* 8 dapat dilihat pada Gambar 4.23.

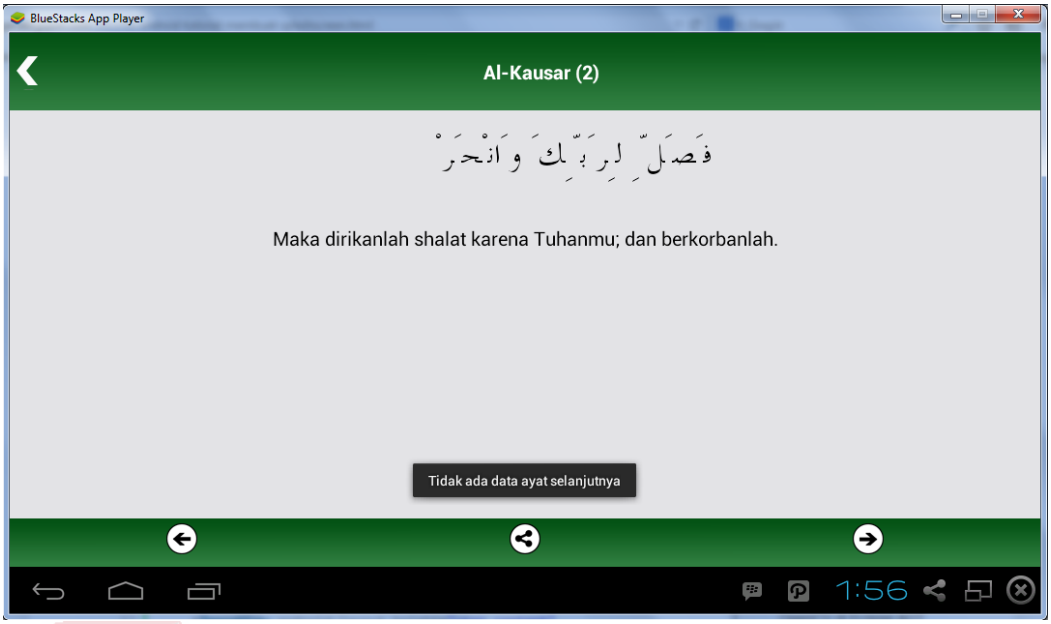


Gambar 4.23 Pembuktian *Test Case* 8

Tabel 4.12 Hasil Uji Coba *Test Case* 9

No Tes Case	Tujuan	Input	Output Yang Diharapkan	Hasil Keluaran Sistem
9	Menampilkan pesan jika tidak ada ayat selanjutnya.	Pilih *button next* dengan kondisi tidak ada ayat selanjutnya.	Aplikasi menampilkan pesan tidak ada ayat selanjutnya.	Sesuai

Pada Tabel 4.12 merupakan hasil uji coba *test case* 9, dimana hasil keluaran sistem sesuai dengan output yang diharapkan. Pencarian menggunakan kata kunci shalat kemudian memilih surat al-baqarah ayat 43 sesuai hasil pencarian, ayat terakhir hasil pencarian adalah surat al-kausar ayat 2, dan tidak ada ayat selanjutnya. Pembuktian *test case* 9 dapat dilihat pada Gambar 4.24.

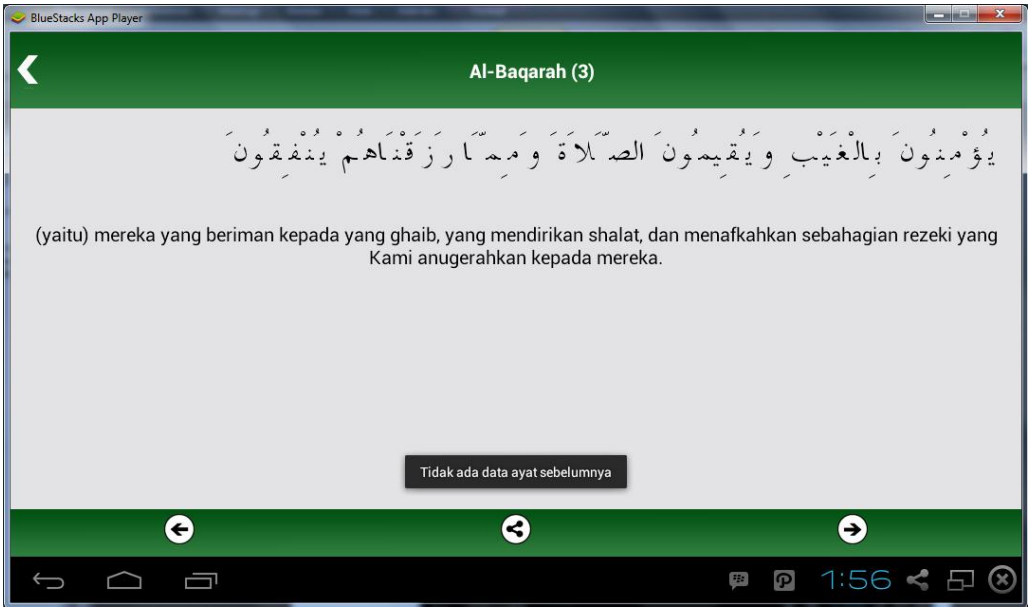


Gambar 4.24 Pembuktian *Test Case* 9

Tabel 4.13 Hasil Uji Coba *Test Case* 10

No Tes Case	Tujuan	Input	Output Yang Diharapkan	Hasil Keluaran Sistem
10	Menampilkan pesan jika tidak ada ayat sebelumnya	Pilih *button next* dengan kondisi tidak ada ayat sebelumnya.	Aplikasi menampilkan pesan tidak ada ayat sebelumnya.	Sesuai

Pada Tabel 4.13 merupakan hasil uji coba *test case* 10, dimana hasil keluaran sistem sesuai dengan output yang diharapkan. Pencarian menggunakan kata kunci shalat kemudian memilih surat al-baqarah ayat 43 sesuai hasil pencarian, ayat pertama dari hasil pencarian adalah surat al-baqarah ayat 3, dan tidak ada ayat sebelumnya. Pembuktian *test case* 10 dapat dilihat pada Gambar 4.25.



Gambar 4.25 Pembuktian *Test Case* 10

4.2.2 System Testing

A. Compability Testing

Pada uji coba kompabilitas ini bertujuan untuk mengetahui apakah aplikasi pencarian ayat al-Qur'an ini dapat berjalan dengan baik pada berbagai macam versi android. Versi android yang akan digunakan untuk uji coba ini yaitu versi 2.3 (*Gingerbread*) sampai dengan 4.4 (*KitKat*).

Tabel 4.14 Daftar Versi Android

No	Versi Android
1	Android 2.3 (*Gingerbread*)
2	Android 4.0 (*Ice Cream Sandwich*)
3	Android 4.1 (*Jelly Bean*)
4	Android 4.4 (*KitKat*)

Uji coba dilakukan menggunakan beberapa *emulator* dan *smartphone* android versi 4.4 (*KitKat*) dengan menguji fungsi-fungsi dan fitur pada aplikasi

pencarian ayat Al-Qur'an. Hasil pengujian pada pengujian ini dapat dilihat pada tabel 4.15 berikut.

Tabel 4.15 Hasil Uji Kompatibilitas

No.	Fungsi dan Fitur yang diuji	Versi Android			
		2.3	4.0	4.1	4.4
1.	Mencari ayat	OK	OK	OK	OK
2.	Melihat detail ayat Al-Qur'an	OK	OK	OK	OK
3.	Fitur ayat selanjutnya	OK	OK	OK	OK
4.	Fitur ayat sebelumnya	OK	OK	OK	OK

B. Performance Testing

Uji coba algoritma bertujuan untuk mengetahui kinerja dari aplikasi dalam melakukan pencarian menggunakan algoritma Knuth-Morris-Pratt. Pada uji coba ini akan diberikan 20 kata kunci yang telah ditetapkan kemudian akan hitung waktu lama proses pencarian dan (ketepatan) *precision* dari masing-masing kata kunci. Hasil uji coba kinerja aplikasi dapat dilihat pada tabel 4.18.

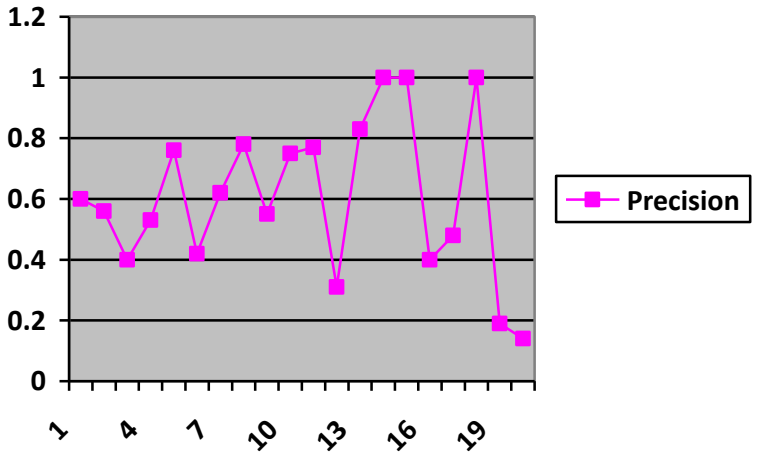
Tabel 4.18 Hasil Uji Kinerja Aplikasi

No	Kata Kunci	A	B	C	D	E
1	Qisas	10	7	1	0.60	1.70
2	Shalat	91	66	15	0.56	2.13
3	Wasiat	10	5	1	0.40	0.95
4	Masjidil Haram	15	9	1	0.53	1.42
5	Puasa	13	10	0	0.76	0.95
6	Thalaq	26	11	2	0.42	1.69
7	Iddah	8	8	3	0.62	0.98
8	Utang Piutang	14	11	0	0.78	1.98

Tabel 4.18 Hasil Uji Kinerja Aplikasi (Lanjutan)

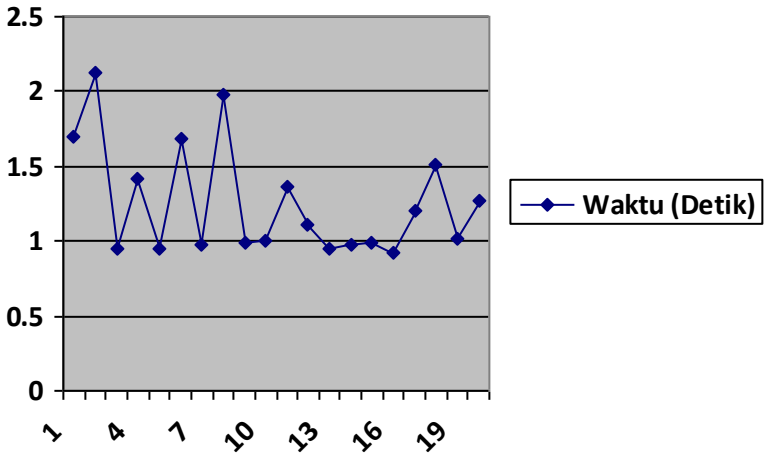
No	Kata Kunci	A	B	C	D	E
9	Putus Asa	18	11	1	0.55	0.99
10	Judi	4	3	0	0.75	1.00
11	Mahar	9	7	2	0.77	1.37
12	Riba	19	6	0	0.31	1.11
13	Nadzar	6	5	0	0.83	0.95
14	Zabur	4	4	0	1.00	0.97
15	Kiblat	5	7	2	1.00	0.99
16	Riya	10	4	0	0.40	0.92
17	Nikah	29	25	11	0.48	1.20
18	Sedekah	12	12	0	1.00	1.51
19	Halal	31	7	1	0.19	1.01
20	Haram	63	10	1	0.14	1.27
	Rata-rata				**0.60**	**1.25**

Dimana A = ∑ ayat diterima, B = ∑ ayat relevan, C = ∑ ayat hilang, D = nilai *precision*, dan E = waktu pencarian (detik). Ayat diterima yaitu ayat dari hasil pencarian aplikasi, ayat relevan yaitu ayat yang ada pada index Al-Qur'an dan ayat hilang yaitu ayat yang ada pada index Al-Qur'an tetapi tidak ada pada hasil pencarian aplikasi.



Gambar 4.26 Grafik Pengujian *Recall* dan *Precision*

Untuk menghitung precision menggunakan persamaan 2.1. Gambar 4.26 menjelaskan grafik nilai *precision*. Dari hasil uji nilai precision, hanya terdapat 3 kata kunci yang memiliki nilai precision 1, 14 kata kunci dengan nilai precision antara 0.4 – 0.8, dan 3 kata kunci dengan precision dibawah 0.4, yaitu kata "riba", "halal", dan "haram". Ketiga kata kunci tersebut memiliki nilai presicion rendah dikarenakan jumlah ayat dari hasil pencarian aplikasi sangat banyak dibandingkan dengan jumlah ayat relevan yang ada pada index Al-Qur'an, walaupun ayat yang dihasilkan oleh aplikasi dapat mencakup semua ayat-ayat pada index Al-Qur'an. Dari hasil uji coba ini, aplikasi ini hanya memiliki nilai rata-rata precision yaitu 0,60 atau sekitar 60%. Hal ini dikarenakan ada beberapa ayat pada kata kunci yang merupakan ayat lanjutan dari ayat sebelumnya. Selain itu juga ada beberapa ayat yang memiliki topik yang sama dengan kata kunci tapi tidak terdapat kata kunci pada terjemahan ayat tersebut.



Gambar 4.27 Grafik Pengujian Kecepatan Algoritma

Pada gambar 4.27 merupakan grafik hasil pengujian kecepatan pencarian pada 20 kata kunci. Sumbu *y* menyatakan waktu pencarian dalam detik, dan sumbu *x* menyatakan urutan kata kunci. Dari hasil uji coba diatas hanya terdapat dua kata kunci yang dengan waktu eksekusi cukup lama yaitu kata "shalat" dan "utang piutang". Pencarian kata "shalat" membutuhkan waktu 2.13 detik, hal ini dikarenakan terdapat banyak ayat yang mengandung kata "shalat", selain itu kata "shalat" juga memiliki padanan kata yang cukup banyak sehingga dibutuhkan waktu yang sedikit lama untuk mencari ayat tersebut, begitu juga pada kata kunci "utang piutang" yang memiliki padanan kata yang banyak. Dari hasil uji coba kecepatan pencarian ayat yang dilakukan dengan 20 kata kunci algoritma Knuth-Morris-Pratt memiliki rata-rata pencarian yaitu 1,25 detik.

4.3 Evaluasi

Dari proses uji coba sebelumnya maka, hasil dari uji coba di rangkum dalam tahap evaluasi ini. Dari uji coba fungsi aplikasi, dapat disimpulkan bahwa aplikasi pencarian ayat Al-Quran menggunakan metode algoritma *string matching*

Knuth-Morris-Pratt ini dapat menghasilkan ayat yang sesuai dengan kata kunci yang diinputkan oleh *user* sesuai dengan yang diharapkan berdasarkan uji coba yang telah dilakukan. Tetapi pada aplikasi ini tidak terdapat fitur *auto complete* dan pembenaran kata karena hanya menangani inputan dari user untuk melakukan pencarian ayat Al-Quran inputan dari user. Selain itu aplikasi ini hanya sebatas menampilkan ayat dan terjemahan Al-Quran, tidak terdapat tafsir dan intisari ayat yang dapat menjelaskan kandungan dari ayat tersebut.

Dari segi kinerja aplikasi, rata-rata ketepatan algoritma Knuth-Morris-Pratt dalam melakukan pencarian ayat sebesar 60% dan rata-rata tingkat kecepatan algoritma Knuth-Morris-Pratt dalam melakukan pencarian ayat adalah 1,25 detik dari uji coba yang telah dilakukan. Untuk itu dapat dilakukan perbandingan dari beberapa algoritma pencarian lainnya untuk mengetahui tingkat efektiffitas dari beberapa algoritma pencarian.