BAB IV

IMPLEMENTASI DAN PENGUJIAN SISTEM

Pada bab ini akan dilakukan implementasi dan pengujian terhadap system yang baru. Pada tahap selanjutnya akan diimplementasikan pada bahasa pemrograman yang akan digunakan. Setelah tahap implementasi selesai dilakukan maka dilakukan pengujian sistem yang baru.

4.1 Implementasi Basis Data

Pembuatan basis data dilakukan dengan mengunakan DBMS (*Database Management System*) MySQL. Implementasi basis data dalam bahasa SQL dapat dilihat pada tabel 4.1 di bawah ini.

Tabel 4.1 Implementasi Basis Data dalam Bahasa SQL

Tabel T_DB	Tabel T_LISTDB
CREATE TABLE `t_db` (`cdb` int(10) unsigned NOT NULL AUTO_INCREMENT, `jenisdb` varchar(20) NOT NULL, PRIMARY KEY (`cdb`)) ENGINE=InnoDB AUTO_INCREMENT=5 DEFAULT CHARSET=latin1 CHECKSUM=1 DELAY_KEY_WRITE=1 ROW_FORMAT=DYNAMIC;	CREATE TABLE `t_listdb` (`cdb` int(10) unsigned NOT NULL, `clistdb` int(10) unsigned NOT NULL AUTO_INCREMENT, `namadb` varchar(50) NOT NULL, `aliasdb` varchar(50) NOT NULL, `ipaddressdb` varchar(20) NOT NULL, `portdb` int(11) NOT NULL, `userdb` varchar(50) NOT NULL, `passdb` varchar(50) NOT NULL, PRIMARY KEY (`clistdb`), KEY `Idx_cdb` (`cdb`), CONSTRAINT `FK_tlistdb_to_tdb` FOREIGN KEY (`cdb`) REFERENCES `t_db` (`cdb`) ON DELETE NO ACTION ON UPDATE CASCADE) ENGINE=InnoDB DEFAULT CHARSET=latin1 CHECKSUM=1 DELAY_KEY_WRITE=1 ROW_FORMAT=DYNAMIC;

Tabel T_LISTKOLOM	Tabel T_LISTQUERY
CREATE TABLE `t_listkolom` (`ctable` int(10) unsigned NOT NULL, `ckolom` int(10) unsigned NOT NULL AUTO_INCREMENT, `namakolom` varchar(50) NOT NULL, `aliaskolom` varchar(50) NOT NULL, `tipedata` varchar(50) NOT NULL, `ukuran` int(10) unsigned DEFAULT NULL, PRIMARY KEY (`ckolom`), KEY `FK_tlistkolom_to_listtable` (`ctable`), CONSTRAINT `FK_tlistkolom_to_listtable` FOREIGN KEY (`ctable`) REFERENCES `t_listtable` (`ctable`) ON DELETE NO ACTION ON UPDATE CASCADE) ENGINE=InnoDB DEFAULT CHARSET=latin1 CHECKSUM=1 DELAY_KEY_WRITE=1 ROW_FORMAT=DYNAMIC;	CREATE TABLE `t_listquery` (`cquery` int(10) unsigned NOT NULL, `clistdb` int(10) unsigned NOT NULL, `clistquery` int(10) unsigned NOT NULL auto_increment, `urutan` int(10) unsigned default NULL, `sqlquery` varchar(500) default NULL, `operand` varchar(50) default NULL, `jumkolom` int(10) unsigned default NULL, `jumrow` int(10) unsigned default NULL, PRIMARY KEY (`clistquery`), KEY `FK_tlistquery_to_transquery` (`cquery`), KEY `FK_t_listquery_to_listdb` (`clistdb`), CONSTRAINT `FK_tlistquery_to_transquery` FOREIGN KEY (`cquery`) REFERENCES `t_transquery` (`cquery`) ON DELETE NO ACTION ON UPDATE CASCADE, CONSTRAINT `FK_t_listquery_to_listdb` FOREIGN KEY (`clistdb`) REFERENCES `t_listdb` (`clistdb`) ON DELETE NO ACTION ON UPDATE CASCADE) ENGINE=InnoDB AUTO_INCREMENT=540 DEFAULT CHARSET=latin1 CHECKSUM=1 DELAY_KEY_WRITE=1 ROW_FORMAT=DYNAMIC;
Table T_RESULTSET	**Table T_TRANSQUERY**
CREATE TABLE `t_resultset` (`cquery` int(10) unsigned NOT NULL, `cresultset` int(10) unsigned NOT NULL auto_increment, `namakolom` varchar(500) default NULL,	CREATE TABLE `t_transquery` (`cquery` int(10) unsigned NOT NULL auto_increment, `starttime` datetime NOT NULL, `endtime` datetime default NULL, `status` varchar(50) default NULL, `sqlquery` varchar(500) NOT NULL,

`isikolom` varchar(500) default NULL, PRIMARY KEY (`cresultset`), KEY `FK_t_resultset` (`cquery`), CONSTRAINT `FK_t_resultset` FOREIGN KEY (`cquery`) REFERENCES `t_transquery` (`cquery`) ON DELETE NO ACTION ON UPDATE CASCADE) ENGINE=InnoDB AUTO_INCREMENT=61 DEFAULT CHARSET=latin1 CHECKSUM=1 DELAY_KEY_WRITE=1 ROW_FORMAT=DYNAMIC;	`jeniserror` varchar(500) default NULL, `posisierror` varchar(500) default NULL, `deskripsierror` varchar(500) default NULL, PRIMARY KEY (`cquery`)) ENGINE=InnoDB AUTO_INCREMENT=573 DEFAULT CHARSET=latin1 CHECKSUM=1 DELAY_KEY_WRITE=1 ROW_FORMAT=DYNAMIC;

4.2 Implementasi Sistem

Implementasi merupakan kelanjutan dari kegiatan perancangan sistem dan dapat dipandang sebagai usaha untuk mewujudkan sistem yang dirancang. Langkah-langkah dari proses implementasi adalah urutan dari kegiatan awal sampai kegiatan akhir yang harus dilakukan dalam mewujudkan sistem yang dirancang. Hasil dari tahapan implementasi ini adalah suatu sistem pengolahan data yang sudah dapat berjalan dengan baik.

4.2.1 Langkah-langkah Penerapan Sistem

Langkah-langkah yang perlu dilakukan dalam penerapan sistem ini adalah sebagai berikut :

1. Pembentukan program aplikasi
2. Memperbaiki dan menguji program aplikasi
3. Mempersiapkan perangkat keras dan perangkat lunak
4. Pembentukan data parameter awal
5. Pendidikan dan latihan sumber daya manusia
6. Ujicoba, evaluasi sistem, dan pengalihan sistem

4.2.1.1. Pembentukan Program Aplikasi

Kegiatan pembentukan program terdiri dari pembuatan program dalam bentuk modul-modul program. Tujuan dari pembentukan modul-modul program yaitu untuk membagi aplikasi menjadi beberapa bagian sesuai dengan fungsi masing-masing yang sudah didefinisikan dalam tahap peracangan.

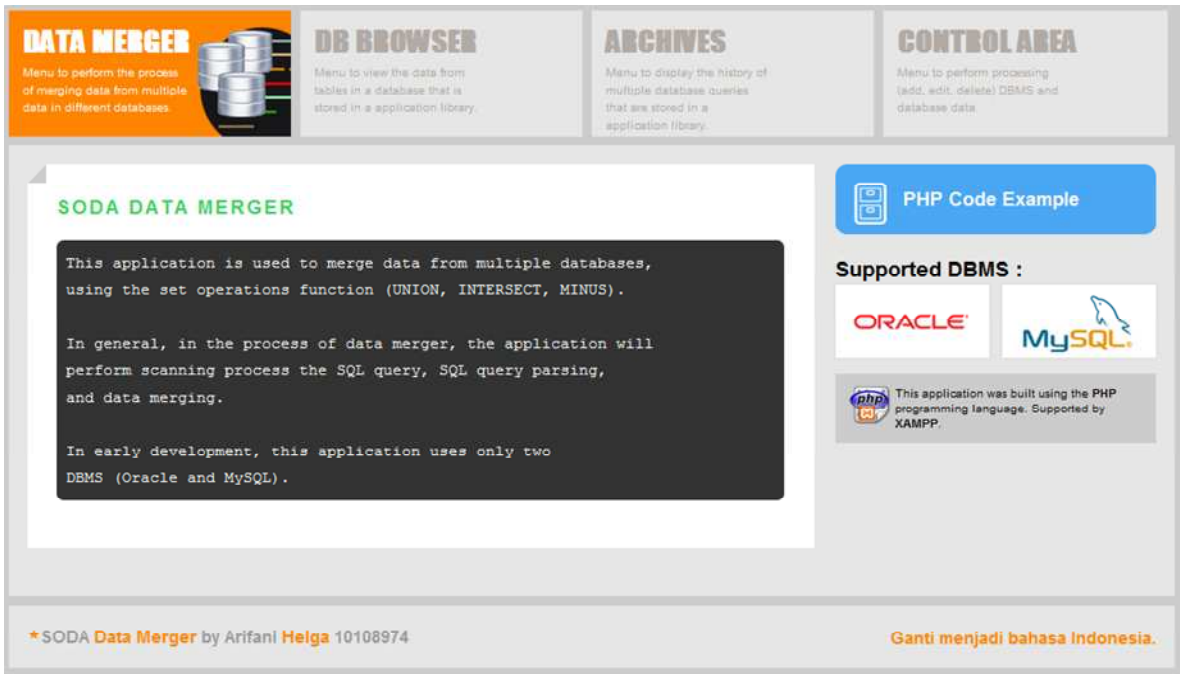
Tabel 4.2 Menu yang dirancang

NO	Menu	Keterangan
1.	HOME	**Tampilan Utama Aplikasi**
2.	DATA MERGER	**Melakukan proses penggabungan data**
3	ARCHIVES	**Menampilkan daftar history proses penggabungan data yang pernah dilakukan sebelumnya.**
4	CONTROL AREA	**Menu untuk memasukkan data *database* yang akan dipakai dalam aplikasi ini**

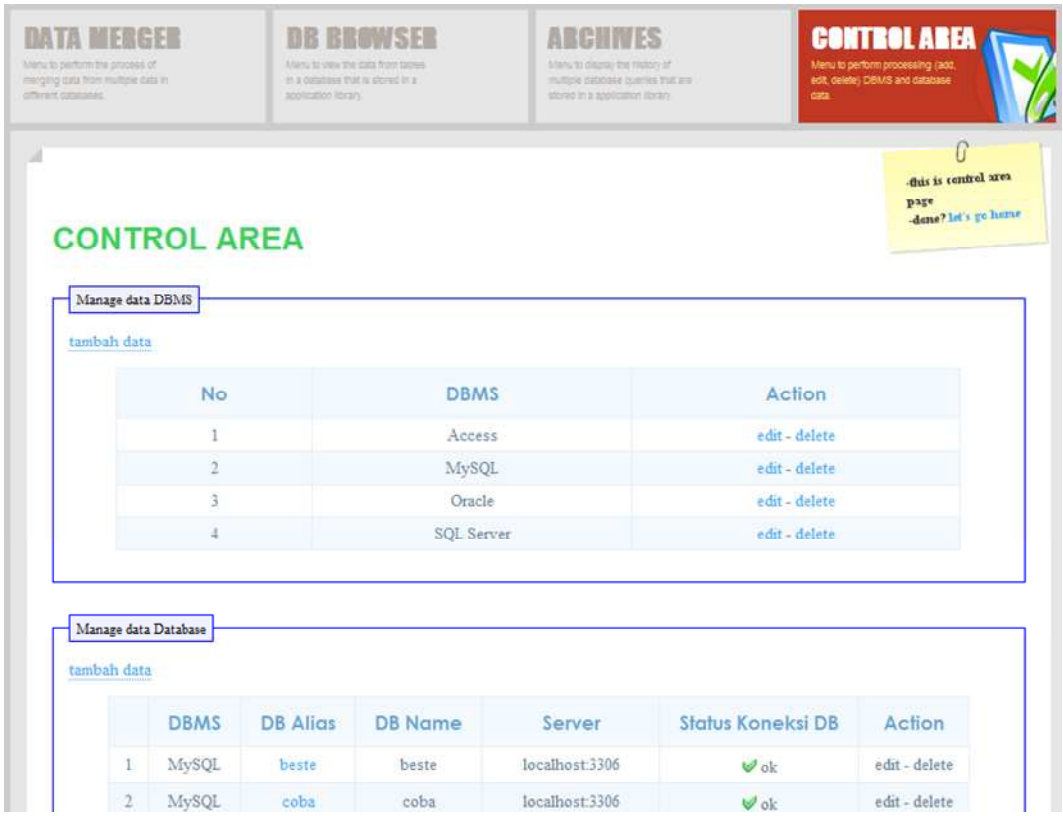
4.2.1.2. Tampilan Program

Untuk memperjelas bentuk dari implementasi antarmuka di atas, berikut ini adalah beberapa tampilan program yang telah dibuat.

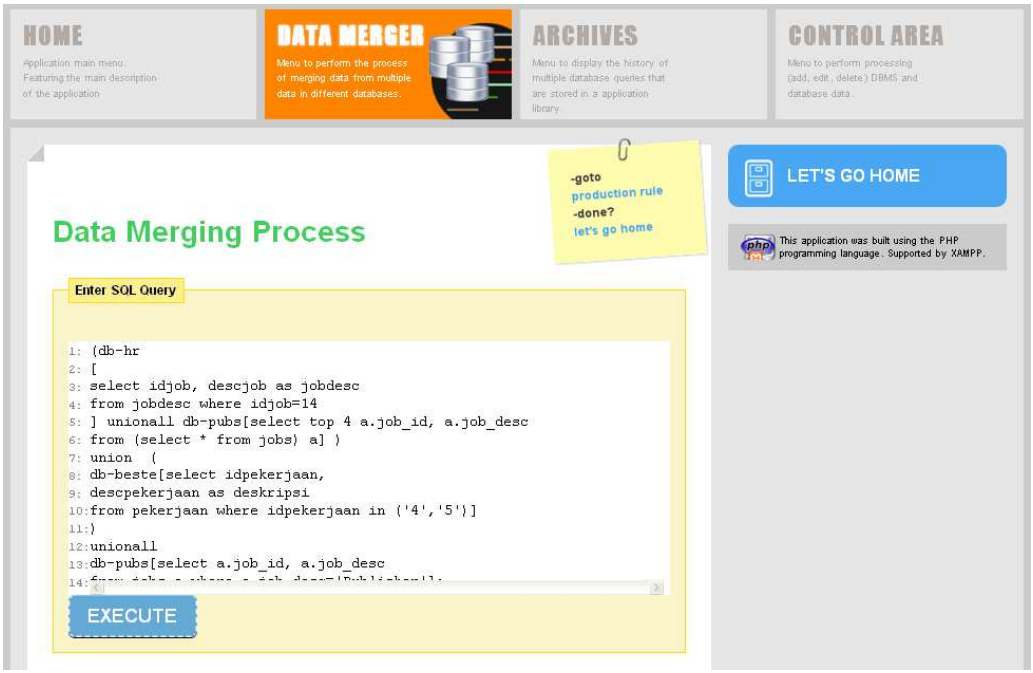
Pada gambar 4.1 dapat dilihat tampilan awal (*homepage*) aplikasi dalam bahasa inggris.



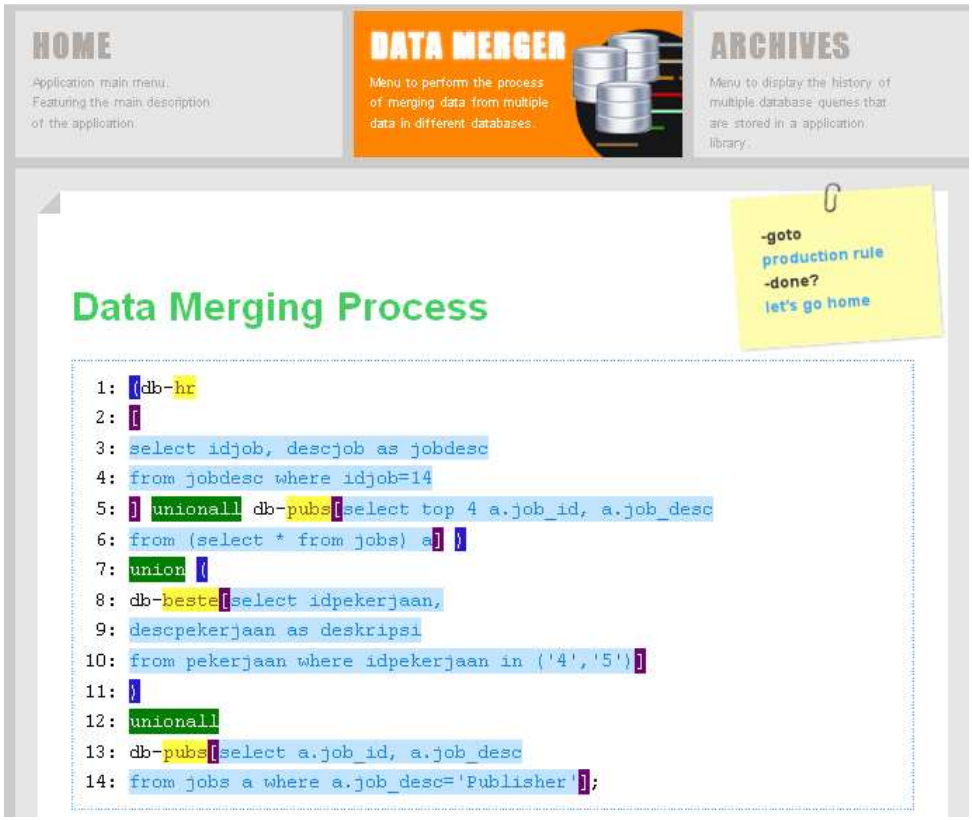
Gambar 4.1 Tampilan halaman awal atau *homepage*



Gambar 4.2 Tampilan halaman *Control Area*



Gambar 4.3 Tampilan halaman masukan SQL *query* menu *Data Merger*



Gambar 4.4 Tampilan *sintax highlighting* pada menu *Data Merger*

4.2.1.3. Memperbaiki dan Menguji Program Aplikasi

Untuk menguji kebenaran program tersebut maka program harus diuji dengan data tes yang telah dipersiapkan sebelumnya, sehingga bila tidak sesuai dengan kebutuhan, kesalahan sintaks, kesalahan logika dan kesalahan pada saat *run time* dapat diketahui secara dini. Pengetesan program dilakukan dengan dua cara yaitu :

1. Pengetesan setiap modul program secara terpisah

   Pengetesan terhadap setiap modul program dimaksudkan apabila ada kesalahan akan lebih mudah dan cepat dalam memperbaikinya.

2. Pengetesan modul program yang telah terintegrasi

   Setelah seluruh modul dites dan tidak terdapat kesalahan, maka selanjutnya dilakukan tes secara keseluruhan menjadi satu unit program. Hal ini dimaksudkan untuk meyakinkan bahwa semua modul dapat terintegrasi tanpa mengalami kesalahan.

4.2.1.4. Mempersiapkan Perangkat Keras dan Perangkat Lunak

Kegiatan ini bertujuan untuk menyiapkan perangkat keras dan perangkat lunak yang akan digunakan dalam aplikasi.

4.2.1.4.1. Kebutuhan Perangkat Keras

Perangkat keras yang akan digunakan oleh aplikasi ini harus memiliki spesifikasi khusus karena dibutuhkan untuk mengelola *database*. Perangkatkeras yang adadi Telkom sudah cukup untuk mendukung sistem yang diusulkan sehingga tidak perlu ada tambahan perangkat keras lagi.

Adapun spesifikasi perangkat keras pada salah satu server yang digunakan di Telkom dapat dilihat pada tabel 4.1 berikut ini.

Tabel 4.1 Perangkas Keras yang digunakan

NO	JENIS	KETERANGAN
1	Processor	2 slot: 1 kosong, 1 isi CPU DualCore Intel Xeon 5130 2000MHz
2	RAM	8 GB
3	Harddisk	1x68GB dan 2x136GB
4	LAN Card	10/100 Mbps
5	Monitor	Resolusi 1024x768

Untuk kebutuhan akan *resource* yang lebih besar, suatu saat akan diperlukan kebutuhan perangkat keras yang lebih tinggi. Ada beberapa hal yang perlu dipertimbangkan dalam mempersiapkan perangkat keras, yaitu :

1. Adanya dukungan dari perangkat lunak yang akan digunakan.
2. Keamanan fisik dari tempat perangkat keras diletakkan.
3. Terdapat ruang khusus server yang cukup pendingin dan kebersihan terjaga dengan baik.
4. Memiliki dukungan vendor atau merk perangkat keras sehingga akan mudah bila terjadi kerusakan.

4.2.1.4.2. Kebutuhan Perangkat Lunak

Perangkat lunak yang digunakan dalam membangun aplikasi ini dapat dilihat pada tabel 4.2 berikut ini:

Tabel 4.2 Perangkas Lunak yang digunakan

NO	PERANGKAT LUNAK	KETERANGAN 1	KETERANGAN 2
1	SistemOperasi	Windows server 2003	ok
2	BahasaPemograman	PHP	install
3	Web Server	Apache	install
4	*Database* Server	Oracle dan MySQL	ok
5	Web Browser	Opera, Mozilla Firefox, Chrome, IE	ok
6	Code Editor	Notepad++	ok
7	Diagram Modeler	Visio, JFLAP, dan Visual Paradigm	install

4.2.1.5. Pendidikan dan Pelatihan Sumber Daya Manusia

Agar sistem dapat diolah secara maksimal maka perlu diadakan pelatihan terhadap pengguna sistem. Metode pelatihan yang diberikan melibatkan personil secara langsung disituasi pekerjaan. Dalam hal mengoperasikan sistem, mempersiapkan data masukan, cara memasukan data dan cara memanfaatkan informasi yang ada pada sistem.

4.3 Pengujian Sistem

Pengujian sistem dilakukan setelah implementasi sistem yang bertujuan untuk mengetahui sejauh mana sistem tersebut dapat memenuhi kebutuhan bagi pengguna sistem juga untuk menguji kemampuan sistem dalam menjalankan fungsionalitas yang ada.

4.3.1 Skenario Pengujian Alpha

Tujuan dari skenario pengujian alpha adalah untuk mengidentifikasi strategi pengujian dan prosedur pengujian yang memungkinkan proses pengujian yang efektif dan efisien. Jenis pengujian yang dilakukan adalah pengujian *black box.*

Tabel 4.3 Skenario Pengujian Aplikasi

No	Kelas Uji	Butir Uji	Jenis Pengujian
1	Eksekusi masukan SQL *Query*	Input dan eksekusi SQL *Query*	*Black box*
		Parsing SQL *Query*	*Black box*
		Error Handling	*Black box*
2	Pengolahan data parameter	Pengolahan data parameter jenis *database*	*Black box*
		Pengolahan data parameter *database*	*Black box*

4.3.2 Kasus dan Hasil Pengujian Alpha

Berdasarkan skenario pengujian alpha yang telah disusun, maka dapat dilakukan pengujian seperti pada tabel 4.6. Untuk memperjelas pengujian aplikasi dalam melakukan atau mengeksekusi masukan SQL *query* yang diberikan, maka penulis memberikan contoh dua buah tabel yang akan diakses. Dengan

menggunakan tabel-tabel tersebut diharapkan pembaca bisa lebih jelas dalam melihat hasil pengujian alpha aplikasi ini.

Tabel pertama adalah tabel TNOMOR yang disimpan dalam *my*SQL *database* dengan alias *database* adalah beste. Nama kolom dan isi dari tabel TNOMOR dapat dilihat pada tabel 4.4 dibawah ini.

Tabel 4.4 Nama kolom dan isi tabel TNOMOR

CNOMOR	KETNOMOR
1	SATU
2	DUA
3	TIGA
4	EMPAT
5	LIMA

Tabel kedua adalah tabel TNUMBER yang disimpan dalam *oracle database* dengan alias *database* adalah customer. Kolom dan isi tabel-table tersebut dapat di lihat pada tabel 4.5 di bawah ini.

Tabel 4.5 Nama kolom dan isi tabel TNUMBER

CNUMBER	KETNUMBER
3	TIGA
4	EMPAT
5	LIMA
6	ENAM
7	TUJUH

Pengujian aplikasi yang dilakukan dapat dilihat pada tabel 4.6 berikut ini.

Tabel 4.6 Pengujian Aplikasi

<table>
<tr><th colspan="4">Kelas dan Hasil Uji</th></tr>
<tr><th>Data Masukan</th><th>Yang Diharapkan</th><th>Pengamatan</th><th>Kesimpulan</th></tr>
<tr><td colspan="4">Masukan SQL Query(masukan benar atau sesuai format yang diterima)</td></tr>
<tr><td>Masukan SQL Query
- hanya satu buah ekspresi
- tanpa ada set operator (union/union all/intersect/minus)
- masukan benar/sesuai format yang diterima
- contoh : db-beste[select * from tnomor where cnomor<4];</td><td>Menghasilkan data keluaran sesuai dengan masukan.
(menghasilkan semua data di tabel tnomor)
Hasil data :
<table><tr><th>Cnomor</th><th>ketnomor</th></tr><tr><td>1</td><td>SATU</td></tr><tr><td>2</td><td>DUA</td></tr><tr><td>3</td><td>TIGA</td></tr></table></td><td>Aplikasi berhasil menampilkan data keluaran sesuai dengan masukan.</td><td>[√] diterima
[ ] ditolak</td></tr>
<tr><td>Masukan SQL Query
- dua buah ekspresi
- sebuah set operator
- masukan benar
- contoh :
db-beste[select tnomor from tnomor]
minus
db-customer
[select cnumber from tnumber];</td><td>Menghasilkan data keluaran sesuai dengan masukan.
Hasil data :
<table><tr><th>cnomor</th></tr><tr><td>1</td></tr><tr><td>2</td></tr></table></td><td>Aplikasi berhasil menampilkan data keluaran sesuai dengan masukan</td><td>[√] diterima
[ ] ditolak</td></tr>
<tr><td>Masukan SQL Query
- tiga buah ekspresi
- dua buah set operator tanpa menggunakan</td><td>Menghasilkan data keluaran sesuai dengan masukan.
Hasil data :</td><td>Aplikasi berhasil menampilkan data keluaran sesuai dengan masukan</td><td>[√] diterima
[ ] ditolak</td></tr>
</table>

tanda kurung buka dan kurung tutup (untuk mengelompokkan ekspresi) - masukan benar/sesuai format yang diterima - contoh: **db-beste[select tnomor from tnomor]** **minus** **db-customer** **[select cnumber from tnumber]** **union** **db-beste[select tnomor from tnomor where tnomor=5]** **;**	**cnomor** 1 2 5		
Masukan SQL *Query* - tiga buah ekspresi - dua buah set operator - dengan menggunakan tanda kurung buka dan kurung tutup - masukan benar/sesuai format yang diterima - contoh : **db-customer [select ccat, '['||ccat||'] '||lcat as category** **from p_ccat where ccat='100']**	Menghasilkan data keluaran sesuai dengan masukan. (menghasilkan data cdb dan jenis di table t_db diminus dengan data ccat dan lcat di tabel p_ccat kemudian hasilnya digabung dengan data ccat dan category di tabel p_ccat)	Aplikasi berhasil menampilkan data keluaran sesuai dengan masukan	[√] diterima [ ] ditolak

union (**db-beste[select cdb, jenisdb as jenis from t_db]** **minus** **db-customer[select a.ccat, a.lcat from (select * from p_ccat) a]** **);**			
Masukan SQL *Query*(masukan salah/ tidak sesuai format yang diterima)			
Masukan SQL *Query* - hanya satu buah ekspresi - tanpa ada set operator (union/union all/intersect/minus) - masukan salah: *dbalias/dbidentifer* yang tidak ada dalam *database* - contoh :	Menampilkan pesan kesalahan bahwa *dbalias/dbidentifier* yang dimasukkan salah dan letak kesalahan ditandai dengan warna merah pada *string* yang salah.	Pesan kesalahan berhasil ditampilkan	[√] diterima [ ] ditolak
Masukan SQL *Query* - hanya satu buah ekspresi - tanpa ada set operator (union/union all/intersect/minus) - masukan salah: hanya ada tanda kurung siku buka ‘[‘ dan tanpa ada kurung siku tutup ‘]’	Menampilkan pesan kesalahan bahwa ditemukan tanda kurung siku buka ‘[’ tetapi tidak ditemukan tanda kurung siku tutup ‘]’ dan letak kesalahan ditandai dengan warna merah pada *string* yang salah.	Pesan kesalahan berhasil ditampilkan	[√] diterima [ ] ditolak

Masukan SQL *Query* - hanya satu buah ekspresi - tanpa ada set operator - masukan salah: *block* SQL *query*salah	Menampilkan pesan kesalahan bahwa SQL *Query* salah dan letak kesalahan ditandai dengan warna merah pada *string* yang salah.Pesan kesalahan sesuai dengan pesan kesalahan masing-masing DBMS.	Pesan kesalahan berhasil ditampilkan	[√] diterima [ ] ditolak
Masukan SQL *Query* - dua buah ekspresi - sebuah set operator - masukan salah: set operator salah	Menampilkan pesan kesalahan bahwa set operator salah dan letak kesalahan ditandai dengan warna merah pada *string* yang salah.	Pesan kesalahan berhasil ditampilkan	[√] diterima [ ] ditolak
Masukan SQL *Query* - tiga buah ekspresi - dua buah set operator - masukan salah : hanya ada tanda kurung tutup ‘)’ dan tanpa ada kurung buka ‘)’	Menampilkan pesan kesalahan bahwa ditemukan tanda kurung tutup ‘)’ tetapi tidak ditemukan tanda kurung buka ‘(’ dan letak kesalahan ditandai dengan warna merah pada *string* yang salah.	Pesan kesalahan berhasil ditampilkan	[√] diterima [ ] ditolak
Pengujian Handling Error pada Parsing SQL *Query*			
Data Masukan	**Yang Diharapkan**	**Pengamatan**	**Kesimpulan**
Pengguna memberikan masukan yang terdapat kesalahan pada kurung	Terdapat pesan kesalahan yang memberikan informasi:	Muncul pesan jenis kesalahan, Posisi kesalahan, dan deskripsi	[√] diterima [ ] ditolak

siku. Misal: diberikan kurung siku buka '[' tanpa menuliskan kurung siku tutup ']'	- Jenis kesalahan format sintaks - Posisi kesalahan, - deskripsi kesalahan.	kesalahan. (pesan kesalahan sesuai dengan yang diharapkan)	
Pengguna memberikan masukan yang terdapat kesalahan pada nama alias *database*. Misalkan alias *database* diisikan sembarangan.	Terdapat pesan kesalahan yang memberikan informasi: - Jenis kesalahan dbalias - Posisi kesalahan, - deskripsi kesalahan.	Muncul pesan jenis kesalahan, Posisi kesalahan, dan deskripsi kesalahan. (pesan kesalahan sesuai dengan yang diharapkan)	[√] diterima [ ] ditolak
Pengguna memberikan masukan yang terdapat kesalahan pada kurung siku. Misal: diberikan kurung tutup ')' tanpa menuliskan kurung buka '('	Terdapat pesan kesalahan yang memberikan informasi: - Jenis kesalahan format sintaks - Posisi kesalahan, - deskripsi kesalahan.	Muncul pesan jenis kesalahan, Posisi kesalahan, dan deskripsi kesalahan. (pesan kesalahan sesuai dengan yang diharapkan)	[√] diterima [ ] ditolak
Pengguna memberikan masukan yang terdapat kesalahan pada *set operator*. Misalkan alias *database* diisikan sembarangan.	Terdapat pesan kesalahan yang memberikan informasi: - Jenis kesalahan set operator - Posisi kesalahan, - deskripsi kesalahan.	Muncul pesan jenis kesalahan, Posisi kesalahan, dan deskripsi kesalahan. (pesan kesalahan sesuai dengan yang diharapkan)	[√] diterima [ ] ditolak
Pengguna memberikan masukan yang terdapat kesalahan pada SQL *query*. Misalkan *query*-1 menghasilkan 3 kolom dan *query*-2 menghasilkan 4 kolom.	Terdapat pesan kesalahan yang memberikan informasi: - Jenis kesalahan jumlah kolom tidak sama - Posisi kesalahan,	Muncul pesan jenis kesalahan, Posisi kesalahan, dan deskripsi kesalahan. (pesan kesalahan sesuai dengan yang diharapkan)	[√] diterima [ ] ditolak

	- deskripsi kesalahan.		
Pengguna memberikan masukan yang terdapat kesalahan pada SQL *query*. Misalkan *query* tidak menghasilkan data.	Terdapat pesan kesalahan yang memberikan informasi: - Jenis kesalahan query tidak menghasilkan data. - Posisi kesalahan, - deskripsi kesalahan.	Muncul pesan jenis kesalahan, Posisi kesalahan, dan deskripsi kesalahan. (pesan kesalahan sesuai dengan yang diharapkan)	[√] diterima [ ] ditolak

4.3.3 Kesimpulan Hasil Pengujian Alpha

Berdasarkan hasil pengujian dengan kasus *sample* uji yang telah dilakukan memberikan kesimpulan bahwa pada proses penggabungan data sudah berjalan dengan benar tetapi masih memungkinkan untuk terjadi kesalahan pada sintaks karena kemungkinan ada bentuk kesalahan lain dalam format masukan ataupun ada kesalahan dalam proses parsing. Secara fungsional system sudah dapat menghasilkan keluaran dan informasi yang diharapkan.

4.3.4 Pengujian Beta

Pengujian beta merupakan pengujian yang dilakukan secara objektif dimana aplikasi diuji secara langsung ke lingkungan atau unit yang bersangkutan. Pengujian beta dilakukan dengan membuat kuisoner yang diberikan kepada sepuluh orang pengguna dari lingkungan atau unit yang berkepentingan dengan aplikasi ini. Kuisoner merupakan daftar pertanyaan atau pernyataan yang distandarisasikan dan menstrukturkan serta memperluas proses pengumpulan fakta. Dengan menggunakan kuisoner akan didapat informasi yang objektif dari pengguna tentang aplikasi ini. Dari hasil kuisoner tersebut akan dilakukan perhitungan untuk dapat diambil kesimpulan terhadap penelitian yang dilakukan oleh penulis.

Kuisoner yang diberikan kepada pengguna aplikasi ini dapat dilihat pada tabel 4.7 di bawah ini.

Tabel 4.7 Kuisoner Pengujian Beta

Bagian A : Kemudahan Koneksi Dan Akses Data Ke Berbagai DBMS	
1.	Cara koneksi ke berbagai DBMS yang beragam lebih baik disimpan ke dalam library aplikasi. Agar jika diperlukan dapat langsung dipanggil dengan parameter tertentu.
	[a] sangat setuju [b] setuju [c] cukup setuju [d] kurang setuju [e] tidak setuju
2.	Jika terdapat beberapa atau banyak *database*, parameter *database* sebaiknya disimpan ke dalam suatu tabel di salah satu *database* yang ada. (parameter *database*: *IP address, port, username, password,* dsb)

	[a] sangat setuju [b] setuju [c] cukup setuju	[d] kurang setuju [e] tidak setuju
3.	Untuk penyimpanan data parameter *database*, *database* yang *free* atau bukan *database* komersil merupakan pilihan yang baik. Penulis menggunakan *database* mysql.	
	[a] sangat setuju [b] setuju [c] cukup setuju	[d] kurang setuju [e] tidak setuju
4.	Coding dengan bahasa pemrograman PHP, sintaks untuk menampilkan data dari *database*, cara penulisan sintaksnya berbeda-beda untuk DBMS yang berbeda. Aplikasi ini menggunakan cara penulisan (sintaks) yang sama untuk menampilkan data dari *database*, meskipun DBMS berbeda-beda. Dengan cara ini *programmer* dimudahkan dalam kegiatan *coding* aplikasi.	
	[a] sangat setuju [b] setuju [c] cukup setuju	[d] kurang setuju [e] tidak setuju
5.	Format sintaks untuk *query* ke *database* adalah db-alias + '[' + sql *query* + ']' Apakah format SQL *query* tersebut mudah dipahami?	
	[a] sangat mudah [b] mudah [c] cukup mudah	[d] cukup sulit [e] sangat sulit

6.	Cara koneksi ke *database* hanya dengan memanggil atau menuliskan alias *database* saja. Apakah cara tersebut mudah digunakan?	
	[a] sangat mudah [b] mudah [c] cukup mudah	[d] cukup sulit [e] sangat sulit
7.	Apakah aplikasi ini memudahkan anda dari sisi kemudahan koneksi dan akses data ke dalam *database* dengan DBMS yang berbeda?	
	[a] sangat setuju [b] setuju [c] cukup setuju	[d] kurang setuju [e] tidak setuju
Bagian B : Fungsionalitas Penggabungan Data antar *Database* denganDBMS yang Berbeda		
1.	Apakah format sintaks SQL *query* ke *database* melalui aplikasi ini mudah dipahami?	
	[a] sangat mudah [b] mudah [c] cukup mudah	[d] cukup sulit [e] sangat sulit
2.	Aplikasi dapat menjalankan fungsi operation set antar tabel dari DBMS yang berbeda-beda. Apakah fungsionalitas ini diperlukan?	
	[a] sangat diperlukan [b] diperlukan [c] cukup diperlukan	[d] kurang diperlukan [e] tidak diperlukan

3.	Aplikasi dapat menjalankan fungsi operation set antar tabel dari DBMS yang berbeda-beda. Apakah fungsionalitas ini membantu pekerjaan anda?	
	[a] sangat membantu [b] membantu [c] cukup membantu	[d] kurang membantu [e] tidak membantu
4.	Aplikasi ini menyajikan dua buah cara untuk melakukan penggabungan data. Cara 1 : dengan memasukkan sintaks SQL *query*. Cara 2 : dengan cara bertahap (step by step wizard). Apakah anda mengerti cara menggunakannya?	
	[a] sangat mengerti [b] mengerti [c] cukup mengerti	[d] kurang mengerti [e] tidak mengerti
5.	Apakah data keluaran yang dihasilkan aplikasi ini sudah sesuai dengan kebutuhan anda?	
	[a] sangat sesuai [b] sesuai [c] cukup sesuai	[d] kurang sesuai [e] tidak sesuai
Bagian C : Kemudahan Format Sintaks PHP dan SQL Query untuk Membangun Aplikasi Web dengan single or multiple DBMS		
1.	Aplikasi dapat memperlihatkan contoh sintaks PHP dan SQL Query untuk menampilkan data dari *database*.	

	[a] sangat setuju [b] setuju [c] cukup setuju	[d] kurang setuju [e] tidak setuju
2.	Apakah anda mengerti atau paham dengan contoh sintaks PHP dan SQL Query untuk menampilkan data dari *database*.	
	[a] sangat mengerti [b] mengerti [c] cukup mengerti	[d] kurang mengerti [e] tidak mengerti
3.	Untuk membangun aplikasi web dengan metode seperti aplikasi ini.Yang diperlukan oleh *programmer* atau *developer* adalah menyalin file soda.php dan menyiapkan *my*SQL *database* untuk menyimpan data parameter *database* apa saja yang digunakan.Apakah anda mengerti tentang cara atau metode aplikasi ini?	
	[a] sangat mengerti [b] mengerti [c] cukup mengerti	[d] kurang mengerti [e] tidak mengerti
4.	Pengaturan parameter *database* dilakukan di *database*. Bisa melalui aplikasi ini atau bisa langsung *insert* ke tabel di *database*.	
	[a] sangat setuju [b] setuju [c] cukup setuju	[d] kurang setuju [e] tidak setuju
5.	Hanya perlu satu baris sintaks PHP untuk mengakses dan menampilkan data dari single or multiple *database*. Cara ini memudahkan *programmer* dalam membangun aplikasi dengan menggunakan *single or multiple database*.	

	[a] sangat setuju [b] setuju [c] cukup setuju	[d] kurang setuju [e] tidak setuju
Bagian D : Kesimpulan dan Saran		

Berdasarkan data hasil kuesioner tersebut, dapat dicari persentase masing-masing jawaban dengan menggunakan rumus :**Y = (P/Q) * 100%**

Keterangan :

P = Banyaknya jawaban responden tiap soal.

Q = Jumlah responden

Y = Nilai persentase

Berikut ini adalah hasil persentase masing-masing jawaban yang sudah dihitung nilainya dengan menggunakan rumus diatas. Kuesioner ini diujikan kepada 10 orang.

Bagian A :

1. Cara koneksi ke berbagai DBMS yang beragam lebih baik disimpan ke dalam library aplikasi. Agar jika diperlukan dapat langsung dipanggil dengan parameter tertentu.

Tabel 4.8 Hasil kuesioner bagian A no 1

Pertanyaan	**No.**	**Keterangan**	**Responden**	**Prosentase (%)**
1	1	Sangat setuju	8	80
	2	Setuju	2	20
	3	Cukup setuju	0	0
	4	Kurang setuju	0	0
	5	Tidak setuju	0	0
Jumlah			**10**	**100**

Berdasarkan kuesioner bagian A nomor 1 di atas diperoleh 80% responden yang sangat setuju, maka dapat disimpulkan bahwa diperlukan suatu library untuk menyimpan parameter *database* apa saja yang digunakan oleh aplikasi.

2. Jika terdapat beberapa atau banyak *database*, parameter *database* sebaiknya disimpan ke dalam suatu tabel di salah satu *database* yang ada.
(parameter *database: IP address, port, username, password,* dsb)

Tabel 4.9 Hasil kuesioner bagian A no 2

Pertanyaan	**No.**	**Keterangan**	**Responden**	**Prosentase (%)**
2	1	Sangat setuju	3	30
	2	Setuju	7	70
	3	Cukup setuju	0	0
	4	Kurang setuju	0	0
	5	Tidak setuju	0	0
Jumlah			**10**	**100**

Berdasarkan kuesioner bagian A nomor 2 di atas diperoleh 70% responden yang setuju, maka dapat disimpulkan bahwa diperlukan satu *database* untuk menyimpan parameter *database* apa saja yang digunakan oleh aplikasi.

3. Untuk penyimpanan data parameter *database*, *database* yang free atau bukan *database* komersil merupakan pilihan yang baik. Penulis menggunakan *database* mysql.

Tabel 4.10 Hasil kuesioner bagian A no 3

Pertanyaan	**No.**	**Keterangan**	**Responden**	**Prosentase (%)**
3	1	Sangat setuju	8	80
	2	Setuju	2	20
	3	Cukup setuju	0	0
	4	Kurang setuju	0	0
	5	Tidak setuju	0	0
Jumlah			**10**	**100**

Berdasarkan kuesioner bagian A nomor 3 di atas diperoleh 80% responden yang sangat setuju, maka dapat disimpulkan bahwa diperlukan *database my*SQL untuk menyimpan data parameter *database*.

4. Coding dengan bahasa pemrograman PHP, sintaks untuk menampilkan data dari *database*, cara penulisan sintaksnya berbeda-beda untuk DBMS yang berbeda. Aplikasi ini menggunakan cara penulisan (sintaks) yang sama untuk menampilkan data dari *database*, meskipun DBMS berbeda-beda. Dengan cara ini programmer dimudahkan dalam kegiatan coding aplikasi.

Tabel 4.11 Hasil kuesioner bagian A no 4

Pertanyaan	**No.**	**Keterangan**	**Responden**	**Prosentase (%)**
4	1	Sangat setuju	7	70
	2	Setuju	3	30
	3	Cukup setuju	0	0
	4	Kurang setuju	0	0
	5	Tidak setuju	0	0
Jumlah			**10**	**100**

Berdasarkan kuesioner bagian A nomor 4 di atas diperoleh 70% responden yang sangat setuju, maka dapat disimpulkan bahwa dengan menggunakan aplikasi ini *programmer* dimudahkan dalam kegiatan *coding* aplikasi.

5. Format sintaks untuk *query* ke *database* adalah

   db-alias + ‘[‘ + sql *query* + ‘]’

   Apakah format SQL *query* tersebut mudah dipahami?

Tabel 4.12 Hasil kuesioner bagian A no 5

Pertanyaan	**No.**	**Keterangan**	**Responden**	**Prosentase (%)**
5	1	Sangat mudah	2	20
	2	Mudah	7	70
	3	Cukup mudah	1	10
	4	Cukup sulit	0	0
	5	Sangat sulit	0	0
Jumlah			**10**	**100**

Berdasarkan kuesioner bagian A nomor 5 di atas diperoleh 70% responden yang memilih jawaban mudah, maka dapat disimpulkan bahwa format SQL *query* tersebut mudah dipahami.

6. Cara koneksi ke *database* hanya dengan memanggil atau menuliskan alias *database* saja. Apakah cara tersebut mudah digunakan?

Tabel 4.13 Hasil kuesioner bagian A no 6

Pertanyaan	**No.**	**Keterangan**	**Responden**	**Prosentase (%)**
6	1	Sangat mudah	9	90
	2	Mudah	1	10
	3	Cukup mudah	0	0
	4	Cukup sulit	0	0
	5	Sangat sulit	0	0
Jumlah			**10**	**100**

Berdasarkan kuesioner bagian A nomor 6 di atas diperoleh 70% responden yang memilih jawaban sangat mudah, maka dapat disimpulkan bahwa cara koneksi ke *database* dalam aplikasi ini mudah dipahami.

7. Apakah aplikasi ini memudahkan anda dari sisi kemudahan koneksi dan akses data ke dalam *database* dengan DBMS yang berbeda?

Tabel 4.14 Hasil kuesioner bagian A no 7

Pertanyaan	**No.**	**Keterangan**	**Responden**	**Prosentase (%)**
7	1	Sangat setuju	6	60
	2	Setuju	4	40
	3	Cukup setuju	0	0
	4	Kurang setuju	0	0
	5	Tidak setuju	0	0
Jumlah			**10**	**100**

Berdasarkan kuesioner bagian A nomor 7 di atas diperoleh 60% responden yang sangat setuju, maka dapat disimpulkan bahwa aplikasi ini memudahkan anda dari sisi kemudahan koneksi dan akses data ke dalam *database* dengan DBMS yang berbeda.

Bagian B :

1. Apakah format sintaks SQL *query* ke *database* melalui aplikasi ini mudah dipahami?

Tabel 4.15 Hasil kuesioner bagian B no 1

Pertanyaan	No.	Keterangan	Responden	Prosentase (%)
1	1	Sangat mudah	1	10
	2	Mudah	8	80
	3	Cukup mudah	1	10
	4	Cukup sulit	0	0
	5	Sangat sulit	0	0
Jumlah			**10**	**100**

Berdasarkan kuesioner bagian B nomor 1 di atas diperoleh 80% responden yang memilih jawaban mudah, maka dapat disimpulkan bahwa format sintaks SQL *query* ke *database* melalui aplikasi ini mudah dipahami.

2. Aplikasi dapat menjalankan fungsi operation set antar tabel dari DBMS yang berbeda-beda. Apakah fungsionalitas ini diperlukan?

Tabel 4.16 Hasil kuesioner bagian B no 2

Pertanyaan	No.	Keterangan	Responden	Prosentase (%)
2	1	Sangat diperlukan	9	90
	2	Diperlukan	1	10
	3	Cukup diperlukan	0	0
	4	Kurang diperlukan	0	0
	5	Tidak diperlukan	0	0
Jumlah			**10**	**100**

Berdasarkan kuesioner bagian B nomor 2 di atas diperoleh 90% responden yang memilih sangat diperlukan, maka dapat disimpulkan bahwa fungsionalitas *operation set* antar tabel dari DBMS yang berbeda-beda sangat diperlukan.

3. Aplikasi dapat menjalankan fungsi operation set antar tabel dari DBMS yang berbeda-beda. Apakah fungsionalitas ini membantu pekerjaan anda?

Tabel 4.17 Hasil kuesioner bagian B no 3

Pertanyaan	**No.**	**Keterangan**	**Responden**	**Prosentase (%)**
3	1	Sangat membantu	9	90
	2	Membantu	1	10
	3	Cukup membantu	0	0
	4	Kurang membantu	0	0
	5	Tidak membantu	0	0
Jumlah			**10**	**100**

Berdasarkan kuesioner bagian B nomor 3 di atas diperoleh 90% responden yang memilih sangat diperlukan, maka dapat disimpulkan bahwa fungsionalitas *operation set* antar tabel dari DBMS yang berbeda-beda sangat membantu pekerjaan responden.

4. Aplikasi ini menyajikan dua buah cara untuk melakukan penggabungan data. Cara 1 : dengan memasukkan sintaks SQL *query*. Cara 2 : dengan cara bertahap (*step by step wizard*).

   Apakah anda mengerti cara menggunakannya?

Tabel 4.18 Hasil kuesioner bagian B no 4

Pertanyaan	No.	Keterangan	Responden	Prosentase (%)
4	1	Sangat mengerti	0	0
	2	Mengerti	7	70
	3	Cukup mengerti	3	30
	4	Kurang mengerti	0	0
	5	Tidak mengerti	0	0
Jumlah			**10**	**100**

Berdasarkan kuesioner bagian B nomor 4 di atas diperoleh 70% responden yang memilih jawaban mengerti, maka dapat disimpulkan bahwa responden mengerti cara menggunakan dan melakukan penggabungan data.

5. Apakah data keluaran yang dihasilkan aplikasi ini sudah sesuai dengan kebutuhan anda?

Tabel 4.19 Hasil kuesioner bagian B no 5

Pertanyaan	No.	Keterangan	Responden	Prosentase (%)
5	1	Sangat sesuai	10	100
	2	Sesuai	0	0
	3	Cukup sesuai	0	0
	4	Kurang sesuai	0	0
	5	Tidak sesuai	0	0
Jumlah			**10**	**100**

Berdasarkan kuesioner bagian B nomor 5 di atas diperoleh 100% responden yang memilih jawaban sangat sesuai, maka dapat disimpulkan bahwa data keluaran yang dihasilkan aplikasi ini sudah sesuai dengan kebutuhan responden.

Bagian C :

1. Aplikasi dapat memperlihatkan contoh sintaks PHP dan SQL Query untuk menampilkan data dari *database*.

Tabel 4.20 Hasil kuesioner bagian C no 1

Pertanyaan	No.	Keterangan	Responden	Prosentase (%)
1	1	Sangat setuju	1	10
	2	Setuju	9	90
	3	Cukup setuju	0	0
	4	Kurang setuju	0	0
	5	Tidak setuju	0	0
Jumlah			**10**	**100**

Berdasarkan kuesioner bagian C nomor 1 di atas diperoleh 90% responden yang memilih jawaban setuju, maka dapat disimpulkan bahwa aplikasi dapat memperlihatkan contoh sintaks PHP dan SQL *Query* untuk menampilkan data dari *database*.

2. Apakah anda mengerti atau paham dengan contoh sintaks PHP dan SQL Query untuk menampilkan data dari *database*.

Tabel 4.21 Hasil kuesioner bagian C no 2

Pertanyaan	No.	Keterangan	Responden	Prosentase (%)
2	1	Sangat mengerti	0	0
	2	Mengerti	8	80
	3	Cukup mengerti	2	20
	4	Kurang mengerti	0	0
	5	Tidak mengerti	0	0
Jumlah			**10**	**100**

Berdasarkan kuesioner bagian C nomor 2 di atas diperoleh 80% responden yang memilih jawaban mengerti, maka dapat disimpulkan bahwa responden

mengerti atau paham dengan contoh sintaks PHP dan SQL *Query* untuk menampilkan data dari *database*

3. Untuk membangun aplikasi web dengan metode seperti aplikasi ini. Yang diperlukan oleh programmer atau developer adalah menyalin file soda.php dan menyiapkan mysql *database* untuk menyimpan data parameter *database* apa saja yang digunakan. Apakah anda mengerti tentang cara atau metode aplikasi ini?

Tabel 4.22 Hasil kuesioner bagian C no 3

Pertanyaan	**No.**	**Keterangan**	**Responden**	**Prosentase (%)**
3	1	Sangat mengerti	0	0
	2	Mengerti	3	30
	3	Cukup mengerti	6	60
	4	Kurang mengerti	1	10
	5	Tidak mengerti	0	0
Jumlah			**10**	**100**

Berdasarkan kuesioner bagian C nomor 3 di atas diperoleh 60% responden yang memilih jawaban cukup mengerti, maka dapat disimpulkan bahwa responden cukup mengerti tentang metode aplikasi ini.

4. Pengaturan parameter *database* dilakukan di *database*. Bisa melalui aplikasi ini atau bisa langsung insert ke tabel di *database*.

Tabel 4.23 Hasil kuesioner bagian C no 4

Pertanyaan	No.	Keterangan	Responden	Prosentase (%)
4	1	Sangat setuju	7	70
	2	Setuju	3	30
	3	Cukup setuju	0	0
	4	Kurang setuju	0	0
	5	Tidak setuju	0	0
Jumlah			**10**	**100**

Berdasarkan kuesioner bagian C nomor 4 di atas diperoleh 70% responden yang memilih jawaban sangat setuju, maka dapat disimpulkan bahwa responden sangat setuju jika pengaturan parameter *database* dilakukan di *database*.

5. Hanya perlu satu baris sintaks PHP untuk mengakses dan menampilkan data dari single or multiple *database*. Cara ini memudahkan programmer dalam membangun aplikasi dengan menggunakan single or multiple *database*.

Tabel 4.24 Hasil kuesioner bagian C no 5

Pertanyaan	No.	Keterangan	Responden	Prosentase (%)
5	1	Sangat setuju	7	70
	2	Setuju	3	30
	3	Cukup setuju	0	0
	4	Kurang setuju	0	0
	5	Tidak setuju	0	0
Jumlah			**10**	**100**

Berdasarkan kuesioner bagian C nomor 5 di atas diperoleh 70% responden yang memilih jawaban sangat setuju, maka dapat disimpulkan bahwa aplikasi ini memudahkan *programmer* dalam membangun aplikasi dengan menggunakan *single or multiple database.*

4.3.5 Kesimpulan Hasil Pengujian Beta

Berdasarkan presentase dari hasil pengujian beta sebelumnya, dapat diambil kesimpulan bahwa manfaat aplikasi ini sudah sesuai dengan tujuan yaitu

1. Memudahkan pengguna dalam melakukan koneksi dan akses data dari aplikasi berbasis web ke dalam *database* dengan DBMS yang berbeda.
2. Membantu pengguna dalam melakukan penggabungan data antar *database* dengan DBMS yang berbeda.
3. Memudahkan *developer* atau *programmer* dalam mengembangkan aplikasi berbasis web yang menggunakan *single* atau *multiple* DBMS.