BAB 4
IMPLEMENTASI DAN PENGUJIAN SISTEM

Dalam bab ini akan dilakukan implementasi dan pengujian terhadap sistem, tahapan ini dilakukan setelah proses analisis dan perancangan sistem selesai dilakukan. Pengimplementasian sistem dilakukan dengan bahasa pemrograman *javascript*, yang kemudian dilakukan proses pengujian dengan tujuan evaluasi sistem yang berguna untuk proses pengembangan sistem di kemudian hari.

4.1. Implementasi Sistem

Implementasi merupakan tahap meletakkan sistem beserta komponen pembentuknya hingga siap untuk dioperasikan. Implementasi bertujuan untuk mengkonfirmasi perancangan, sehinga pengguna dapat memberikan masukan berupa saran ataupun komentar kepada peneliti.

Pada implementasi sistem akan dijelaskan kebutuhan perangkat keras, perangkat lunak, basis data, proses, antarmuka, dan tampilan.

4.1.1. Implementasi Perangkat Keras

Komponen perangkat keras yang digunakan untuk implementasi adalah :

a. *Single core processor* 1,0 GHz
b. *Random access memory* 512 MB
c. *Drive storage* 10 GB
d. *Monitor* 13”
e. *Keyboard* dan *mouse*

4.1.2. Implementasi Perangkat Lunak

Komponen perangkat lunak yang digunakan untuk mengiplementasikan sistem adalah :

a. *Operating system* Ubuntu 14.04.1 LTS
b. *Command-prompt shell* Linux Terminal
c. *Database server* MongoDB 2.6.7 dan MySQL 5.5.40
d. *Web-server* NodeJS 0.10.35
e. *Web-browser* Google chrome canary 42.0.2291.0
f. *Text editor* ACE editor

4.1.3. Implementasi Basis Data

Implementasi basis data yang diterapkan dalam sistem ini menggunakan dua *database management service* yang berbeda yaitu MySQL dan MongoDB, pada subbab selanjutnya akan dijelaskan pengimplementasian untuk masing-masing *database management service* tsb.

4.1.3.1. Implementasi *DBMS* MySQL

Implementasi basis data dengan menggunakan DBMS MySQL bertujuan untuk acuan data awal yang digunakan dalam operasional sistem, *database management system* ini diasumsikan telah memiliki data dan struktur tabel dengan bentuk relasional yang berupa *file backup* data penelitian, berikut implemplementasi basis data MySQL yang dilakukan :

a. Menjalankan MySQL server

   Jalankan program *terminal*, gunakan perintah berikut, kemudian tekan tombol "*Enter*".

   ```
   Nama-komputer:~ Pengguna$ /etc/init.d/mysqld start
   ```

b. *Login* kedalam MySQL server

   Masih menggunakan *terminal*, gunakan perintah berikut untuk *login* kedalam Mysql *server* yang sedang berjalan, kemudian tekan tombol "*Enter*".

   ```
   Nama-komputer:~ Pengguna$ mysql –u root –p
   ```

   jika diminta kata sandi masukkan *password* yang anda gunakan untuk nama pengguna "*root*".

c. Membuat *database* dengan nama "*restaurant*"

   Masih menggunakan *terminal*, gunakan perintah berikut, kemudian tekan tombol "*Enter*".

   ```
   mysql> CREATE DATABASE `restaurant`;
   ```

d. Jadikan "*restaurant*" sebagai *workspace database*

   Masih menggunakan *terminal*, gunakan perintah berikut, kemudian tekan tombol "*Enter*".

   ```
   mysql> USE `restaurant`;
   ```

e. Melakukan import data penelitian "restauran.sql"

Masih menggunakan *terminal*, gunakan perintah berikut, kemudian tekan tombol "*Enter*".

```
mysql> source path_to_your/restaurant.sql;
```

4.1.3.2. Implementasi *DBMS* MongoDB

Implementasi basis data dengan menggunakan DBMS MongoDB merupakan *service* manajemen data yang selanjutnya didistribusikan terhadap sistem pada proses diluar penelitian, MongoDB dalam penelitian ini digunakan untuk memproses pemodelan data baru yang berasal dari DBMS MySQL pada proses analisis data. Berikut implementasi yang diterapkan terhadap manajemen basis data ini adalah :

a. Menjalankan MongoDB server

   Jalankan program *terminal*, gunakan perintah berikut, kemudian tekan tombol "*Enter*".

```
Nama-komputer:~ Pengguna$ mongod
```

b. *Login* kedalam MongoDB server

   Masih menggunakan *terminal*, gunakan perintah berikut untuk *login* kedalam Mysql *server* yang sedang berjalan, kemudian tekan tombol "*Enter*".

```
Nama-komputer:~ Pengguna$ mongo
```

c. Membuat *database* dengan nama "*restaurant*"

   Masih menggunakan *terminal*, gunakan perintah berikut, lalu "*Enter*".

```
> use `restaurnat`;
```

   kemudian untuk memastikan database tsb sudah dibuat masukan perintah

```
> db.createCollection('testingKoleksi');
```

4.1.4. Implementasi Konversi Data

Implementasi konversi data dilakukan pada DBMS MySQL ke DBMS MongoDB dengan menggunakan pemodelan yang telah dibuat, dalam penelitian ini proses seluruhnya dilakukan pada sisi *server* (NodeJS). Penjelasan lebih lengkap dijelaskan pada Tabel 4. 1 yang merupakan konteks pemrosesan tertentu dengan menggunakan pengkodean *javascript* yang berupa *file* program dengan melibatkan dependesi *library* diluar sistem, berikut tabel yang dimaksud :

Tabel 4. 1. Implementasi kode program pemrosesan

Pemrosesan	**Nama *class* *(file)***	**Dependensi** (*node package module*)	**Keterangan**
Manajemen *request*	*Socket* (*socket.js*)		Menangani permintaan yang dilakukan *client-side* dalam sistem
Pemodelan	*Model* (*model.js*)		Menangani pemodelan data bentukan dengan metode pengokodean *javascript Object*
Olah data MySQL	*Mysql* (*mysql.js*)	*knex* *native-mysql*	Menangani pengolahan data DBMS MySQL
Olah data MongoDB	*Mongo* (*mongo.js*)	*native-mongodb*	Menangani pengolahan data DBMS MongoDB
Export data	*Export* (*export.js*)		Melakukan export data yang dibuat dengan pemodelan data (*model.js*) kemudian mengambil data yang dibutuhkan melalui MySQL DBMS lalu mejadikannya bentukan data dalam pemodelan MongoDB
Main / Starter	*App* (*app.js*)	*path* *cookie-parser* *body-parser* *morgan* *express.io* *performance-now* *memwatch*	Pemicu jalannya aplikasi, dapat disebut juga sebagai *main program*

4.1.5. Implementasi Antarmuka

Implementasi antarmuka dilakukan berdasarkan perancangan sistem yang telah dijelaskan sebelumnya. Implementasi antarmuka yang dijelaskan dalam subbab ini merupakan yang halaman aplikasi sistem yang berjalan dalam client-side yang dibuat dengan pengkodean dalam bentuk *file* program. Berikut ini

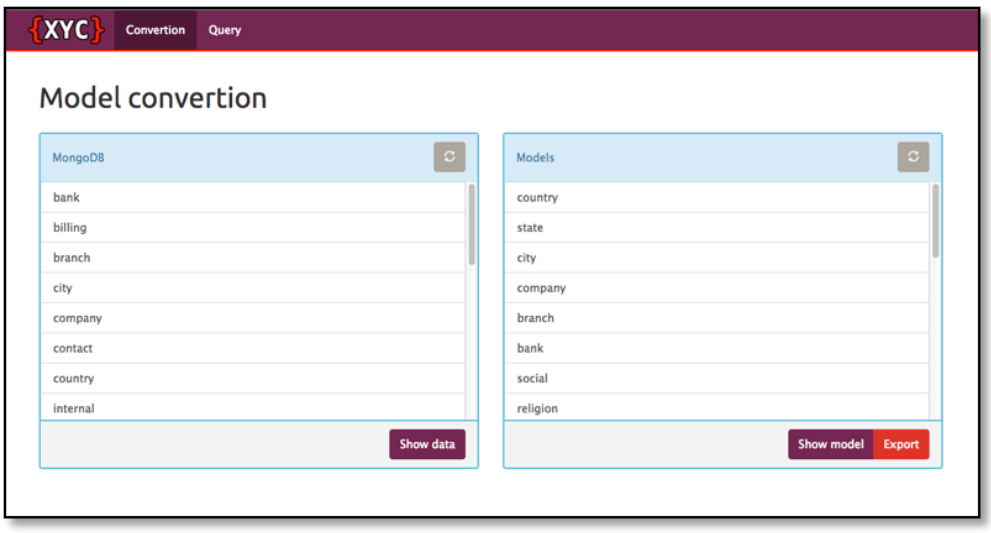
adalah implementasi antarmuka dengan nama *file* yang bersangkutan beserta dependesi *library* yang digunakan, dapat dilihat pada Tabel 4. 2.

Tabel 4. 2. Implementasi kode program antarmuka

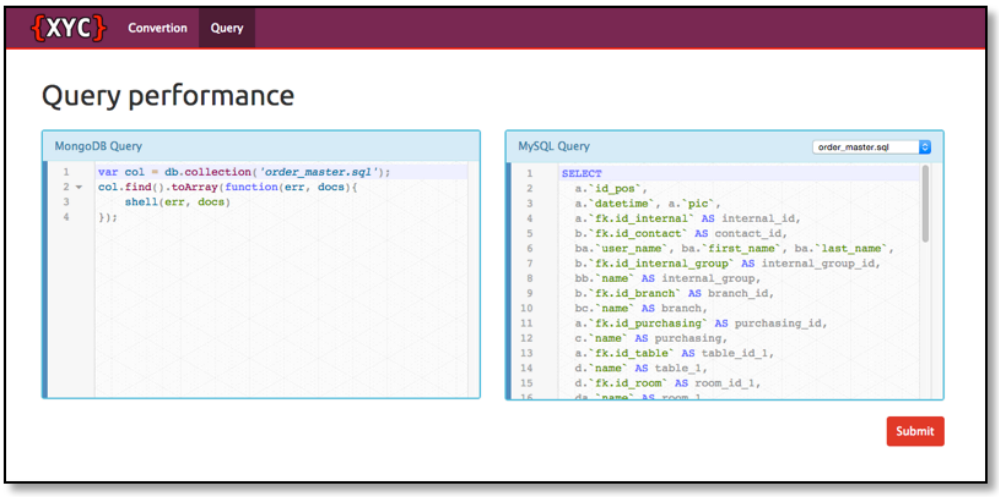
Tampilan	Nama *file*		Dependensi	Keterangan
	View	*Controller class*		
Konversi pemodelan	*convertion.jade*	*ConvertionCtrl* (*convertion.js*)		Halaman yang dapat digunakan untuk konversi / *export* melalui pemodelan data
Performansi *query*	*query.jade*	*QueryCtrl* (*query.js*)		Halaman yang digunakan untuk uji *query* MySQL ataupun MongoDB
Index	*index.jade*		*bootstrap* *bootswatch* *jquery* *chartjs* *underscore* *backbone* *pretty-json* *codemirror*	Template halaman yang digunakan sebagai index atau penampung sub halaman.

4.1.5.1. Tampilan

Untuk memperjelas antarmuka yang diimplementasikan dalam penelitian ini berikut tampilan disajikan pada Gambar 4. 1, Gambar 4. 2, dan Gambar 4. 3 yang merupakan anatarmuka yang diimplementasikan berdasarkan perancangan sistem yang telah dilakukan pada bab sebelumnya :



Gambar 4. 1. Tampilan halaman konversi



Gambar 4. 2. Tampilan halaman performansi *query*



Gambar 4. 3. Tampilan pesan

4.1.6. Implementasi penggunaan *query*

Implemetasi penggunaan *query* diperlukan untuk pengujian performansi perintah *query* yang akan dilakukan pada bagian pengujian, pengimplementasian ini dilakukan setelah implementasi basis data (yang telah dijelaskan pada subbab **4.1.3**) sudah dilakukan lengkap dengan proses konversi data dengan menggunakan teknik pemrograman (dapat dilihat pada subbab **4.1.4**), dan hasil konversi yang telah dilakukan menghasilkan beberapa *collection* yang sudah dianalisis sebelumnya pada subbab **3.1.4**.

Implementasi penggunaan *query* akan disimulasikan dengan menggunakan salah satu skema yaitu *product*. Berdasarkan padanan *query* pada materi subbab **2.4.5.1** yang telah dijelaskan sebelumnya, skema terebut akan disimulasikan untuk penggunaan *query* terhadap :

1. Pemeliharaan data yang meliputi pembuatan (*create*) dan perubahan struktur (*alter*).
2. Pengolahan data yang meliput penambahan (*insert*), pembacaan (*select*), peremajaan (*update*), dan penghapusan (*delete*) data.

4.1.6.1. Implementasi *query* pembuatan struktur (*create*)

Perintah *query* berikut ini akan mensimulasikan pembuatan *collection* pada MongoDB dan tabel pada MySQL lengkap dengan dependensi tabel yang dimiliki, terkait tentang produk beserta harga jual dan harga beli maka dapat diimplementasikan :

MySQL

```
CREATE TABLE `product` (
  `id_product` varchar(30) NOT NULL,
  `fk.id_level` varchar(30) DEFAULT NULL,
  `fk.id_unit` varchar(30) DEFAULT NULL,
  `fk.id_department` varchar(30) DEFAULT NULL,
  `datetime` timestamp NOT NULL DEFAULT '0000-00-00 00:00:00',
  `name` varchar(100) NOT NULL DEFAULT '',
  `recommend` enum('0','1') DEFAULT '0',
  `active` enum('0','1') NOT NULL DEFAULT '0',
  `notes` varchar(100) DEFAULT '',
  `update` timestamp NULL
    DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,
  PRIMARY KEY (`id_product`),
  KEY `fk.id_level` (`fk.id_level`),
  KEY `fk.id_unit` (`fk.id_unit`),
  KEY `fk.id_department` (`fk.id_department`),
  CONSTRAINT `product_ibfk_1` FOREIGN KEY (`fk.id_level`)
    REFERENCES `level` (`id_level`) ON UPDATE CASCADE,
```

```
  CONSTRAINT `product_ibfk_2` FOREIGN KEY (`fk.id_unit`)
    REFERENCES `unit` (`id_unit`) ON UPDATE CASCADE,
  CONSTRAINT `product_ibfk_3` FOREIGN KEY (`fk.id_department`)
    REFERENCES `department` (`id_department`) ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

```
CREATE TABLE `product_purchase_price` (//one to many cardinality
  `id_product_purchase_price` int(11) unsigned NOT NULL AUTO_INCREMENT,
  `fk.id_product` varchar(30) DEFAULT NULL,
  `datetime` timestamp NULL DEFAULT CURRENT_TIMESTAMP,
  `value` double unsigned NOT NULL DEFAULT '0',
  `active` enum('0','1') NOT NULL DEFAULT '0',
  `notes` varchar(100) DEFAULT NULL,
  PRIMARY KEY (`id_product_purchase_price`),
  KEY `fk.id_product` (`fk.id_product`),
  CONSTRAINT `product_purchase_price_ibfk_1` FOREIGN KEY (`fk.id_product`)
    REFERENCES `product` (`id_product`) ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

```
CREATE TABLE `product_sale_price` (//one to many cardinality
  `id_product_sale_price` int(11) unsigned NOT NULL AUTO_INCREMENT,
  `fk.id_product` varchar(30) DEFAULT NULL,
  `datetime` timestamp NULL DEFAULT CURRENT_TIMESTAMP,
  `value` double unsigned NOT NULL DEFAULT '0',
  `active` enum('0','1') NOT NULL DEFAULT '0',
  `notes` varchar(100) DEFAULT NULL,
  PRIMARY KEY (`id_product_sale_price`),
  KEY `fk.id_product` (`fk.id_product`),
  CONSTRAINT `product_sale_price_ibfk_1` FOREIGN KEY (`fk.id_product`)
    REFERENCES `product` (`id_product`) ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

MongoDB

```
var name = "product";
db.createCollection(name);
```

4.1.6.2 Implementasi *query* penambahan data (*insert*)

Perintah *query* berikut ini akan mensimulasikan penambahan *documents* pada *collection* MongoDB dan *rows* pada tabel MySQL, terkait tentang data produk beserta harga jual dan harga beli maka dapat diimplementasikan :

MySQL

```
INSERT INTO `product` (
  `id_product`, `fk.id_level`, `fk.id_unit`, `fk.id_department`,
  `datetime`, `name`, `recommend`, `active`, `notes`, `update`
) VALUES (
  '54e20fb9d0f8380a9b8e6066', '26c7c31a', "pcs", "kitchen",
  CURRENT_TIMESTAMP(), 'Ayam Suwir', '0', '1', NULL, CURRENT_TIMESTAMP()
);
```

```
INSERT INTO `product_purchase_price` (
  `fk.id_product`, `datetime`, `value`, `active`, `notes`
) VALUES (
  '54e20fb9d0f8380a9b8e6066', CURRENT_TIMESTAMP(), 17000, '1', NULL
);
```

```
INSERT INTO `product_sale_price` (
  `fk.id_product`, `datetime`, `value`, `active`, `notes`
) VALUES (
  '54e20fb9d0f8380a9b8e6066', CURRENT_TIMESTAMP(), 34000, '1', NULL
);
```

MongoDB

```
db.product.insert({
  "id_level": "26c7c31a",
  "id_unit": "pcs",
  "id_department": "kitchen",
  "datetime": new Date(),
  "name": "Ayam Suwir Balado",
  "recommend": "1",
  "sale_price": [{ //one to many cardinality, embedded.
      _id: ObjectId(), datetime: new Date(),
      value: 17000, active: "1", notes: null
  }],
  "purchase_price": [{ //one to many cardinality, embedded.
    _id: ObjectId(), datetime: new Date(),
      value: 5399.4, active: "1", notes: null
  }],
  "active": "1",
  "notes": null
});
```

4.1.6.3. Implementasi *query* pembacaan data (*select*)

Perintah *query* berikut ini akan mensimulasikan pembacaan *documents* pada *collection* MongoDB dan *rows* pada tabel MySQL, terkait tentang produk beserta harga jual dan harga beli maka dapat diimplementasikan pembacaan data tanpa kriteria dan dengan kriteria parameter harga jual dan harga beli terakhir untuk setiap produk :

MySQL

```
SELECT product.*, sale_price.*, purchase_price.*
FROM product
LEFT JOIN product_sale_price AS sale_price
  ON product.`id_product` = sale_price.`fk.id_product`
LEFT JOIN product_purchase_price AS purchase_price
  ON product.`id_product` = purchase_price.`fk.id_product`;
```

```
SELECT product.*, sale_price.*, purchase_price.*
FROM product
LEFT JOIN (
  SELECT * FROM (
    SELECT * FROM product_sale_price
    WHERE active = "1"
    ORDER BY `datetime` DESC
  ) AS x GROUP BY `fk.id_product`
) AS sale_price ON product.`id_product` = sale_price.`fk.id_product`
LEFT JOIN (
  SELECT * FROM (
    SELECT * FROM product_purchase_price
    WHERE active = "1"
    ORDER BY `datetime` DESC
  ) AS x GROUP BY `fk.id_product`
) AS purchase_price
  ON product.`id_product` = purchase_price.`fk.id_product`;
```

MongoDB

```
db.product.find();
```

```
db.product.aggregate( //aggregate is advance .find() function;
  { $unwind: '$sale_price' },
```

```
  { $match: { 'sale_price.active': "1", 'purchase_price.active': "1" } },
  { $sort: { 'sale_price.datetime': -1, 'sale_price.datetime': -1  } }
);
```

4.1.6.4. Implementasi *query* peremajaan data (*update*)

Perintah *query* berikut ini akan mensimulasikan peremajaan documents pada *collection* MongoDB dan *rows* pada tabel MySQL, terkait tentang produk beserta harga jual dan harga beli maka dapat diimplementasikan perubahan data *product* dengan kriteria *id_product*, dan *name* tertentu dengan peremajaan *id_product*, dan *name*. Kemudian setelah itu perubahan juga dilakukan pada *sale_price* untuk kriteria *fk.id_product* dan *id_sale_price* tertentu dengan peremajaan *value* :

MySQL

```
UPDATE `product_sale_price ` SET
  `value` = 7800
WHERE
  `fk.id_product` = "54e20fb9d0f8380a9b8e6066" AND
  `id_sale_price` = "54e20fb9d0f8380a9b8e6064";
```

```
UPDATE `product` SET
  `id_product` = "c0d3pr0d03ct"
WHERE
  id_product = "54e20fb9d0f8380a9b8e6066" AND
  name LIKE "Ayam Suwir%";
```

MongoDB

```
db.product.update(
  {
    "_id" : ObjectId("54e20fb9d0f8380a9b8e6066"),
    "name" : /Ayam Suwir/,
    "sale_price._id" : ObjectId("54e20fb9d0f8380a9b8e6064")
  },
  { $set: {
      "_id" : "c0d3pr0d03ct",
      "name" : "Ayam Suwir Pedas",
      "sale_price.$.value" : 7800
  }},
  { multi: true }
);
```

4.1.6.5. Implementasi *query* penghapusan data (*delete*)

Perintah *query* berikut akan mensimulasikan penghapusan *documents collection* MongoDB dan *rows* tabel MySQL, terkait tentang produk beserta harga jual dan harga beli maka dapat diimplementasikan perubahan data *product* dengan kriteria *id_product*, dan *name* tertentu. Kemudian setelah itu perubahan juga dilakukan pada *sale_price* untuk kriteria *fk.d_product* dan *id_sale_price* tertentu:

MySQL

```
DELETE FROM `product_sale_price ` WHERE
  `fk.id_product` = "54e20fb9d0f8380a9b8e6066" AND
  `id_sale_price` = "54e20fb9d0f8380a9b8e6064";
DELETE FROM `product` WHERE
  id_product = " c0d3pr0d03ct " AND
  name LIKE "Ayam Suwir Ped%";
```

MongoDB

```
db.product.remove({
    "_id" : "c0d3pr0d03ct",
    "name" : /Ayam Suwir Ped/,
    "sale_price._id" : ObjectId("54e20fb9d0f8380a9b8e6064")
});
```

4.1.6.6. Implementasi *query* perubahan struktur (*alter*)

Perintah query berikut ini akan mensimulasikan perubahan struktur *collection product* MongoDB dan tabel *product* MySQL dengan kriteria :

a. Menambahan *field/column* dengan nama *note*, kemudian
b. Menambah *index* pada *field/column* dengan nama *notes*, dan *active*, kemudian
c. Menghapus *index* pada *field/colum notes*, kemudian
d. Menghapus *field/colum notes*, dan terakhir
e. Merubah *field/column note* menjadi *notes*

Berikut pengimplementasian *query* perubahan struktur berdasarkan kriteria yang dimaksud pada masing-masing DBMS adalah :

MySQL

```
ALTER TABLE `product`
  ADD COLUMN `note` VARCHAR(100) DEFAULT 'aku notes' AFTER `notes`;
```

```
ALTER TABLE `product`
  ADD INDEX `active` (`active`),
  ADD INDEX `notes` (`notes`);
```

```
ALTER TABLE `product`
  DROP INDEX `notes`;
```

```
ALTER TABLE `product`
  DROP COLUMN `notes`;
```

```
ALTER TABLE `product`
  CHANGE `note` `notes` VARCHAR(100) DEFAULT 'aku notes';
```

MongoDB

```
db.product.update({}, {
  $set : { "note": "aku notes"}
}, { multi: true });
```

```
db.product.ensureIndex({active: 1}, {unique: false});
db.product.ensureIndex({notes: 1}, {unique: false});
```

```
db.product.dropIndex("idx_notes");
```

```
db.product.update({}, {
  $unset : { "notes": 1 },
  $rename: { "note": "notes" }
}, {multi: true});
```

4.2. Pengujian

Pengujian dilakukan untuk menguji sistem yang telah dibangun, dalam penelitian ini dilakukan tahap pengujian performansi, dan pengujian *black box*.

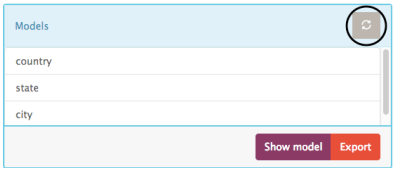
4.2.1. Pengujian *Black box*

Metode pengujian pertama yang digunakan adalah metode *black box*, metode ini berfokus pada persyaratan fungsional dari perangkat lunak yang dibangun. Berikut rincian pengujian *black box* berdasarkan fungsionalitas yang terdapat pada sistem :

1. Muat ulang daftar model

Pengujian fungsionalitas *muat ulang daftar model* dilakukan untuk mengetahui keluaran dari masukan yang dilakukan pengguna, fungsionalitas ini terdapat pada halaman *Convertion.* Rincian terhadap proses ini disajikan pada Tabel 4. 3.

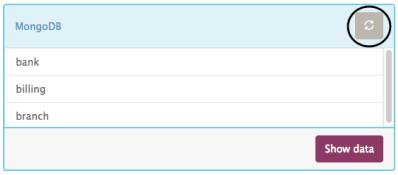
Tabel 4. 3. Hasil pengujian muat ulang daftar model

Antarmuka	
Data masukan benar	
Masukan pengguna	Klik tombol logo muat ulang daftar model
Harapan	Memuat ulang daftar model
Pengamatan	Daftar model berhasil dimuat ulang
Kesimpulan	Diterima

2. Muat ulang daftar *collection*

Pengujian fungsionalitas *muat ulang daftar collection* dilakukan untuk mengetahui keluaran dari masukan yang dilakukan pengguna, fungsionalitas ini terdapat pada halaman *Convertion.* Rincian terhadap proses ini disajikan pada Tabel 4. 4.

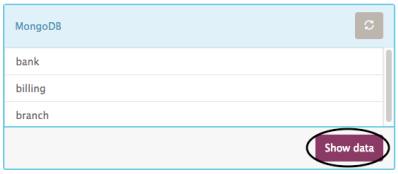
Tabel 4. 4. Hasil pengujian muat ulang daftar *collection*

Antarmuka	
Data masukan benar	
Masukan pengguna	Klik tombol logo muat ulang daftar *collection* MongoDB
Harapan	Memuat ulang daftar *collection* MongoDB
Pengamatan	Daftar *collection* berhasil dimuat ulang
Kesimpulan	Diterima

3. Lihat data *collection* MongoDB

Pengujian fungsionalitas *lihat data collection MongoDB* dilakukan untuk mengetahui keluaran dari masukan yang dilakukan pengguna, fungsionalitas ini terdapat pada halaman *Convertion.* Rincian terhadap proses ini disajikan pada Tabel 4. 5.

Tabel 4. 5. Hasil pengujian lihat data *collection* MongoDB

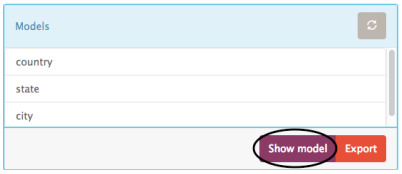
Antarmuka	
Data masukan benar	
Masukan pengguna	1. Memilih salah satu *collection* pada daftar *collection* MongoDB. 2. Klik tombol *show data*.
Harapan	Menampilkan pesan yang berisi data suatu *collection* MongoDB
Pengamatan	Data *collection* MongoDB berhasil ditampilkan dalam pesan
Kesimpulan	Diterima
Data masukan salah	
Masukan pengguna	1. Tidak memilih salah satu *collection* pada daftar *collection* MongoDB. 2. Klik tombol *show data*.
Harapan	Tidak menampilkan pesan yang berisi data suatu *collection* MongoDB
Pengamatan	Tidak ada respon
Kesimpulan	Diterima

4. Lihat model

Pengujian fungsionalitas *lihat model* dilakukan untuk mengetahui keluaran dari masukan yang dilakukan pengguna, fungsionalitas ini terdapat

pada halaman *Convertion*. Rincian terhadap proses ini disajikan pada Tabel 4. 6.

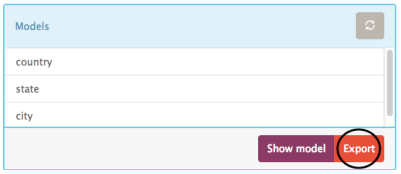
Tabel 4. 6. Hasil pengujian lihat model

Antarmuka	
Data masukan benar	
Masukan pengguna	1. Memilih salah satu model pada daftar model. 2. Klik tombol *show model*.
Harapan	Menampilkan pesan yang berisi data pemodelan MongoDB
Pengamatan	Data pemodelan berhasil ditampilkan dalam pesan
Kesimpulan	Diterima
Data masukan salah	
Masukan pengguna	1. Tidak memilih salah satu model pada daftar model. 2. Klik tombol *show model*.
Harapan	Tidak menampilkan pesan yang berisi data pemodelan MongoDB
Pengamatan	Tidak ada respon
Kesimpulan	Diterima

5. *Export* model

Pengujian fungsionalitas *export model* dilakukan untuk mengetahui keluaran dari masukan yang dilakukan pengguna, fungsionalitas ini terdapat pada halaman *Convertion*. Rincian proses ini disajikan pada Tabel 4. 7.

Tabel 4. 7. Hasil pengujian *export* model

Antarmuka	
Data masukan benar	
Masukan pengguna	1. Memilih salah satu model pada daftar model. 2. Klik tombol *export*.
Harapan	Menampilkan pesan yang berupa data hasil konversi *collection* MongoDB
Pengamatan	Data data hasil konversi *collection* MongoDB berhasil ditampilkan dalam pesan
Kesimpulan	Diterima
Data masukan salah	
Masukan pengguna	1. Tidak memilih salah satu model pada daftar model. 2. Klik tombol *show data*.

Harapan	Tidak menampilkan pesan yang berupa data hasil konversi *collection* MongoDB
Pengamatan	Tidak ada respon
Kesimpulan	Diterima

6. Performansi *query*

Pengujian fungsionalitas *performansi query* dilakukan untuk mengetahui keluaran dari masukan yang dilakukan pengguna, fungsionalitas ini terdapat pada halaman *Convertion.* Rincian terhadap proses ini disajikan pada Tabel 4. 8.

Tabel 4. 8. Hasil pengujian performansi *query*

Antarmuka	
Data masukan benar	
Masukan pengguna	1. Mengisi *query editor* MongoDB 2. Mengisi *query editor MySQL* 3. Klik tombol *submit*.
Harapan	Menampilkan pesan yang berisi respon *query* dari masing-masing DBMS
Pengamatan	Respon *query* dari masing-masing DBMS berhasil ditampilkan
Kesimpulan	Diterima
Data masukan salah (1)	
Masukan pengguna	1. Tidak mengisi semua *query editor* baik (MongoDB/MySQL) 2. Klik tombol *submit*
Harapan	Tidak menampilkan pesan
Pengamatan	Tidak ada respon
Kesimpulan	Diterima
Data masukan salah (2)	
Masukan pengguna	1. Hanya mengisi salah satu *query editor* (MongoDB/MySQL) 2. Klik tombol *submit*
Harapan	Menampilkan pesan yang berisi respon *query* dari *query editor* DBMS yang diisi
Pengamatan	Respon *query* dari *query editor* DBMS yang diisi berhasil ditampilkan.
Kesimpulan	Diterima
Data masukan salah (3)	
Masukan pengguna	1. Mengisi perintah *query* yang memang tidak *valid* baik dalam *query editor* MongoDB ataupun MySQL 2. Klik tombol *submit*
Harapan	Menampilkan pesan yang berisi respon *query* dari *query editor* DBMS yang memang *valid*
Pengamatan	Respon *query* dari *query editor* DBMS yang memang *valid* berhasil ditampilkan.
Kesimpulan	Diterima

4.2.1.1. Kesimpulan Pengujian *Alpha*

Berdasarkan hasil dari pengujian *black box*, dapat disimpulkan bahwa funsgsionalitas sistem sudah berjalan baik, meski demikian tidak dapat dipastikan bahwa sistem sudah bebas dari kesalahan sehingga diperlukan pengamatan lebih lanjut, khususnya selama proses operasional.

4.2.2. Pengujian Performansi

Pengujian pertama yang direncanakan untuk menguji sistem adalah pengujian performansi, pengujian ini menitik-beratkan komparasi terhadap masing-masing DBMS, dalam pengujian performansi dilakukan juga rencana pengujian yang dapat dilihat pada Tabel 4. 9.

Tabel 4. 9. Rencana pengujian performansi

No	Pengujian	Detil pengujian	
		Deskripsi	**Objek model**
1	*Create*	Membuat tabel atau *collection* beserta *constraint* (jika ada) untuk setiap objek model	Dilakukan pada DBMS MongoDB dan MySQL dengan ukuran informasi yang berskala sama (model data *flat*) yaitu model harga jual produk, harga beli produk, pemesanan (*master*), pemesanan (*detail*) dan rekap pendapatan (*billing*). Pengujian menitik-beratkan pada performa pemrosesan data dalam ukuran waktu dengan satuan mili-detik
2	*Insert*	Menambah data tabel atau *collection* beserta data *constraint* (jika ada) untuk setiap objek model. Pemrosesan tambah data yang digunakan dalam 1x uji adalah satu baris data.	
3	*Select*	Membaca data untuk setiap objek model.	
4	*Update*	Meremajakan data tabel atau *collection* beserta data *constraint* (jika ada) untuk setiap objek model. Pemrosesan tambah data yang digunakan dalam 1x uji adalah satu baris data.	
5	*Alter*	Meremajakan salah satu struktur tabel atau *collection* setiap objek model. Pemrosesasan ini dilakukan penambahan *column/field*, penambahan *index*, penghapusan *index*, penghapusan *column/field*, dan perubahan *column/field*.	
6	*Delete*	Penghapusan data untuk setiap objek model.	

4.2.2.1. Hasil Pengujian Performansi

Berdasarkan rencana pengujian yang telah disusun, maka dapat dilakukan pengujian sebagai berikut :

1. *Create query*, berikut hasil pengujian terhadap pemrosesan *create query* dapat dilihat pada Tabel 4.10 :

4.10. Hasil pengujian *create query*

Objek model		**Harga beli produk**		**Harga jual produk**		**Pemesanan (*master*)**		**Pemesanan (*detail*)**		**Rekap pendapatan (*billing*)**	
DBMS		**MongoDB**	**MySQL**	**MongoDB**	**MySQL**	**MongoDB**	**MySQL**	**MongoDB**	**MySQL**	**MongoDB**	**MySQL**
Pengamatan (*ms*)	**1**	2.83	5.02	2.78	16.58	2.85	32.61	3.00	38.51	2.83	23.23
	2	1.80	3.03	1.96	12.15	2.16	28.96	2.90	29.28	2.99	19.82
	3	1.57	3.06	1.59	13.06	1.66	25.16	2.64	32.54	2.65	12.44
	4	1.77	5.81	2.69	14.91	2.21	26.71	2.80	30.45	2.81	14.50
	5	1.56	3.03	1.70	12.33	1.56	21.86	2.99	28.91	2.64	16.19
	6	2.02	3.11	1.57	13.94	1.77	26.95	2.77	26.64	2.56	18.67
	7	1.56	5.15	1.62	13.13	1.64	26.87	3.12	32.85	2.62	10.67
	8	1.59	3.37	1.79	12.37	1.77	28.49	2.62	28.33	2.74	19.03
	9	1.66	3.48	1.71	15.14	1.75	31.72	2.80	19.61	2.83	21.49
	10	1.56	4.04	1.50	14.73	1.56	25.05	3.59	24.20	2.69	18.94
Rata-rata (*ms*)		**1.79**	**3.91**	**1.89**	**13.83**	**1.89**	**27.44**	**2.92**	**29.13**	**2.74**	**17.50**
Rows		n		n		n		n		n	
Table*/*collection		1	1	1	1	1	1	1	1	1	1
Kesimpulan		Diterima		Diterima		Diterima		Diterima		Diterima	

2. *Insert query*, berikut hasil pengujian terhadap pemrosesan *insert query* dapat dilihat pada Tabel 4.11 :

Tabel 4.11. Hasil pengujian *insert query*

Objek model		**Harga beli produk**		**Harga jual produk**		**Pemesanan (*master*)**		**Pemesanan (*detail*)**		**Rekap pendapatan (*billing*)**	
DBMS		**MongoDB**	**MySQL**	**MongoDB**	**MySQL**	**MongoDB**	**MySQL**	**MongoDB**	**MySQL**	**MongoDB**	**MySQL**
Pengamatan (*ms*)	**1**	13.25	1004.01	21.60	892.21	73.80	626.56	86.37	9257.20	13.37	769.71
	2	4.86	461.98	14.38	641.78	51.07	463.52	65.03	807.21	14.58	614.88
	3	4.91	452.22	1.76	477.13	34.78	448.68	75.32	616.92	14.77	472.00
	4	5.97	529.71	15.64	815.30	35.95	372.96	56.17	416.02	15.57	525.12
	5	5.62	422.72	15.65	505.99	44.90	486.39	45.81	443.20	15.58	648.44
	6	7.71	463.91	17.23	981.29	37.75	511.01	58.24	628.15	16.85	467.85
	7	5.33	508.25	15.39	467.99	26.08	555.16	55.65	571.24	14.67	455.78
	8	4.42	390.39	14.31	602.61	44.29	422.38	64.80	348.79	14.10	399.97
	9	5.21	455.98	15.50	814.21	55.52	446.19	45.14	457.06	15.31	429.36
	10	5.48	633.96	15.82	468.14	35.73	354.32	45.93	475.62	15.24	433.61
Rata-rata (*ms*)		**6.28**	**532.31**	**14.73**	**666.67**	**43.99**	**468.72**	**59.85**	**1402.14**	**15.00**	**521.67**
Rows		0		0		0		0		0	
Table*/*collection		1	1	1	1	1	1	1	1	1	1
Kesimpulan		Diterima		Diterima		Diterima		Diterima		Diterima	

3. *Select query*, berikut hasil pengujian terhadap pemrosesan *select query* dapat dilihat pada Tabel 4.12 :

Tabel 4.12. Hasil pengujian *select query*

Objek model		**Harga beli produk**		**Harga jual produk**		**Pemesanan (*master*)**		**Pemesanan (*detail*)**		**Rekap pendapatan (*billing*)**	
DBMS		**MongoDB**	**MySQL**	**MongoDB**	**MySQL**	**MongoDB**	**MySQL**	**MongoDB**	**MySQL**	**MongoDB**	**MySQL**
Pengamatan (*ms*)	**1**	12.35	2046.22	12.50	7166.89	18.51	5402.69	43.45	428.25	12.11	2194.93
	2	5.30	1709.31	14.78	1369.31	8.47	3382.75	24.03	630.08	5.23	2528.95
	3	6.36	1351.66	16.32	974.13	24.20	2894.29	24.39	319.72	5.80	2728.92
	4	24.99	1770.25	28.15	2089.06	24.83	456.46	26.46	1083.62	24.65	1675.51
	5	4.70	1271.28	15.23	1405.83	20.88	4864.88	26.57	340.91	6.62	1865.11
	6	20.33	1603.47	21.90	1552.01	24.04	3355.76	30.08	401.17	21.10	2312.89
	7	4.49	1350.82	14.19	1479.24	22.68	4382.46	22.11	295.21	4.22	1586.37
	8	19.38	1421.81	18.25	1554.13	23.17	5415.97	34.43	253.81	4.36	1933.81
	9	4.90	1364.72	14.90	1560.42	22.55	4022.68	25.73	295.32	20.73	2270.63
	10	20.16	1365.64	19.13	1471.64	21.92	3127.36	26.22	354.11	4.22	2871.53
Rata-rata (*ms*)		**12.29**	**1525.52**	**17.53**	**2062.26**	**21.12**	**3730.53**	**28.35**	**440.22**	**10.90**	**2196.86**
Rows		n		n		n		n		n	
Table*/*collection		1	1	1	1	1	1	1	1	1	1
Kesimpulan		Diterima		Diterima		Diterima		Diterima		Diterima	

4. *Update query*, berikut hasil pengujian terhadap pemrosesan *update query* dapat dilihat pada Tabel 4.13 :

Tabel 4.13. Hasil pengujian *update query*

Objek model		**Harga beli produk**		**Harga jual produk**		**Pemesanan (*master*)**		**Pemesanan (*detail*)**		**Rekap pendapatan (*billing*)**	
DBMS		**MongoDB**	**MySQL**	**MongoDB**	**MySQL**	**MongoDB**	**MySQL**	**MongoDB**	**MySQL**	**MongoDB**	**MySQL**
Pengamatan (*ms*)	**1**	15.95	17.06	10.25	7.76	888.62	403.06	455.60	451.40	4.93	58.57
	2	5.87	10.19	5.11	8.04	299.62	228.53	415.91	433.97	0.88	38.25
	3	4.79	6.87	5.01	6.62	303.70	237.72	434.59	446.18	0.51	39.98
	4	3.61	5.73	4.15	5.95	315.36	213.25	413.84	435.88	0.61	42.98
	5	3.98	4.99	3.47	4.35	310.32	235.37	429.69	465.48	1.72	52.17
	6	4.43	6.53	3.73	4.87	308.37	220.79	497.39	474.62	0.57	41.38
	7	5.98	6.45	4.50	6.08	297.43	215.70	445.09	443.81	0.48	39.46
	8	4.35	5.63	4.03	6.70	312.60	228.51	426.12	451.60	0.50	35.43
	9	4.72	6.08	4.42	4.82	323.89	244.26	429.39	446.19	0.54	38.15
	10	4.44	6.39	5.13	7.29	288.61	209.91	437.88	474.88	1.07	40.65
Rata-rata (*ms*)		**5.81**	**7.59**	**4.98**	**6.25**	**364.85**	**243.71**	**438.55**	**452.40**	**1.18**	**42.70**
Rows		337		300		8133		32424		6	
Table*/*collection		1	2	1	2	1	9	1	2	1	9
Kesimpulan		Diterima		Diterima		Diterima		Diterima		Diterima	

5. *Alter query*, berikut hasil pengujian terhadap pemrosesan *alter query* dapat dilihat pada Tabel 4.14 :

Tabel 4.14. Hasil pengujian *alter query*

Objek model		**Harga beli produk**		**Harga jual produk**		**Pemesanan (*master*)**		**Pemesanan (*detail*)**		**Rekap pendapatan (*billing*)**	
DBMS		**MongoDB**	**MySQL**	**MongoDB**	**MySQL**	**MongoDB**	**MySQL**	**MongoDB**	**MySQL**	**MongoDB**	**MySQL**
Pengamatan (*ms*)	**1**	3.54	19.48	3.21	18.73	6.62	56.54	6.27	91.78	3.24	82.86
	2	1.08	8.00	0.92	13.25	1.49	15.88	1.15	48.80	0.97	106.89
	3	0.96	8.60	0.75	8.21	0.78	44.57	1.15	53.27	1.17	71.62
	4	1.04	8.85	0.64	8.64	0.89	53.80	0.88	44.39	0.63	47.06
	5	0.97	8.03	0.69	8.02	1.73	36.19	1.60	49.91	1.29	49.07
	6	0.65	8.71	0.68	8.95	0.85	60.87	1.51	87.75	0.67	45.81
	7	0.76	8.58	0.93	8.55	1.71	34.51	1.21	43.54	0.90	47.16
	8	3.25	14.56	0.69	12.36	0.83	30.13	10.41	88.22	0.59	43.53
	9	0.88	7.72	1.18	8.45	0.88	32.41	1.05	94.92	2.68	48.85
	10	0.64	7.48	0.76	9.06	0.79	29.88	1.11	47.61	0.77	111.29
Rata-rata (*ms*)		**1.37**	**10.00**	**1.04**	**10.42**	**1.66**	**39.48**	**2.63**	**65.02**	**1.29**	**65.41**
Rows		1		1		1		1		1	
Table*/*collection		1	4	1	5	1	15	1	21	1	27
Kesimpulan		Diterima		Diterima		Diterima		Diterima		Diterima	

6. *Delete query*, berikut hasil pengujian terhadap pemrosesan *delete query* dapat dilihat pada Tabel 4.15 :

Tabel 4.15. Hasil pengujian *delete query*

Objek model		**Harga beli produk**		**Harga jual produk**		**Pemesanan (*master*)**		**Pemesanan (*detail*)**		**Rekap pendapatan (*billing*)**	
DBMS		**MongoDB**	**MySQL**	**MongoDB**	**MySQL**	**MongoDB**	**MySQL**	**MongoDB**	**MySQL**	**MongoDB**	**MySQL**
Pengamatan (*ms*)	**1**	5.71	1109.39	6.64	1754.47	7.29	4917.67	9.38	4094.07	7.56	5571.99
	2	3.77	1489.49	4.20	1208.75	3.86	4077.83	4.07	4198.54	3.57	6903.17
	3	3.57	1018.57	4.03	1033.52	4.40	4180.77	4.83	4205.29	3.90	7769.41
	4	4.13	1088.52	4.94	1681.29	4.46	4727.71	3.81	4162.85	5.45	6558.96
	5	3.86	1106.82	3.60	1186.74	4.43	4200.94	4.07	4124.07	3.45	6568.41
	6	3.64	1147.88	3.81	1335.48	4.79	4695.90	3.76	4119.87	3.82	6846.13
	7	3.43	1398.38	3.82	1278.13	4.68	4495.68	5.46	4092.37	4.06	7546.43
	8	3.18	1079.15	5.15	1037.66	3.56	3815.30	3.53	4442.36	3.12	6610.08
	9	3.47	2110.14	3.72	995.88	4.20	4134.48	4.99	9935.06	3.54	9585.70
	10	26.36	1208.43	4.49	1289.47	6.37	3994.69	3.41	9526.76	5.04	7678.61
Rata-rata (*ms*)		**6.11**	**1275.68**	**4.44**	**1280.14**	**4.80**	**4324.10**	**4.73**	**5290.13**	**4.35**	**7163.89**
Rows		0		0		0		0		0	
Table/collection		1	5	1	5	1	19	1	21	1	34
Kesimpulan		Diterima		Diterima		Diterima		Diterima		Diterima	

4.2.2.2. Kesimpulan Hasil Pengujian Performansi

Berdasarkan hasil pengujian terhadap performansi *query* pada subbab sebelumnya dapat ditarik kesimpulan bahwa pada proses pemeliharaan dan pengolahan data MongoDB relatif lebih unggul dalam hal kecepatan pemrosesan *query* dibandingkan dengan RDBMS MySQL, terkecuali pada proses pembacaan data, MongoDB akan menghasilkan waktu yang relatif lebih lama dibandingkan MySQL seiring kebutuhan baris data yang besar, MongoDB akan efektif jika melakukan pembacaan data yang sedikit.