Implementation of ElasticSearch Search Engine on Order Management System Data

Devi Fitrianah
Faculty of Computer Science
Universitas Mercu Buana,
Jakarta, Indonesia

Thomson Palito Napitupulu
Faculty of Computer Science
Universitas Mercu Buana,
Jakarta, Indonesia

Umniy Salamah
Faculty of Computer Science
Universitas Mercu Buana,
Jakarta, Indonesia

ABSTRACT
Elasticsearch is a search engine, generally searching full-text formatted data. It organizes the data and makes them more accessible. Elasticsearch is built with Java programming language, open-source, and under the Apache license. Elasticsearch is utilized technically by querying searched keywords, which communicated via API. It is installed in a standalone database server using HTTP/JSON protocol, retrieved and stored data in optimized form. It becomes a reliable technology in today's IT industries that needs optimization in searching full-text formatted data. The bigger size of data, the slower the accessibility, meantime, recent requirements need faster access for very large transaction. The research discussed the influence of adding Elasticsearch in web-based system and the non-adding Elasticsearch. The study implemented Agile Scrum methodology in developing the system. The result of this study is the data access becomes faster by 10.01% when implemented Elasticsearch.

Keywords
Elasticsearch; MongoDB; Replica Set; PHP

1. INTRODUCTION
Some companies have implemented Elasticsearch search engines to improve the performance of their systems. Elasticsearch relates to optimized index storage by Lucene and the actual algorithm for text matching. Elasticsearch is run using Application Programming Interface (API) method which has high scalability. The Elasticsearch installation on the server is quite simple although some configurations need to adjust the environment.

The Order Management System which is the observing object of the Elasticsearch application is a system used by users such as online merchants or wholesalers or merchants. Order Management System that runs at this time has applied NoSQL as their transaction database. In day-to-day operations, the company has not met the satisfaction in the performance of system usage, so it needs to be done from the system that has been running to improve the performance of the system. Based on the above conditions, then conducted a study titled "Implementation Elasticsearch Search Engine on Order Management System". With this implementation, it is expected to facilitate the company in conducting activities related to user order management.

2. LITERATURE VIEW
In this section, we give some reviews on Elasticsearch experiment to compare the system's performance of some NoSQL and SQL databases on a single machine, and conceptual working of Elasticsearch.

Presented benchmarking an experimental evaluation of the four NoSQL Systems using various workloads, they are ElasticSearch, MongoDB, Redis and OrientDB. By using Yahoo Cloud Serving Benchmark (YCSB), they vary the parameters that affect the performance of the databases execution time. Their work do not consider the implementation of Elasticsearch in real industrial system (Abubakar, Adeyi, & Auta, 2014).

Presented theory and working of Elasticsearch. They also presented benefits of Amazon Elasticsearch service. Their work is limited to conceptual working of Elasticsearch only (Gupta & Nair, 2016).

Search engines are a practical application of information retrieval techniques to large-scale text collections. Web search engines, such as Google and Yahoo!, must be able to capture a number of terabytes of data, and then provide sub-second response times to the millions of queries that are sent daily from around the world.

Open source search engines are another important system class that has somewhat different design goals than commercial search engines. The three systems of interest are Lucene, Lemur, and Galago. Lucene is a popular Java-based search engine and has been used for a variety of commercial applications (Croft, Metzler, & Strohman, 2015). Elasticsearch is an open-source search engine built on Apache Lucene ™, a full-text search engine library. Lucene can be said to be the latest, high performing, full-featured, and full-featured search engine library (Gormley & Tong, 2015). Some of the concepts that come with Elasticsearch that can help a full understanding of how elasticsearch works can be described as follows (Gupta & Nair, 2016):

1. Index
   Elasticsearch uses the library's Apache Lucene library to write and read data from the index. The elasticsearch index may be built on more than one Apache Lucene index using "Shards".

2. Documents
   Document is the main entity in the world of elasticsearch. The document consists of fields, and each field is identified by its name and may contain one or more values.

3. Type
   Each document in Elasticsearch has a defined type. It is possible to store different types of documents in one index and different mappings for different document types.

4. Mapping
   All documents are analyzed before being indexed. Text input is divided into tokens, which tokens should be filtered out, or what additional processes, such as deleting HTML tags, are needed. This is where the mapping stage begins to play; it holds all the information about the chain

of analysis.

A way to improve the performance performed on Elasticsearch is to do the data denormalization in the index. By having a copy of the data in each document, it can reduce the need for a join among indexes. The advantage of data denormalization is speed. This is because each document contains all the necessary information in determining whether the data found matches what is searched in query form so there is no need for a complex join index (Gormley & Tong, 2015).

3. OBJECTIVE

The aim of this study is to improve the processing performance and appearance of data on the system. Besides, to keep the maintainability of data on the system to easily adapt to system changes.

4. RESEARCH METHODOLOGY

4.1 Data Collection Technique

Methods of data collection conducted in this study is by observation (direct observation in the environment of Information Technology division and Operational division), interviews (conducting question and answer with staff on related division), and literature study (data collection through books, e-books and journals related with research).

4.2 System Development Method

The method used to build this system is the Agile Scrum Model. This model is an approach to systematic software development, with several steps, namely: Planning, Design, Coding, and Testing.

5. ANALYSIS

5.1 Current System Analysis

The running business process system can be explained as follows:

1. The customer wraps the item want to send.
2. The customer enters the information of the goods to be sent in one Excel file and sends it by e-mail to the Agent.
3. Agent receives email and inserts item information from Excel file into system.
4. Driver picks up goods from customer and will update the status of goods through mobile apps provided.
5. Items are then delivered to the Shipper warehouse.
6. Agent will check-in goods delivered into the system and sort by logistics type. Goods are divided into 2 batches (day and night).
7. Once the item has been sorted, Agent will checkout the goods.
8. Driver will deliver the goods that have been batched with AWB list to the logistics.
9. The logistics party will register the goods into their system and receive the AWB number.
10. Agent will update order tracking status of goods.

5.2 Target System Analysis

Based on the analysis on the system running, then the system development on order management will be built with the following specifications:

1. Agent required to login first before making new order, see order until update order status.
2. To create a new order, the agent must fill all the necessary data into the system before making the process of taking the goods.
3. Agent will also update the status of order and airwaybill number (awb) into the system when knowing the change of order delivery status and also when it has received airwaybill number.
4. Agent is also able to see the entire order data that goes online through the system.
5. On the system will be implemented Elasticsearch search engine to improve performance in online data order search through the system.

5.3 User System Analysis

Users who will be involved in the system as well as its role in the system is Agent: User who has the task to create order, view order, checkin order, checkout order, and input AWB number.

5.4 Database Design

The database design of the system can be seen in the following table.

Table 1. List of Table on Database

#	**Tables Name on Database**
1	Agent
2	Backend_users
3	Role
4	City
5	Drivers
6	Logistic
7	Province
8	Order_tracking
9	Order
10	Rate
11	Rateshow
12	Track_status
13	Address
14	Users
15	Logistic_status
16	Suburb
17	Area
18	Route
19	Pricing

6. RESULT ON TESTING

6.1 Implementation Environment

In the implementation of elasticsearch on the order management system, hardware and software are required, in which hardware specifications can be explained as follows:

1. Processor Intel Core i3
2. RAM 4GB
3. Harddisk 500GB

While the software specifications used are as follows:

1. Operating system Windows 10
2. System development tools Microsoft Visual Studio Code.
3. The programming language PHP CI.
4. The database application program My SQL and MongoDB
5. Web browser Google Chrome.

6.2 Sprint 1

Here are sprint backlogs, system interfaces, and burndown chart results in Sprint 1.

Backlog Item	*Story Points (1 point = 2 hours)*	*Original Estimate (in hour)*	*Day 1*	*Day 2*	*Day 3*	*Day 4*	*Day 5*	*Day 6*	*Day 7*	*Day 8*	*Day 9*	*Day 10*	*Sprint Review and Retrospective*
User Story #1	**4**												
Login	3	6	2	0	0	0	0	0	0	0	0	0	0
Home	1	2	2	0	0	0	0	0	0	0	0	0	0
User Story #2	**7**												
Select Agent	2	4	2	2	0	0	0	0	0	0	0	0	0
Create Order	5	10	10	10	9	4	1	0	0	0	0	0	0
User Story #3	**5**												
Order List	3	6	6	6	6	6	6	3	0	0	0	0	0
Order Detail	2	4	4	4	4	4	4	4	2	0	0	0	0
Total	**16**	**32**	**28**	**22**	**19**	**14**	**11**	**7**	**2**	**0**	**0**	**0**	**0**

Fig 1: Sprint Backlog

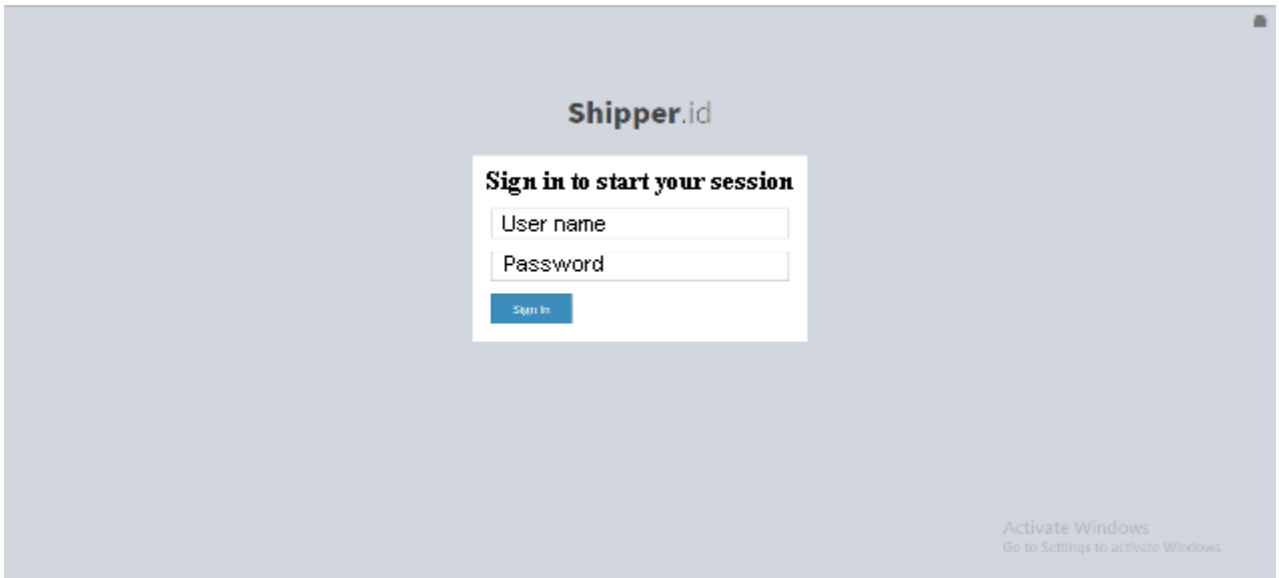

Fig 2: Login View

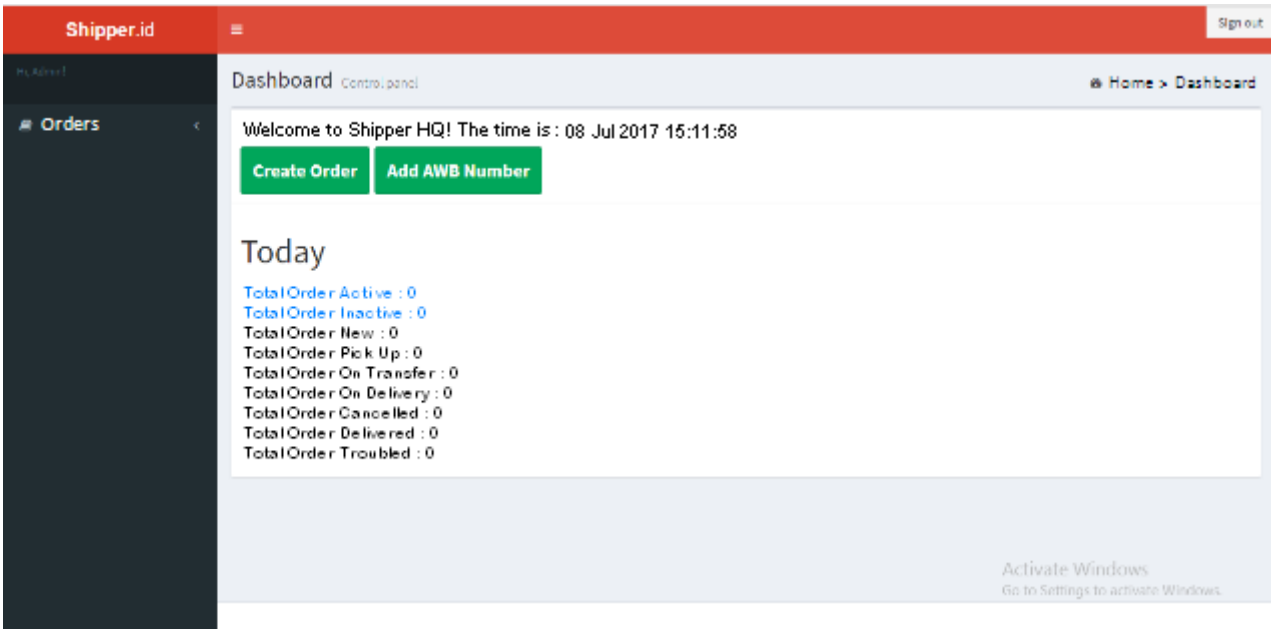

Fig 3 : Home View

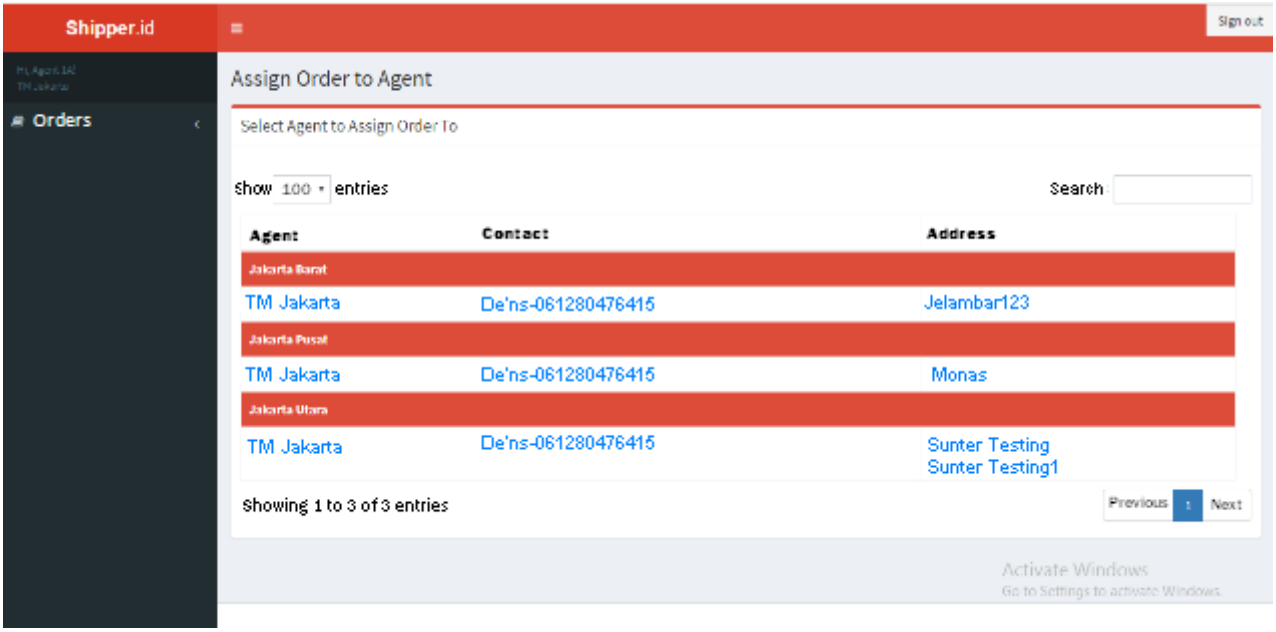

Fig 4 : Select Agent view

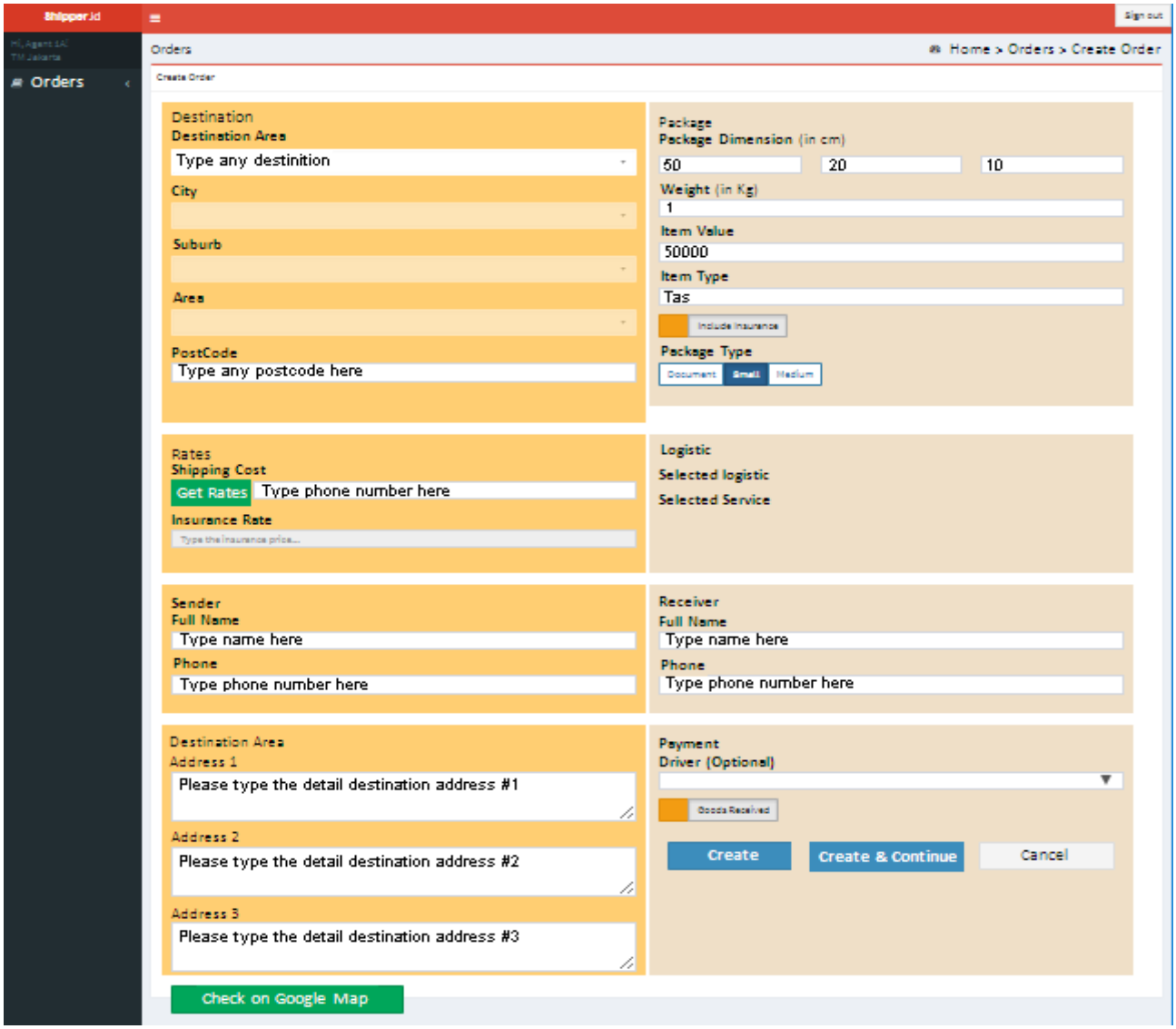

Fig 5 : Create Order View

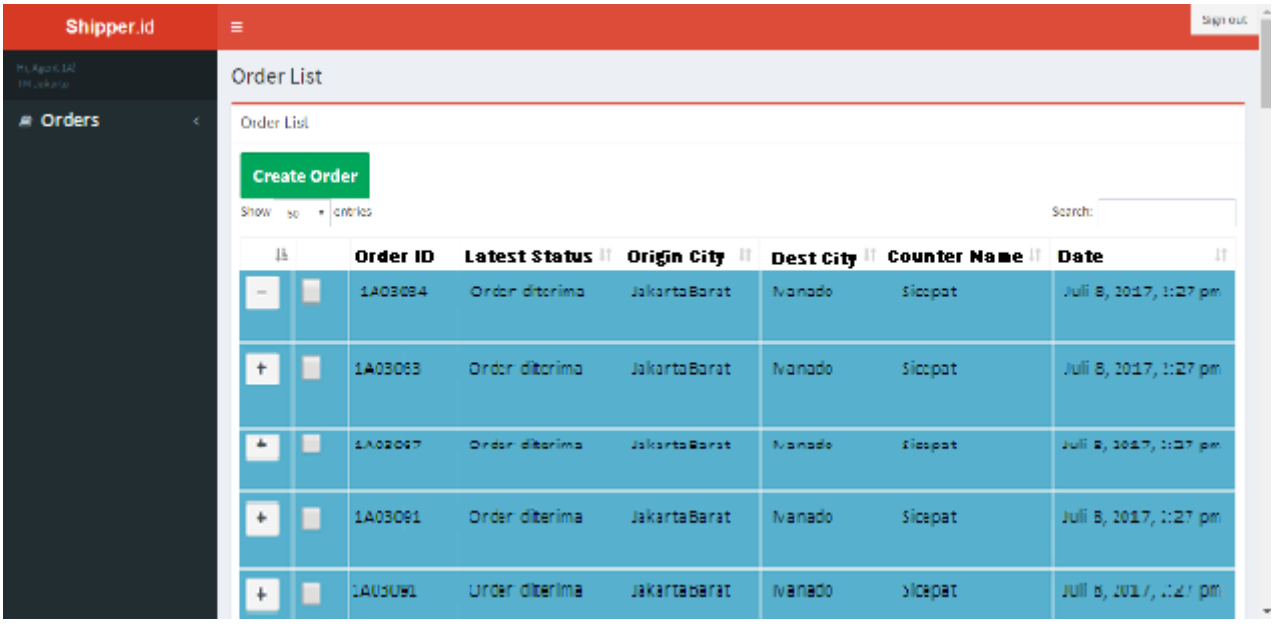

Fig 6 : Order List View

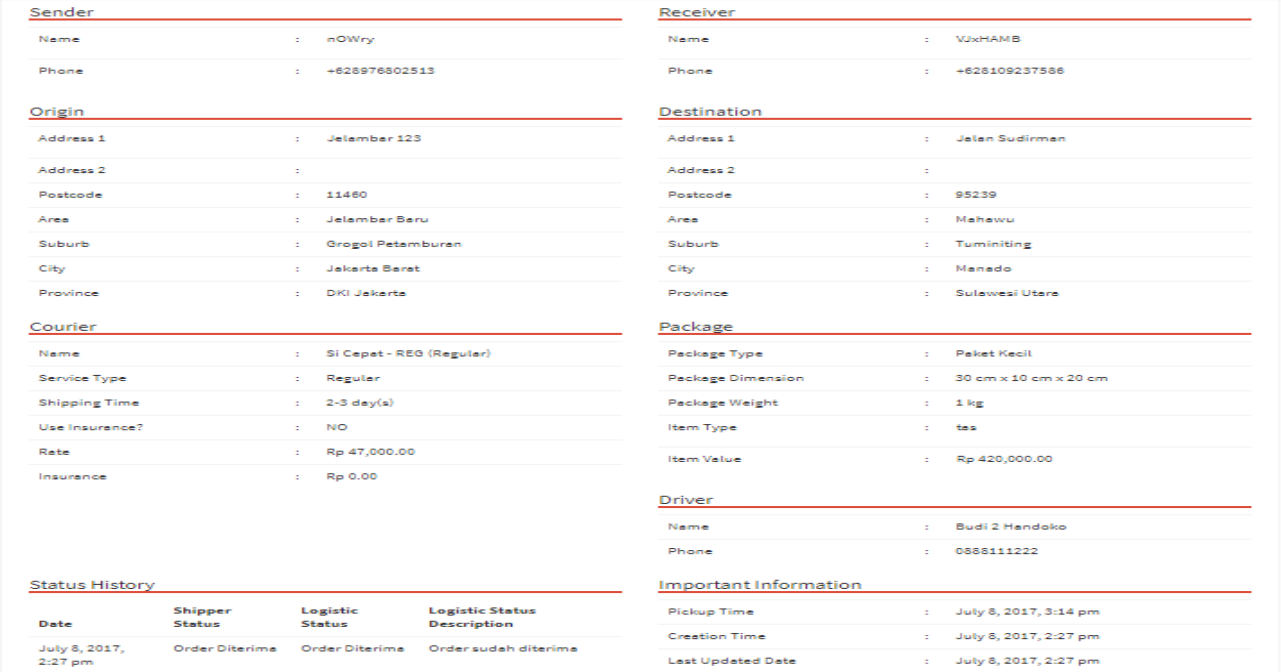

Fig 7 : Order Detail View

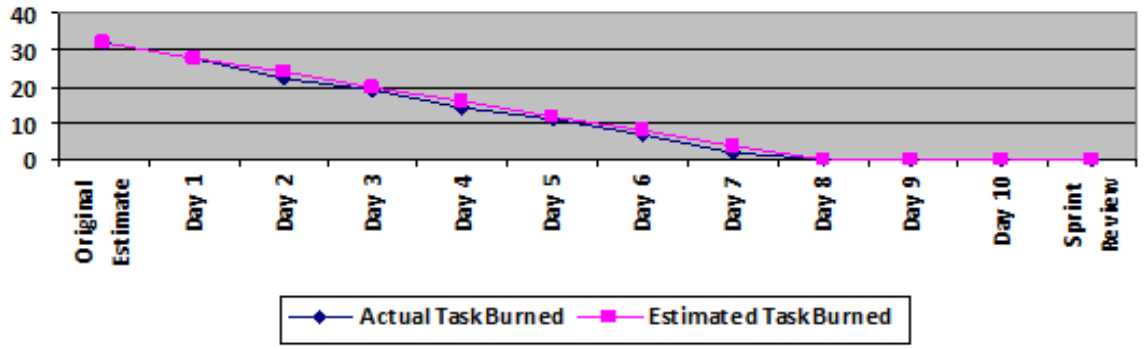

Fig 8 : Burndown Chart

From the burndown chart above can be seen that the execution of each task given is faster than the planned estimate. This can be seen from the Actual Task Burned (blue) line below the Estimated Task Burned (purple) line. The total task before doing is 32 and is estimated to be reduced by 4 hours per day.

In the process, the task is more reduced on the 2nd, 3rd, 4th, 5th, 6th and 7th day and on the 8th day the entire task is done.

6.3 Sprint 2

Here are sprint backlogs, system interfaces, and burndown chart results in Sprint 2.

Backlog Item	*Story Points (1 point = 2 hours)*	*Original Estimate (in hour)*	*Day 1*	*Day 2*	*Day 3*	*Day 4*	*Day 5*	*Day 6*	*Day 7*	*Day 8*	*Day 9*	*Day 10*	*Sprint Review and Retrospective*
User Story #4	**5**												
Checkin Order	5	10	8	5	2	0	0	0	0	0	0	0	0
User Story #5	**6**												
Checkout order	6	12	12	12	12	10	6	2	0	0	0	0	0
User Story #6	**4**												
Input AWB Number	4	8	8	8	8	8	8	8	4	1	0	0	0
Total	**15**	**30**	**28**	**25**	**22**	**18**	**14**	**10**	**4**	**1**	**0**	**0**	**0**

Fig 9 : Sprint Backlog

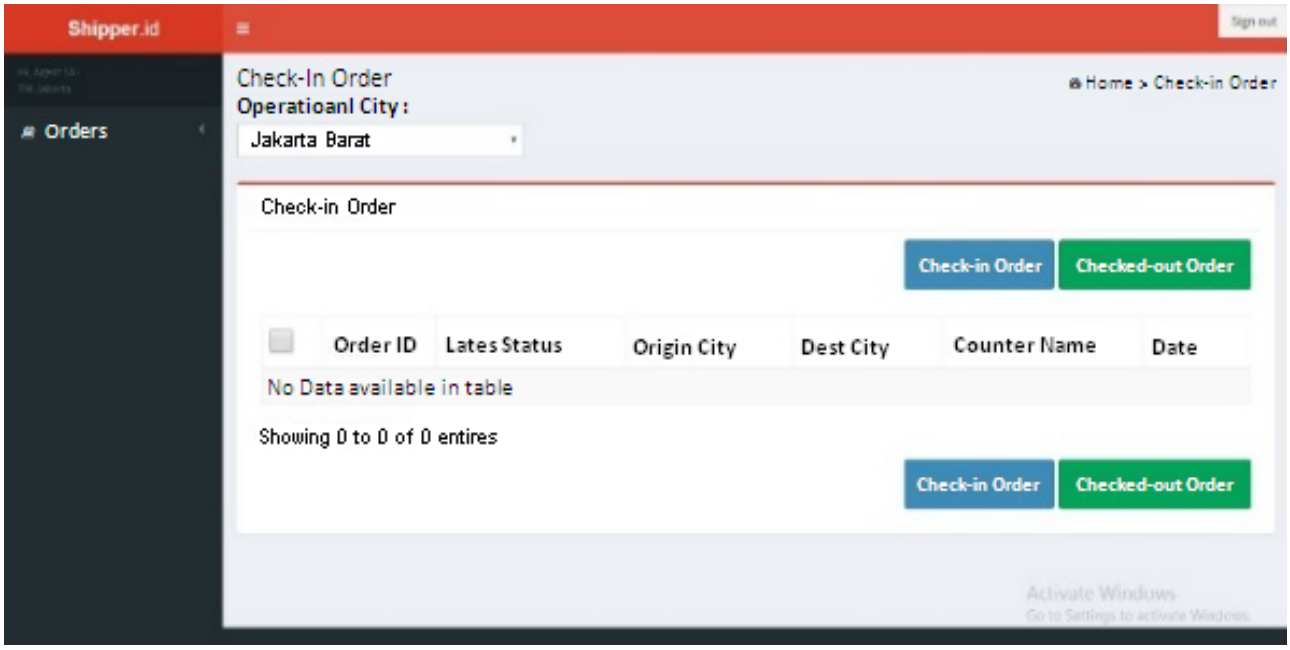

Fig 10 : Check-in Order View

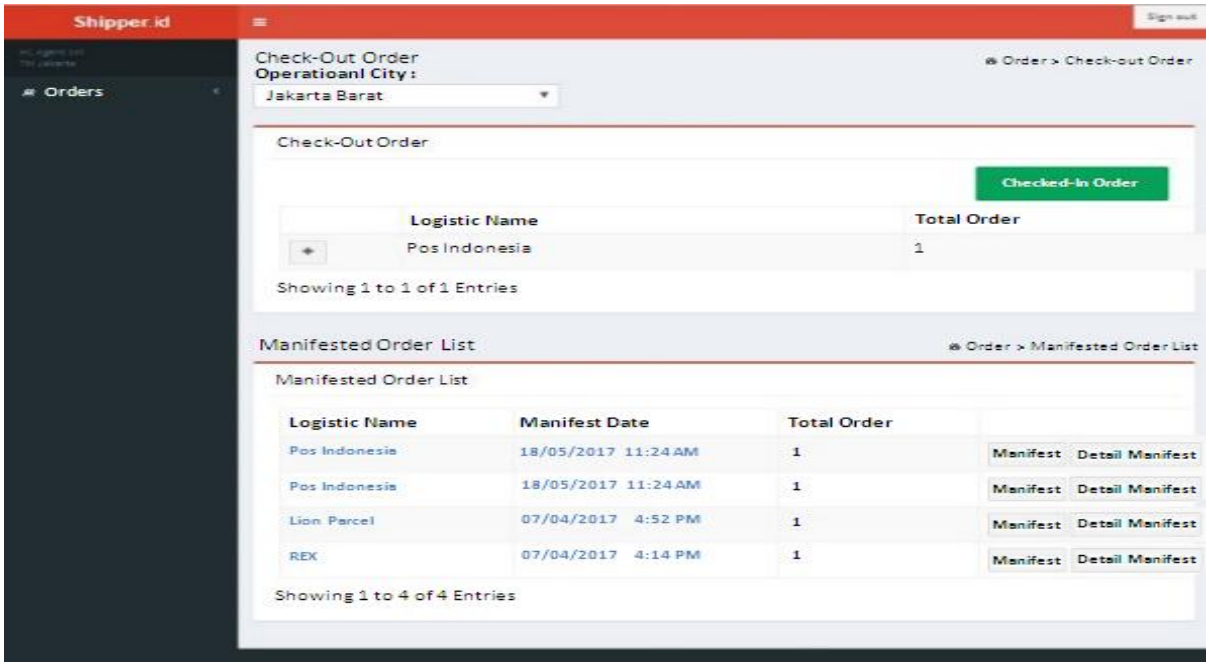

Fig 11 : Check-out Order View

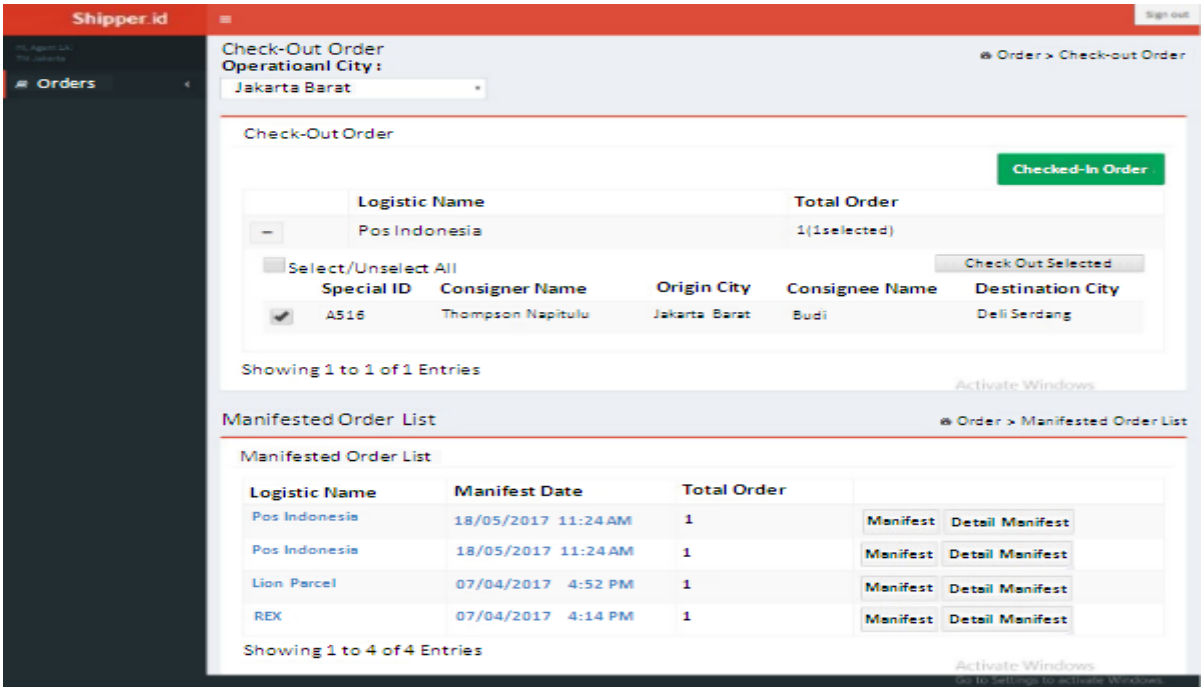

Fig 12 : Check-out Order Detail View

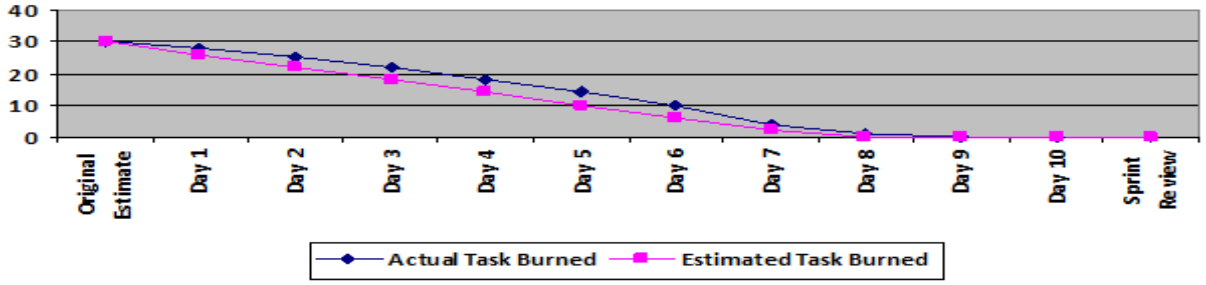

Fig 13 : Burndown Chart

6.4 Testing

Testing is done using Blackbox Testing method which the result can be seen as follows:

1. View Order List

Nama Fungsi	*View Order List*
URL	http://localhost/shipper-skripsi/order/jsonForOrderData
Metode	GET
Ukuran Data yang diterima	20.9 KB
Jumlah Data Tersimpan	16275 dokumen

No. Percobaan	MongoDB Connection		MongoDB Indexing dengan ElasticSearch		Keterangan
	Status	Lama Waktu (dalam detik)	Status	Lama Waktu (dalam detik)	
1.	200	19.64	200	19.29	Lebih Baik
2.	200	14.29	200	14.56	Kurang Baik
3.	200	14.72	200	14.40	Lebih Baik
4.	200	14.41	200	14.21	Lebih Baik
5.	200	14.62	200	14.40	Lebih Baik
6.	200	19.64	200	19.29	Lebih Baik
7.	200	14.56	200	14.29	Lebih Baik
8.	200	14.67	200	14.43	Lebih Baik
9.	200	14.90	200	14.63	Lebih Baik
10.	200	14.58	200	14.40	Lebih Baik
Rata-rata waktu		15.60		15.39	Lebih Baik

Fig 14 : Test Result – View Order List

From the test results table above can be seen that the average access time for demand data function View Order List faster 0.21 seconds or by 1.34%.

The results obtained from the average non-Elasticsearch of 15.60 seconds subtracted by the average Elasticsearch for 15.39 seconds.

2. View Order Detail

Nama Fungsi	*View Order Detail*
URL	http://localhost/shipper-skripsi/order/read/1B06840
Metode	GET
Ukuran Data yang diterima	12.4 KB
Jumlah Data Tersimpan	16275 dokumen

No. Percobaan	MongoDB Connection		MongoDB Indexing dengan ElasticSearch		Keterangan
	Status	Lama Waktu (dalam detik)	Status	Lama Waktu (dalam detik)	
1.	200	7.2	200	8.9	Kurang Baik
2.	200	7.30	200	7.47	Kurang Baik
3.	200	7.28	200	7.23	Lebih Baik
4.	200	7.46	200	7.26	Lebih Baik
5.	200	7.39	200	7.25	Lebih Baik
6.	200	7.36	200	7.21	Lebih Baik
7.	200	8.05	200	7.50	Lebih Baik
8.	200	12.72	200	7.66	Lebih Baik
9.	200	9.0	200	7.68	Lebih Baik
10.	200	7.40	200	7.57	Kurang Baik
Rata-rata waktu		8.11		7.57	Lebih Baik

Fig 14 : Test Result – View Order Detail

From the test results table above can be seen that the average access time for demand data function View Order Detail faster 0.54 seconds or by 6.65%.

The results obtained from the average non-Elasticsearch of 8.11 seconds deducted by the average Elasticsearch for 7.57 seconds.

3. Search Order (For 1 letter)

Nama Fungsi	*Search Order*
URL	http://localhost/shipper-skripsi/order/jsonForOrderData?sSearch=X
Metode	GET
Ukuran Data yang diterima	1.0 KB
Jumlah Data Tersimpan	16275 dokumen
Kata kunci yang dicari	X

No. Percobaan	MongoDB Connection		MongoDB Indexing dengan ElasticSearch		Keterangan
	Status	Lama Waktu (dalam detik)	Status	Lama Waktu (dalam detik)	
1.	200	7.65	200	6.89	Lebih Baik
2.	200	7.58	200	7.22	Lebih Baik
3.	200	7.67	200	7.11	Lebih Baik
4.	200	7.56	200	7.14	Lebih Baik
5.	200	7.55	200	7.03	Lebih Baik
6.	200	7.68	200	7.21	Lebih Baik
7.	200	7.89	200	7.34	Lebih Baik
8.	200	7.66	200	7.11	Lebih Baik
9.	200	7.54	200	7.13	Lebih Baik
10.	200	7.56	200	7.02	Lebih Baik
Rata-rata waktu		7.63		7.12	Lebih Baik

Fig 15 : Test Result – Search Order 1

From the test results table above can be seen that the average access time for data query Search Order function with search keywords as much as 1 letter faster 0.51 seconds or equal to 6.66%.

The result is obtained from the average non-Elasticsearch of 7.63 seconds deducted by Elasticsearch average of 7.12 seconds.

4. Search Order (For 5 letters)

Nama Fungsi	*Search Order*
URL	http://localhost/shipper-skripsi/order/jsonForOrderData?sSearch=bUxCH
Metode	GET
Ukuran Data yang diterima	1.0 KB
Jumlah Data Tersimpan	16275 dokumen
Kata kunci yang dicari	bUxCH

No. Percobaan	MongoDB Connection		MongoDB Indexing dengan ElasticSearch		Keterangan
	Status	Lama Waktu (dalam detik)	Status	Lama Waktu (dalam detik)	
11.	200	8.1	200	7.52	Lebih Baik
12.	200	8.29	200	7.20	Lebih Baik
13.	200	8.14	200	7.28	Lebih Baik
14.	200	7.90	200	7.20	Lebih Baik
15.	200	7.85	200	7.23	Lebih Baik
16.	200	8.09	200	7.21	Lebih Baik
17.	200	8.11	200	7.24	Lebih Baik
18.	200	8.10	200	7.22	Lebih Baik
19.	200	8.05	200	7.20	Lebih Baik
20.	200	8.12	200	7.31	Lebih Baik
Rata-rata waktu		8.07		7.26	Lebih Baik

Fig 16 : Test Result – Search Order 5

From the test results table above can be seen that the average access time for data query Search Order function with search keywords as much as 5 letters faster 0.81 seconds or 10.01%. The results obtained from the average non-Elasticsearch of 8.07 seconds deducted by Elasticsearch average of 7.26 seconds.

5. Search Order (For 10 letters)

Nama Fungsi	*Search Order*
URL	http://localhost/shipper-skripsi/order/jsonForOrderData?sSearch=RDJzHuBUfI
Metode	GET
Ukuran Data yang diterima	1.0 KB
Jumlah Data Tersimpan	16275 dokumen
Kata kunci yang dicari	RDJzHuBUfI

No. Percobaan	MongoDB Connection		MongoDB Indexing dengan ElasticSearch		Keterangan
	Status	Lama Waktu (dalam detik)	Status	Lama Waktu (dalam detik)	
21.	200	7.85	200	7.29	Lebih Baik
22.	200	8.10	200	7.35	Lebih Baik
23.	200	8.13	200	7.46	Lebih Baik
24.	200	7.91	200	7.26	Lebih Baik
25.	200	7.86	200	7.22	Lebih Baik
26.	200	8.13	200	7.24	Lebih Baik
27.	200	8.05	200	7.21	Lebih Baik
28.	200	8.11	200	7.22	Lebih Baik
29.	200	8.20	200	7.31	Lebih Baik
30.	200	7.96	200	7.23	Lebih Baik
Rata-rata waktu		8.03		7.27	Lebih Baik

Fig 16 : Test Result – Search Order 5

From the test results table above can be seen that the average access time for data query Search Order function with search keywords as much as 10 letters faster 0.76 seconds or by 46%. The results obtained from the average non-Elasticsearch of 8.03 seconds deducted by Elasticsearch average of 7.27 seconds.

7. CONCLUSION

Based on the results of research on "Implementation Elasticsearch Search Engine on Order Management System" can be concluded as follows:

1. In this research has been implemented search-engine Elasticsearch into Order Management System in a company. Elasticsearch has been applied to the system in the form of libraries that have been used in search order data. Therefore, it can be said that Elasticsearch can be applied and used well into the system.
2. In this study also can be concluded that Elasticsearch has an influence in improving the performance Order Management System. It is concluded from the comparison made between order management systems that have been

implemented Elasticsearch with similar system but not implement Elasticsearch. From the comparison results found that the system that has implement Elasticsearch faster in the access time about 10.01% of the system that is not implement Elasticsearch for search function based on keyword or keyword.

8. ACKNOWLEDGMENT

Authors please to acknowledge Mercu Buana University, Jakarta for every support to write this paper.

9. REFERENCES

[1] Abubakar, Y., Adeyi, T. S., & Auta, I. G. (2014). Performance Evaluation of NoSQL Systems Using YCSB in a Resource Austere Environment. *International Journal of Applied Information Systems (IJAIS), 7 - No. 8*, 23-27.

[2] Croft, B., Metzler, D., & Strohman, T. (2015). *Search Engines - Informational Retrieval in Practice.* Massachusetts: Pearson Education, Inc.

[3] Gormley, C., & Tong, Z. (2015). *Elasticsearch The Definitive Guide - A Distributed Real-time Search and Analytics Engine.* Sebastopol: O'Reilly Media, Inc.

[4] Gupta, P., & Nair, S. (2016). Survey Paper on Elastic Search. *International Journal of Science and Research (IJSR), 5*(1), 333-336.