Design and Implementation of Web Based Real Time Chat Interfacing Server

Diotra Henriyan [#1], Devie Pratama Subiyanti[#2], Rizki Fauzian[#3], Dian Anggraini[#4], M. Vicky Ghani Aziz[#5], Ary Setijadi Prihatmanto[#6]

[#] *Control and Computer System Laboratory*
School of Electrical Engineering and Informatics
Institut Teknologi Bandung, Indonesia
[1]diotrahenriyan@gmail.com
[2]deviepsubiyanti@gmail.com
[3]rizkifauzian@gmail.com
[4]yandiyan29@gmail.com
[5]vickyaziz@live.com
[6]ary.setijadi@gmail.com

***Abstract*— Chat application is a feature or a program on the Internet to communicate directly among Internet users who are online or who were equally using the internet. Chat applications allow users to communicate even though from a great distance. Therefore, this chat application must be real-time and multi platform to be used by many users. The development of information and communication technologies are rapidly making one of the reasons for Indonesia, especially Bandung to develop this chat application. That's because Indonesia does not always rely on outsiders. It is important for Indonesia to develop this chat application for themselves. This chat application in the manufacture begins with the collection of relevant data that will be displayed in the web and mobile versions. The programming language used to build server is Node.js with express framework and MongoDB database.**

***Keywords*— real time chat, multi-platform, node js**

I. INTRODUCTION

At this time, the development of information and communication technology is growing. With the information and communication technologies to facilitate the public to obtain information and communicate from anywhere, anytime. In the development of information and communication technologies, Indonesia cannot always depend on outsiders. Indonesia is also expected to contribute to participate in the development of information and communication technology. One of them is the provider of chat applications. Chatting in Indonesian means a chat or talk in both directions between one or several people. In the world of computers, chatting means talking to other people using the computer [2].

Currently, the chat application has been developed by outsiders. A wide variety of chat application has advantages and disadvantages. Indonesia has not had a chat application that was developed by themselves. Therefore, from the above problems, it is the background to build a chat application themselves.

Chat applications that will be built is web-based and mobile applications. This chat application is an application that is used to communicate and developed to support especially the city of Bandung and the Indonesian people in general. In this chat application, built using Node.js, Socket.io , MongoDB and java.

Node.js is a software platform that is used to build serverside flexible applications in a network application [1]. Socket.IO is a javascript library is an implementation of web-socket protocol and various other necessary improvisation for real-time web [4]. JavaScript is a language used to create a program so the HTML document displayed in the browser becomes more interactive [6]. MongoDB is an open source database based on document (Document-Oriented Database) which was originally created in C ++ [3]. Therefore, it is expected this chat application can run in real time.

II. MAIN PROBLEM

Based on the description above, the problem is how to design and build a dedicated web-based and mobile chat applications in the city of Bandung in real time to make share and communicate easier and faster.

III. OBJECTIVE

An objective then can be defined from the main problem above that is build a dedicated real-time multi-platform chat application which can be used by Bandung's people to make they share the information and communicate easier and faster.

IV. SCOPE

The scope of system should be declared before move advancing to the next step. System scope are as follows:

1. Design and construction of this chat application only web-based applications and mobile applications.
2. This system was developed uses Node.js and socket.io.
3. Database of this chat application using MongoDB.

4. Used by less than 100 people (for testing purpose)

V. System Design

The system will be built in two sides; server and client side. Server side has service and middleware, while the client side has two multi-platform; website and mobile phone. Website is a collection of information pages that provided via Internet that can be accessed around the world over the network connected to the Internet [5]. Client will communicate to server through JSON which act as a middleware of server. JSON (JavaScript Object Notation) is a lightweight data interchange format, readable and writable by humans, as well as easily translated and made (generated) by the computer [4]. The service has four services which are node.js, socket.io, express.js, and mongo db. All server – client data communication will communicate in both ways.

Server's services are using the Node.js with Express framework, Socket.io and MongoDB, while Middleware are using JSON. If the user uses a web-based chat application and take action on the web, the web will interact with JSON JSON would then access the service, after getting the desired functionality then JSON will return the results to the user.

A. Block Diagram

Here is a block diagram of server – client communication

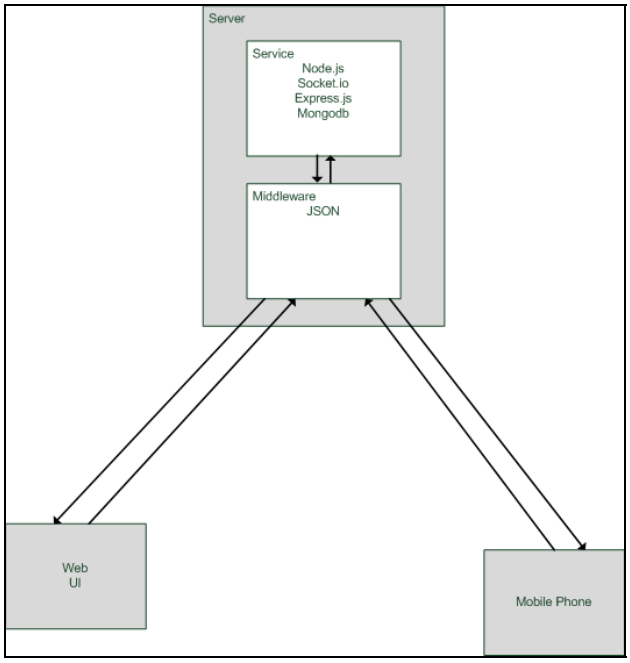

Fig. 1 Server – client block diagram

B. Flow Chart

Here is a flow of chat overall process.

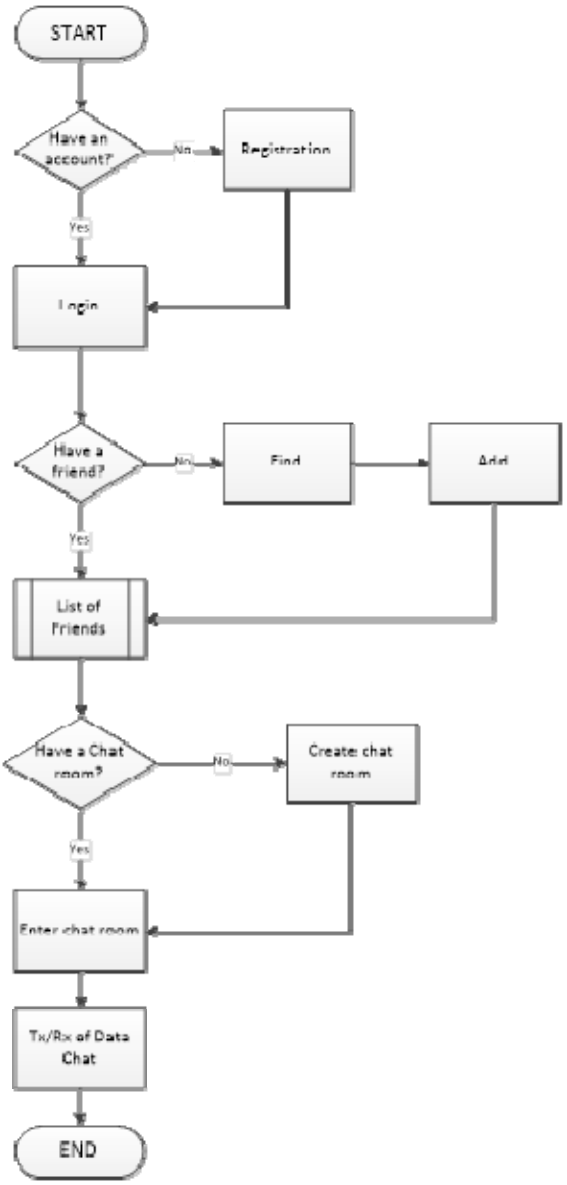

Fig. 2 User interface design of web based chat application

The explanation for each flow from picture above is as follow:

1. When the main page is accessed by user it will show the login page first, instead of show the main page directly. User login credentials is required. If the user doesn't have an account yet, then they must register first.
2. After login, user must have friend before they can start a chat. Therefore, if user has no friend yet, they can find them directly from the application. Afterwards, user can add friend to their friendlist, and waiting for friend request confirmation.
3. However, if user has friends in their friendlist, then they can immediately start a chat.
4. Next step, the system will checks if user has a chatroom with another user selected. If has not, then system will create the chatroom and if they already have a chatroom, they will enter that chatroom to start chat.

5. Finally, data will be sent and received.

C. Data Flow Diagram

Here is a data flow diagram to describe overall process of chat application data.

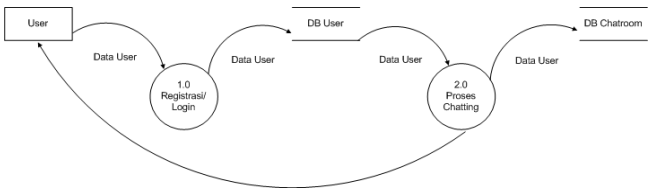

Fig. 3 Chat application data flow diagram

D. Database Table Data Structure

Here is database table data structure for each table user and chatroom.

TABLE I
USER DATA STRUCTURE

No	Field Name	Data Type	Explanation
1	_id	ObjectId	User Id
2	username	String	Username
3	password	String	Password
4	email	String	Email
5	active	Boolean	Status
6	on_last_login	Date	Last login
7	img_profile	String	Profile picture
8	privilege	Int	Status
9	Friends	Array	Friendlist array status

TABLE II
CHATROOM DATA STRUCTURE

No	Field Name	Data type	Explanation
1	_id	ObjectId	Id Chatroom
2	on_created	Date	Chatroom date created
3	messages	Array	Message, status, sender, and send time array.

E. User Interface Design

The web based chat application would has two main feature that are Friends and Chatroom.

Here is a user interface design of Friends page.

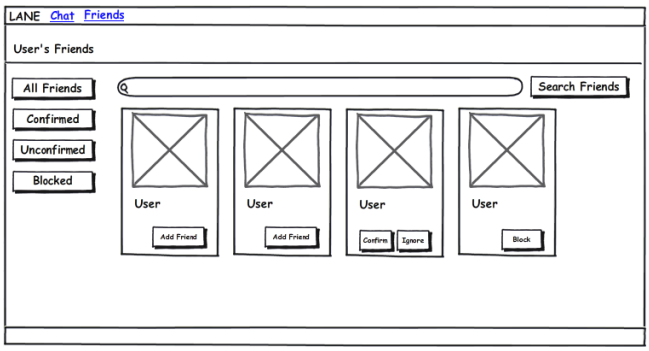

Fig. 4 User interface design of Friends page

Here is a user interface design of chatroom page.

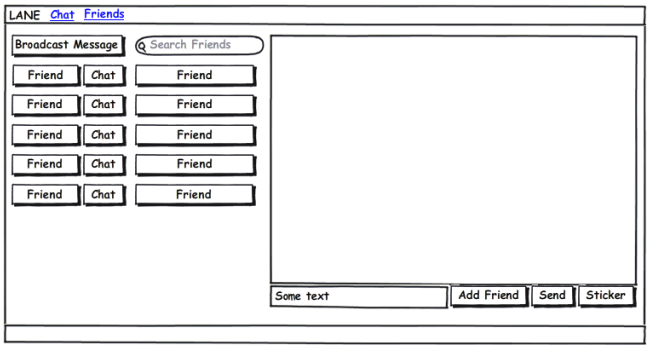

Fig. 5 User interface design of Chatroom page

VI. IMPLEMENTATION

F. Friendlist Management

In Friendlist Management, users can search your friends, add friends, confirm friend requests, block or unblock friends.

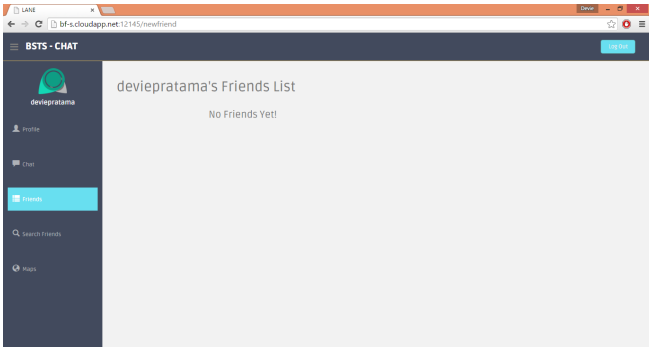

Fig. 6 Friends page

The figure shown above is the display of friends menu. On the menu user can see who his friends and when the user wants to send a message, they can start make a chatroom there.

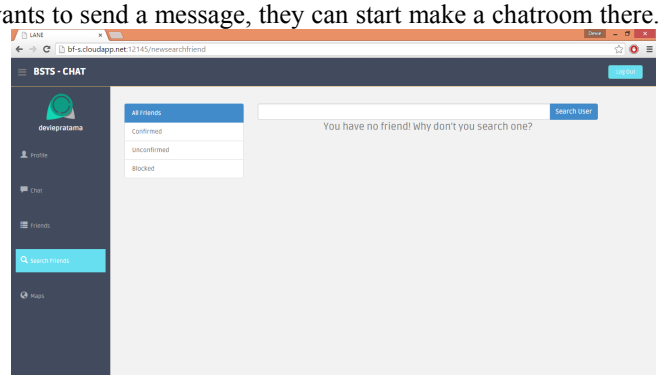

Fig. 7 Search Friends page

The figure shown above is the display from the Search Friends menu. On the menu user can search for another users based on the user id. Search Friends menu also has another menu; All Friends, Confirmed, Unconfirmed, and Blocked.

G. Chat room / Groupchat

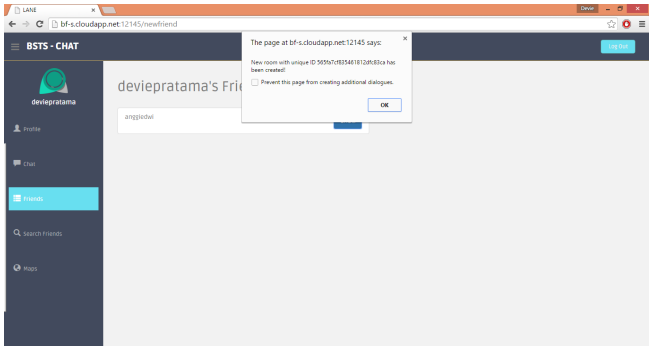

Fig. 8 Add friend to a chat room

Pictured above is a display when a user wants to create a group chat. Users must make room first and then add more users to join in the room.

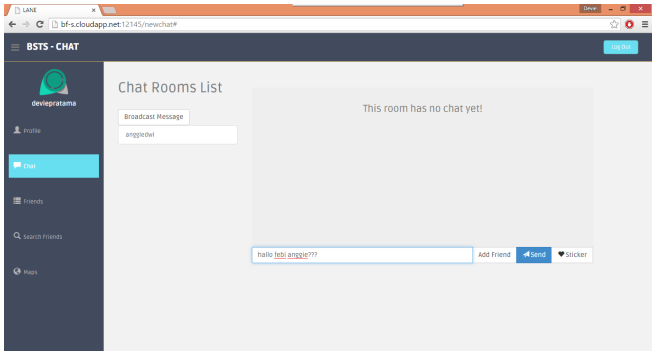

Fig. 9 Group chat

The figure shown above is the view of the user that has been added to a group chat. So in the room there are three users. And in the display above one user sends a message.

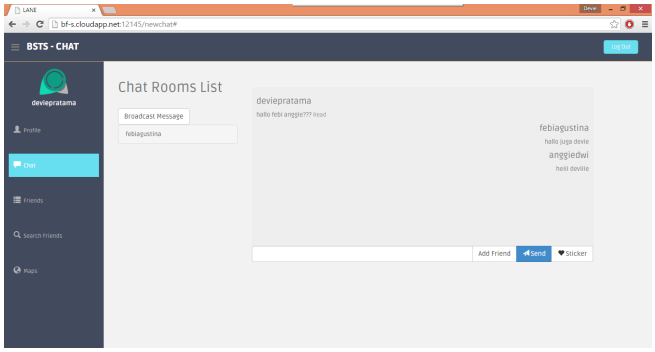

Fig. 10 Each users sending a message

H. Broadcast Message

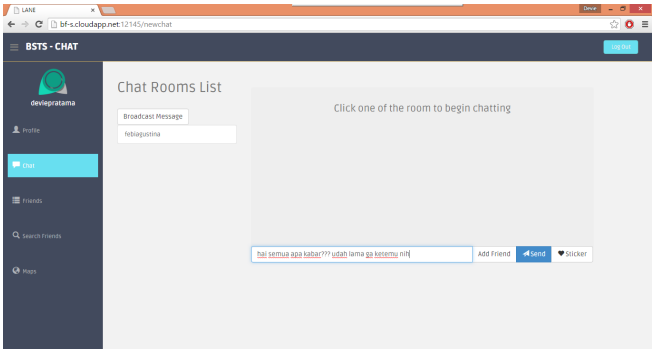

Fig. 11 Broadcast message button pressed

The figure shown above is a picture of the broadcast message. Once the user presses the selection and enter the Broadcast message and send the message, then the message will be received by all users who become friends.

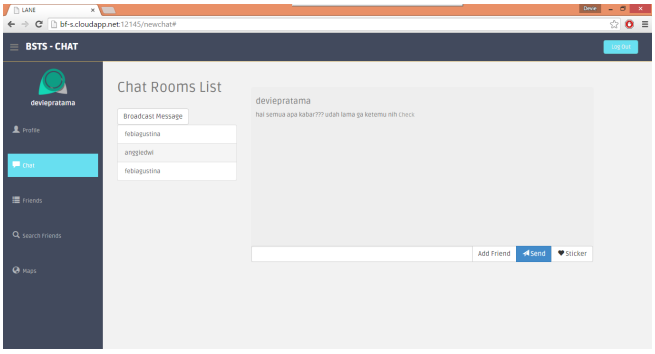

Fig. 12 Users sends a broadcast message

Figure shown above is a display of the user who sent a broadcast message.

VII. TEST RESULT

As the test result, the main features of these chat application sends and receives the messages very well. Chat application built with Node.js, MongoDB and Socket.io running much faster in order to achieve real-time chat applications compared with the chat application built with PHP and MySQL. Test result of the chat application shows the same results when tested on localhost, in the network or when in hosting. Here is a comparison table of chat application built with Node.js, MongoDB, and socket.io with chat application built with PHP and MySQL.

TABLE III
TESTING NODE.JS, SOCKET.IO AND MONGODB WITH PHP AND MYSQL

No	Chat with Node.js Mongodb and Socket.io based	Chat with PHP dan MySQL based
1	0.13	1.56
2	0.15	1.44
3	0.18	2.03

4	0.13	1.44
5	0.13	1.34
6	0.16	1.55
7	0.17	1.53
8	0.22	1.39
9	0.18	2.00
10	0.19	1.34

VIII. CONCLUSION

Based on the results of tests performed, it can produce an conclusion that

1. The chat application built with Node.js, Socket.io, and MongoDB is running in real time very well with a speed less than one second (6 times faster) than the chat application built with PHP and MySQL. So it can be ascertained if data more than 200 chat, Node.js, Socket.io and MongoDB will have a speed 1200 faster than PHP and MySQL.
2. CombSort Strict CPU Test

Table IV
CombSort Strict CPU Test

	CPU time	System time	RAM
PHP 5.6.4	102.69s	104.20s	2497508 KB
Node.js v0.10.35	2.64s	2.64s	92240 KB

Node.js is more than 35 times faster than plain PHP (by system time). In terms of RAM usage, Node.js is much more efficient than PHP.

REFERENCES

[1] Croucher, T. H., & Wilson, M. (2012). Node: Up and Running. United States.
[2] Haryadi, M. F. (2010). ANALISA DAN PERANCANGAN APLIKASI CHATTING BERBASIS WEB MENGGUNAKAN FLASH CS3. 1-2.
[3] Kurniarin, K. (2012). PENERAPAN MONGODB DAN CODEIGNITER PADA PERMAINAN KOKOLOGY. 3-4.
[4] Purnomosidi, B. (2013). Pengembangan Aplikasi Cloud Computing Menggunakan Node.js.
[5] Rahmadini, A. (2013). Pembangunan Sistem Informasi Pengelolaan Inventaris Barang Divisi Pustekin Berbasis Web. Bandung: Politeknik Telkom.
[6] Sidik, B. (2011). JavaScipt. Bandung: Informatika.
[7] Kiessling, Manuel. 2012. The Node Beginner Book. lulu.com, United Stated.
[8] Mardan, Azat. 2012. Practical Node.js: Building Real-World Scalable Web Apps. Appress.
[9] Teixeira, Pedro. 2012. Professional Node.js: Building Javascript Based Scalable Software Kindle Edition. Wrox.
[10] Teixeira, Pedro. 2012. Hands-on Node.js. Wrox.