In the name of ALLA

Network Project 2

A simple mail box with
NodeJS,Express,Socket.io

for : Dr. Siamak Sarmadi

Developed by :

Mohammad Sharifi
&
Amin Ettehadi

UUT
June – 2015

Why NodeJS?
What is Express and socket.io?
How socket.io works?

Our project + screen shots

- background: +how to run project
- Client side
- Server side
- How we use protocols

Why NodeJS ?

NodeJS gives me the ability to write back-end code in one of my favorite language: JavaScript. It's the perfect technology for building real time applications. In this project, we build a web mail application, using ExpressJS and Socket.io

What is Express and Socket.io ?

1. **ExpressJS** - this will manage the server and the response to the user.

2. **Socket.io** - allows for real time communication between the front-end and back-end.

How Socket.io works ? ^

In this project we have to sections, Client-side and Server-side

Server-side

Sending Data From the Server To the Client:

If we want to send some data from the server to *index.html*. All data transactions in socket.io, like in most of node.js, can be handled with callbacks primarily we will utilize the `on` method. The on method in simplistic terms is used to map a method name to an annonymous function. Because we are using socket.io the on method simply uses the listener on the websocket connection for the method name and when it is found it executes the mapped annonymous function.

The flip side of the *on* method is the ***emit*** method. The *emit* method sends the mapped method name to the client or the server. It takes two arguments. The mapped method name and the data to be fed to the annonymous function.

So here is a simple server.js:

```
io.listen(server);

io.sockets.on('connection', function(socket){
        socket.emit('message', {'message': 'hello world'});
});
```

To retrieve the emitted “message” action on socket.html we just add an on method listener:

```
var socket = io.connect();

socket.on('message', function(data){
        console.log(data.message);
});
```

Client-side

For using socket.io in client-side we must have this minimum code in *views/index.html :*

```
<html>

    <head>
        <script src="/socket.io/socket.io.js"></script>
    </head>
    <body>
        <script>
            var socket = io.connect();
        </script>
        <div>This is our socket.html file</div>
    </body>
</html>
```

Sending Data From the Client To the Server:

this process is completely the same as what we just did! For example lets say we want to print every letter the user types into a textbox to the server console.

```
<html>

<head>
    <script src="/socket.io/socket.io.js"></script>
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/1.7.2/jquery.js"></script>
</head>
<body>
    <script>
    var socket = io.connect();
    $(document).ready(function(){
        $('#text').keypress(function(e){
            socket.emit('client_data', {'letter': String.fromCharCode(e.charCode)});
        });
    });
    </script>
    <textarea id="text"></textarea>
</body>
</html>

//recieve client data
socket.on('client_data', function(data){
    console.log(data.letter);
});
```

Our Project

The first thing to do is get **NodeJS installed** on your system. If you are a Windows or Mac user, you can visit nodejs.org and download the installer. If you instead prefer Linux, I'd suggest that you refer to this link.

After installation, from root of project directory open the terminal and type :

```
node server.js
```

finally open this address in your browser: http://127.0.0.1:1337

And you see some thing like this!:

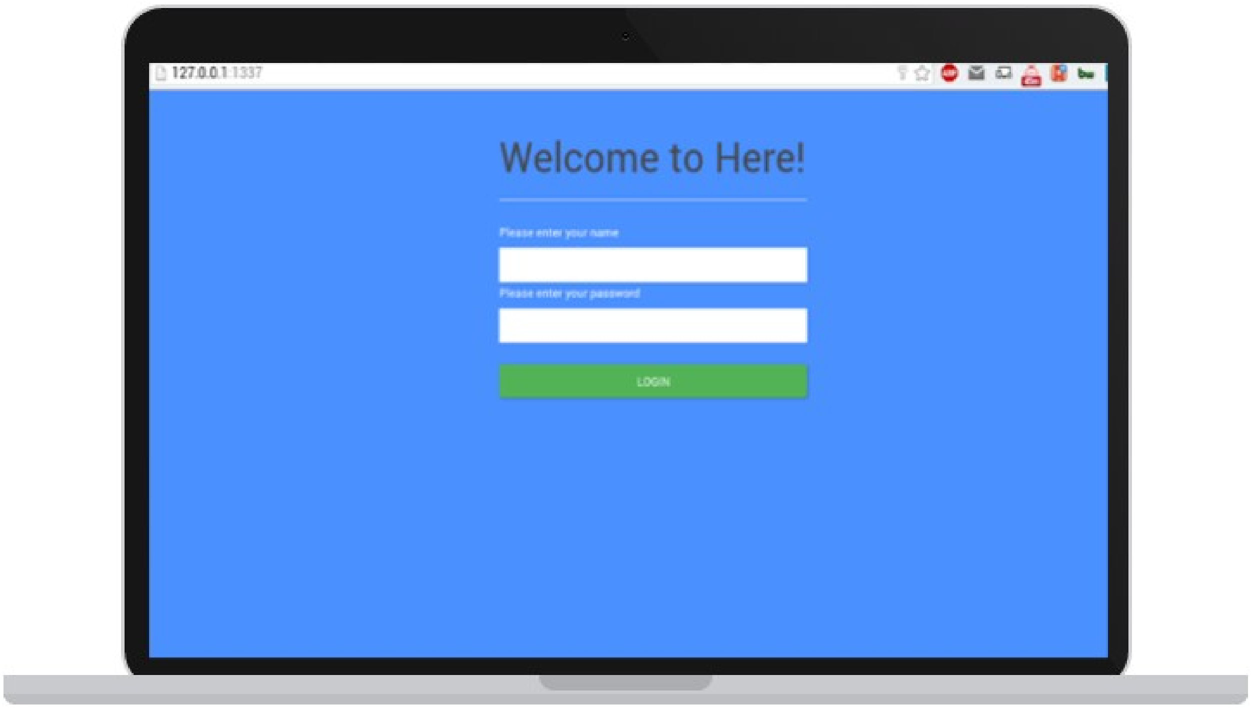
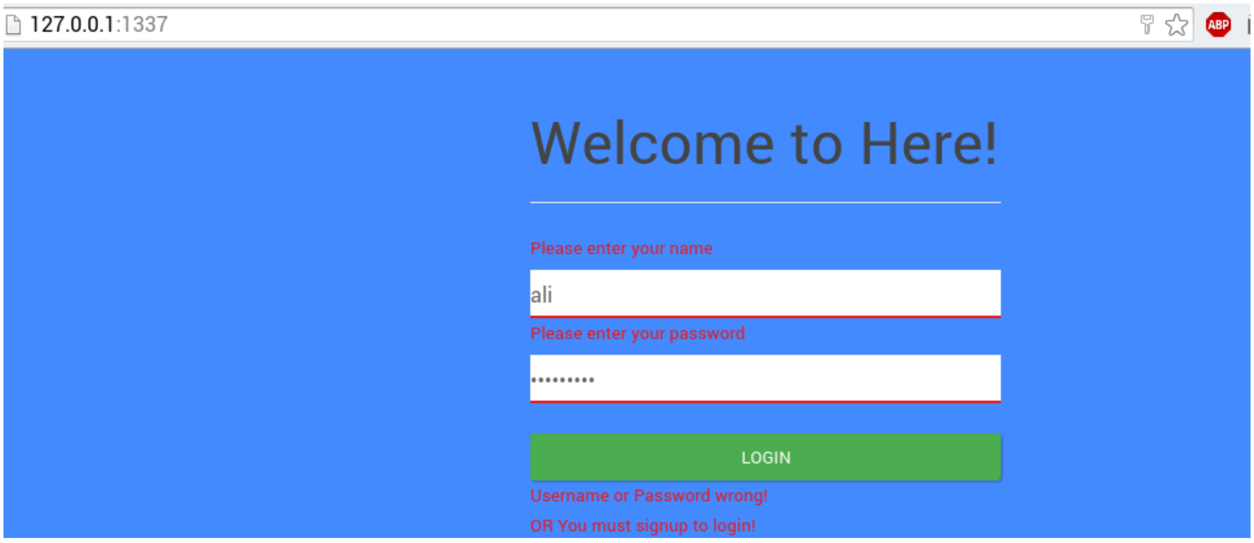


Image 1 – Login form

From above image first things that user see is a login form. If you enter user name and password correctly you will login to app successfully but otherwise you see this error:



Username or Password wrong!
OR You must signup to login!

Image 2 – Username or password wrong or user not existed in database!

We save username passwords in *DB/users.txt* , you can try this valid username & password to login:

"ali , 123 " … "ahmad , 321" … "babak , asd"

after login:

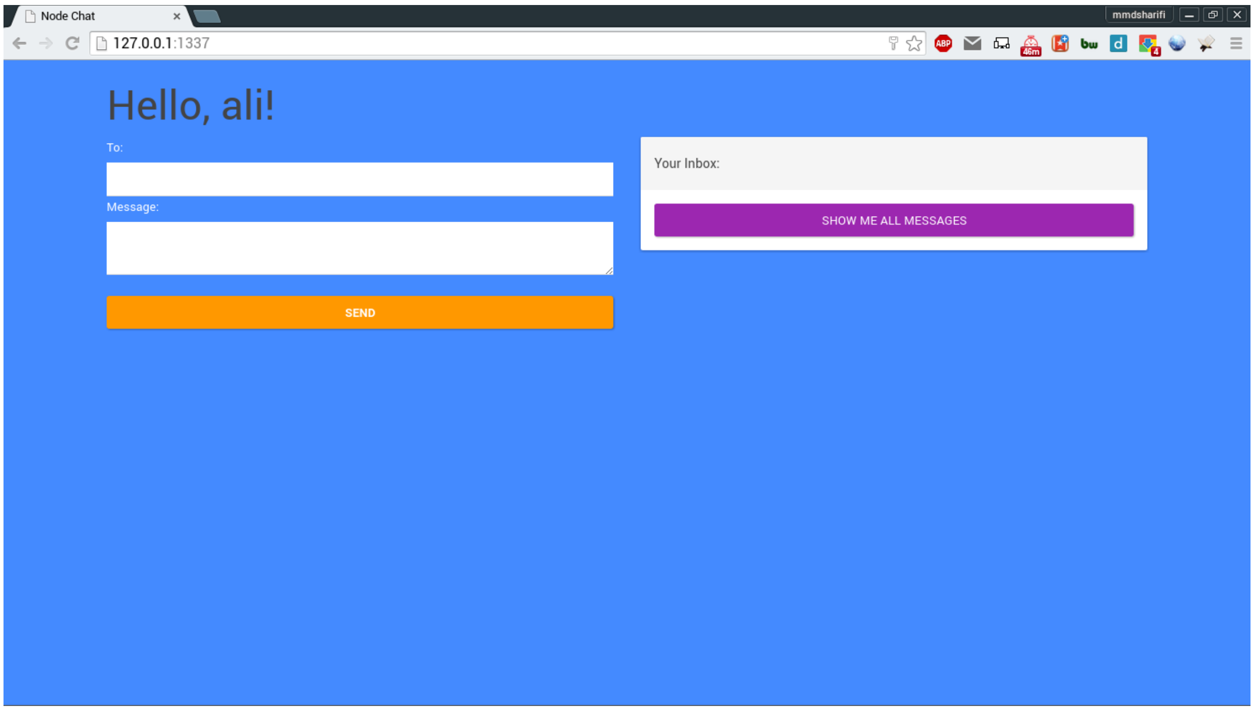


Image 3 – successful login for username,password : ali,123

after you loged in app you see 2 sections, the left section for sending message and other to see all messages that comes to ali.let's send a message!

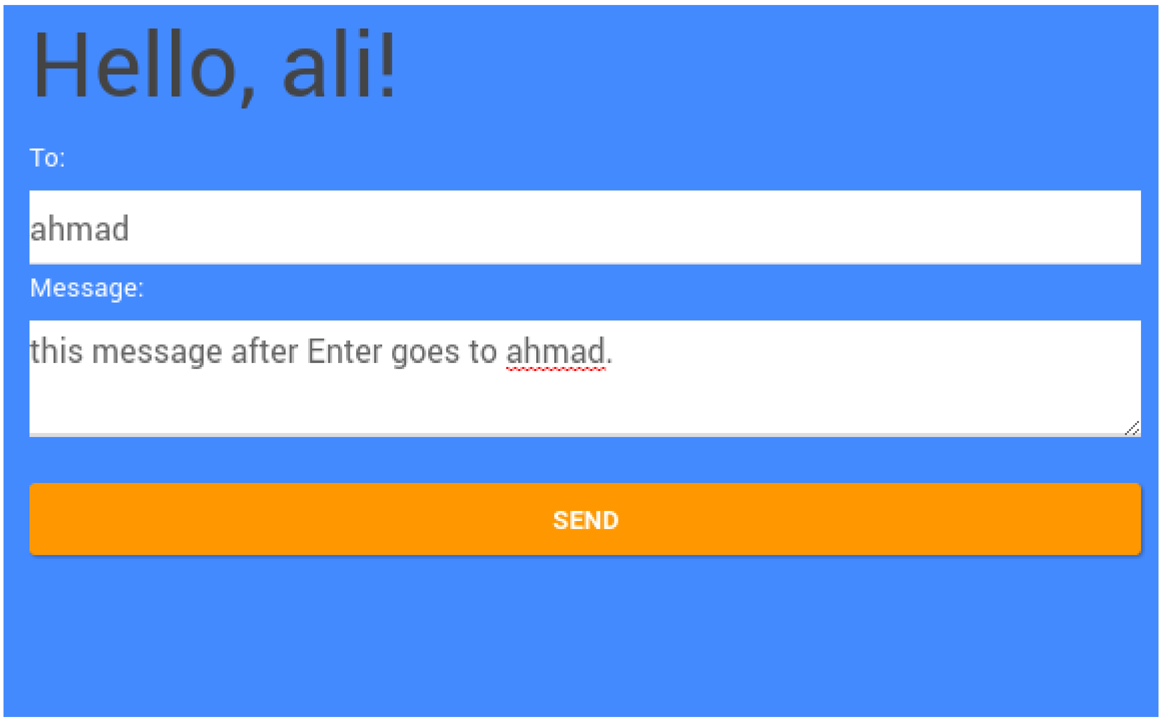
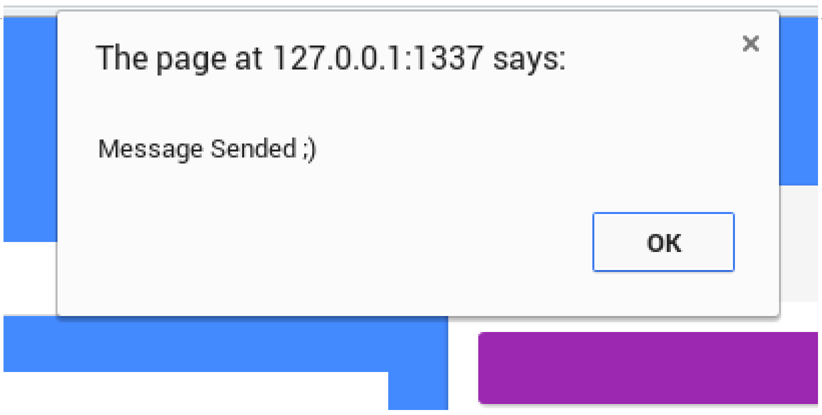


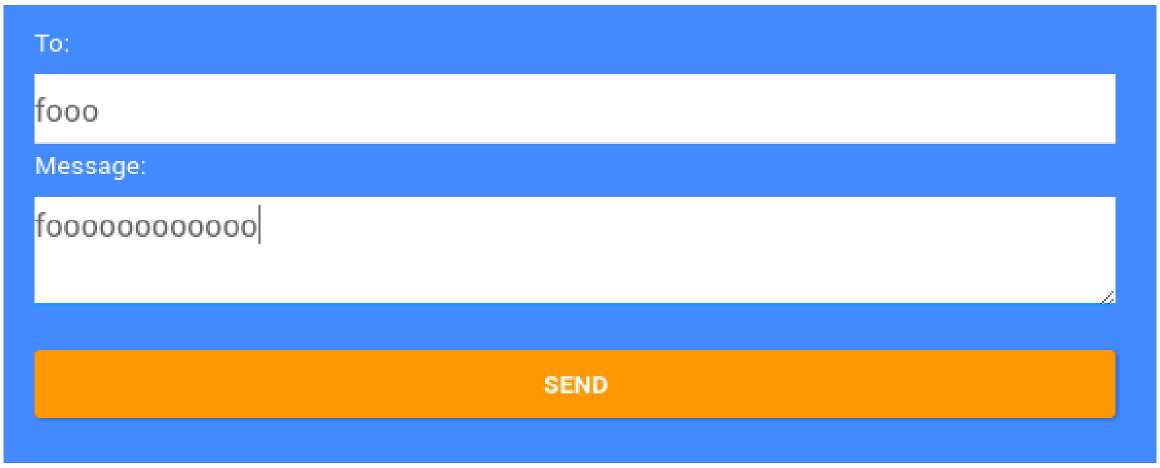
Image 4 – send a message for ahmad(a valid username)


The page at 127.0.0.1:1337 says:

Message Sended ;)

Image 5 – after sending message successfully show a alert to user…

Receiver not found example:

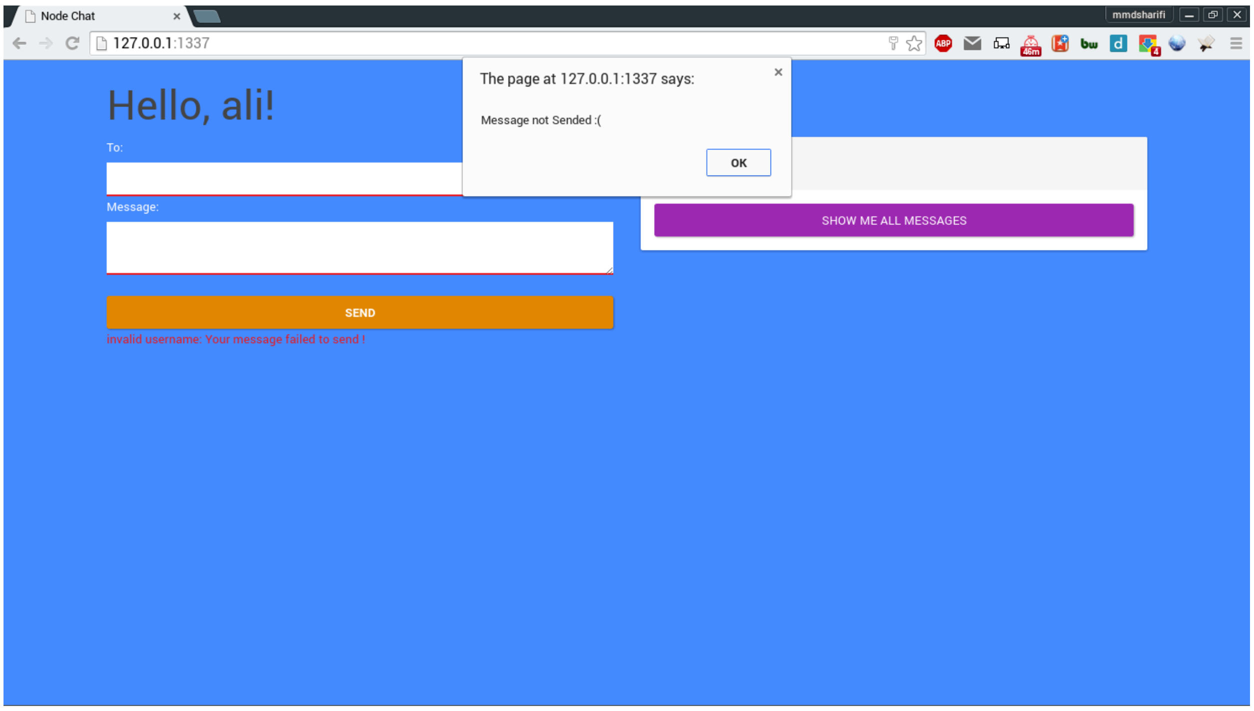

To:

fooo

Message:

fooooooooooo

Image 6 – After user press Enter to send message.


Hello, ali!

To:

Message:

invalid username: Your message failed to send !

The page at 127.0.0.1:1337 says:

Message not Sended :(

Image 7 – resiver not found and fail to send message alert.

Also user can see all messages in his/her inbox,by click the purple button:

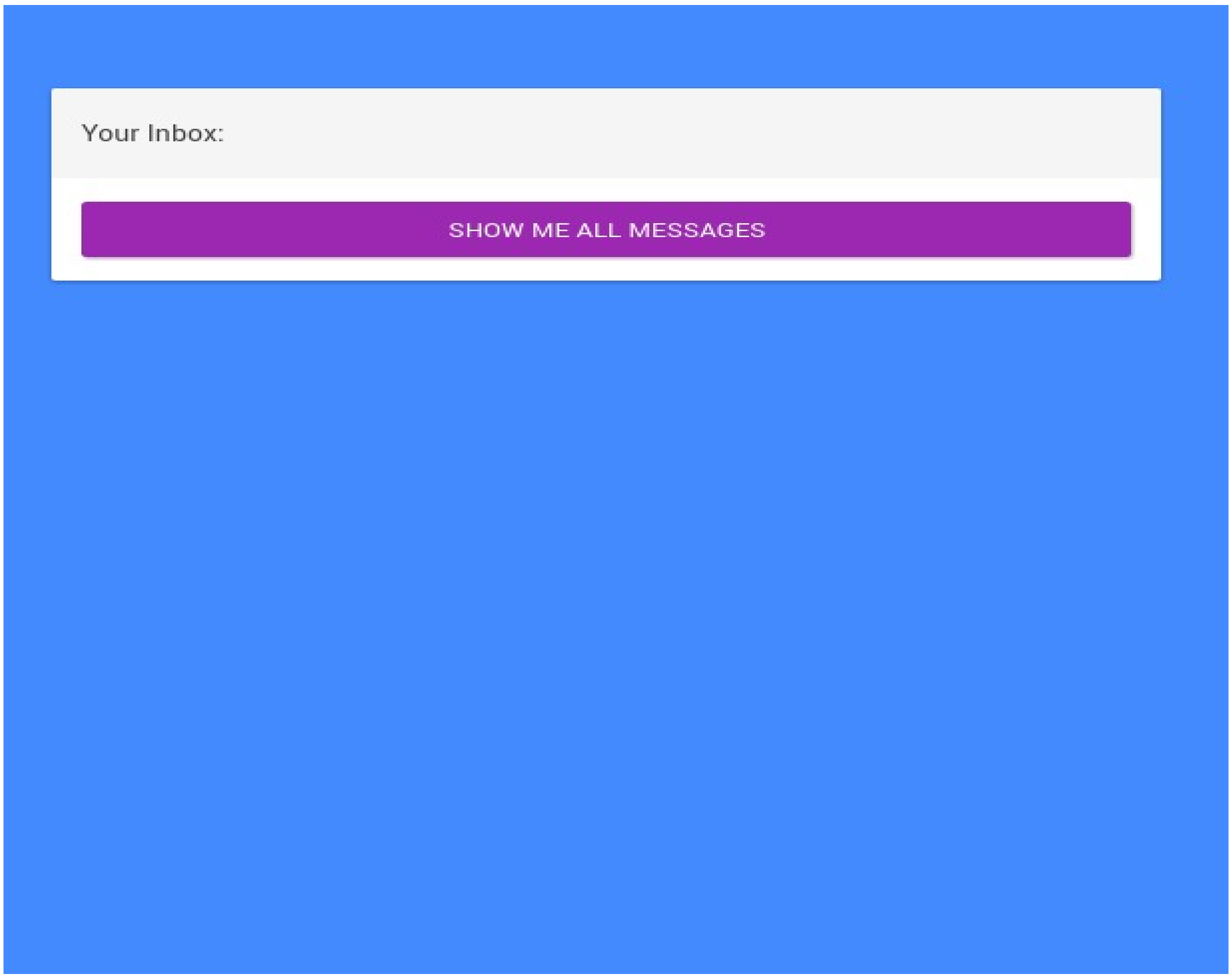
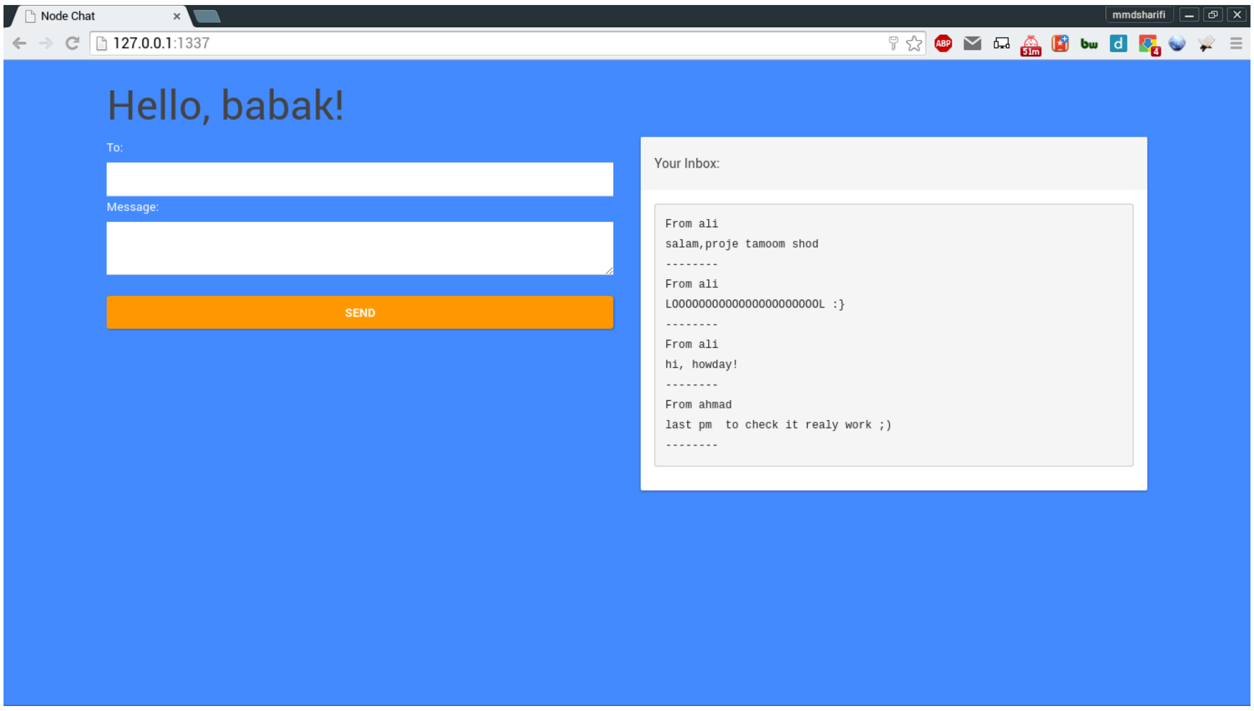


Image 8 – User request to see his/her inbox



```
From ali
salam,proje tamoom shod
--------
From ali
LOOOOOOOOOOOOOOOOOOOOOOL :}
--------
From ali
hi, howday!
--------
From ahmad
last pm  to check it realy work ;)
--------
```

Image 9 - all messages shows .

Back-end and Codes

we have to main files : 1.server.js for server-side codes and 2.cilent.js for client side.

Server.js

We use emit and on methods,for send and receive data with client. And all of code have comments .

Technologies: NodeJS,Express,socket.io.

Client.js

For user interface (UI) we use bootstrap framework with awesome theme. And Jquery for client-side validations , socket.io for sending and receive data.

Technologies: JavaScript,HTML5,CSS3,Bootstrap,jquery.

With best Regards,

Mohammad Sharifi, Amin Ettedi