Peer2Chat - Decentralized Chat App

A project report submitted in partial fulfillment of the requirements for the award of the degree of

Bachelor of Technology

in

Computer Science & Engineering

Submitted by

Dharwish Raj - FIT17CS046
Adhyaksh Guhan - FIT17CS007
Alen Raphaelnotrealnumber - FIT17CS042
Ajay Dijinotrealnumber - FIT17CS045

Federal Institute of Science And Technology (FISAT)®
Angamaly, Ernakulam

Affiliated to

APJ Abdul Kalam Technological University
CET Campus, Thiruvananthapuram

Federal Institute of Science And Technology (FISAT)®

Mookkannoor(P.O), Angamaly-683577

CERTIFICATE

This is to certify that report entitled "**Peer2Chat - Decentralized Chat App** " is a bonafide report of the network lab project (CS334 - Network Programming Lab) presented during VIth semester by **Dharwish Raj(FIT17CS046), Adhyaksh Guhan(FIT17CS007), Alen Raphaelnotrealnumber(FIT17CS042), Ajay Dijinotrealnumber(FIT17CS045)**, in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology (B.Tech) in Computer Science & Engineering.

Dr. Prasad J C

Staff in Charge | Project in Charge | Head of the Department

ABSTRACT

A decentralised application is one that does lies outside the purview and control of a single authority. This enables the app to allow for transfer of data with complete anonimity and interruptions from third-parties. This project uses a peer-to-peer (p2p) network to enable communication between multiple parties in a decentralised fashion.

ACKNOWLEDGMENT

If words are considered as symbols of approval and tokens of acknowledgment , then let the words play the heralding role of expressing our gratitude.

Ms. Anitha P, Chairman, Governing Body-FISAT, who provided us with vital facilities required by the project right from the inception to completion.

We express our deepest appreciation towards **Dr. George Issac**, Principal ,FISAT, for providing amenities, which helped us in the fulfillment of our project.

Dr. Prasad J.C , Head of Department of Computer Science and Engineering, FISAT, guided us and rendered his help in all phases of our project.

We would also like to express our gratitude to **Ms. Sruthy Suresh , Ms. Divya John**, had been a pillar of support for the successful completion of our project.

Last but not the least, we express our sincere gratitude to all the staffs of the Department of Computer Science and Engineering, who helped us in the course of work.

Dharwish Raj(FIT17CS046) Adhyaksh Guhan(FIT17CS007) Alen Raphaelnotrealnumber(FIT17CS042) Ajay Dijinotrealnumber(FIT17CS045)

Contents

Chapter 1

INTRODUCTION

1.1 Overview

Decentralised networks, as opposed to centralised networks, are organised in a much more distributed fashion. Each node within the network functions as a separate authority with independent decision-making power regarding how it interacts with other systems. These networks also distribute processing power and workload functions among connected servers. This creates a lack of any central power that can maintain control over the whole network and control how information flows within it. The P2P based chat app utlises this philosophy and can connect one peer to another to start a conversation in complete anonimity.

Chapter 2

RELATED WORK

- **1. Augur** Augur combines decentralized networking and financial prediction markets to create powerful forecasting. It's built upon the Ethereum blockchain. In its current guise, Augur allows you to make predictions about real-world events not limited to financial markets. The platform turns your prediction into "shares" that other users can buy or sell.

- 2. Golem

  Golem was one of the first global marketplaces for your idle computing power. The platform styles itself as a "global, open source, decentralized supercomputer that anyone can access." What does that mean? Well, it means that if you have any unused computing power, you can lend it to the network. In turn, that unused or idle computing power is available for purchase from the Golem network as part of a combined bundle.

- 3. Aragon

  Aragon is an ambitious decentralized management platform, also built on the Ethereum blockchain. It wants to break down the traditional barriers that restrict the creation and maintenance of organizational structures. In other words, Aragon wants to make it easier to create private Decentralized Autonomous Organizations (DAOs), along with everything you need to succeed. This means arbitration, token management and transfers, role assignments, fundraising, and much more.

- 4. Sia

  Sia is a promising decentralized storage platform that leverages "underutilized hard drive capacity around the world," creating a first-of-its-kind blockchain-based data storage marketplace. The platform turns those empty hard drives into cheap cloud storage that almost anyone can use. Prices are cheap, especially when compared to other major cloud storage providers.

- 5. SAFE Network

  The SAFE Network uses a decentralized approach to protecting consumer data and private communication. SAFE, which stands for Secure Access For Everyone, uses peer-to-peer technology to share that computing power between connected users. This creates a secure private network, rather than relying on centralized servers.

Chapter 3

DESIGN and IMPLEMENTATION

3.1 Design

The announcement broadcast system has been implemented using :

- A socket.io server in the backend to handle incoming messages and broadcast them to all connected clients.
- A JavaScript socket.io client to send messages from the user, and listen to and display messages from other users..
- A web application server to serve the HTML page, the socket.io server and any static files we need.

3.1.1 How this works

The data flows back and forth between the client and the server. When a client sends a message to a server the server receives the message and to broadcast this message to all the clients, we use the io.sockets.emit method.

Basically Socket.IO is of two parts:

- A server that integrates mounts on Node.JS HTTP Server socket.io
- A client library that loads on the browser side socket.io-client

3.1.2 User Interface

The front end bit is done using html for the basic layout and CSS to stylize the html page.

3.2 Implementation

Socket.IO is a bidirectional communication javascript library, Usually the server needs to be refreshed for a message to be delivered to a client but when using socket.IO the communication becomes instantaneous. After creating a basic layout using html and css we move on to the sending broadcasts part. In the file chat.js we query the DOM elements from the html file so that we can create references for them.

- const message refers to the box area where we enter the message
- handle is where the username is entered
- Send sends the message when pressed.
- output is where the messages will be displayed

Now next what happens is when a message is send by a client with a handle name to the server the server accepts this message and emits the same message to all connected clients on the single press of the send button, thus basically broadcasting the message to all connected users.

The emit method basically takes in two arguments.

- The name of the message event
- And the text entered in the chat field.

Next at index.js we receive this message that was emitted by the client.We have to recieve this chat and emt it to all other clients too. For that this small snippet of code is used

```
        socket.on('chat',function(data){
            io.sockets.emit('chat',data);
    });
```

Whats basically happening here is that when a socket connection is esteblished with a client we are using on method to listen to the callback function, this function takes data as a parameter and receives the data we sent and send out the recieved message to all connected sockets using io.sockets.emit.

Next part is displaying the sent chat on all the html pages that have the client for that n innerhtml is used in chat.js

```
        socket.on('chat',function(data){
    feedback.innerHTML = '';
    output.innerHTML +='<li>'+'<div class="chat-body clearfix">'+'<div c
});
```

Chapter 4

TESTING

The project was implemented by testing at multiple stages. First a testing was done as soon as a client could be connected to the server, which displayed the client was connected as soon as a server client connection was established.

Then multiple clients were connected to the same server and tested

After this an app with a similar UI to our familiar chat apps were made, with a handle so that the user can identify who is broadcasting the given message.

Chapter 5

RESULTS

The final result of the project is a broadcasting app that basically comprises of a server and multiple clients. Multiple clients can connect to a single server, can give their own names and also send and receive messages.The broadcasting app uses Socket.io to connect the server to the clients.Such a system can be used in institutions for Broadcasting important information between the faculty and the students using a LOCAL AREA NETWORK.

Chapter 6

CONTRIBUTIONS

Each member of team had contributed equally to the project. The files style.css and index.html was worked on by all four members of the group,each member made changes to html and css file as thought would improve the user experience of the app, The other two main files the index.js and chat.js each person tried a version for themselves and the one that best worked was chosen at the end.

Chapter 7

CONCLUSION

Broadcasting can thus be considered as a relevant and important technology in day today life of small businesses, enterprises, institutions and media companies. The broadcasting app we implement here leverages the technology to make announcements within Organizations easier than what we are used to. This app can be further modified with a login system and with features to enable sending of large multimedia files in the future if necessary which would then enable for a really secure was of communication using the LAN for the respective firm.

Bibliography

[1] Rohit Rai. *Socket. IO Real-time Web Application Development*. Packt Publishing Ltd, 2013.

[2] Jake Spurlock. *Bootstrap: Responsive Web Development*. " O'Reilly Media, Inc.", 2013.

[3] Stefan Tilkov and Steve Vinoski. Node. js: Using javascript to build high-performance network programs. *IEEE Internet Computing*, 14(6):80–83, 2010.

References -[1, 2, 3]

Appendices

Appendix A

Sample Code

A.1 index.js

```
const express = require('express');
const socket = require('socket.io');
let clients = 0;
const app = express();
const server = app.listen(4000,function(){
    console.log("listening to the request on port 4000");
});

//static files

app.use(express.static('public'));

const io = socket(server);


io.on('connection',function(socket){

    socket.on('chat',function(data){
            io.sockets.emit('chat',data);
    });

    socket.on('typing', function(data){
        socket.broadcast.emit('typing', data);
    });
});
```

A.2 chat.js

```
const socket = io.connect('http://localhost:4000');

//Query DOM
const message = document.getElementById('message');
const handle = document.getElementById('handle');
const sendButton = document.getElementById('send');
const output = document.getElementById('output');
const announcements = document.querySelectorAll('.announcements');
const feedback = document.getElementById('feedback');
const rightPanel = document.getElementById('right-panel');
//create date object
const date = new Date().toDateString();

//emit events

sendButton.addEventListener('click',function(){
    if(message.value.length > 0 & handle.value.length > 0){
        socket.emit('chat', {
            message: message.value,
            handle: handle.value
        });
      }
      message.value = "";
});

message.addEventListener('keypress', function(){
    if(handle.value.length > 0){
      socket.emit('typing', handle.value);
    }
  });

//listen for events

socket.on('chat',function(data){
    feedback.innerHTML = '';
    output.innerHTML +='<li>'+'<div class="chat-body clearfix">'+'<div c
});

socket.on('typing', function(data){
    feedback.innerHTML = '<p><em>' + data + ' is typing a message...</em
});
```

A.3 index.html

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <meta name="description" content="Chat">
        <meta name="keywords" content="HTML,CSS,JavaScript,SOCKET.IO">
        <meta name="viewport" content="width=device-width, initial-scale=1
        <title>Broadcast App</title>
        <script src="/socket.io/socket.io.js"></script>
        <!-- Latest compiled and minified CSS -->
        <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/boots
        <link href="https://maxcdn.bootstrapcdn.com/font-awesome/4.4.0/css
        <link rel="stylesheet" href="https://use.fontawesome.com/releases/
        <link href="/styles.css" rel="stylesheet" >
    </head>
    <body>
        <div class="container-fluid header-container px-0">
            <div class="row mx-0">
                <div class="col-sm-12 px-0">
                    <div class="row">
                        <div class="col-sm-3">
                        </div>
                        <div class="col-sm-6">
                            <br> <br>
                            <h1 class="header-text">Broadcast app</h1>
                        </div>
                    </div>
                </div>
                <!-- end of col-sm-12 -->
            </div>
            <!-- end of row -->
        </div>
        <!-- end of container> -->
         <div>
         <p id="feedback"></p>
        </div>
        <div class="container-fluid" id="output-container">
            <div class="row no-gutters">
                <div class="col-sm-3 side" id="left-panel"></div>
                <div class="col-sm-6" id="main-output">
                    <div class="row output-row no-gutters">
                        <div class="col-sm-12" id="output">
                            <p class="announcements"></p>
                        </div>
```

```
                </div>
                <!-- end of row -->
                <div class="row no-gutters">
                    <div class="col-sm-10" id="changecolour">
                        <textarea id="message" type="text" placeholder="Mes
                    </div>
                    <!-- end of col-sm-6-->
                    <div class="col-sm-2 no-gutters" id="action-here">
                        <input id="handle" type="text" placeholder="Handle"
                        <button class="btn-btn-success btn-block" id="send"
                    </div>
                    <!--end of col-sm-12 -->
                </div>
                <!-- end of nested row -->
            </div>
            <!-- end of col-sm-8 -->
            <div class="col-sm-3 side" id="right-panel"></div>
        </div>
        <!-- end of row -->
    </div>
    <!-- end of container -->
    <script src="/chat.js"></script>
    <!-- jQuery library -->
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jq
    <!-- Latest compiled JavaScript -->
    <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.0/js/bo
  </body>
</html>
```

A.4 style.css

```
@import url("https://fonts.googleapis.com/css?family=Montserrat:400,400i
body{
  font-family: Montserrat, sans-serif;
  color: rgb(255, 255, 255);
  background-color: rgb(253, 253, 253);
  overflow-x: hidden;
}

.header-container{
  background-image: url("images/kidda.png");
  height:150px;
  border-top: 3px solid rgb(255, 255, 255);

}
.header-text{
  text-transform: uppercase;
  font-weight: 900;
  opacity: 0.7;
  color: #000000;
}

#main-output{
  background-color: rgb(0, 0, 0);
  height: 100%;
}
#output{
  height: 450px;
  overflow-y: scroll;
  width: 50%;
  background-color:#bbbbbb;
  border-bottom: 3px solid rgb(78, 78, 78);
}

.chat
{
    list-style: none;
    margin: 0;
    padding: 0;
}


.chat li .chat-body p
{
    margin: 0;
    color: #777777;
```

```
}

.panel .slidedown .glyphicon, .chat .glyphicon
{
    margin-right: 5px;
}
.chat li
{
    margin-bottom: 10px;
    padding-bottom: 5px;
    border-bottom: 1px dotted rgb(169, 178, 179);
}

#message {
    width: 100%;
    height: 100%;
    background-color:#ffffff;
    color: #000000;
}

#send{
  background-color: #3b5998 ;
  color: #FFFFFF;
  margin-left: auto;
  margin-right: auto;
  width: 100%;
  height: 70%;
  margin-top: 0px;
  border: none;
  opacity: 0.7;
}

#changecolour
{
  background-color: #929292;
  height: 75px;
}

#handle{
  width: 100%;
  background-color:rgb(255, 255, 255);
  opacity: 0.9;
  margin-top: 0px;
  margin-left: auto;
  margin-right: auto;
  margin-bottom: 0px;
  height: 30%;
  color:#000000;
}
```

```
#date{
font-style: oblique;
color:rgb(0, 0, 0);
font-size: 14px;
}

#style-handle{
   color: rgb(0, 0, 0);
}
.announcements{
     color: #000000;
     text-transform: full-width;
}
#right-panel{
   padding-top: 3px;
   padding: 30px;
   text-align:center;
   color: #ffffff;
   border-top: 2px solid #7289DA;
}
#left-panel{
   padding-top: 3px;
   text-align:center;
   color: #7289DA;
   border-top:2px solid #7289DA;
}
#action-here
{
   background-color: #929292;
}
```