TECHNIUM
SOCIAL SCIENCES JOURNAL

Vol. 37, 2022

A new decade for social changes

ISSN 2668-7798

www.techniumscience.com

TECHNIUM SOCIAL SCIENCES JOURNAL

Technium Social Sciences Journal
Vol. 37, 142-152, November, 2022
ISSN: 2668-7798
www.techniumscience.com

Design and Implementation of Peer-to-Peer Video and Chat Communication

Muath Abdullah Saeed[1], Naktal Moaid Edan[2]

[1]Software Department, College of Computer Science and Mathematics, The University of Mosul, Iraq, [2]Software Department, College of Computer Science and Mathematics, The University of Mosul, Iraq.

muataljambl@gmail.com, naktal.edan@uomosul.edu.iq

Abstract. Web Real-Time Communication (WebRTC) technology permits real-time media and data exchange between browsers. A connection is proven via a detection procedure named signalling. However, signalling has no exact definition in WebRTC. This paper aims to design and implement WebRTC chat, video communication, and recording between peers (browser-to-browser) in a real application using Chrome and Firefox based on peer-to-peer topology. Thus, a signalling channel between two peers for chat and video conferencing using the following: Socket.io mechanism, Node.JS platform, and Express.JS has been produced and performed. Including, Vanilla JS, Axios (HTTP JSON) and JavaScript as the main programing language utilized. The results of this work have ensured that a signalling channel has been built and implemented.

Keywords: Web Real-Time Communication (WebRTC); Session Description Protocol (SDP); Socket.io protocol; Node. JS, JavaScript (JS); Local Area Network (LAN)

1. Introduction

Web Real-Time Communication (WebRTC) was announced in 2011 as an HTML5 standard and industry-wide open-source collection of JavaScript libraries. WebRTC offers real-time voice, video and data communications between browsers and is done with mobile applications and interfaces. Also, it enables the combination of voice and video contact within websites without installing any additional plugins on the website [1]. In addition, is considered one of the important tools as long as it offers peer-to-peer direct file sharing between browsers without the requirement of facilitating servers. WebRTC API has three core basics: (a) PeerConnection creates straight messages between peers, (b) DataChannel is a data transport service through bidirectional linking between peers, and (c) MediaStream manages and generates audio and video streams [2]. In [3], the authors indicated that the WebRTC application can be written using HTML5, CSS, and JavaScript language, and also can cooperate with numerous web browsers through the WebRTC (Application Programming Interface (API), thus it enables correct practice.

Accordingly, [4][5] emphasised that WebRTC requires signalling mechanism support to establish communication across multiple users or devices. However, the (IETF) and (W3C)

Technium Social Sciences Journal
Vol. 37, 142-152, November, 2022
ISSN: 2668-7798
www.techniumscience.com

have not yet decided on a formal protocol for testing WebRTC and regulating communication architecture [6]. Figure (1), shows the WebRTC signalling mechanism architecture.

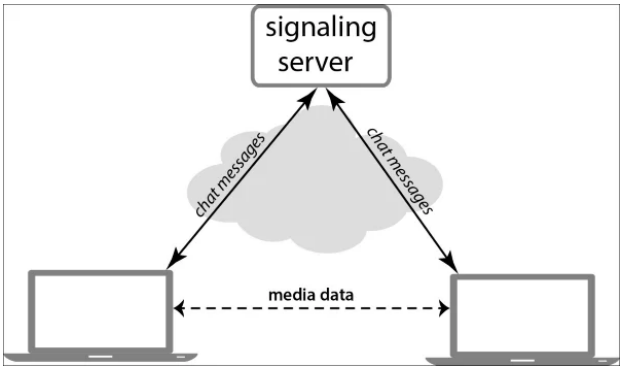

Figure (1): the WebRTC signalling mechanism architecture.

As a result, signalling is WebRTC's major implementation issue, so the peer discovery approach relies on signalling to detect any existing peers and then communicate between browsers [6]. On other hand, WebRTC employs the Interactive Connectivity Establishment (ICE) technique for navigating Network Address Translation (NATs), which depends on Session Traversal Utilities for NAT (STUN) to determine the exterior location assigned to a particular peer and Traversal Using Relay NAT (TURN) to assist STUN and overcome the Symmetric NAT limitation by establishing a connection with a TURN server and retransmitting all details via that server [7].

Streams with audio and video materials can be sent and received using WebRTC. During a call, streaming could be introduced and withdrawn at any moment. They can also be independent or mixed. Using Real-Time Communication (RTC) for collaboration often entails recording a computer's desktop as a video feed and adding voice and video from the webcam and microphone. Codec independence is a general trait of the WebRTC protocol. Nevertheless, the WebRTC configure file abilities with relation to multimedia codecs have been committed to regulation and are properly described. The fundamental transport has been built to accommodate any codec type [7].

The advancement of digital technology has made communication simpler than ever. Applications exist that facilitate communication by sending texts, photos, and other content from one participant to another. There are a number of these applications that can be used to reach a broad audience. These programs frequently appeal to the regular populace and improve society overall. Few programs promote communication among institutions, and other groups that have a limit on the total of users and maintain the privacy of the content shared between users [8]. In order to solve this issue and give users a far better system that holds text at a distance and is contained within a border, the online chatting application was created. React JS, a JavaScript library for creating user interfaces was used to create this web application. Mobile or single-page applications can be built using React as a foundation. And it has made use of an API which manages all of the chat. As explained in [7] that the year 2020 was unique. Covid-19 has raised awareness of the need for RTC as individuals all over the world have discovered new ways to work, learn, and communicate with friends and family via video chat.

The main goal of this work is to create a physical implementation of WebRTC chat, video communications, and recording employing socket.io, Node.js platform, Express.JS,

Technium Social Sciences Journal
Vol. 37, 142-152, November, 2022
ISSN: 2668-7798
www.techniumscience.com

Vanilla JS, Axios (HTTP JSON) and JavaScript language. This paper's explanation will assist interested users in learning WebRTC functionality and interacting between client and server. Besides, this demonstration is useful for users who want to design WebRTC chat and video signalling mechanisms in real-time applications.

This paper is structured as follows: part 2 contains definitions. Part 3 contains survey comments on some WebRTC concerns (related work). Part 4 describes the implementation. Part 5 explains the evaluation process. Finally, in part 6, the conclusion is discussed.

2. Definitions

2.1. WebSocket (WS)

It is a web-based protocol that enables a web client and a remote/webserver to connect in a continual and bi-directional TCP or secured Transport Layer Security (TLS) fashion. It keeps the connection available to both sides for communication. As a result, customers will have a large number of binary or textual messages running across both ways [9].

2.2. Socket.io and Node.JS

It is a JavaScript-written transport protocol that enables real-time, two-way interaction across web browsers and a host. Node.js served as the foundation for Socket.io and served as the main backend ever since [10]. Since its inception, Socket.io has used Node.js as its primary backend [11]. Node.JS is Google's open-source increased JavaScript interpreter, which is built as a server platform. Node.JS is a good framework for developing network web applications. Because of its non-blocking input/output and event-driven approach, Node.JS can manage maximum throughput and performance [12].

2.3. Vanilla JS

A quick, light, cross-platform platform for creating amazing, powerful JavaScript apps is called Vanilla JS. Also, every line of code in the platform is maintained by the Vanilla JS team, who also put a lot of effort into keeping it simple and understandable. Indeed, nowadays Vanilla JS is utilized more than Prototype JS, MooTools, YUI, and Google Web Toolkit on websites [13].

2.4. Express.JS

Express is a thin foundation that appears at the top of the web server capability of Node.js to increase its APIs and improve collaborative new structures. With middleware and routing, it is simpler to structure the functionality of the application. It improves HTTP classes in Node.js with useful functions, and also it makes displaying dynamic HTTP objects easier [14].

3. Literature review

Many designers strained to create a signalling mechanism for chat and video conferencing. The purpose behind that is there are web applications that permit users to exchange messages and share photographs, text or videos with a substantial audience online. A picture-sharing logic was incorporated into the architecture of some of these programs. This means that any user who owns the shared image will get a notification when it is shared. This may be a massive method to stay in touch with friends and family, as well as a chance to learn a lot about different people. This application is entirely web-based, so clients do not require any suitability of the device for space to store; all they need is a reliable internet connection. Also, this helps the users who do not have a compatible device with more storage and space that might control the database for content sharing and file receiving. Not only that but also, [15] claimed that chat application is beneficial for many different aims, such as speed (while it

Technium Social Sciences Journal
Vol. 37, 142-152, November, 2022
ISSN: 2668-7798
www.techniumscience.com

allows user to connect with others in real-time), familiarity (people can send messages and deliberate about whatever), convenience (can send a message or reply to a query from a computer or device, change the topic, and carry on with what it was happening), segmented target advertising (can publish any information with anyone, either privately or publicly), file storage and sharing, employee engagement, privacy, easy to set up, and less troublesome (Messaging through a chat program is less distracting than making a phone call. Multiple calls at once are more challenging to manage than multiple chat windows.). Also, the webRTC video, chat and screen-sharing applications were proposed, however, the signalling protocol does not explain and the authors' used TURN and STURN servers for signalling [16]. In other words, chat software is a service that allows users to interact with one another over computers or mobile devices, such as Facebook Messenger, WeChat, WhatsApp, and numerous other chat programs. Besides, video applications will be grown up to $19.8 billion by 2023. So, this is not astonishing when people reflect on what way video conferencing assistance industries is faster decisions and appropriate involvement, cooperative effort and document distribution, more contented staff and advanced work maintenance, lower costs, and unique contacts [17].

4. Implementation

The actual application was set up to execute WebRTC chat and video communication through LAN of (wired and wireless) network and using Chrome and Firefox. As stated below, this architecture can be separated into three types: chat and video including signalling:

4.1. Chat Communication

This research has been built based on a specific structure. For instance, Express.js, node.js, socket.io, RTCDataChannel, server.js, and so on. Figure (2), shows the screenshot of the main environment. Messages are sent via an “RTCDataChannel” which uses part of the index.html interface to show sent and received messages, a text box to write messages and a send button.

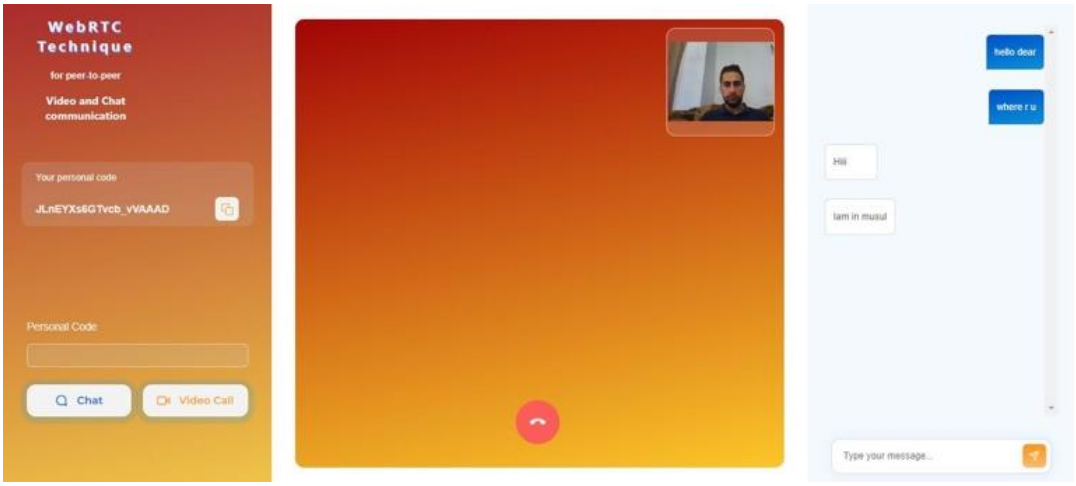

Figure (2): the screen shoot of the chat.

4.1.1. To create and display a message should keep an eye on the following steps

1. main.js file

It has created a function named “newMessageInput” that takes the value from the textbox of the message input that is specified with ID (new_message_input) and then executes the “sendMessageUsingDataChannel” function from inside the webRTCHandler file.

Technium Social Sciences Journal
Vol. 37, 142-152, November, 2022
ISSN: 2668-7798
www.techniumscience.com

2. WebRTCHandler.js file

It has been configured the data channel API with the following functions:

- **onDataChannel:** to run the API of the data channel.
- **onOpen:** this function is used to tell the server that the client is ready to receive messages through the data channel.
- **onMessage:** this function receives the message and changes its syntax from JSON to JavaScript format. Also, create a function called sendMessageUsingDataChannel to send messages by taking the message and changing its format from javaScript to JSON format.
-

4.1.2. To view messages in the recipient's destination

1. ui.js file: a function is created to display messages sequentially and control the direction of the message through the file element, which contains two functions right/lift. If the message is sent, it displays it on the right, and if the message is received, it displays it on the left.

2. element. file: it has created a div(display area) to show messages. Figure (3), shows the streaming of signalling.

Various libraries have been used in this project for the chat, such as Express.js (to configure the server and make it easier to be used with node.js. Also, socket.io (for client communication with the server relies on events to transfer information between the two parties (signalling).

4.2. Video Communication and Signalling Technique

The signalling system in this study employs a socket.io mechanism based on WebSocket for chat and video connections. It created four types of control messages in this signalling solution: Initiator receiving broadcasting stream, "peerChannel", and interchange Session Description Protocol (SDP). Peer X will submit the "request" to the server at first, and then after receiving the control message receiving the contact broadcasting stream, it will have the server generate the user media. Peer X now waits for Peer Y to respond. Peer Y will transmit a reply signalling message until it is activated, and if it is discovered that it is not the originator, it will then get "peer Channel" and then "became access media stream". Both sides can then begin constructing a peer-to-peer connection. The offer and response messages are sent in SDP format, which includes information on the type of media, CODECs, RTP (Real-Time Protocol), RTCP (Real-Time Control Protocol), and all other attributes that can be used in a media session.

Technium Social Sciences Journal
Vol. 37, 142-152, November, 2022
ISSN: 2668-7798
www.techniumscience.com

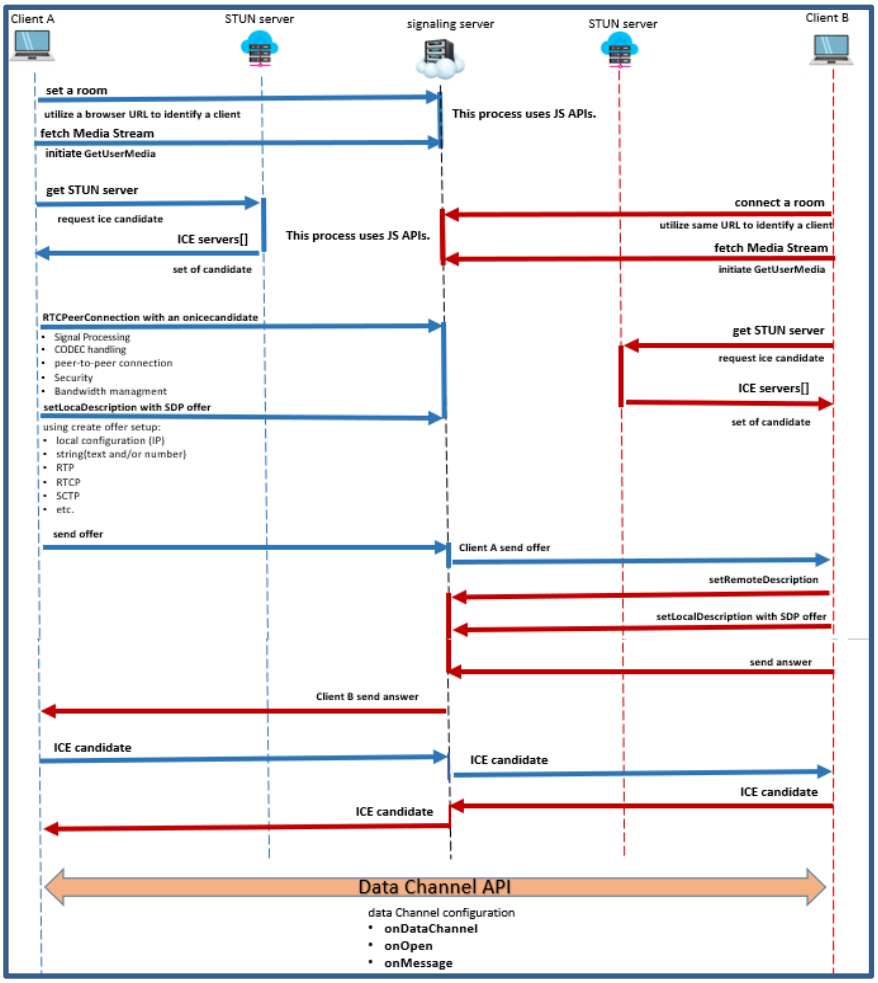

Figure (3): the data flow of the signalling mechanism for a chat.

4.3. Other Techniques used in both Chat and Video

The node_modules is a file that enables npm data that is downloaded to be saved on the used PC for supporting the application to be run. In addition, Package.json is an essential component of many applications where the code is based on the node environment. Including, it records important metadata about the requested project before deploying to npm, and also defines the functional attributes of the project that npm uses to install dependencies, run scripts, and define the entry point to our package. Thus, the interconnection was established via a LAN network (wired and wireless). Including, many functions have been created and set up in the main HTML of this implementation, with establishing chat communication between two distinct users (PCs)/mobiles employing Chrome and Firefox (audio and video), activating full-screen, employing volume slider, and taking a screenshot. The main web consists of some major files as the following:

1. Index.html: The main interface is divided into three parts:

The left part contains a socket ID representing personal code, especially for contact and contact identification by the recipient and a text box to enter the ID for connectivity as well as buttons to determine the type of connection video, chat or chat only.

The middle part displays the contact picture for both the caller and the recipient and the call buttons such as disconnect, mute, camera off, screen share and screen recording.

Technium Social Sciences Journal
Vol. 37, 142-152, November, 2022
ISSN: 2668-7798
www.techniumscience.com

The right part has space to view messages sent and received and a text box at the bottom to enter messages and send them via the send button next to the box insert text.

2. App.js: This is a backend file for the project where the primary server is built using express.js library, a free library that provides ready tools that facilitate the configuration of the server and contains many features that are useful for developers to create their apps and contain the HTTP library.Folder js contains several files as follows:

A. Main.js: This is the main file which is front end application where all the functions used in the project are connected and called and sent to the server which is the bridge between client and server and it sends offers and receives an answer to configure the connection via browser and send messages directly.

B. Constant.js: Contains constant values of application such as connection type (video - chat), offer/answer status (accept/reject - available/not available) and contact status (available/not connected).

C. Store.js: This file is used to store preliminary values implemented at the start of the application and also includes functions to update the values placed if later changed.

D. wss.js: This file is used to set up and configure WebSocket by the client and set up an event to transfer offers/answers.

E. Element.js: When you start the connection, a pop-up interface appears from the caller, showing the type of connection (video/chat) and buttons to answer or reject the connection, either from the recipient, showing an interface containing the type of connection with the button to reject the connection.

F. webRTCHandler.js: This file is used to configure and setup contact via webRTC such as using getUserMedia to access the connected camera and microphone using RTCPeerConnection to open contact with the recipient directly and use RTCDataChannel to send and receive messages via chat, set up offer and answer, convert the screen to screen sharing and return it video.

G. Ui.js: Used to manage changes in the user interface based on actions that make any change to button status and user interface.

H. Recording.js: This file is used to record the screen and use the library MediaRecorder to set up the recording as well as the file containing settings and selecting codecs and video storage format.

The exchange of local and remote characteristics takes place before a connection is created (the audio and video media information) and chat. The (SDP) will be used to describe how the signalling request and reply are exchanged.

First, a peer uses the "setLocalDescription()" function to set the local description, then uses the RTCPeerConnection create Offer () method to transmit the session description to Peer Y over the signalling server. Second, Peer Y uses the "setRemoteDescription()" method to set the description that Peer X delivered as a remote description. Peer Y uses the "setLocalDescription()" method to set the local description, then uses the RTCPeerConnection "createAnswer()" function to make an answer, and delivery channels this event description to Peer X. Peer X uses the "setRemoteDescription()" method to set the Peer B connection summary as the remote description. The server, for example, will permit WebRTC peer-to-peer reliable communication via (URL http://localhost:3000/) if installed within an internal network.

Technium Social Sciences Journal
Vol. 37, 142-152, November, 2022
ISSN: 2668-7798
www.techniumscience.com

5. Evaluation

Based on using Chrome and Firefox, the implementation results have proved that using Chrome demonstrated a quality higher than Firefox. So, using Firefox presented that the communication was taking time more than Chrome and the communication was not constant as it was using Chrome. Figures (4, 5, 6, 7 and 8) show the designed application of chat and video conferencing as shown in the following:

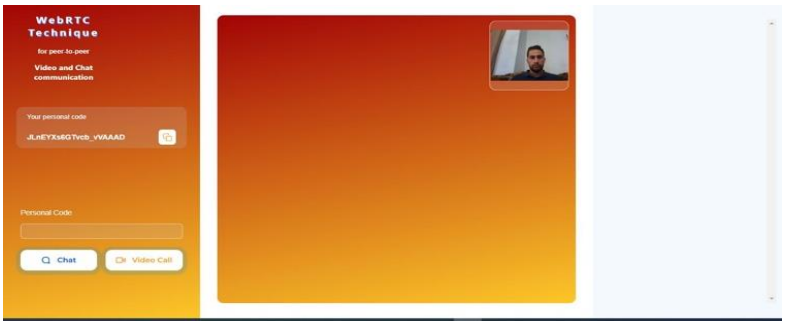

Figure (4): the main interface of the start application.

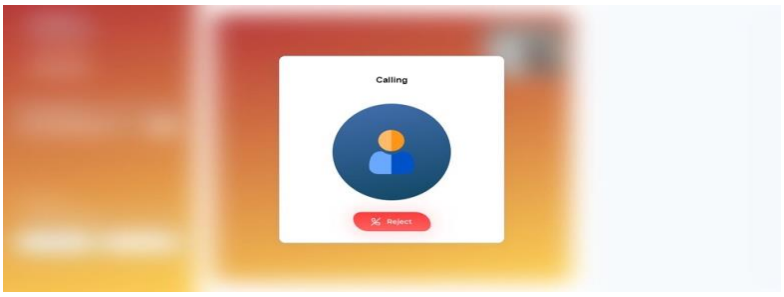

Figure (5): begin the video call on the caller's side.

Figure (6): begin video call on the callee side

Technium Social Sciences Journal
Vol. 37, 142-152, November, 2022
ISSN: 2668-7798
www.techniumscience.com

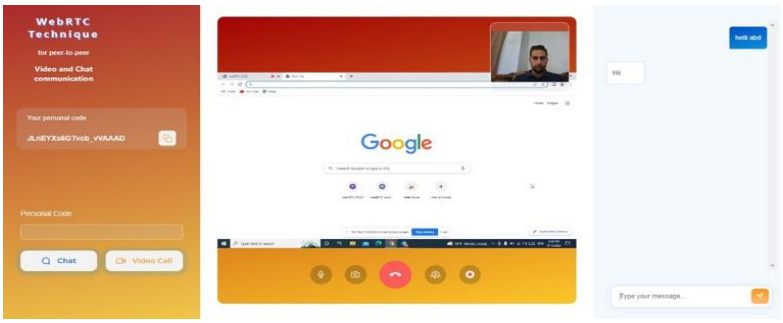

Figure (7): a caller Sharing the screen

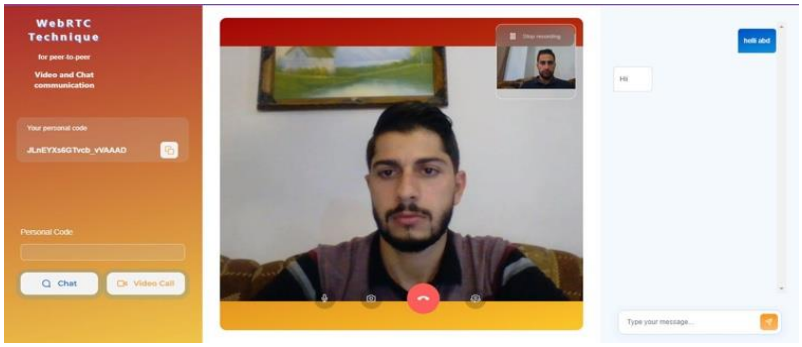

Figure (8): the recording screen

People are seeking new ways to connect in real-time since connections are more crucial than ever. Using JavaScript APIs, WebRTC is a cutting-edge technology that enables real-time audio, video, and data transfer across web browsers without the requirement for a plugin (Application Programming Interface). The WebRTC framework, HTML5, and Node.js server addresses are used in this study to describe a web peer-to-peer real-time connectivity that enables users to communicate across a channel of communication with high-speed data transmission. The outcome indicates that the system is reliable and stable. The following outcomes were attained after the WebRTC video chat system was finished:

1. A web-based video conferencing system was developed.
2. Users have access to voice and video calls.
3. Users can interact with one another while they are in identical communication.
4. The individual who starts the chat session is capable of answering the phone.
5. To use this, the system lessens the need for physical exertion. The usage of video conferencing technology replaces in-person meetings.
6. It enables clear and uncomplicated communication.
7. The user can call or attend from any place at any time.
8. Users can record the calls and have their screenshots at any time to save the data.
9. Users can get chat with each other.

Based on using Chrome and Firefox, the implementation results have proved that using Chrome demonstrated a quality higher than Firefox.

6. Conclusion

Web Real-Time Communication (WebRTC) is a new standard for supporting RTC between users using different web browsers without the need for additional software. However, there is a significant hurdle in implementing WebRTC: it does not specify a

communication protocol across various browsers, which prohibits WebRTC from working properly owing to incompatibility issues with multi-browser connections. This work used the socket.io protocol, node.js (as a server to fully interact between two separate browsers), and express.js, overcoming the aforementioned challenge. It also designed and deployed the WebRTC video call, chat messages, screen sharing and records application, which allows for bi-directional communication over LAN (wired and wireless) network using Chrome and Firefox. The goal of this research is to lessen the effort and challenges associated with communicating while also creating a video. calls and texts, video chat, file sharing, and recording are all possible during a video conference.

7. Acknowledgment

This research was funded by the College of Computer Sciences and Mathematics/software department to sponsor the first author to pursue his master's research.

References

[1] A. Paudyal, "Developing Video Chat Application with ReactJs And WebRTC," Metropolia University, 2021. [Online]. Available: https://www.theseus.fi/bitstream/handle/10024/500565/Paudyal_Abhinav.pdf?sequence=2

[2] E. A. Tarim and H. C. Tekin, "Performance evaluation of WebRTC-based online consultation platform," *Turkish J. Electr. Eng. Comput. Sci.*, vol. 27, no. 6, pp. 4314–4327, 2019, doi: 10.3906/ELK-1903-44.

[3] Z. T. Nayyef, S. Faris Amer, and Z. Hussain, "Peer to Peer Multimedia Real-Time Communication System based on WebRTC Technology," *Researchgate.Net*, vol. 7, no. February, pp. 125–130, 2018.

[4] H. V. Cola. Cristian, "On multi-user web conference using WebRTC," in *18th International Conference on System Theory, Control and Computing (ICSTCC)*, 2014, pp. 430–433. doi: 10.1109/ICSTCC.2014.6982454.

[5] A. Albas and G. Auguets, "WebRTC," Politècnica de Catalunya, 2016. [Online]. Available: https://upcommons.upc.edu/bitstream/handle/2117/106694/alex.albas_117680.pdf

[6] R. Raswa, S. Sumarudin, and E. Ismantohadi, "WebRTC Signaling Mechanism Using npRTC Topology for Online Virtual Classroom," in *Proceedings of the 5th FIRST T1 T2 2021 International Conference (FIRST-T1-T2 2021)*, 2022, vol. 9, pp. 264–270. doi: 10.2991/ahe.k.220205.047.

[7] N. Blum, S. Lachapelle, and H. Alvestrand, "WebRTC: Real-Time Communication for the Open Web Platform," *Commun. ACM*, vol. 64, no. 8, pp. 50–54, 2021, doi: 10.1145/3453182.

[8] S. Jagtap, F. Pathan, and S. Jadhav, "WEB CHAT STATION – A REVIEW," *Int. J. Eng. Appl. Sci. Technol.*, vol. 6, no. 11, pp. 33–35, 2022.

[9] B. Sredojev, D. Samardzija, and D. Posarac, "WebRTC technology overview and signaling solution design and implementation," in *38th International Convention on Information and Communication Technology, Electronics and Microelectronics, MIPRO - Proceedings*, 2015, no. May, pp. 1006–1009. doi: 10.1109/MIPRO.2015.7160422.

[10] M. Grinberg, "socketio Documentation," 2017. [Online]. Available: https://media.readthedocs.org/pdf/python-socketio/latest/python-socketio.pdf

[11] R. Rai, *Socket. IO Real-time Web Application Development*. BIRMINGHAM - MUMBAI: PACKT, 2013. [Online]. Available: http://books.google.com/books?hl=en&lr=&id=YgdbZbkTDkoC&oi=fnd&pg=PT9&dq=Sock

et+.+IO+Real-time+Web+Application+Development&ots=TVve7ogNAQ&sig=K0Sf8yhHjskpEYta799u3UtMcY4
[12] A. S. Karl. Bissereth, Billy B. L. Lim, "An Interactive Video Conferencing Module for e-Learning using WebRTC," in *Internetional Conferences*, 2014, pp. 1–4. [Online]. Available: http://www.cita.my/cita2015/docs/shortpaper/31.pdf
[13] Shehroz Azam, "What is 'Vanilla JavaScript'?," *linkhint*, 2021. https://linuxhint.com/what-is-vanilla-javascript/ (accessed Jun. 04, 2022).
[14] Shubhadarshie, "Node.js vs Express.js," *GeeksforGeeks*, 2022. https://www.geeksforgeeks.org/node-js-vs-express-js/ (accessed Jul. 01, 2022).
[15] Amanda Holmes, "11 Reasons Why A Chat Application Is Great For Business," *HeroBot*, 2021. https://herobot.app/messenger-chatbot/chat-application/ (accessed Jun. 15, 2022).
[16] A. Kasetwar, N. Balani, D. Damwani, A. Pandey, A. Sheikh, and A. Khadse, "A WebRTC Based Video Conferencing System with Screen Sharing," *Int. J. Res. Appl. Sci. Eng. Technol.*, vol. 10, no. 5, pp. 5061–5066, 2022, doi: 10.22214/ijraset.2022.43595.
[17] Galyna Yavorskaya, "Advantages and Disadvantages of Video Conferencing," *my own conference*, 2021. https://myownconference.com/blog/en/advantages-disadvantages-video-conferencing/ (accessed Jun. 20, 2022).