MINAR International Journal of Applied Sciences and Technology
ISSN: 2717-8234
Article type: Research Article

Received: 28/09/2022 **Accepted:** 01/11/2022 **Published:** 01/12/2022

DESIGN AND IMPLEMENT VIDEO AND CHAT APPLICATION USING MESH NETWORK

Muath Abdullah SAEED [1]

University of Mosul, Iraq

Naktal Moaid EDAN [2]

University of Mosul, Iraq

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 the real application using Chrome and Firefox based on mesh 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, Axios (HTTP JSON), and JavaScript as the main programing language utilised. 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).

http://dx.doi.org/10.47832/2717-8234.13.7

[1] muataljambl@gmail.com, https://orcid.org/0000-0002-1499-9965

[2] naktal.edan@uomosul.edu.iq

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 as shown in Figure (1).

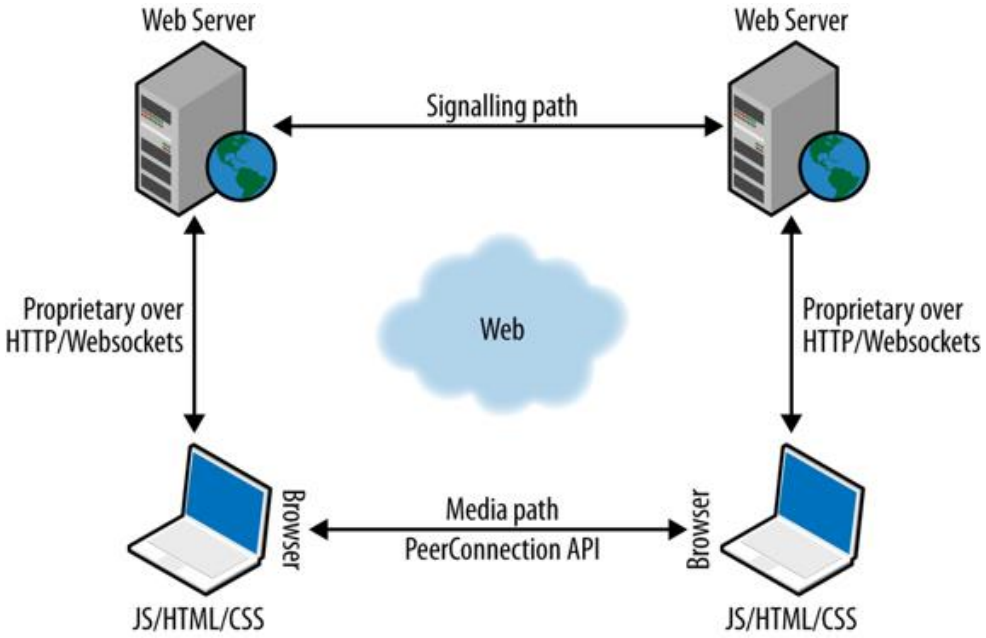

Figure (1), The architecture via the WebSocket

Accordingly, [4][5] emphasised that WebRTC requires signalling mechanism support to establish communication across multiple users or devices. However, the (IETF) and (W3C) have not yet decided on a formal protocol for testing WebRTC and regulating communication architecture [6]. Figure (2), shows the WebRTC signalling mechanism architecture.

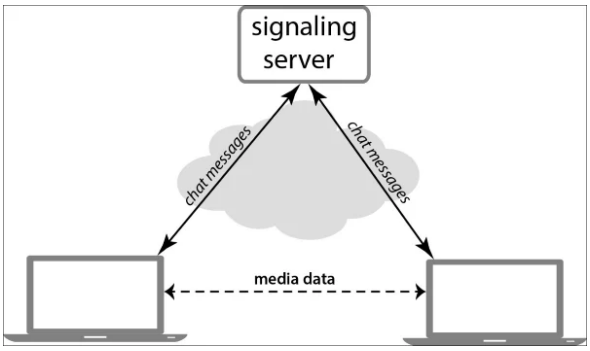

Figure (2), The WebRTC signalling mechanism architecture.

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]. Figure (3), shows the WebRTC library components.

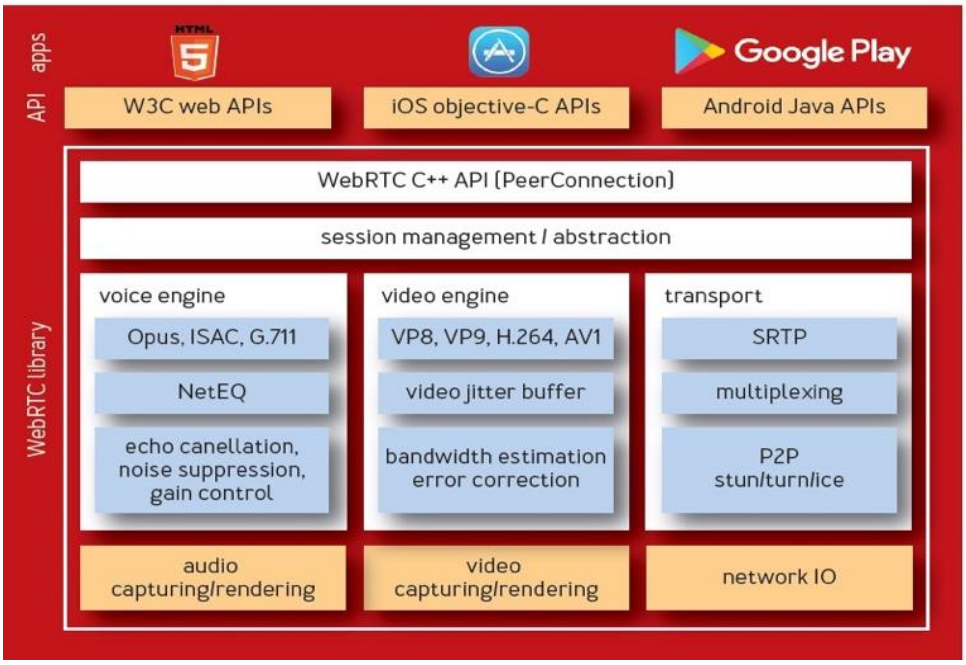

Figure (3), The WebRTC library components.

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, Vanilla JS, Axios (HTTP JSON), React User Interface libraries 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 II contains definitions. Part III contains survey comments on some WebRTC concerns (related work). Part IV describes the implementation. Part V explains the evaluation process. Finally, in part VI, the conclusion is discussed.

2. Definitions

A. 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].

B. 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].

C. 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 [13].

D. jQuery.js

It is one of the light, fast and feature-packed JavaScript libraries, facilitating the modification of HTML documents, dealing with events, and creating animations, making the use of javascript easier [14].

E. Bootstrap.css

It is a ready-made library that helps developers to coordinate their applications without the need to write code from scratch because it contains ready-to-use tools to save developers time and effort [14].

3. LITERATURE REVIEW

Many developers tried to establish a signalling mechanism for chat and video conferencing. The reason behind that is there are web applications that allow 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, such as Spark Matrix. 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 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 application 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 conferencing becomes widely used everywhere, especially in the enterprise market, education, etc. Thus, 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 while it is faster decisions and appropriate involvement, cooperative effort and document distribution, more contented staff and advanced work maintenance, lower costs, and unique contacts [17]. However, using HTTP to achieve real-time is a basic approach because HTTP headers consume extra data that is typically useless for real-time applications. As well as, the application's usage of continuous polling has the potential to increase communication overhead. In contrast, But, [18] emphasized that WebRTC has not yet specified signalling management designed to facilitate users to tweak, repurpose current systems, and design their signalling; also, to reduce redundancy and improve flexibility with existent technologies [19]. But, WebRTC uses the WebSocket protocol which is designed for it because it provides a full-duplex, bidirectional line of communication for use that is carried out with a specific web socket and can create scalable, real-time Web services. As well as, the application's usage of continuous polling has the potential to increase communication overhead. Accordingly, using the WebSocket protocol offers a full-duplex bi-directional channel that is accomplished with a single socket, it enables a robust real-time online application [20]. Based on the mentioned connected opinions; understandably chat and video applications are requested. Nevertheless, signalling between browsers is not defined in WebRTC [21][22][23]. Therefore, in this research chat, video and recording applications with a signalling mechanism based on WebRTC will be designed and implemented.

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:

c. Chat Communication

This research has been built based on a specific structure. For instance, Express.js, node.js, socket.io, JQuery, Bootstrap, index.html, server.js, and so on. Figure (4), shows the screenshot of the main environment. Messages are sent via a socket which uses part of the index.html interface to show sent and received messages, a text box to write messages and a send button as shown in figure (4).

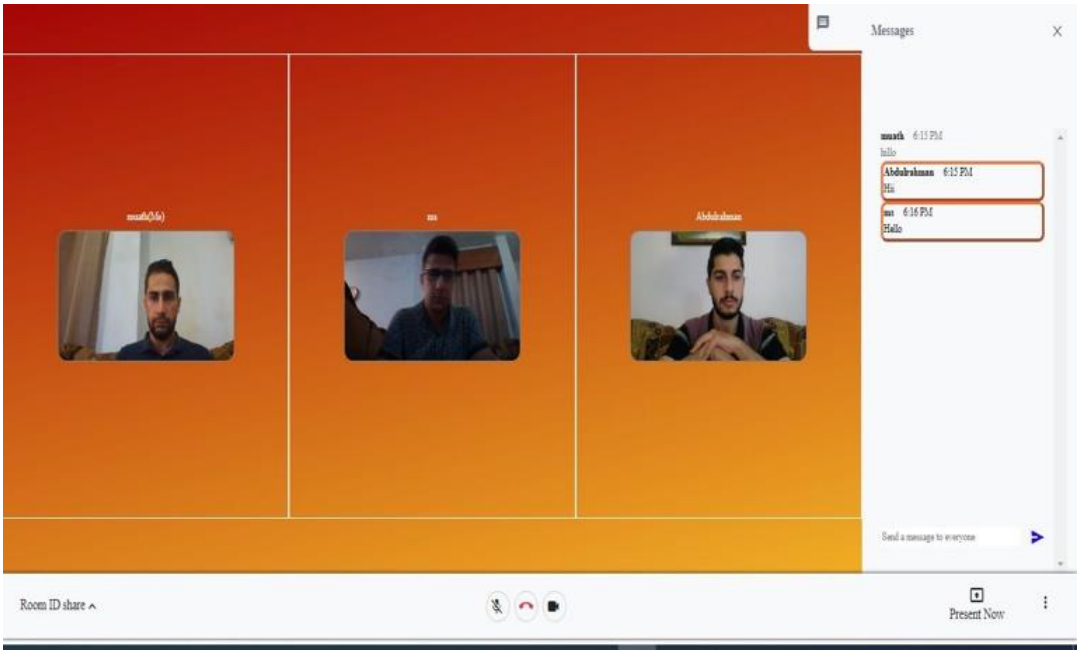

Figure (4), shows a screenshot of the chat communication between 3 peers.

In the app.js file, the value is taken from the text box and sent via socket to server.js in turn server.js sends it to all callers whose id is stored in the peer_connection_id array. Figure (5) shows the main structure with the data flow of the chat part.

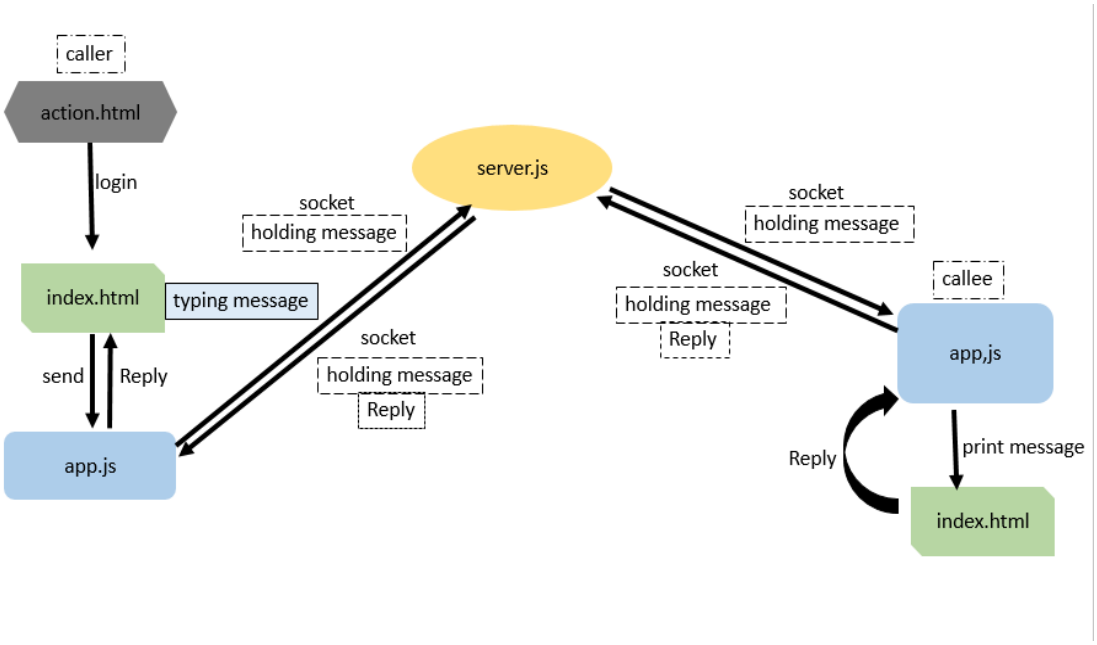

Figure (5). The main interface with the data flow of the chat part

Various libraries have been used in this project for the chat part, 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). Besides, JQuery (to facilitate linking and programming between HTML and JavaScript language), and Bootstrap (for supporting using of the library to format pages). Not only that but also, specific steps were used to achieve text chat communication as the following:

- Label of user ID
- Textarea: id= user-Id
- Label Other ID
- Textarea: id= other-Id
- Button id= connect
- Label Enter Message
- textarea id= user-Message
- Button id= send (pre id= messages)

d. 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.

e. 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 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. Action.html: it is the main interface that is used to start the call or to join in for a pre-existing connection. The connection is initiated in two ways, the first is to select a random room ID, and the second is to choose a room ID by the caller, and there are two buttons for a new connection or for joining the connection
2. Index.html: it is the interface that interacts with callers. It contains three parts: the left part has the largest area to display the callers' image, the right part is a drop-down part to display messages between callers, and the lower part contains buttons to control the call.
3. **server.js**: This file is the backend of the application where the server is built using the node.js environment, express.js library and the socket.io library based on the WebSocket protocol to communicate between the server/client and transfer the information required to configure the connection based on the roomsID, userID and socketID.
4. public\js\app.js: this file is the frontend of the application and it consists of two main parts, the first is to initialize the initiator connection. It contains variables that store the meeting_ID and user_ID information. It also contains an event to tell the server to connect, disconnect, and notify others to connect and exchange SDP with the server. It contains a function to change the state of the buttons when clicked and a function to add a connection New and screen recording settings, as well as a function for exchanging chat between callers through the server by sending socket handle messages The second part is about communicating with the server by fetching data from other clients and sending it to the client side. It contains a variable that stores the id of the connection and arrays to store both the information (callers - remote video stream - remote audio stream) and information about the connection state and contains the serverProcess function that passes The information to the server and eventProcess is the function that is used when the server sends a specific event and the webRTC properties such as getUserMedia are used to take permission to use the camera and audio and peerConnection to set up the connection and send an offer and receive an answer through the server.
5. public\css\style.CSS: is the basic file for formatting the interfaces and adding colours and sizes for our application

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.

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 (6, 7, 8, 9 and10) show the designed application of chat and video conferencing as shown in the following:

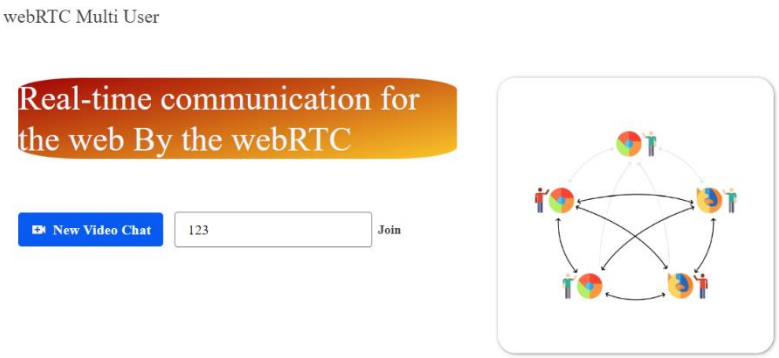

Figure (6), shows a screenshot of the main interface for starting or joining the session.

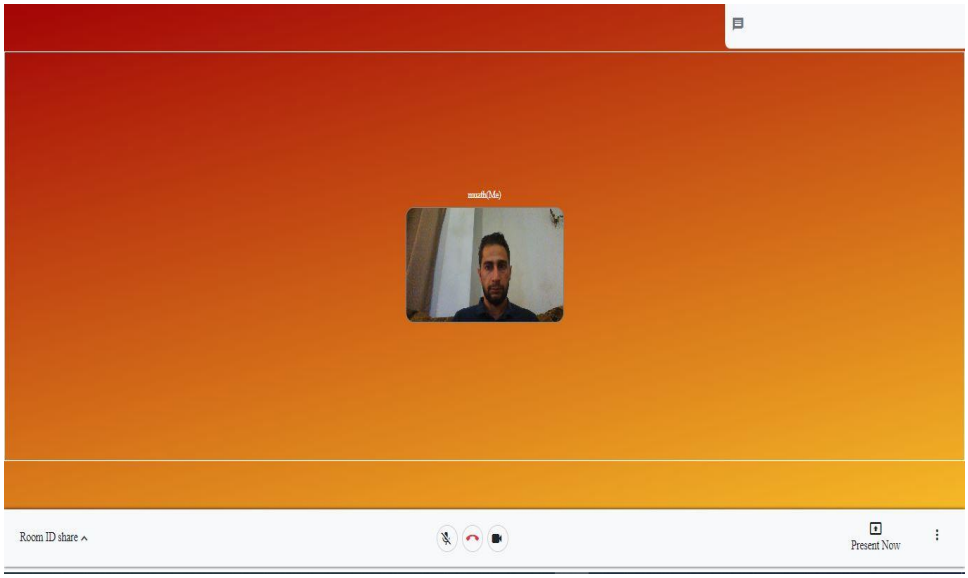

Figure (7), shows a screenshot of a video call after accessing the camera and microphone from the main initiator.

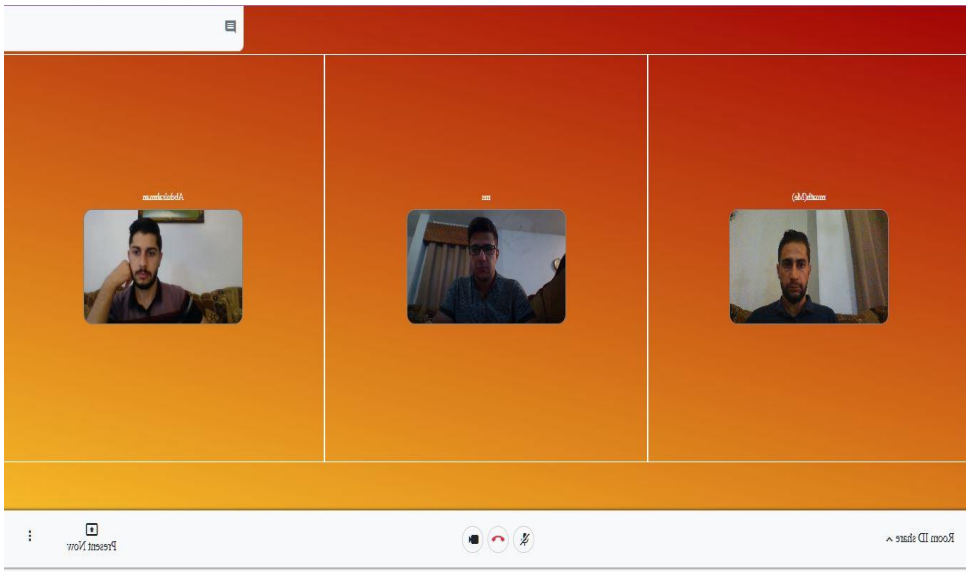

Figure (8), shows a screenshot of joining different participants in the video call among 3 peers.

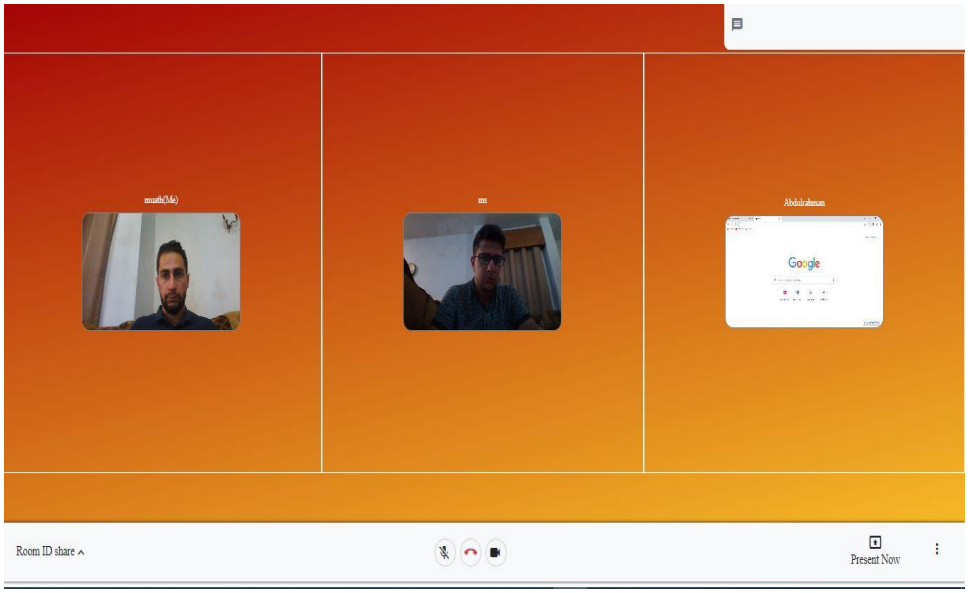

Figure (9), shows a caller Sharing the screen between 2 peers.

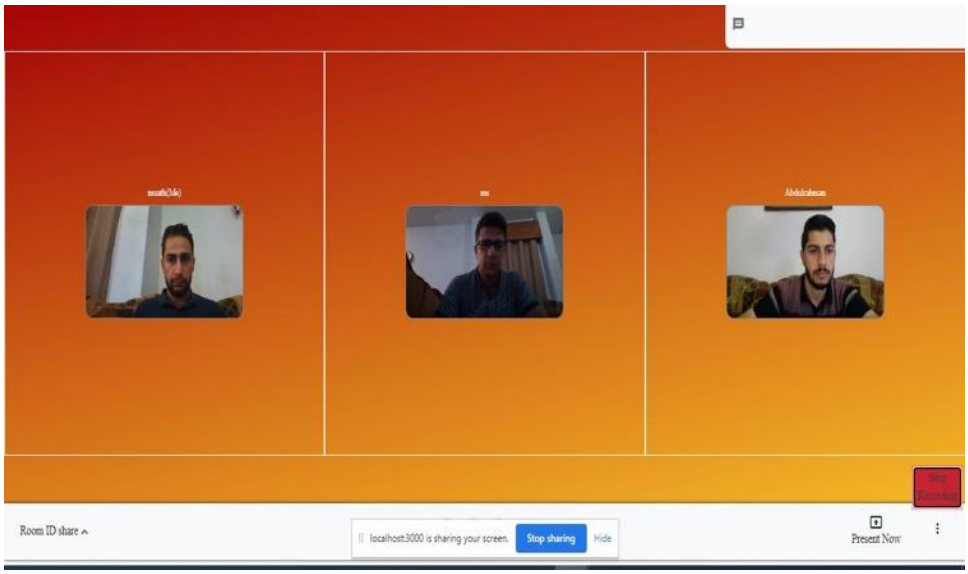

Figure (10), shows the recording screen among 3 peers.

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 can make and join rooms.
3. Users have access to voice and video calls.
4. Users can interact with one another while they are in the identical communication room.
5. The individual who starts the chat session is capable of answering the phone.
6. To use this, the system lessens the need for physical exertion. The usage of video conferencing technology replaces in-person meetings.
7. It enables clear and uncomplicated communication.
8. The user can call or attend from any place at any time.
9. Users can record the calls and have their screenshots at any time to save the data
10. This application does not need external software or licence

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 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.

REFERENCE:

[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=Socket+.+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] Shubhadarshie, "Node.js vs Express.js," *GeeksforGeeks*, 2022. https://www.geeksforgeeks.org/node-js-vs-express-js/ (accessed Jul. 01, 2022).

[14] M. Mohan, "Difference Between JavaScript and jQuery," *GreeksforGeeks*, 2022. https://www.geeksforgeeks.org/difference-between-javascript-and-jquery/ (accessed Sep. 15, 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).
[18] F. Paganelli, T. Ambra, A. Fantechi, and D. Giuli, "Formalizing REST APIs for web-based communication and SIP interworking," *Telecommun. Syst.*, vol. 66, no. 1, pp. 75–93, 2017, doi: 10.1007/s11235-016-0271-2.
[19] T. S. Edan. N, Al-sherbaz. A, "Desing and Implement A Hybrid WebRTC Signalling Mechanism for Unidirectional & Bi-directional Video Conferencing," *Int. J. Electr. Comput. Eng.*, vol. 8, no. i, pp. 1–8, 2018, doi: 10.11591/ijece.v8i1.pp390-399.
[20] K. E. Ogundeyi and C. Yinka-Banjo, "WebSocket in real time application," *Niger. J. Technol.*, vol. 38, no. 4, p. 1010, 2019, doi: 10.4314/njt.v38i4.26.
[21] B. Y. Julian. Jang-Jaccard, Surya. Nepal, Branko. Celler, "WebRTC-based video conferencing service for telehealth," *Computing*, vol. 98, no. 1–2, pp. 169–193, 2016, doi: 10.1007/s00607-014-0429-2.
[22] N. Edan and S. A. Mahmood, "Design and implement a new mechanism for audio, video and screen recording based on WebRTC technology," *Int. J. Electr. Comput. Eng.*, vol. 10, no. 3, pp. 2773–2778, 2020, doi: 10.11591/ijece.v10i3.pp2773-2778.
[23] N. Edan and E. Y. Abdullah, "Design and implementation of a novel secured and wide WebRTC signalling mechanism for multimedia over internet," *Int. J. Electr. Comput. Eng.*, vol. 10, no. 5, pp. 5430–5435, 2020, doi: 10.11591/ijece.v10i5.pp5430-5435.