INTERNATIONAL JOURNAL OF Telecommunications
AIR DEFENSE COLLEGE

# Real-Time Facial Expression Recognition and Speech Transcripts over an on-premise Video Conference Application

**Sally Eltenahy[1, *], Nihal F. F. Areed[1], Marwa Obayya[1], and Fahmi Khalifa[1]**

**Citation:** Eltenahy, S.; Areed, N.; Obayya, M.; Khalifa, F. Inter. Jour. of Telecommunications, **IJT'2022, Vol. 02, Issue 02, pp. 1-14, July 2022**. https://ijt-adc.org/articles/2805-3044/849368

**Academic Editor: Youssef Fayad**

**Received: 2022-04-26.**

**Accepted: 2022-08-18.**

**Published: 2022-08-21.**

**Publisher's Note:** The International Journal of Telecommunications, IJT, stays neutral regard-ing jurisdictional claims in published maps and institutional affiliations.



[1] Mansoura University Electronics and Communication Engineering Department, Faculty of Engineering, Mansoura University, Mansoura 35516 Egypt (email: sallyahmed2011@gmail.com, nahoolaf@mans.edu.eg, omnya@mans.edu.eg, fahmikhalifa@mans.edu.eg)
* Correspondence: sallyahmed2011@gmail.com; Tel.: +201096175303

**Abstract:** Since Covid-19 pandemic outbreak, organizations and individuals have had to use video conference applications increasingly. However, the commercial video conference applications are expensive, and feature limited. This paper discusses how to enable organizations to host on-premise video conference applications. Then, it explores assisting organization's stakeholders with making decisions based on facial expressions of video conference attendees. Moreover, it facilitates transcribing speech into text to enable deaf persons to participate in online conferences. Technologies and tools used in addressing these challenges respectively are: (i) Web Real Time Communication (WebRTC) project, (ii) Tensorflow.js library, (iii) and Web Speech Application Programming Interface (API). This paper depends on integration between a collection of technologies, libraries, standards, and protocols. Most of them can be managed using JavaScript framework. Hence, load of the performance is distributed on each client-side device. The proposed on-premise video conference application has been enhanced through including facial expression recognition with 66% high accuracy while the speech-into-text feature with Word Error Rates (WER) are 0 and 0.12 for British English and Egyptian Arabic, respectively.

**Keywords:** WebRTC; Video conferencing; Facial Expression Recognition; Speech Recognition; Computer Vision; ML; TensorFlow.js; OpenVidu; Speech-to-Text

## 1. Introduction

On the one hand, this paper is an extended work of our three conference papers [1-3]. Indeed, video conference applications are widely used in various aspects of our lives, including, but not limited to, business meetings, e-Learning, and healthcare. However, the commercial video conferencing applications are costly and feature limited. Some corporations and organizations may require an on-premise video conferencing application because of their security and privacy concerns. Moreover, some corporations and organizations need to make decisions based on expressions feedback during the video conference whether natural, sad, happy, fearful, or surprised. This is beneficial in a number of scenarios: one example is when an instructor wants to know whether each student is paying attention; when presales or salesmen want to know if a client is enthusiastic; and when an employer or supervisor wants to analyze the right performance based on customer and staff facial expressions. On the other hand, according to a World Health Organization's report, over 5% of the total population endure hearing loss. Deaf people need to have the ability to engage in video conference meetings and understand what is being said.

In 2011, Web Real Time Communication (WebRTC) technology [4-6] was released as an open-source project built in modern browsers for real-time communication. It provides JavaScript application program interfaces

(APIs) that enable application developers to create applications for transmitting and receiving video, audio, and data in real-time. As a result, there are many open source WebRTC-based video conference applications available to be deployed on-premise and be developed. This paper presents one of these applications and the two proposed enhancements executed on it. The first proposed solution is facial expression recognition in real-time and storing results in the database. The second is transcription of speech in real-time.

This paper is indexed as follows: In Section 2, we review literature work related to hosting on-premise video conferencing applications, facial emotion recognition, and speech-to-text conversion. Section 3 presents the data about the technologies used: WebRTC technology, TensorFlow.js[7], and Web Speech API[8], respectively. Section 4 presents the methodology required to integrate facial emotion detection function over video conference in real-time, as well as the methodology used to integrate speech-to-text feature through video conference in real-time. Evaluation of the performance is presented in Section 5 while Section 6 holds the conclusion.

## 2. Related Work

In this Section, we introduce reverent literature works in three main parts: (1) Comparative analysis among common WebRTC-based applications, which allows for choosing the most suitable on-premise video conferencing applications in resource utilization; (2) facial emotion recognition, which detects facial expressions; and (3) speech recognition, which converts speech into text. Further detail comes next.

### *2.1. Comparative Analysis among WebRTC-based Applications*

WebRTC enables developers to make applications that send and receive data, video, and audio in real-time. As a result, many open-source WebRTC-based apps are being created by application developers all around the world. Typically, open-source software provides the capability for developers to deploy self-host, develop, and redistribute that software without the need for more effort [9]. The conducted comparative analysis, that has been executed by Eltenahy, et al [1], is done among Big Blue Button [10], JITSI MEET [11], and OpenVidu-call [12] to choose optimal resource-efficiency and also customization capability. BigBlueButton, JITSI MEET, and OpenVidu-call are the most common on-premise open-source WebRTC-based video conference applications used. BigBlueButton is licensed under LGPL and is supported by BigBlueButton Inc [10]. JITSI MEET is licensed under Apache License v2 [13] and is supported by 8x8, Inc [11]. As well, OpenVidu-call is under the Apache License v2 [14] and is also supported by the Ministry of Economy, Finance and Competitiveness of Spain, and the European Regional Development Fund [15]. Hence, it is allowed to be reused and upgraded up to the latest releases. In the experiment, the three applications have been deployed in three virtual machines following the instructions in their official websites. Two users had accessed each virtual machine and a peer-to-peer communication had been established. During that, observations of resource utilizations in each machine have been recorded and compared [1]. Figure 1 represents a comparative analysis regarding consumption of power and memory and the rate of transmitting and receiving in each virtual machine. According to the results, the Openvidu-call application's machine has the best resource usage.

Hence, the OpenVidu framework has been selected to fulfill our operational needs. It is also compatible with common programming languages through using APIs. Therefore, the OpenVidu framework has been used as the core of the two proposed solutions, facial expression recognition and speech-into-text conversion over video conference in real-time, and will be presented in this paper.

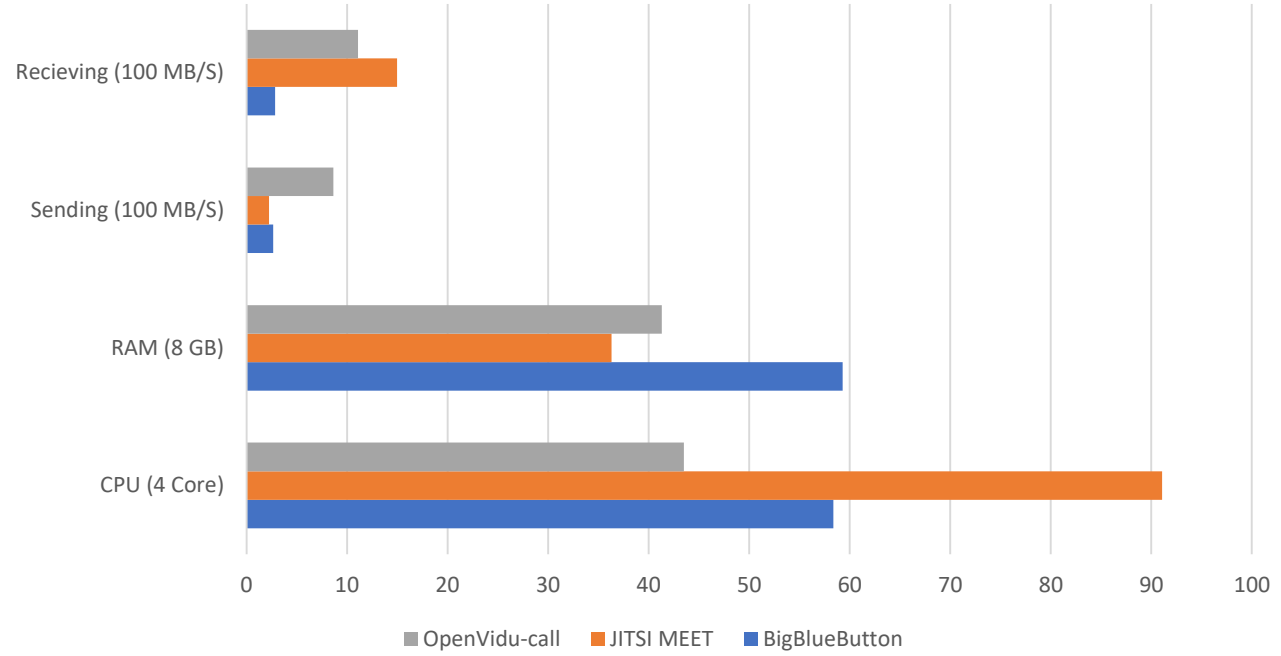


**Figure 1.** The resource utilization in BigBlueButton, Jitsi-meet, and OpenVidu-call applications.

### *2.2. Facial Emotion Recognition*

Facial Emotion Recognition (FER) is a stream of work in which researchers in the area of computer vision, affective computing, human–computer interaction, and human behavior deal with the prediction of emotions using facial expressions in images or videos [16]. FER is a broad and well-studied research stream which has recently received a lot of attention due to the rising development of deep learning-based methods [9]. Audience's impressions, such as sad, happy, natural, or surprised, are very helpful information for who is responsible for the meeting or the stakeholders to make a decision based on the video conference participants' feedback. The advantages of FER in decision-making have been realized as a result of the experiment findings that had been executed by Peter Gloor et al [17]. In this experimental setting, 35 people worked in eight teams over Zoom in a one-semester course on Collaborative Innovation Networks, with bi-weekly video meetings where each team presented their progress. Zoom face video snapshots were used to track emotions, with facial emotion recognition recognizing six different moods (happiness, sadness, fear, anger, neutrality, and surprise). Certain combinations of emotions expressed by the presenter or demonstrated by the audience have been proven to be predictors of higher perceived presentation quality. Therefore, it is approved that facial emotion recognition over video conference can assist in decision-making. In their experiment, Zoom application was used, and Zoom face video snapshots were used to analyze recorded video's facial expressions. This paper, though, has differently performed a real-time analysis throughout the integration within a WebRTC-based video conference application with a pre-trained facial expression recognition model.

### *2.3. Speech Recognition*

Automatic Speech Recognition (ASR) technologies have been evolving steadily in recent years. Monthly progress has been facilitated by the massive development of cloud-based speech recognition services. Among the various providers of speech recognition APIs (Google Web/Cloud Speech API, IBM Watson, Microsoft Azure,

Amazon Alexa API, Nuance Recognizer, Wit.ai, Houndify), some provide the possibility to use their speech recognition API also within personal products or with own speech files [18]. To find the best cloud-based speech recognition service, Ingo Siegert et al [9] analyzed the performance of Google Speech API [19], IBM Watson [20], and Wit.ai [21] on German samples of high-quality spontaneous human-directed and device-directed speech, as well as noisy device-directed speech, over an eight-month period. The experimental results [18] have shown that word error rate is quite low in Google Cloud Speech and Wit.ai with 7% to 8% absolute. Furthermore, there is only a 3.3% difference in word error rate between Google Cloud Speech and Google Web Speech. Hence, Google Web Speech service has been used, in the proposed solution, to achieve an acceptable word error rate and can be used for personal purposes to transcript participants' speech.

## 3. Data

This section includes main technologies, libraries, and APIs that are the core of this study. All of them can be managed using JavaScript language.

### *3.1. WebRTC Technology*

WebRTC is a collection of standards, protocols, and application programming interfaces (APIs) that enables secure peer-to-peer (P2P) high-quality audio, video, and data sharing among browsers [22]. Those standards and protocols are built in modern browsers and are supported by Apple, Google, Microsoft, and Mozilla, amongst others [23]. It is called a browser-to-browser communication which is used for providing P2P connections in real-time without the need to install any plugins or tools on user's side. The open-source implementation and tutorials for the WebRTC platform can be found on WebRTC's official website [24]. Indeed, real time communication opens the door to a whole new range of peer-to-peer communication, including text-based chat, file sharing, screen sharing, gaming, sensor data feeds, audio calls, video chat, and more [5]. Figure 2 shows WebRTC project architecture in modern browsers [24]. WebRTC technology is divided into two layers: the WebRTC library layer and the API apps layer. The WebRTC library layer is being developed by browser makers. It includes three components: voice engine, video engine, and transport component. The voice engine and video engine comprise codecs for decreasing latency, enhancing quality, and reducing background noise. The transport component handles secure real-time bidirectional communication across the network address translators (NATs) and firewalls, while the API apps layer hides the complicated work and enables the application developers to deal with the WebRTC protocols and standards through JavaScript APIs. Furthermore, WebRTC needs four types of servers [6]: a web server, a signaling server, a media server, and a NAT traversal server. The web server, such as NGINX [25] or Apache [26], is used to host web application's files. The signaling server exchanges session data, such as IP address, codec, bandwidth, and media type. The media server, meanwhile, is used for providing a toolbox of capabilities which include group communications, recording, routing, transcoding, and mixing recording, routing, transcoding and mixing [27]. Furthermore, the NAT traversal servers, which are STUN [28] or TURN [29] servers, are used to enable connection without having to reveal the IP addresses and to overcome the firewall blocking issues [30].

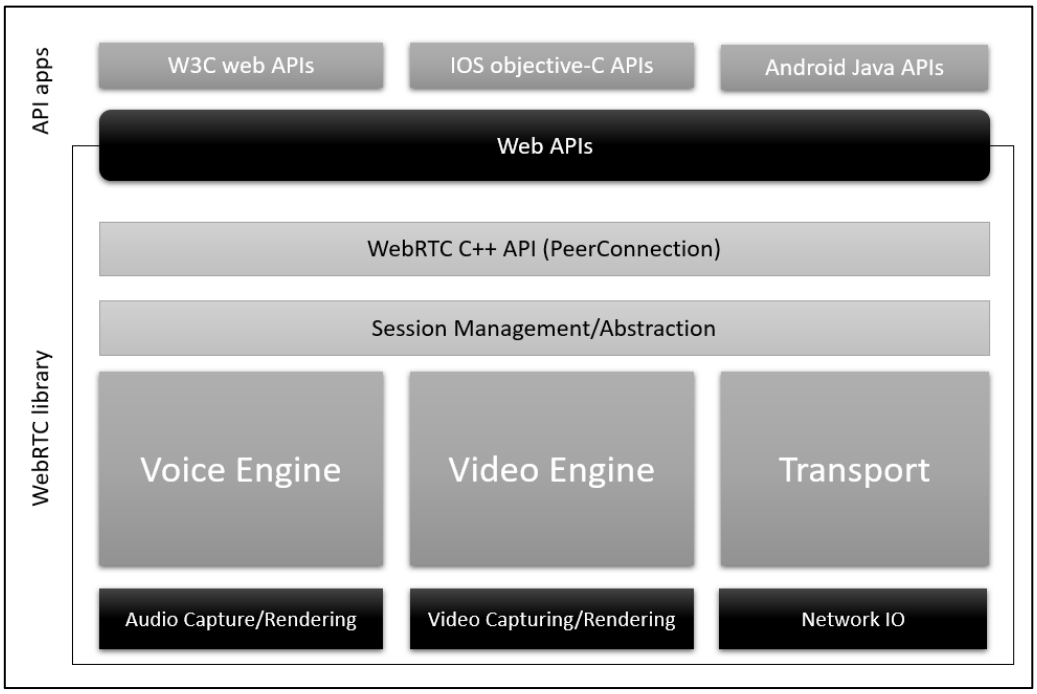


**Figure 2.** The WebRTC architecture.

### *3.2. TensorFlow.js library*

TensorFlow.js is a JavaScript library for building machine learning models in the browser or within the Node.js [31], whose stable version was released in 2019. TensorFlow.js hides the complexity level of machine learning such as Tensors or Optimizers behind APIs in modern browsers. Hence, developers can build on top of TensorFlow.js with face-api.js [32] as an example. The face-api.js is a JavaScript API for face detection and face recognition. Pre-trained models included in faceapi.js are: Face Detection Models, 68 Point Face Landmark Detection Models, Face Recognition Model, Face Expression Recognition Model, and Age and Gender Recognition Model. In this study, these models are executed to detect facial expressions over the video conference application in real-time.

### *3.3. Web Speech API*

According to the related work section, Web Speech API has been recommended for speech-into-text conversion. Work on the Web Speech API can be traced back, at least, to the end of 2011.[33] Web Speech API provides two functions: speech recognition, and speech synthesis. Speech recognition is the capability to receive the context of a voice from an audio input, such as the device's microphone, then checked by a speech recognition service against a list of grammar, while response results are like a text string. Speech synthesis is the capability to receive synthesizing text, convert it into speech, and play it using the device's speaker or audio output. In other words, it allows voice to be converted to text and vice versa. There are many NPM packages which use Web Speech API. The most used one is react-speech-recognition [34]. It simplifies the development of speech-driven web apps and allows developers to focus their efforts on voice command handling. It is a React Hook that converts speech captured from the microphone into text and makes it available to your React components.[34] Furthermore, react-speech-recognition can be configured to convert several human languages, as presented in [35], into text. Hence, the react-speech-recognition has been chosen in this study to convert the speech into text of the video conference in real-time.

## 4. Methodology

The comparative analysis mentioned in Sub-Section 2.1 approved that the OpenVidu framework is a good choice for building on-premise video conference application with less resource utilization and high performance compared to others. The OpenVidu-Call-React application, one of the demos of the OpenVidu framework, has been selected to be the base of the two novel aspects. Figure 3 represents the OpenVidu-Call-React application's components as in the official website [36]. It is written in ReactJS[37]. The stream component is the main component that controls the Webcam and the microphone and streams video and voice in the browser. Hence, the two proposed solutions' modification and integration have occurred in the stream component. In the following, facial expression recognition and automatic speech recognition over video conference solutions in real-time will be represented in detail.

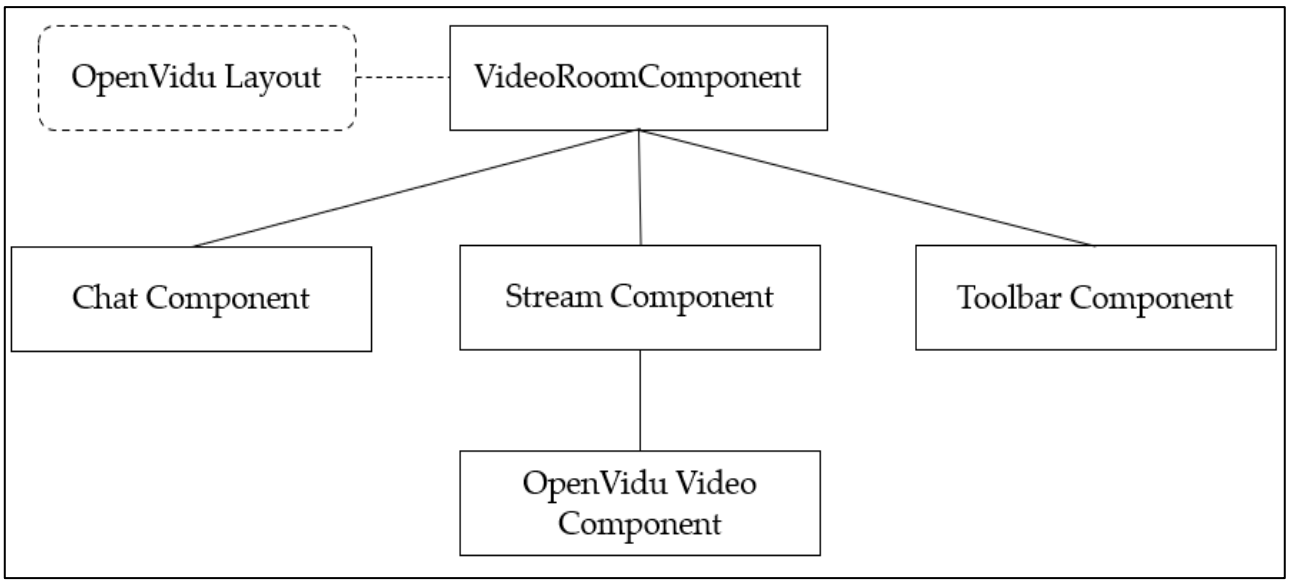


**Figure 3.** OpenVidu-Call-React's components.

### *3.1. Facial Expressions Recognition over Videoconference.*

The TensorFlow.js converter, COCO-SSD, and TensorFlow.js Core API are the needed dependencies that have to be installed in order to proceed. The TensorFlow.js converter is an open-source library to load a pre-trained TensorFlow Saved Model or TensorFlow Hub Module into the browser and run inference through TensorFlow.js [38]. The COCO-SSD is an object detection model that aims to localize and identify multiple objects in a single image [39]. The TensorFlow.js Core API provides low-level, hardware-accelerated linear algebra operations and an eager API for automatic differentiation [40].

Figure 4 represents the flowchart of deployment and development of the detection and recording of facial expressions over the video conference application in real-time. Figure 5 represents the modifications and the new components added compared to the one in Figure 3. The application has been enhanced to include a backend interface written in ExpressJS[41] and connects to a MongoDB[42] database server for recording facial expression recognition outputs. Figure 6 represents the two interfaces of the application. The last one is the detection interface of the facial expressions, while the right interface is that of the recorded outputs.

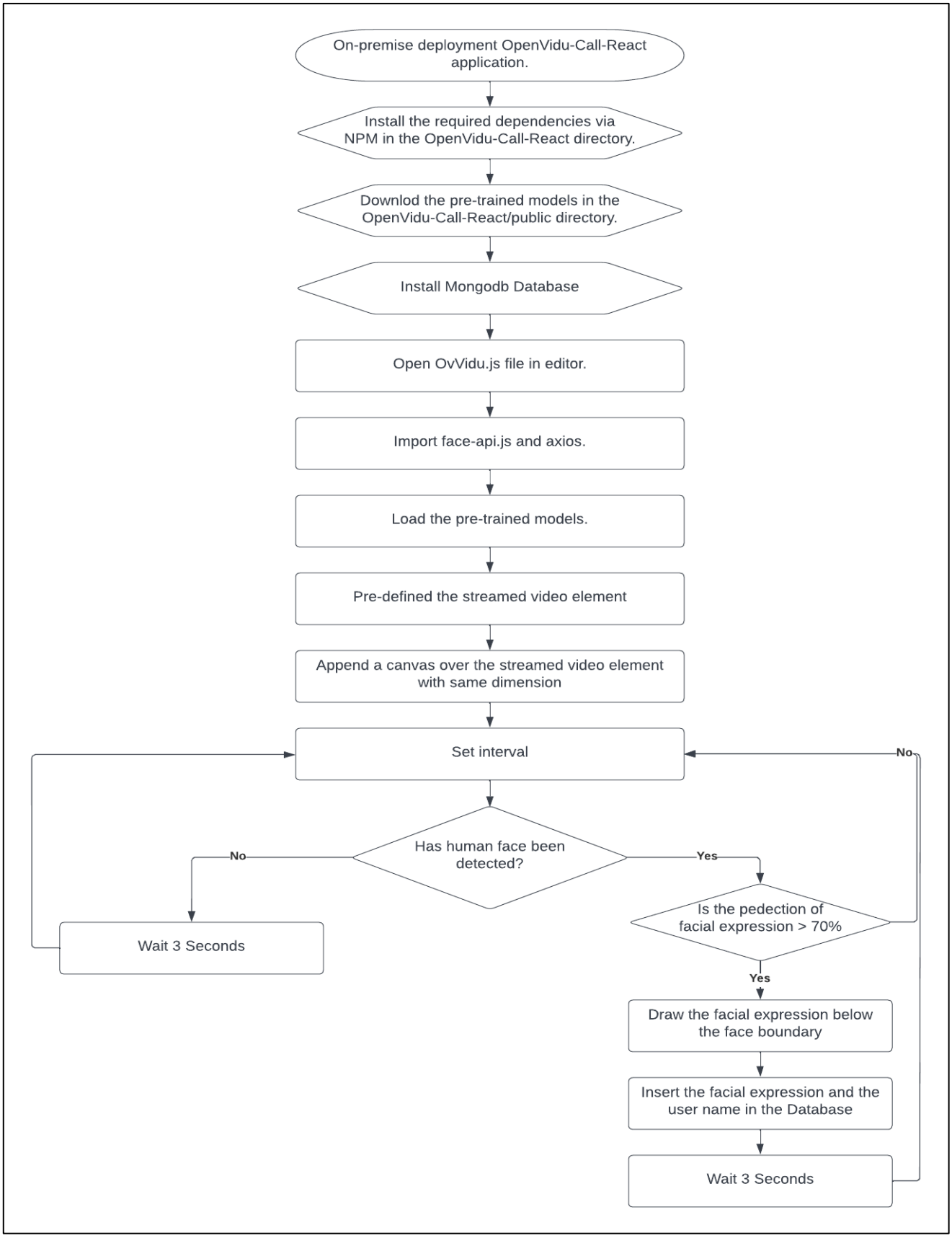


**Figure 4.** The flowchart of the facial expression detection deployment over a video conference application in real-time and the recorded results.

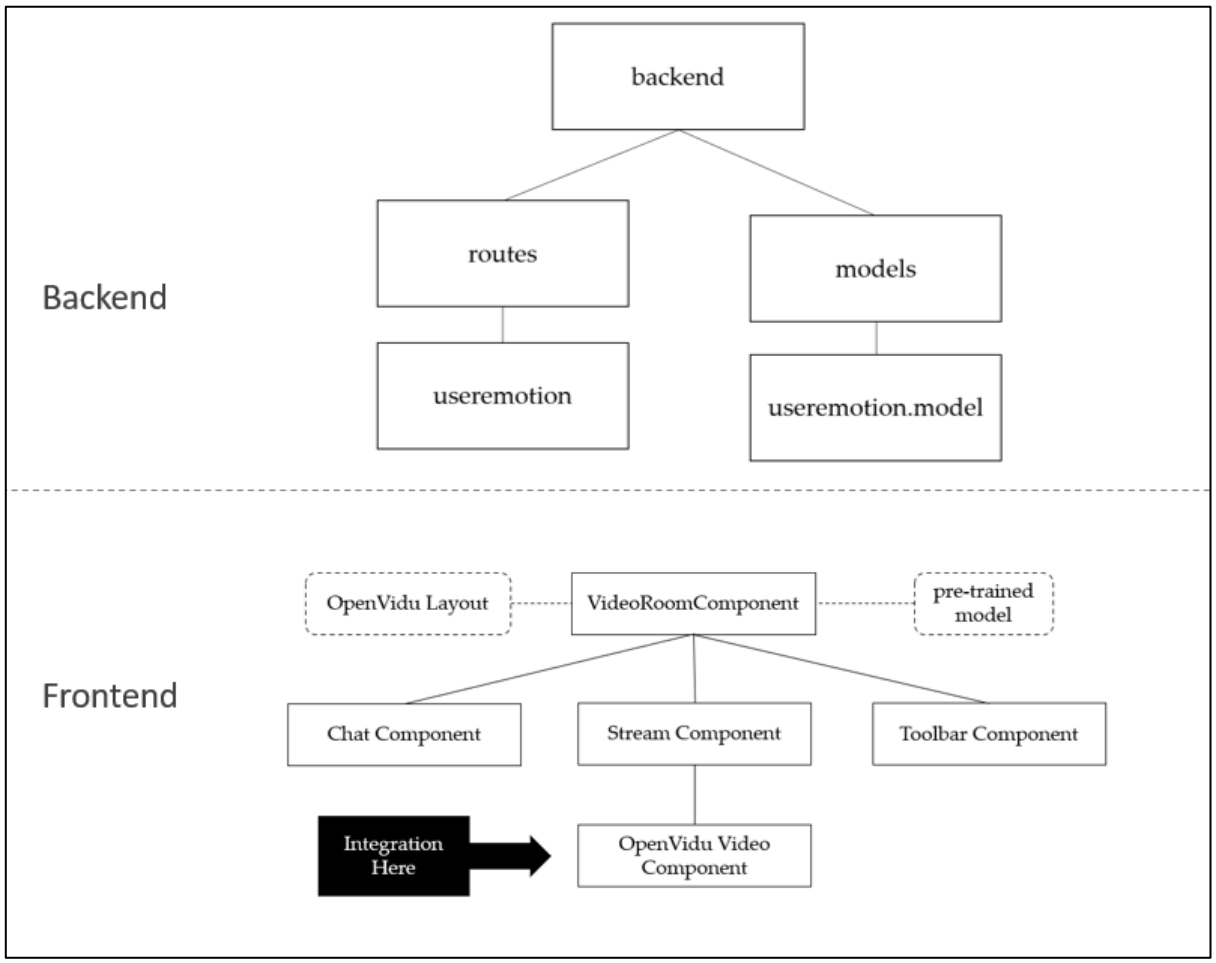


**Figure 5.** The main components of the proposed solution for facial expression recognition over a video conference application.

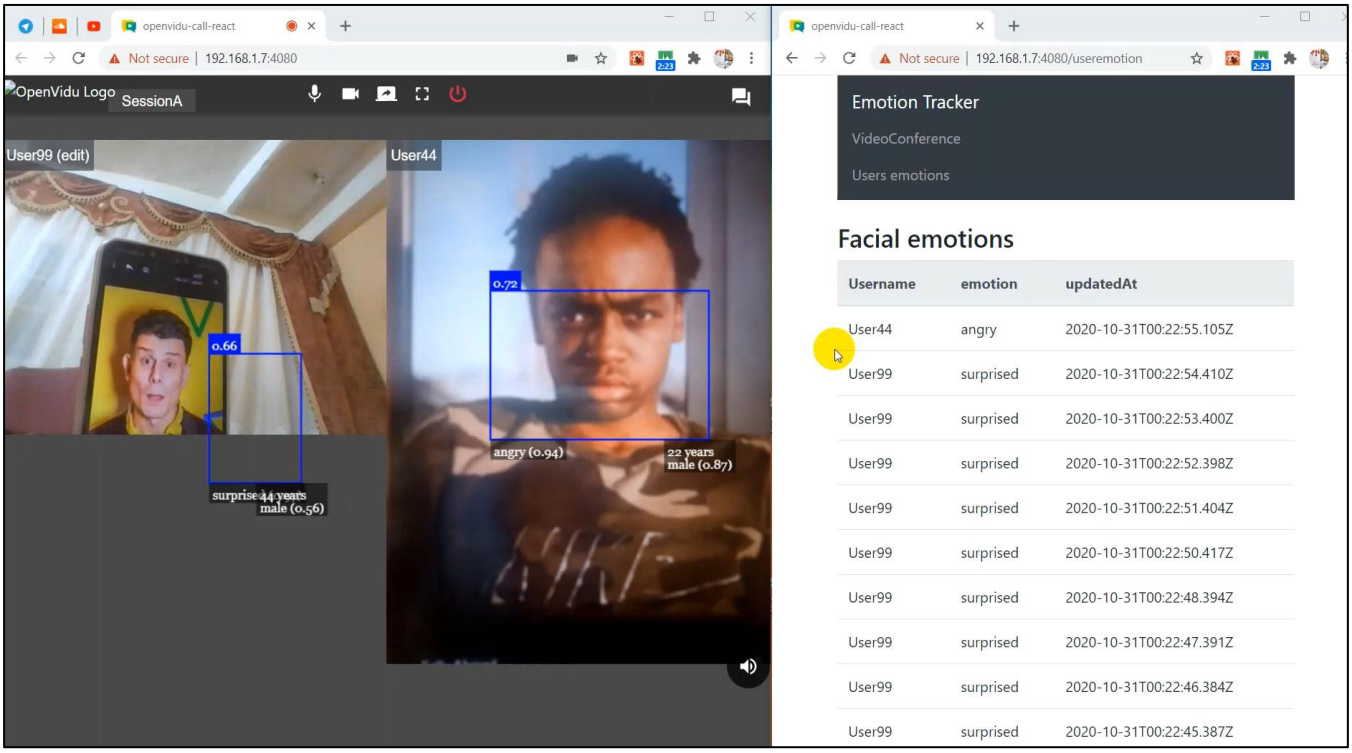


**Figure 6.** The facial expression recognition over video conference solution interface and the emotion tracker interface.

### *3.2. Automatic Speech Recognition over Video conference*

The steps proposed are presented in Figure 7. First, the react-speech-recognition has been installed via NPM[43] in the application directory. Then, a new component entitled Transcript Component has been created in the src/components directory. In the Transcript Component file, Speech Recognition and use Speech Recognition are imported from react-speech-recognition and the standard code line explained in the official website is used to control the microphone and transcript the output of the process of converting speech into text. Next, a code is deployed to render a transcript div element. After that, the Transcript Component in the OpenVidu Video component is imported and the transcript div element is integrated with the video. You can change the speech-to- text language throughout the language configuration line in the code. The proposed application in British English and Egyptian Arabic interfaces are represented in Figure 8 and Figure 9.

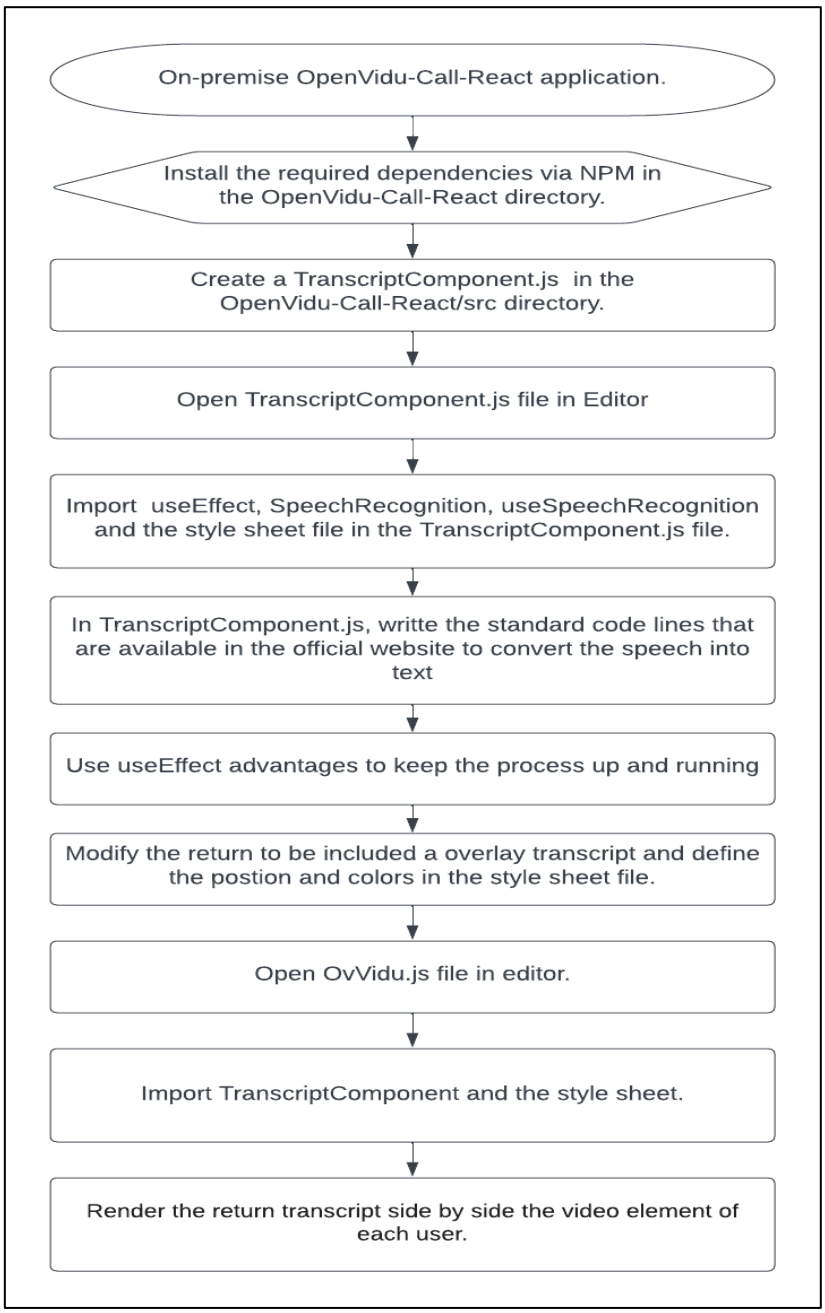


**Figure 7.** The flowchart of speech-to-text conversion over video conference application in real-time.

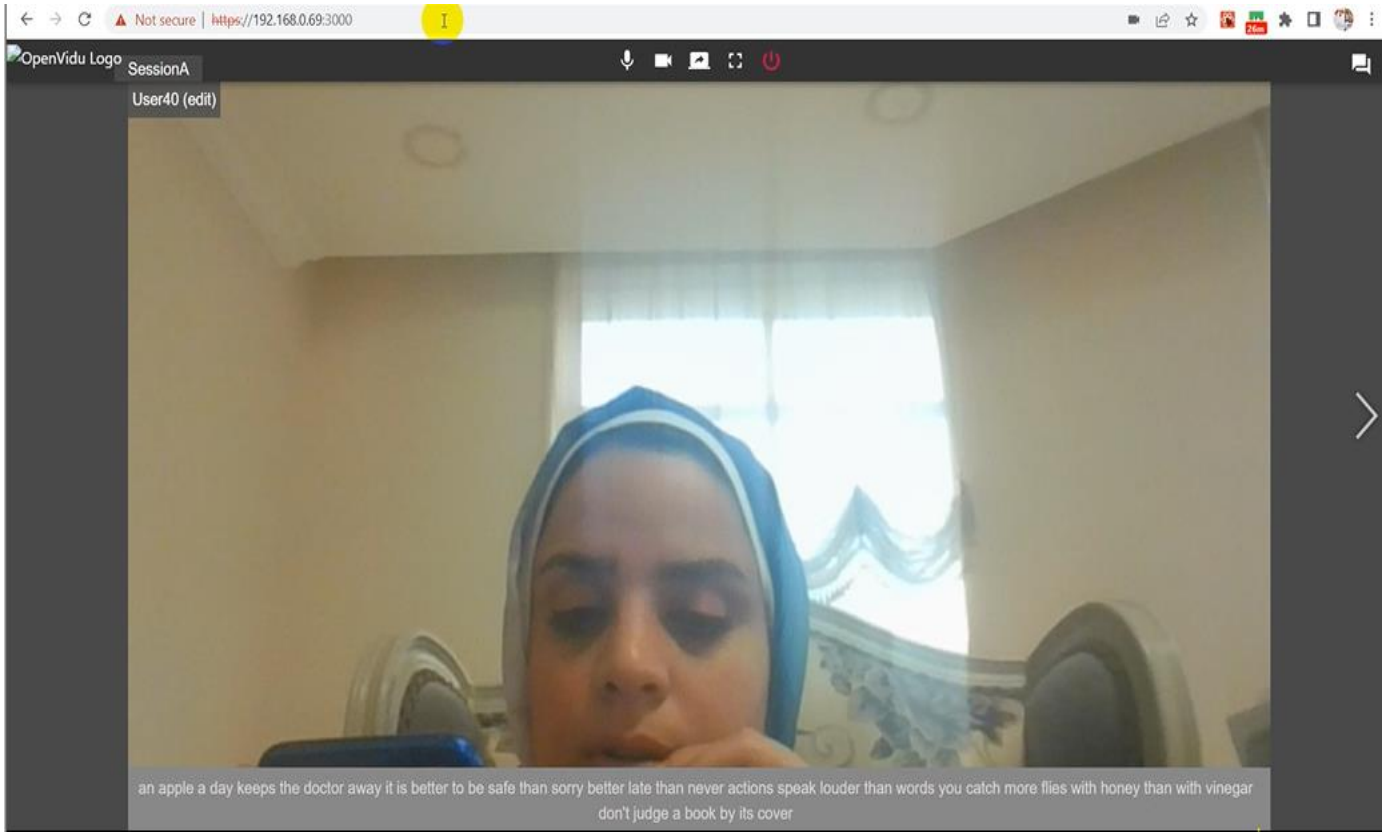


**Figure 8.** The English interface of speech-into-text conversion over the video conference application.

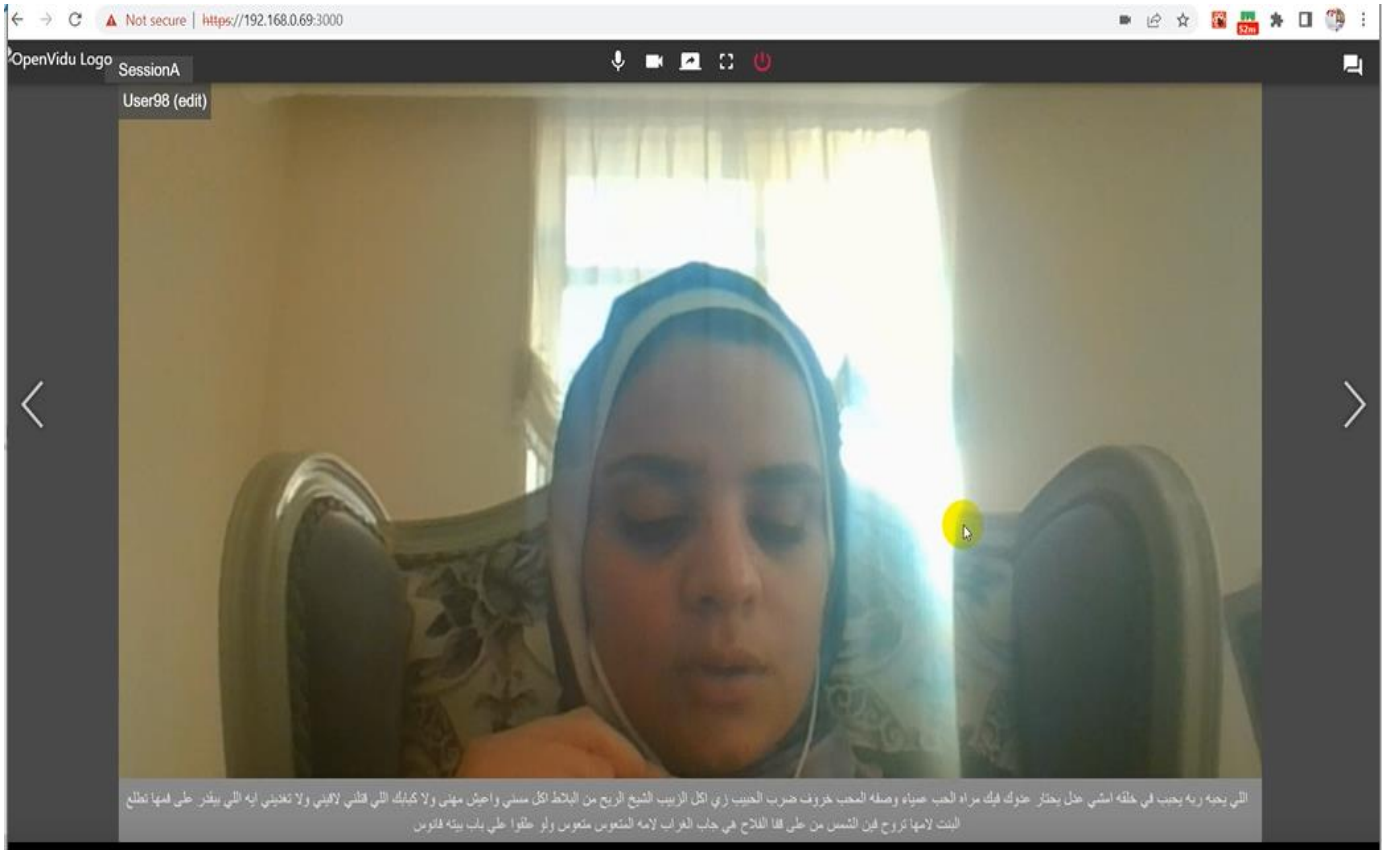


**Figure 9.** The Arabic interface of speech-into-text conversion over the video conference application.

## 5. Evaluation and Results

Firstly, the performance of the first proposed solution of facial expression recognition over the video conference application can be identified through examining the application using Kaggle [44] datasets. The human facial expressions datasets of natural [45], sad [46], angry [47], fearful [48], happy [49], disgusted [50], and surprised [51] have been downloaded and the images are presented one by one in front of the browser's used camera as it is presented in Figure 6.

The predicted expressions outputs have been stored in the database. The comparative analysis between the predictions expressions stored in the database and the original expressions of each image have been presented using the confusion matrix or error matrix in Figure 10.

Additionally, performance metrics of the classification model, such as accuracy, precision, recall and F1 measures, have been used to quantitatively evaluate the proposed solution. The metrics can be calculated in terms of TP, TN, FP and FN. Where TP (true positive) occurs when the model predicts the positive class correctly, TN (true negative) is a result in which the model correctly predicts the negative class, FP (false positive) occurs when the model predicts the positive class inaccurately, and FN (false negative) is an outcome in which the model predicts the negative class inaccurately. Therefore, the formulas of the performance metrics are as follows.

Accuracy is calculated as the ratio of the total number of correct predictions made to the total number of the observations made:

$$Accuracy = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}} \tag{1}$$

In addition to accuracy, precision has been estimated, which represents the number of the correct predictions that were correct and is calculated as the ratio of the correct prediction of an expression to the predicted positive cases:

$$Precision = \frac{\mathrm{TP}}{\mathrm{TP} + FP} \tag{2}$$

Further, recall/sensitivity that estimates the number of positives were accurately identified and is calculated as the ratio of the correct prediction of an expression to the actual positive cases of this expression:

$$Recall = \frac{TP}{TP + \mathrm{FN}} \tag{3}$$

Finally, the F1-score is calculated as the harmonic mean of both precision and recall and is ranged between 0,1

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{4}$$

If an F1 score is close to 1, it signifies high performance.

In Figure 11, it represents the performance metrics using the classification report of the sklearn.metrics [52] library.

Indeed, the results of the confusion matrix shown in figure 10 and the classification report shown in figure 11 indicate that the performance is good in happy, angry, surprised, natural, and sad expressions, while the performance is acceptable in the disgusted expression. However, the performance is poor in the fearful expression. The overall accuracy is 66%.

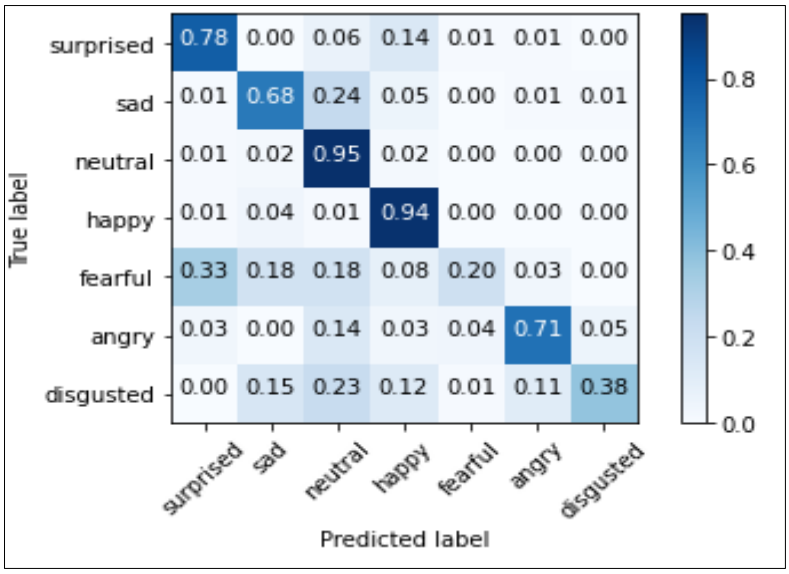


**Figure 10.** The confusion matrix of facial expressions recognition over the video conference application.

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| angry | 0.82 | 0.71 | 0.76 | 100 |
| disgusted | 0.86 | 0.38 | 0.53 | 100 |
| fearful | 0.77 | 0.20 | 0.32 | 100 |
| happy | 0.68 | 0.94 | 0.79 | 100 |
| neutral | 0.52 | 0.95 | 0.68 | 100 |
| sad | 0.64 | 0.68 | 0.66 | 100 |
| surprised | 0.67 | 0.78 | 0.72 | 100 |
| accuracy | | | 0.66 | 700 |
| macro avg | 0.71 | 0.66 | 0.64 | 700 |
| weighted avg | 0.71 | 0.66 | 0.64 | 700 |

**Figure 11.** The classification report of facial expressions recognition over the video conference application.

Secondly, the accuracy of the speech-to-text conversion in the second contribution can be calculated using the Word Error Rate (WER) [53] formula:

$$WER = \frac{\mathrm{S} + \mathrm{D} + \mathrm{I}}{\mathrm{S} + \mathrm{D} + \mathrm{C}} \quad (5)$$

where S (Substitution) is the number of substituted words, D (Deletion) is the number of undetected words, I (Insertion) is the number of incorrect words added, and C (Correct) is the number of correct words. Figures 7 and 8 represent the transcript of some popular proverbs in English and Arabic separately. According to the examinations that are represented, the British English WER is 0 and the Egyptian Arabic WER is 0.12.

## 6. Conclusions

This paper discussed the ability to host an on-premise video conference application based on WebRTC technology and the methods to append Artificial Intelligence (AI) pre-trained models over the video conference application. Indeed, WebRTC technology enables application developers to create on-premise video conference applications with less effort throughout JavaScript APIs. Individuals and organizations may need to cus-

tomize this application to meet their business needs. Besides, the availability of the artificial intelligence pre-trained models as JavaScript APIs enables application developers also to integrate AI features in their applications. This paper presented two proposed solutions resulted from the integration between the WebRTC APIs and AI pre-trained model APIs. The first proposed solution is the facial expression recognition over a video conference application and the results stored in the database. The second proposed solution is converting speech into text and transcribing the output below of the video conference application interface. The frameworks and tools used in implementing the two proposed solutions are a WebRTC-based application called OpenVidu-Call-React, facapi.js for facial expressions recognition, and react-speech-recognition for speech-into-text conversion. The evaluation metrics of facial expression recognition prove that the performance in detecting happy, angry, surprised, natural, and sad expressions is highly accurate. The disgusted expression is acceptably accurate, while the fearful expression is poorly accurate. Meanwhile, the British English and Egyptian Arabic speech-into-text conversion WERs are 0 and 0.12, respectively.

## References


1. Eltenahy, S., et al. *Comparative Analysis of Resources Utilization in Some Open-Source Videoconferencing Applications based on WebRTC*. in *2021 International Telecommunications Conference (ITC-Egypt)*. 2021. IEEE.
2. Eltenahy, S.A.M. *Facial Recognition and Emotional Expressions Over Video Conferencing Based on Web Real Time Communication and Artificial Intelligence*. 2021. Singapore: Springer Singapore.
3. Eltenahy, S., et al. *Conversion of Videoconference Speech into Text based on WebRTC and Web Speech APIs*. in *2021 International Telecommunications Conference (ITC-Egypt)*. 2021. IEEE.
4. Johnston, A.B. and D.C. Burnett, *WebRTC: APIs and RTCWEB protocols of the HTML5 real-time web*. 2012: Digital Codex LLC.
5. Manson, R., *Getting Started with WebRTC*. 2013: Packt Publishing Ltd.
6. Dutton, S.J.H.R., *Getting started with WebRTC.* 2012. **23**.
7. *Tensorflow.js*. Available from: https://www.tensorflow.org/js.
8. *Web Speech API*. Available from: https://developer.mozilla.org/en-US/docs/Web/API/Web_Speech_API.
9. Min, D., B. Park, and J. Park, *Artificial Engine Sound Synthesis Method for Modification of the Acoustic Characteristics of Electric Vehicles.* Shock and Vibration, 2018. **2018**: p. 5209207.
10. *BigBlueButton ·: Open Source Web Conferencing*. Available from: https://docs.bigbluebutton.org/.
11. *Jitsi*. [cited 2022/ 7/19]; Available from: https://jitsi.org/.
12. *openvidu-call*. Available from: https://docs.openvidu.io/en/2.15.0/demos/openvidu-call/#openvidu-call.
13. *Scaling Jitsi Meet in the Cloud Tutorial*. 10 Jul 2018; Available from: https://www.youtube.com/watch?v=Ij8a6ZRgehI.
14. *What is OpenVidu?* [cited 2022/ 7/19]; Available from: https://docs.openvidu.io/en/2.15.0/.
15. *Acknowledgments*. [cited 2022/ 7/19]; Available from: https://docs.openvidu.io/en/2.15.0/#acknowledgments.
16. Jain, D.K., P. Shamsolmoali, and P.J.P.R.L. Sehdev, *Extended deep neural network for facial emotion recognition.* 2019. **120**: p. 69-74.
17. Rößler, J., J. Sun, and P.J.F.I. Gloor, *Reducing Videoconferencing Fatigue through Facial Emotion Recognition.* 2021. **13**(5): p. 126.
18. Siegert, I., et al. *Recognition Performance of Selected Speech Recognition APIs–A Longitudinal Study*. in *International Conference on Speech and Computer*. 2020. Springer.
19. *Speech-to-Text*. January 6, 2022]; Available from: https://cloud.google.com/speech-to-text/.
20. High, R.J.I.C., Redbooks, *The era of cognitive systems: An inside look at IBM Watson and how it works.* 2012. **1**: p. 16.
21. *Wit.ai*. Available from: https://wit.ai/.
22. García, B., et al., *Understanding and estimating quality of experience in WebRTC applications.* Computing, 2019. **101**(11): p. 1585-1607.
23. team, G.W. *Real-time communication for the web*. [cited 2022; Available from: https://webrtc.org/.

24. Blum, N., S. Lachapelle, and H.J.Q. Alvestrand, *WebRTC-Realtime Communication for the Open Web Platform: What was once a way to bring audio and video to the web has expanded into more use cases we could ever imagine*. 2021. **19**(1): p. 77-93.
25. *NGINX*. January 6, 2022]; Available from: https://www.nginx.com/.
26. *APACHE*. Available from: https://httpd.apache.org/.
27. López, L., et al. *Kurento: The WebRTC modular media server*. in *Proceedings of the 24th ACM international conference on Multimedia*. 2016.
28. Rosenberg, J., et al., *STUN-simple traversal of user datagram protocol (UDP) through network address translators (NATs)*. 2003.
29. Mahy, R., P. Matthews, and J. Rosenberg, *Traversal using relays around nat (turn): Relay extensions to session traversal utilities for nat (stun)*. 2010, RFC 5766.
30. El Hamzaoui, A., H. Bensaid, and A. En-Nouaary. *A Formal Model for WebRTC Signaling Using SDL*. 2016. Cham: Springer International Publishing.
31. *Node.js*. Available from: https://nodejs.org/en/.
32. *face-api.js*. Available from: https://justadudewhohacks.github.io/face-api.js/docs/index.html.
33. Adorf, J.J.K.R.I.o.T., *Web speech API*. 2013.
34. *react-speech-recognition*. [cited 2022/ 07/19]; Available from: https://github.com/JamesBrill/react-speech-recognition.
35. *API docs*. Available from: https://github.com/JamesBrill/react-speech-recognition/blob/master/docs/API.md.
36. *openvidu-call-react*. Available from: https://docs.openvidu.io/en/2.17.0/demos/openvidu-call-react/.
37. *React*. Available from: https://reactjs.org/.
38. *tfjs/tfjs-converter/*. Available from: https://github.com/tensorflow/tfjs/tree/master/tfjs-converter.
39. *tfjs-models/coco-ssd/*. Available from: https://www.npmjs.com/package/@tensorflow-models/coco-ssd.
40. *tfjs/tfjs-core/*. Available from: https://github.com/tensorflow/tfjs/tree/master/tfjs-core.
41. *Express*. Available from: https://expressjs.com/.
42. *MongoDB*. Available from: https://www.mongodb.com/.
43. *NPM*. Available from: https://www.npmjs.com/.
44. *kaggle*. Available from: https://www.kaggle.com/.
45. *human-expression-natural*. Available from: https://www.kaggle.com/datasets/sallyahmed20/human-expression-natural.
46. *human-emotionssad-faces*. Available from: https://www.kaggle.com/datasets/jafarhussain786/human-emotionssad-faces.
47. *human-emotionsangry-faces*. Available from: https://www.kaggle.com/datasets/jafarhussain786/human-emotionsangry-faces.
48. *human-emotionsfear-faces*. Available from: https://www.kaggle.com/datasets/jafarhussain786/human-emotionsfear-faces.
49. *human-emotionshappy-faces*. Available from: https://www.kaggle.com/datasets/jafarhussain786/human-emotionshappy-faces.
50. *human-expression-disgusted*. Available from: https://www.kaggle.com/datasets/sallyahmed20/human-expression-disgusted.
51. *human-emotionssuprise-faces*. Available from: https://www.kaggle.com/datasets/jafarhussain786/human-emotionssuprise-faces.
52. *scikit-learn*. Available from: https://scikit-learn.org/stable/.
53. *Word_error_rate*. Available from: https://en.wikipedia.org/wiki/Word_error_rate.