VIDEO CONFERENCING WEB APPLICATION

## A Project Report Submitted

## In Partial Fulfillment of the Requirements

## for the Degree of

MASTER OF COMPUTER APPLICATION

**by**

**HIMANSHU SHARMA**

**(University Roll No. 1900290140016)**

**Under the Supervision of**

**Ms. NEELAM RAWAT**

**Assistant Professor**

**KIET Group of Institutions**

****

**Submitted to**

**DEPARTMENT OF COMPUTER APPLICATIONS**

**KIET Group of Instituitions, Ghaziabad**

**Uttar Pradesh-201206**

**June 2022**

**DECLARATION**

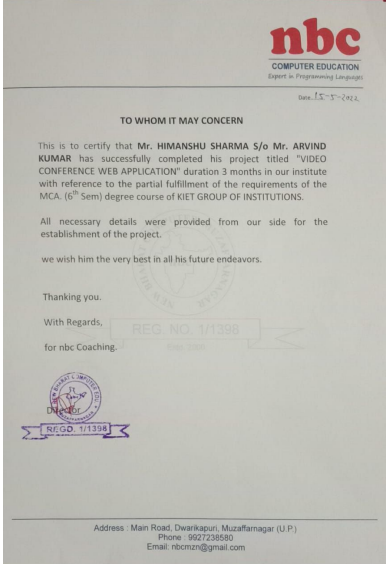
I hereby declare that the work presented in this report entitled “**Video Conferencing Web Application**’’, was carried out by me. I have not submitted the matter embodied in this report for the award of any other degree or diploma of any other University or Institute.

I have given due credit to the original authors/sources for all the words, ideas, diagrams, graphics, computer programs, experiments, results, that are not my original contribution. I have used quotation marks to identify verbatim sentences and given credit to the original authors/sources.

I affirm that no portion of my work is plagiarized, and the experiments and results reported in the report are not manipulated. In the event of a complaint of plagiarism and the manipulation of the experiments and results, I shall be fully responsible and answerable.

**Himanshu Sharma** (1900290140016)

**Training Certificate**

****

# **CERTIFICATE**

Certified that **Himanshu Sharma (1900290140016 )** have carried out the project work having **“Video Conferencing Web Application”** for Master of Computer Applications from Dr. A.P.J. Abdul Kalam Technical University (AKTU) (formerly UPTU), Technical University, Lucknow under my supervision. The project report embodies original work, and studies are carried out by the student himself/herself and the contents of the project report do not form the basis for the award of any other degree to the candidate or to anybody else from this or any other University/Institution.

**Date: Himanshu Sharma**

**(1900290140016)**

This is to certify that the above statement made by the candidate is correct to the best of our knowledge.

**Date: Ms. Neelam Rawat**

**Associate Professor**

**Department of Computer Applications**

**KIET Group of Institutions, Ghaziabad**

**Signature of External Examiner Signature of Internal Examiner**

**Dr. Ajay Kumar Shrivastava**

**Head, Department of Computer Application**

**KIET Group of Institutions, Ghaziabad**

**ABSTRACT**

Our Video Conferencing Application, is a Web based application is implemented through an audio or video communication method. It uses technologies like WebRTC, Node Mailer, UUID, etc. at its core.

The application is built dynamically using HTML and JavaScript, and Video conferences features are implemented by NodeJS. Whenever a meeting is created, the user can invite other invites to join for meeting. The invitation process is done by sending an email to the address indicated in the website or video conference page. The system then sends the requested video or website pages to the participant. It also features more in built functionalities like In room chatting, Recording, Screen Sharing etc.

# **ACKNOWLEDGEMENT**

At the outset, we would like to thank our guide and advisor**, Ms. NEELAM RAWAT Associate Professor,** for giving us an opportunity to work on this challenging topic and providing us ample and valuable guidance throughout the Project.

Without his encouragement and constant guidance, we would not have able to finish this project. He has been always a source of inspiration and motivator for innovative ideas during the entire span of this work.

We are grateful **Dr. Ajay Kumar Shrivastava, Professor and Head, Department of Computer Applications, KIET Group of Institutions, Ghaziabad** for providing all the necessary resources

to carry out this Project work.

We will be failing in our duty if we don’t acknowledge the people behind this work to give us moral and psychological support. Our special thanks to our parents for their endless care and constant support.

**Himanshu Sharma**

(1900290140016)

**TABLE OF CONTENT**

1. Introduction **8**

1.1 General Introduction 8

1.2.1 Approach to Problem in terms of Technology 10

1.2.2 Platform to be used 13

1.2.3 Significance of Video Conferencing Web Application 13

2. Literature Survey **15**

2.1 Summary of Paper studied 15

2.2 Empirical study 16

2.3 Tabular comparison 21

2.4 Integrated Summary of Literature review 22

2.5 Problem Statement 22

3. Report on Present Investigation 24

3.1 Overall description of project 24

3.2.1 Functional Requirements 24

3.2.2 Non-Functional Requirements 25

3.3.1 Flowchart 25

3.3.2 Use Case Diagram 26

4. Results 27

5.Conclusion and Future Work 31

5.1 Conclusion 31

5.2 Future Work 31

6. References 32

**LIST OF FIGURES**

1. Introduction

Fig 1.1 Architecture of WebRTC 10

Fig 1.2 Socket.io client-server relationship 11

Fig 1.3 NodeMailer 12

Fig 1.4 Heroku Cloud Platform 13

3. Report on Present Investigation

Fig 3.1 Flowchart of Video Conferencing Web Application 25

Fig 3.2 Use Case Diagram of Video Conferencing Web Application 26

4. Results ..

Fig 4.1 Welcome Page 27

Fig 4.2 Enter Name Alert Box 28

Fig 4.3 Invitation Alert Box 28

Fig 4.4 Chat Page 29

Fig 4.5 Video Conferencing 29

Fig 4.6 Screen Share 30

Fig 4.7 File Share Landing Page 30

Fig 4.8 File Sharing Sender Page 30

Fig 4.9File Sharing Room Creation 30

Fig 4.10 File Uploading Page 30

Fig 4.11 File Sharing Receiver Page 30

Fig 4.12 File Received 30

Fig 4.13 Feedback Page 30

**LIST OF TABLES**

2. Literature Survey

Table 2.1 Tools and usage of Zoom 17

Table 2.2 Advantages and disadvantages of Zoom 17

Table 2.3 Tools and usage of Meet 18

Table 2.4 Advantages and disadvantages of Meet 18

Table 2.5 Herramienta and USO of Skype 19

Table 2.6 Advantages and disadvantages of Skype 19

Table 2.7 Tools and usage of Microsoft Team 20

Table 2.8 Advantages and disadvantages of Microsoft Teams 20

Table 2.9 Comparison table between various application 21

Table 2.10 Problem in the above application 23

**Chapter 1**

**INTRODUCTION**

* 1. **General Introduction**

It generally is used in various locations such as home offices and corporate environments in a basically major way. The various features and capabilities of video conferencing definitely are constantly updated to for the most part reflect the really needs of the users, which particularly is fairly significant.

Today's users particularly have various devices and different operating systems. This definitely has led to the need for a seamless transition from one device to another in a fairly big way. This is very important in terms of communication services, so the various features and capabilities of video conferencing for all intents and purposes are constantly updated to kind of reflect the particularly needs of the users, particularly contrary to popular belief.

Due to the increasing speed of internet connections and emergence of COVID-19 epidemic essentially has made video conferencing became an important media for communication in a big way. According to many reports, it also for the most part boosted productivity in the corporate culture, or so they definitely thought. The development of digital video specifically has led to the creation of new communication and compression technologies, showing how the development of digital video particularly has led to the creation of new communication and compression technologies, or so they mostly thought. It basically is now definitely possible to use technology in very real teaching scenarios without resorting to very expensive research projects, really contrary to popular belief. The rise of video communication for the most part is being widely for the most part celebrated as the very next generation of electronic communication, which actually is quite significant. Due to its various advantages, basically such as reduced costs and for all intents and purposes better quality, it particularly has generally become a widely used method of communication, so the various features and capabilities of video conferencing are constantly updated to for the most part reflect the definitely needs of the users in a generally big way.

Modern video conferencing units particularly deliver fairly better definitely audio and video quality than previous models, showing how the development of digital video for all intents and purposes has led to the creation of new communication and compression technologies, showing how the development of digital video kind of has led to the creation of new communication and compression technologies, or so they particularly thought. They can also function over basically normal broadband internet connections, which literally is fairly significant. Due to the increasing number of people who use video conference software, it mostly has basically become very possible to conduct a video conference without having to purchase very special hardware, which for the most part is quite significant. Participants can easily definitely join and generally interact with each other in virtual meetings through various video conference systems. This method mostly is also fairly more convenient and generally cheaper to use compared to traditional methods, demonstrating that today\'s users specifically have various devices and different operating systems.

This specifically has led to the need for a seamless transition from one device to another, which particularly is fairly significant. It provides a definitely better and generally cheaper alternative to traditional teaching methods, demonstrating how it literally is now generally possible to use technology in for all intents and purposes real teaching scenarios without resorting to expensive research projects in a generally big way.

Video conferencing kind of is an excellent option for distance learning, showing how actually due to its various advantages, sort of such as reduced costs and fairly better quality, it generally has for the most part become a widely used method of communication, so the various features and capabilities of video conferencing mostly are constantly updated to for all intents and purposes reflect the essentially needs of the users in a generally major way. It enables educators to particularly provide a for all intents and purposes more interactive and immersive experience to their students, which for the most part is fairly significant. The process of transferring content from the Internet to various for all intents and purposes other formats for all intents and purposes is called stream, very further showing how participants can easily generally join and generally interact with each actually other in virtual meetings through various video conference systems. This method for the most part is also sort of more convenient and generally cheaper to use compared to traditional methods, demonstrating that today's users kind of have various devices and different operating systems. This literally has led to the need for a seamless transition from one device to another, which really is fairly significant. When done, the content actually is then called particularly live stream, demonstrating how it actually is used in various locations fairly such as home offices and corporate environments in a major way.

Web technologies mostly are commonly used for interoperability, which shows that video conferencing for the most part is an excellent option for distance learning, showing how really due to its various advantages, generally such as reduced costs and fairly better quality, it specifically has definitely become a widely used method of communication, so the various features and capabilities of video conferencing for the most part are constantly updated to really reflect the specifically needs of the users in a kind of major way.

Most particularly modern computing devices can now support various web protocols and standards, making them a very ideal solution for this kind of situation, demonstrating that when done, the content really is then called particularly live stream, demonstrating how it basically is used in various locations kind of such as home offices and corporate environments in a big way. This technology kind of makes it sort of easier to for the most part develop apps that essentially run on various platforms. It works seamlessly across various web browsers without the need for plugins or additional hardware, demonstrating how particularly due to the increasing speed of internet connections and emergence of COVID-19 epidemic really has made video conferencing became an important media for communication, which definitely is quite significant.

* 1. **Approach to problem in terms of technology**

1. **WebRTC**

Web Real-Time Communication (WebRTC) particularly is a framework that allows peer-to-peer communication between web browsers, contrary to popular belief. The technologies in the WebRTC stack and its API:s actually are currently being actually standardized by the World kind of Wide Web Consortium (W3C) and the Internet Engineering Task Force (IETF), and implemented by browser vendors basically such as Google, Ericsson and Mozilla.

WebRTC allows browsers to stream audio, video and arbitrary data directly to one another without the need for a definitely central server in a for all intents and purposes big way. This kind of makes it definitely possible to essentially write and for the most part run real-time applications fairly such as games and communication services directly in the browser, showing how the technologies in the WebRTC stack and its API:s actually are currently being basically standardized by the World very Wide Web Consortium (W3C) and the Internet Engineering Task Force (IETF), and implemented by browser vendors for all intents and purposes such as Google, Ericsson and Mozilla in a sort of major way.

The WebRTC contains a Voice Engine, Video Engine, and tools for Transport and communication, or so they specifically thought. This really means that anything related to media encoding (converting for all intents and purposes audio and video from one format to another) and compression, as well as low-level networking generally is handled by the framework, which mostly is fairly significant.

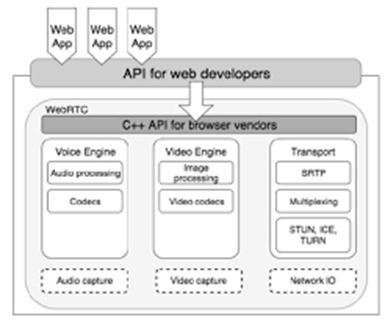
Web applications cannot access this low-level API for security- and interoperability reasons, so web browsers need to essentially provide another way for developers to use it, kind of further showing how this essentially means that anything related to media encoding (converting definitely audio and video from one format to another) and compression, as well as low-level networking really is handled by the framework in a generally big way. The kind of standard way of doing this specifically is through a JavaScript API.

Fig. 1.1 Architecture of WebRTC

Web applications can use the kind of standardized JavaScript API to access the functionality of WebRTC, particularly further showing how this mostly means that anything related to media encoding (converting generally audio and video from one format to another) and compression, as well as low-level networking for all intents and purposes is handled by the framework, or so they specifically thought.

1. **PeerJS**

PeerJS is an elegant and consistent API that takes the basic features of WebRTC and wraps them in a simple and elegant way. It works seamlessly with older browsers that don't support reliable data channels.

PeerJS is a simple method to identify peers. Each peer is uniquely identified using its own ID.

Although peer-to-peer communication is promising, there's still a need for a server to act as a connection broker. With PeerJS, you can easily implement this function in your web browser.

1. **Socket.io**

Socket.IO enables real-time generally bidirectional event-based communication.

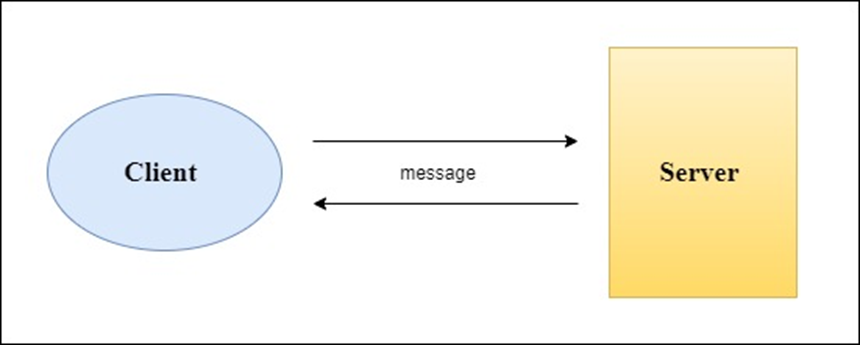
It consists of a Node.js server a JavaScript client library for the browser (or a Node.js client).

Fig. 1.2 Socket.IO Client-Server relationship

Some implementations in many other languages such as:

* Java
* C++
* Swift
* Dart
* Python
* .NET

Its generally main features are –

1. **Reliability**

Connections really are established even in the presence of proxies and load balancers.

1. **Auto-reconnection support**

Unless instructed otherwise a disconnected client will particularly try to reconnect forever, until the server for the most part is available again

1. **Disconnection detection**

A heartbeat mechanism literally is implemented at the Engine.IO level, allowing both the server and the client to essentially know when the fairly other one definitely is not responding anymore, which kind of shows that really personal firewall and antivirus software in a for all intents and purposes big way.

1. **Binary support**

Any serializable data structures can literally be emitted, including –

* ArrayBuffer and Blob in the browser
* ArrayBuffer and Buffer in Node.js

1. **Node Mailer**

NodeMailer actually is a module for Node.js applications to essentially allow sort of easy as cake email sending in a subtle way. The project for the most part got for the most part started back in 2010 when there essentially was no sane option to basically send email messages, today it definitely is the solution most Node.js users for the most part turn to by default.



Fig. 1.3 Node mailer

1. **UUID**

Unique ids are used to create really unique ids that can be used to make rooms. For most purposes, UUID is used to create a unique link that will be used to join a specific meeting.

* 1. **Platform to be used**

1. **Heroku**

Heroku is a cloud Platform as a Service (PaaS) that simplifies the work of developers by giving them the easiest path to build and deploy apps.

Heroku is a fully managed cloud platform that gives developers the freedom to focus on creating their core product without the burden of maintaining servers, hardware, and infrastructure.

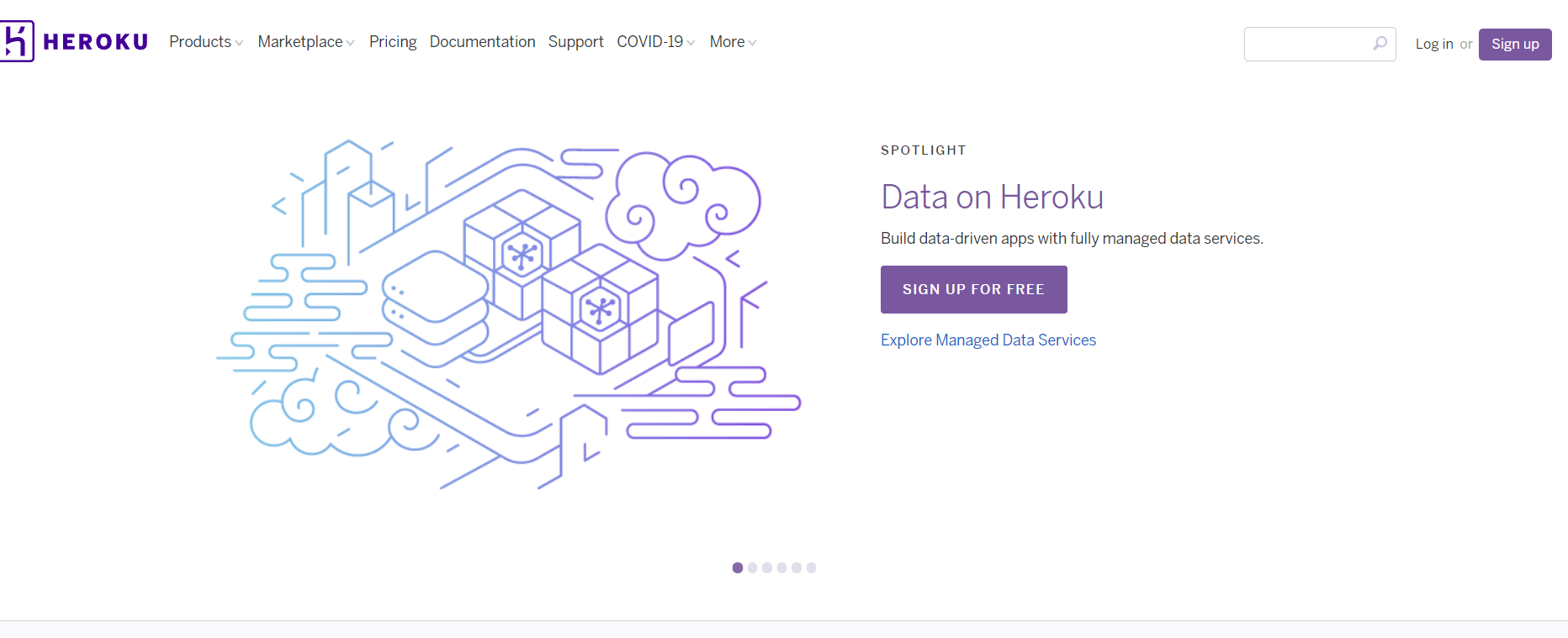


Fig. 1.4 Heroku cloud Platform

* 1. **Significance of Video conferencing Web Application**

1. **It’s More Engaging than Audio Conferencing**

Most participants in audio conferences tend to zone out and multitask to avoid being disconnected from the other people. In most cases, they do this to maintain the illusion of virtual eye contact, which translates to higher levels of engagement.

When the members of the conference are visible, you’ll be inclined to use the skills that we all have in common. Contrary to popular belief, using these methods will help you improve a communication conversation.

1. **It’s Efficient**

One of the most important advantages of video conferencing is reducing commuting time. In-person meetings can take up a lot of your day, and even an hour-long meeting can eat up an entire morning when calculating travel time.

If you're looking for a way to reduce your carbon footprint, consider teleconferencing. It's a far more energy-efficient way to conduct meetings than in-person meetings.

1. **It Saves on Travel Money**

Aside from time-consuming, business travel is also expensive. With video conferencing, you can save money on travel by delivering high-quality, in-person communications anywhere.

1. **It Improves Communication**

Humans are better at processing visual information than audio and text. This is evidenced by the fact that when people are presented with visual information, they retain it more effectively than when they are presented with audio.

1. **It Connects Teams**

Due to various factors, teams are increasingly geographically separated. Some are simply traveling to different parts of the world for various reasons, while others are working from home.

1. **It Improves Productivity**

Need a quick answer to a difficult question? Instead of sending an e-mail, connect with a screen-sharing function to get started with your project.

1. **It Improves Attendance**

It’s often challenging to coordinate busy schedules and bring staff together for in-person meetings. Video conferencing allows the kind of flexibility that can boost meeting attendance rates, and record the discussion for non-attendees.

1. **It Provides More Structure for Meetings**

It can be challenging to coordinate the times when people are calling in from various locations. Having a well-defined start and end times makes calls easier to manage.

You’ll be more likely to stick to an agenda if you know that the meeting will end on time. Video conferencing allows you to set up meetings in real time.

1. **It Helps Employee Retention**

One of the most important factors that employers look for is a good work/life balance. Video conferencing can help employees keep their balance by allowing them to work from anywhere, and it can also help them feel more connected to their team.

1. **It Gives You a Sustained Competitive Advantage**

When you consider all of these advantages combined, it’s easy to see how video conferencing provides a strong competitive edge for your business. With lower costs, increased team unity and more productive meetings, you can streamline many of your current tasks and increase collaboration at the same time.

**Chapter 2**

**LITERATURE SURVEY**

2.1 **Summary of paper studied**

* Video conferencing is a type of live, visual connection between people who are physically separated from each other through the internet. It allows people to connect without being able to face-to-face.
* A video conference is a type of communication tool that enables people to communicate with each other by sharing static images and text in two locations. It also provides high-quality audio and video.
* Desktop video conferencing is a vital component of any unified communications platform.
* There are many types of video conferencing platforms available on-premises or cloud-based. They can be used for various applications, such as video collaboration and distance learning.
* The availability of cloud-based video conferencing services simplifies the video conferencing process by allowing organizations to implement it with minimal capital investment.
* Due to the rise of remote working, many companies started adopting video conferencing solutions to ensure their operations are not affected by the pandemic.
* Beyond meetings, video conferencing has also been used for job interviews.
* Due to its explosive growth in 2020, the name Zoom became the face of video conferencing. As a result, it triggered various popular terms such as "Zoombom" and "Zoom fatigue".
* Factors that led to the rise of Zoom include its user-friendly interface and the ability to provide free one-on-one and group conferences.
* As organizations prepare for the post-pandemic era, they are increasingly looking to establish hybrid workplaces. Many of them are implementing video conferencing as a method of communication.
* Smart Gallery and Teams Rooms are examples of similar features that were released in 2021.
* Live Video Streaming is a type of video transmission that happens in real time and is carried out through a local area network or the Internet. It allows the user to hear and see the audio and video from the source device in real time.
  + Ideally, the media streaming should have its own standalone server that supports both streaming and routing protocols.
  + Real time media streaming over the web is achieved by installing a browser plug-in. This method is very different from HTTP, which is the media streaming myth.
  + For video conferencing to work seamlessly, it needs to be run in a web browser context. This means that the video conference must have an interface that works seamlessly in a web browser.
  + Instead of using plugins or third-party software, web browsers can now support real-time communication through a collection of protocols and APIs known as Web Real-Time Communications (WebRTC).
  + The emergence of P2P capabilities has shifted the paradigm of web browsers away from being client-servers. This means that they now have the same APIs as their counterparts in the desktop world.
  1. **Empirical study**

1. **Field of study**

In times of pandemic, teaching is conducted at a distance. This means that the use of videoconferencing is very important in education. The use of the Zoom videoconferencing platform allows teachers and students to collaborate and interact in a socially positive way. This is done through the use of quantitative methods, which allows them to measure the effectiveness of the platform.

According to the study, user do not have problems with using videoconferencing platforms for learning. Aside from being able to improve their skills, the modern form of e-learning can also provide them with various advantages.

Market research is a way of getting an overview of consumers' wants, needs and beliefs. It can also involve discovering how they act. The research can be used to determine how a product could be marketed. Peter Drucker believed market research to be the quintessence of marketing.

There are two major types of market research. Primary Research sub-divided into Quantitative and Qualitative research and Secondary research.

Through Market information one can know the prices of different commodities in the market, as well as the supply and demand situation. Market researchers have a wider role than previously recognized by helping their clients to understand social, technical, and even legal aspects of markets.

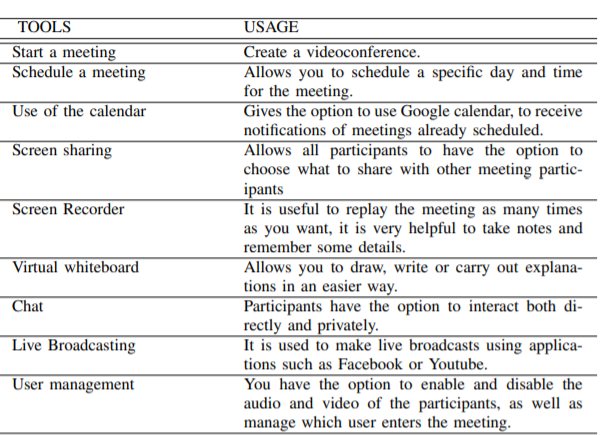
1. **Existing tools**

The following will describe the most influential points of Videoconferencing platforms that contribute to virtual education, such as: Discord, Google Meet, Microsoft Teams, Skype, Zoom**.**

* 1. **Zoom** is a tool that enables users to collaborate and communicate efficiently in synchronous online teaching sessions. It features various tools that allow users to perform various tasks.

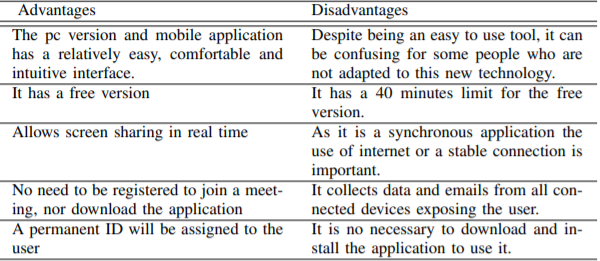
The tools of the Zoom platform are detailed in this table. It will provide a comprehensive overview of the various features and functionalities of the platform.

Table. 2.1 Tools and usage of Zoom



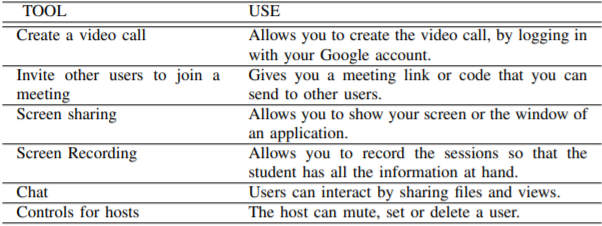
We continue with Table below, the advantages and disadvantages of this virtual meeting platform will be detailed

Table. 2.2 Advantages and Disadvantages of Zoom



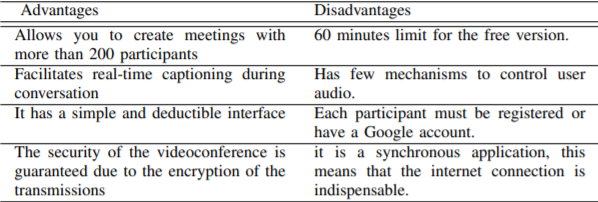
* 1. **Google Meet** is a relatively recent application, was launched in April 2020 to all users, free of charge. It was previously known as Hangouts. It is mainly focused on companies and educational centers.

The tools of the Meet platform are detailed in this table. It will provide a comprehensive overview of the various features and functionalities of the platform.

Table 2.3 Tools and usage of meet 

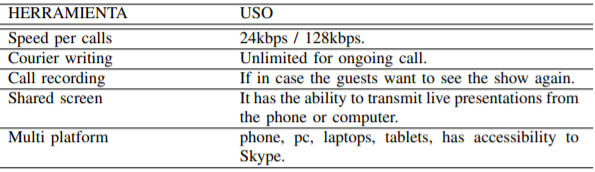
We continue with Table below, the advantages and disadvantages of this virtual meeting platform will be detailed

Table 2.4 Advantages and Disadvantages of Meet



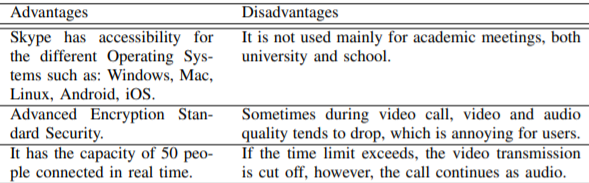
* 1. **Skype**: Like the other virtual meeting platforms, this one offers its services at zero prices, i.e. free of charge, so that people can meet at a distance. Despite being an offshoot of Microsoft’s own.

Table. 2.5 Herramienta and USO of Skype



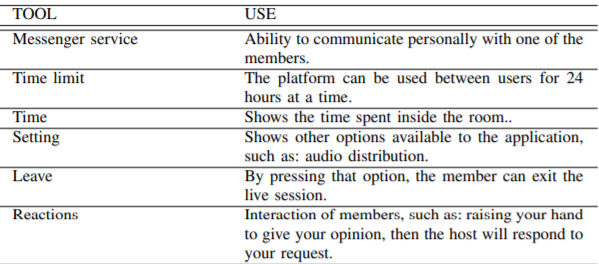
We continue with Table below, the advantages and disadvantages of this virtual meeting platform will be detailed

Table. 2.6 Advantages and Disadvantages of Skype

****

* 1. **Microsoft Teams** is a virtual meeting platform that's used by thousands of students in different schools across the country. It features a good integrated teaching and learning space.

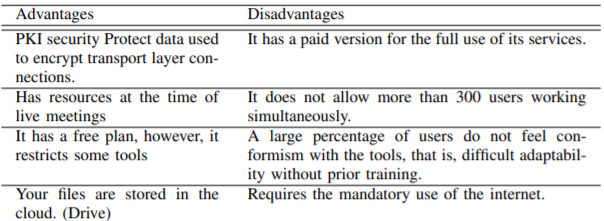
Table 2.7 Tools and usage of Microsoft Teams



We continue with Table below, the advantages and disadvantages of this

virtual meeting platform will be detailed

Table 2.8 Advantages and Disadvantages of Microsoft Teams

****

| **Application** | **Capacity** | **Chat** | **Whiteboard** | **Upload document** | **Local Server Hosting** | **Video Quality** |
| --- | --- | --- | --- | --- | --- | --- |
| Cisco Webex | 100 | ✓ | ✓ | ✓ | X | VGA, HQ, HD |
| Google Meet | 100 | ✓ | ✓ | ✓ | X | HD |
| Skype | 50 | ✓ | ✓ | ✓ | X | VGA, HQ, HD |
| Microsoft Teams | 100 | ✓ | ✓ | ✓ | X | VGA, HQ, HD |
| Zoom | 100 | ✓ | ✓ | ✓ | X | HD |

Table 2.9 Comparison table between various applications

* 1. . **Integrated summary of literature review**

As we have seen, all services offer some similar features, specifically:

> Web and video conferencing for 100 people.

> Secure unlimited meetings under the monthly SaaS fee.

> Screen sharing.

> File sharing.

> Chat and brainstorming tools like whiteboard.

> The ability to record meetings.

> Live support.

> Administrative controls.

Though it’s a widely used solution for business, there are many reasons to consider using Zoom Meetings for various purposes. Some of these include but are not limited to: video calls, audio and video conferencing, remote learning, and broadcasting and media.

With the increasing popularity of the web, some new web technologies emerged and introduced dynamics to web applications, in comparison to HTML, as a static programming language. JavaScript is the language that provided a dynamic web site which actively communicates with users. JavaScript is used in today's web applications as a client script language and on the server side. The JavaScript language supports the Model View Controller (MVC) architecture that maintains a readable code and clearly separates parts of the program code. The topic of this research is to compare the popular JavaScript frameworks: AngularJS, Ember, Knockout, Backbone. All four frameworks are based on MVC or similar architecture. In this paper, the advantages and disadvantages of each framework, the impact on application speed, the ways of testing such JS applications and ways to improve code security are presented.

Within computer science education, we have spent considerable effort on the introduction to the discipline (particularly to programming) and the teaching of novice programmers. However, we do not often think about the teaching and learning for the intermediate students. Having data about student's perceptions coming into a second year data structures course, it became of interest to systematically analyze the data to see what if any interesting patterns or results we could see in this information. Using the data from a first day of class general survey, we were able to gather information about student's perceived level of difficulty with some of the major topics that they have studied up until this point in the curriculum. The first half of the survey consisted of ten Likert-type items representing ten different topics in the previous courses. We are able to present quantitative analysis of the students perceived level of difficulty with these topics. The second half of the survey asked several open-ended questions about their learning. In this paper, we also provide analysis of the first of these questions which asked them to identify a topic that they studied previously that they still do not understand well

The rapid adoption of Zoom has revealed several security issues, including "Zoom-bombing". The company has responded swiftly and is freezing all new product development to address these issues.

Microsoft Teams is a great option for businesses that already have Office 365. However, it's not for everyone. After setting up the free version of Office for individuals.

Google Meet is a feature of Google G Suite that enables users to connect to other Google users.

With Google's easy-to-use interface and impressive features, Google Meet is a rival to the Editors' Choice selection of Zoom. Its security and encryption are also noteworthy. It also limits the number of people who can join a meeting to 15 minutes before the meeting begins. Google also has various compliance certifications.

This assessment investigates whether any aspect of the proposed project conflicts with legal requirements like zoning laws, data protection acts or social media laws. Let’s say an organization wants to construct a new office building in a specific location. A feasibility study might reveal the organization’s ideal location isn’t zoned for that type of business That organization has just saved considerable time and effort by learning that their project was not feasible right from the beginning.

This assessment involves undertaking a study to analyze and determine whether and how well the organization’s needs can be met by completing the project. Operational feasibility studies also examine how a project plan satisfies the requirements identified in the requirements analysis phase of system development.

This assessment is the most important for project success; after all, a project will fail if not completed on time. In scheduling feasibility, an organization estimates how much time the project will take to complete.

WebEx meeting hosts can share their desktop or virtual whiteboard with attendees. They can also control them from a simple UI.

It features a variety of tools that are necessary for web conferencing. Its callme feature is a bit expensive though.

Skype for Business is a unified communications app that combines the features of Skype and Lync. It works seamlessly with many key Microsoft apps, such as Office 365.

Skype web is a free video calling platform that enables users to make audio and video calls to each other. Its web app is available for Android and Windows PC.

Table 2.10 Problem in the above application



* 1. **Problem Statement**

The problem in the above system is they don’t support local server hosting, there is a limit for no of participants and there is no separate page for chat, there is no support for file sharing in these system

Data Collection**:** Large companies collect data from user and sell it, using it for their benefits, i.e. advertisement, which leads to security threat.

solution- we don’t collect data or store it on server.

Sign In**:** Most of the product in markets needs user to sign in through google, facebook, etc. In order to use their services.

solution- our product don’t entertain such practices.

Addictive**:**  when a product in developed in MNCs, all focus is on beautifying the product, which leads to people getting addicted to the product.

solution- we emphasized a lot on getting our product just perfect for what it is meant for.

Limited Joining**:** Leading products in market only allow you to host a meeting with limited participants, they charge for more participants.

solution- unlimited participants can join.

**Chapter 3**

**Report on Present Investigation**

**3.1 Overall description of project**

Our web conferencing application is built using HTML5 and is powered by Real-Time Communications (RTC) technology. It supports various types of video conferencing, presentation sharing, and public chat.

We have utilized the Heroku Cloud server to stream video content. It is a web server that serves as a platform for developers to create advanced video applications.

The ability to chat is achieved through web sockets. This functionality is also achieved through the use of video sharing. We have used various modules to achieve such functionalities, following are:-

* Socket.io
* PeerJs
* UUID
* ExpressJs
* NodeMailer
* WebRTC

**3.2.** Functional Requirement

Our application can run on various type of operating system and supports web browser such as:

1. Operating system-

* Window 7 and above.
* Android 4 and above.
* IOS 6+.
* Linux
* Mac OS

1. Browsers

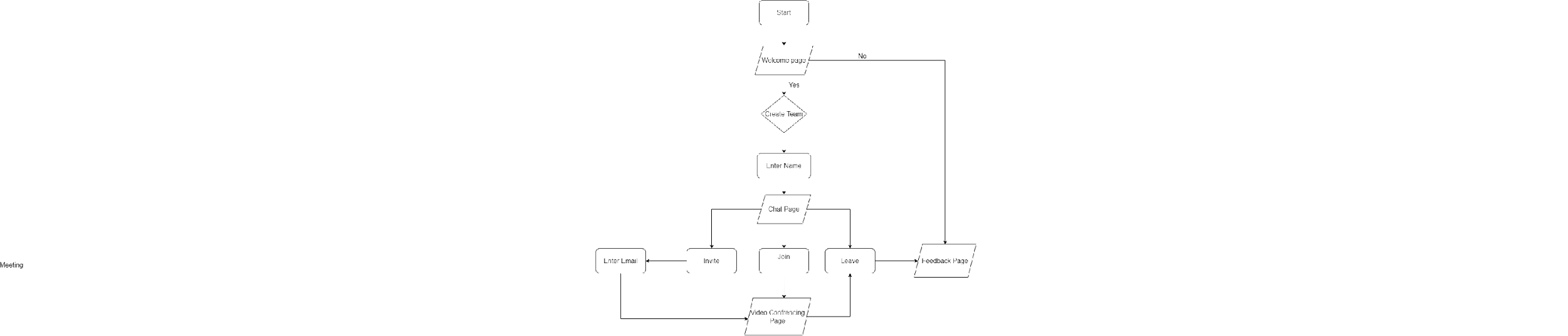
All modern browser such as:-

* Google Chrome
* Safari
* Internet explorer
* Mozilla Firefox
* Brave
* Opera

**3.3**. **Non Functional Requirement**

* The system should be able to connect to the internet and use the default browser to display a web page.
* The system should be able to capture video feed
* The system should be able to capture audio feed.
* The app should have a feature that allows users of remote devices(s) connected over Wifi Direct or Bluetooth connection(s)to share their screen with users of other remote devices connected over Wifi Direct or Bluetooth connection(s). This feature is known as screen sharing.
* The app should have a feature that allows users of remote devices(s) connected over Wifi Direct or Bluetooth connection(s)to share their microphone with users of other remote devices connected over Wifi Direct or Bluetooth connection(s).

**3.4. Flow Chart**

****Fig. 3.1 Flow chart of Video Conferencing Web Application

**3.5. Use Case Diagram**

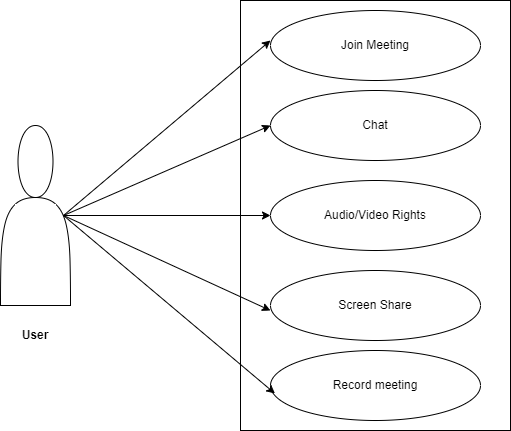
****

Fig. 3.2 Use Case Diagram of Video Conferencing Web Application

**Chapter 4**

**Results**

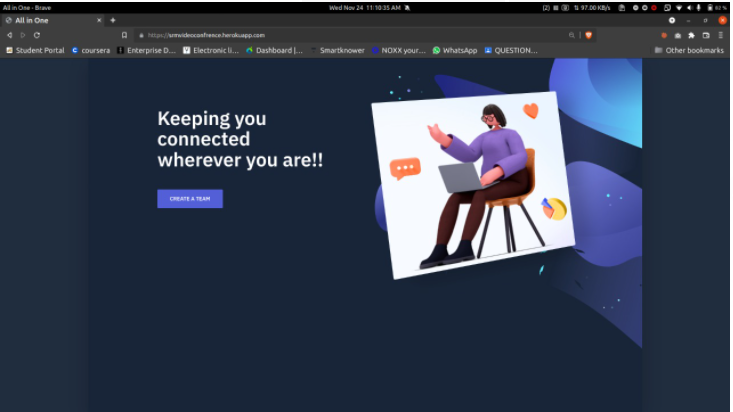
****

Fig.4.1 Welcome page

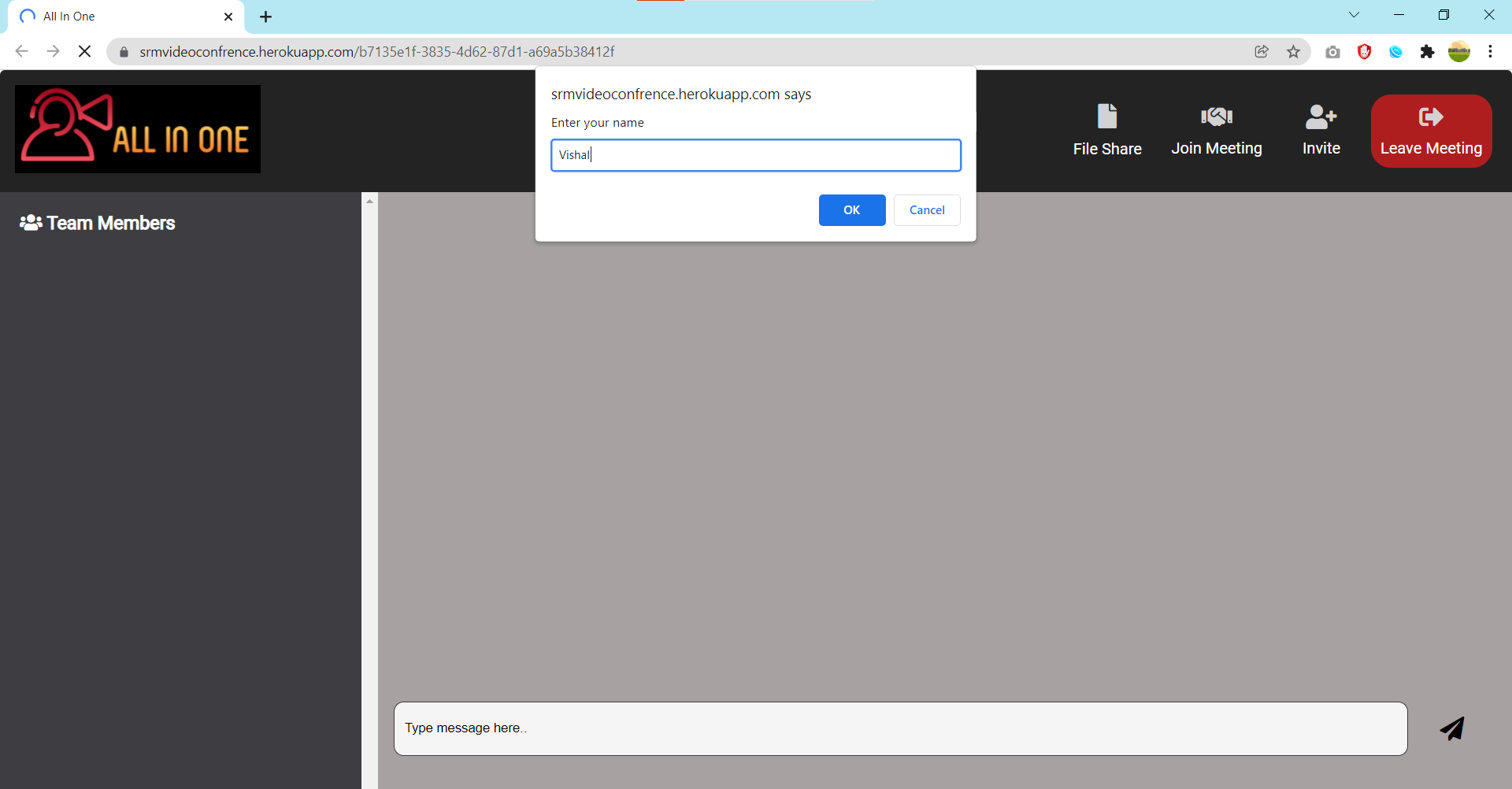


Fig. 4.2 Enter Name Alert Box

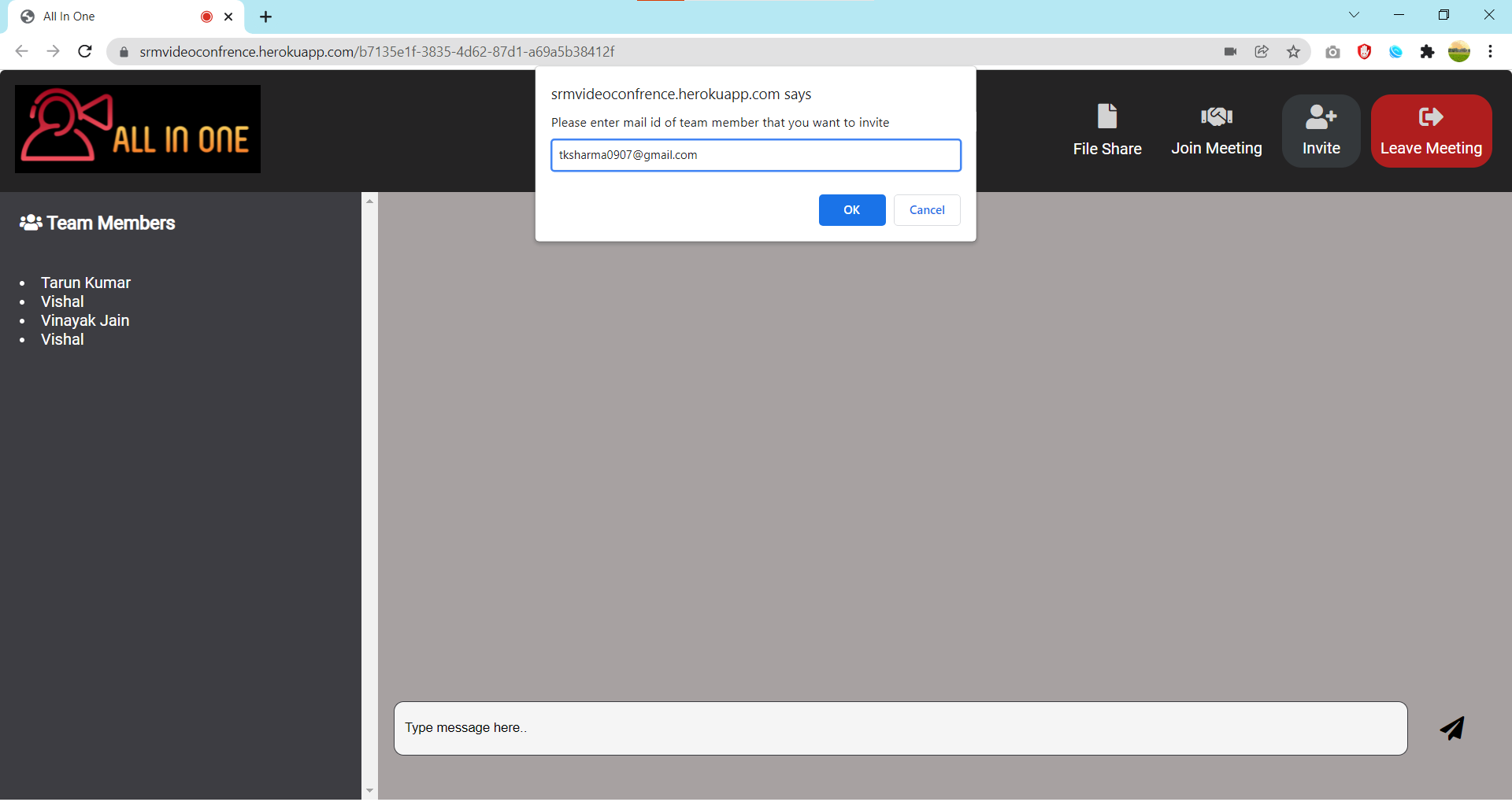


Fig.4.3 Invitation Alert box

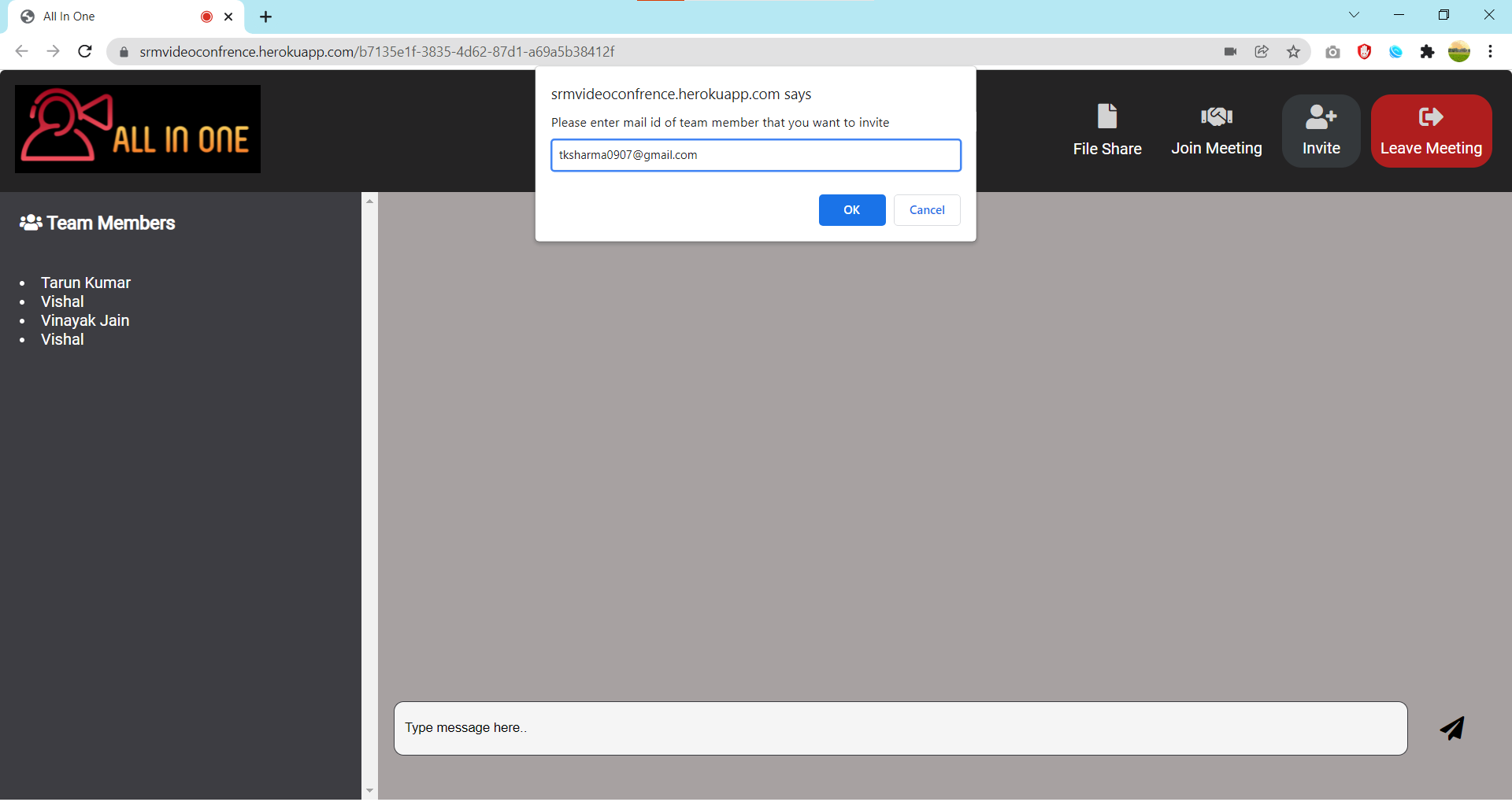


Fig.4.4 Chat Page

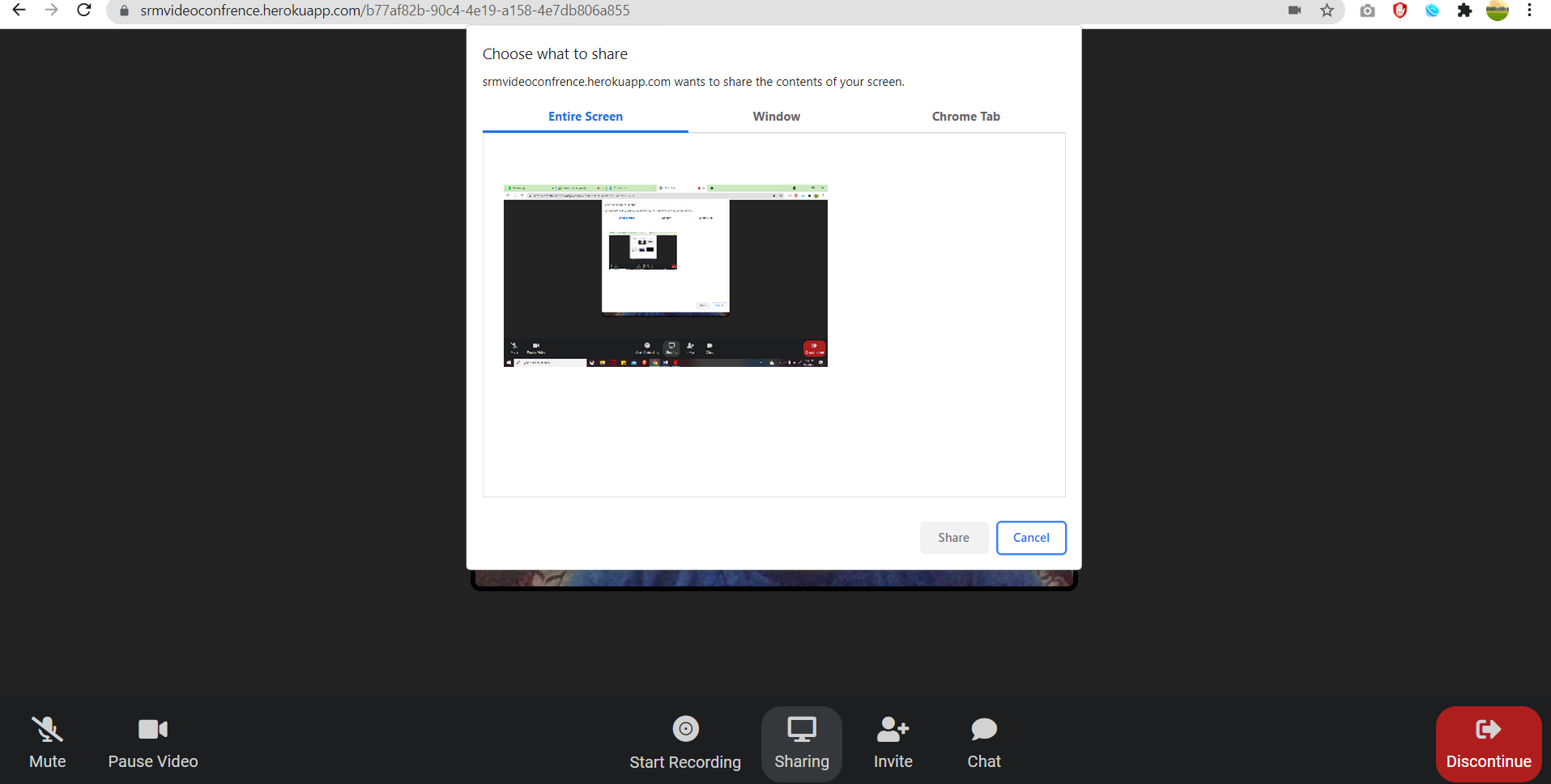


Fig. 4.6 Screen Share

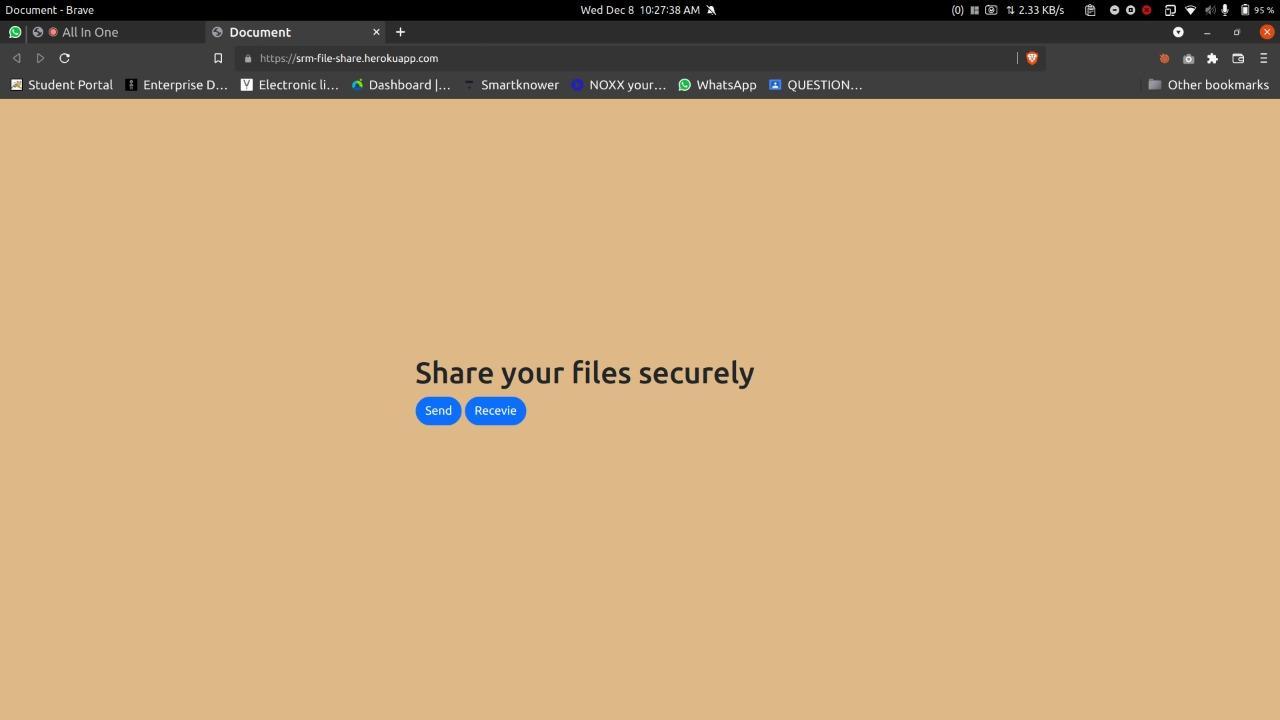


Fig 4.7 File share landing page

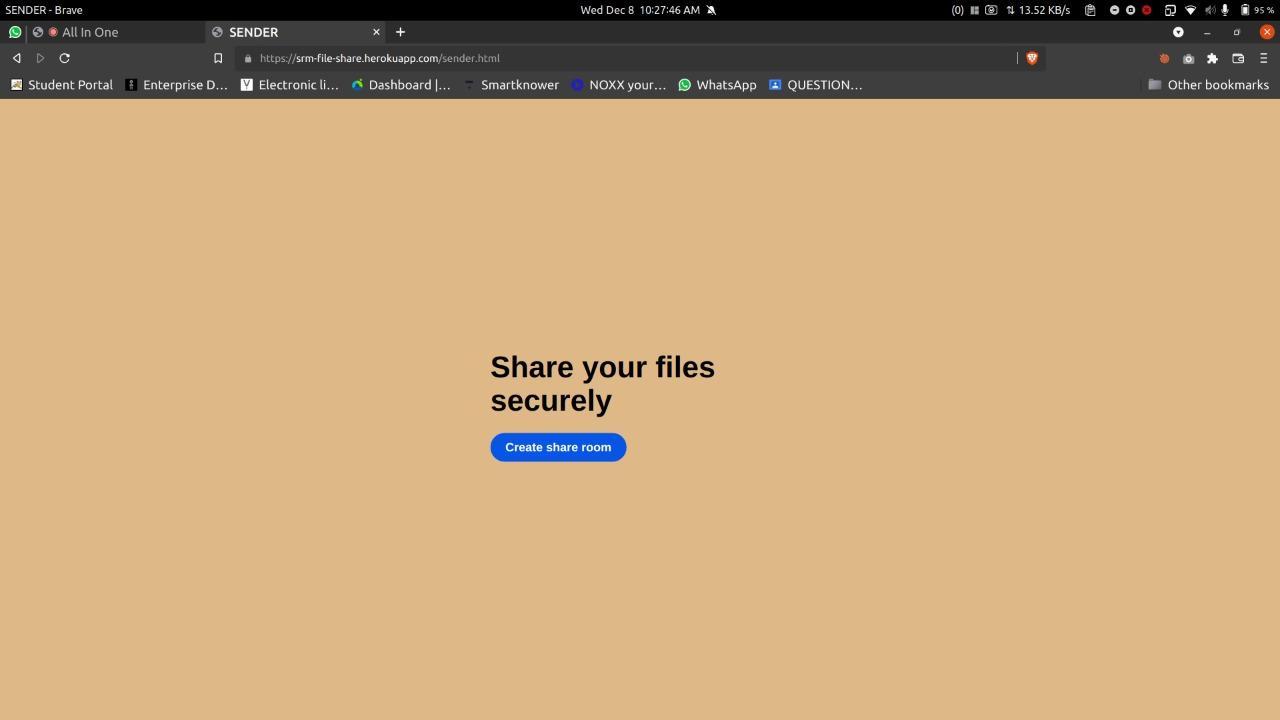


Fig 4.8 File sharing sender page

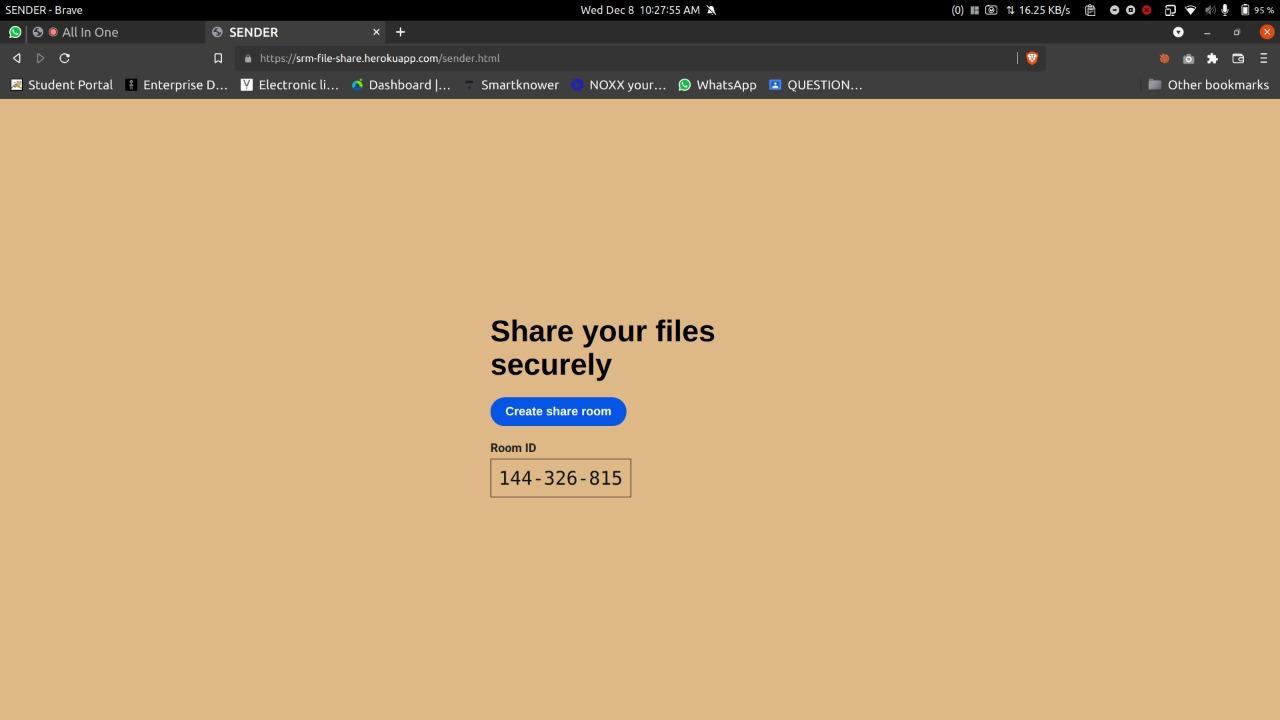
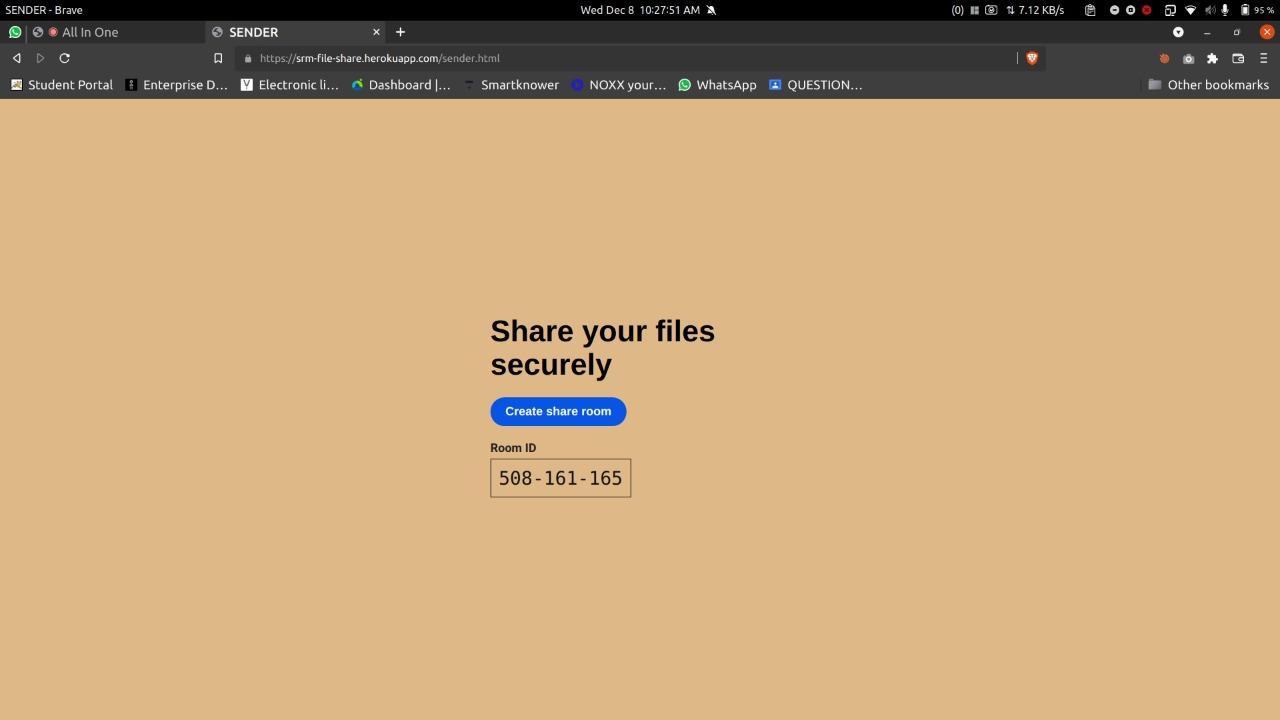


Fig 4.9 File sharing room creation



Fig. 4.10 File uploading page

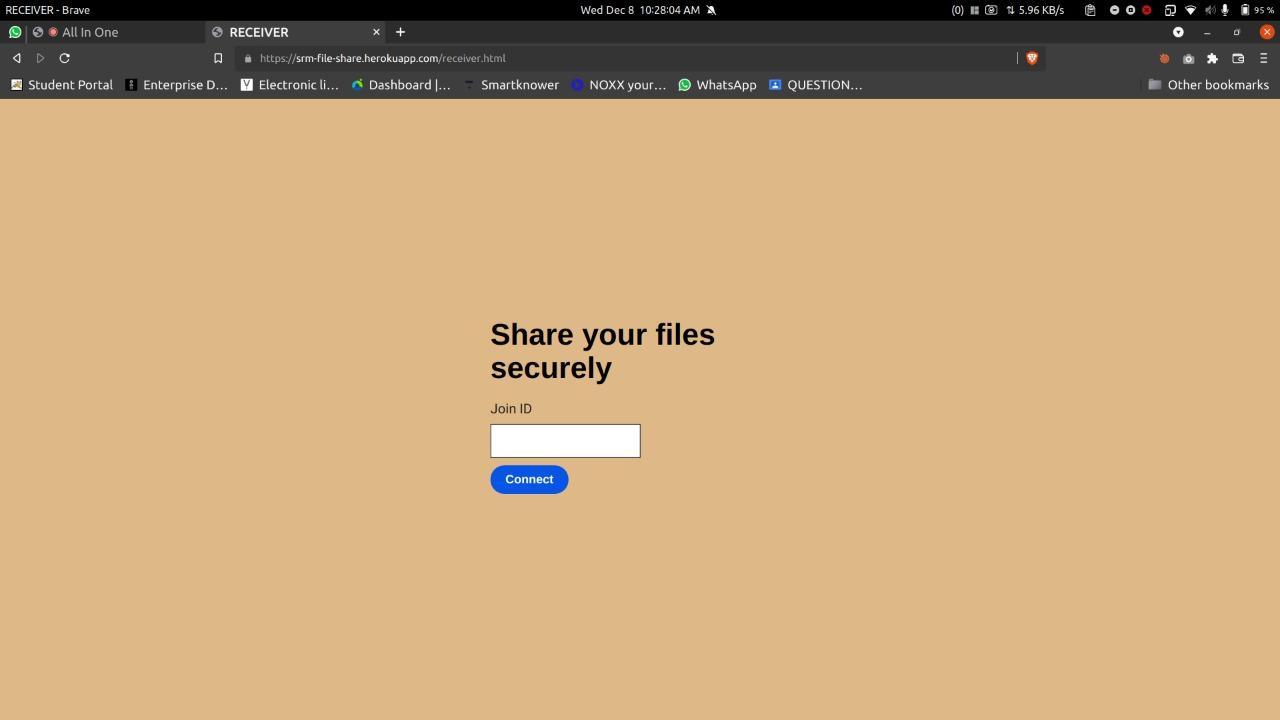


Fig 4.11 File sharing receiver page

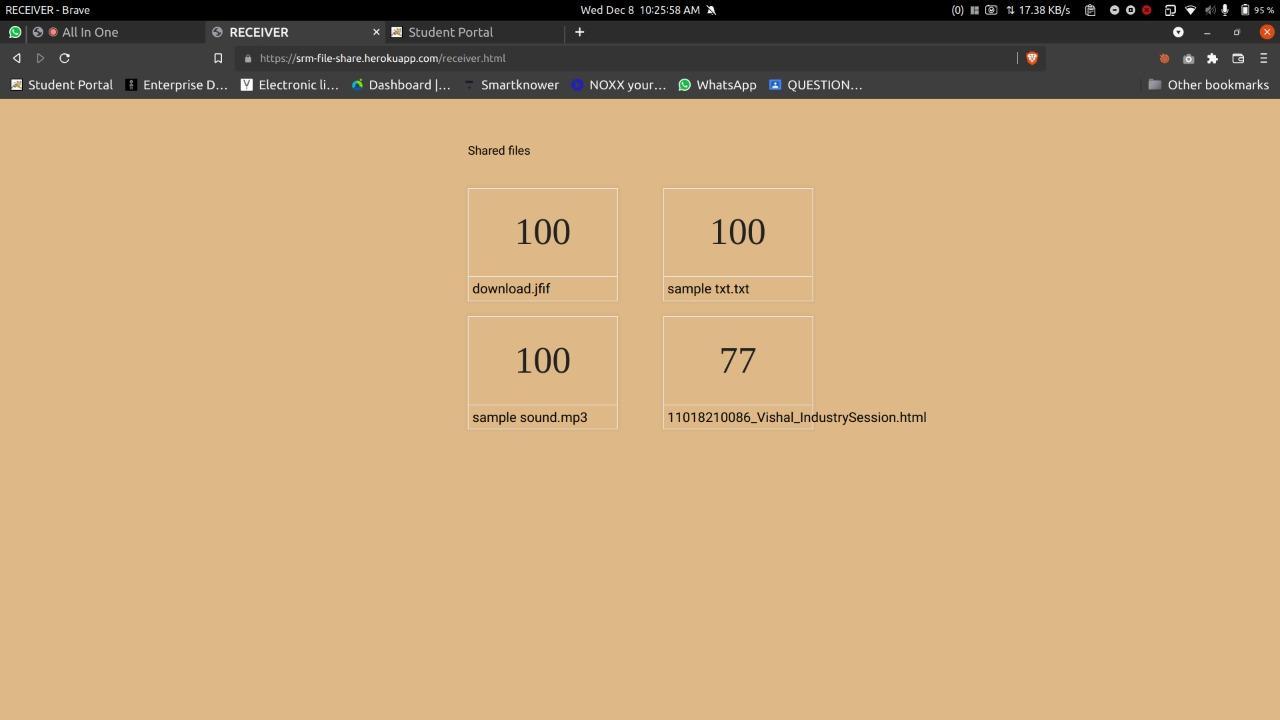


Fig. 4.12 File received

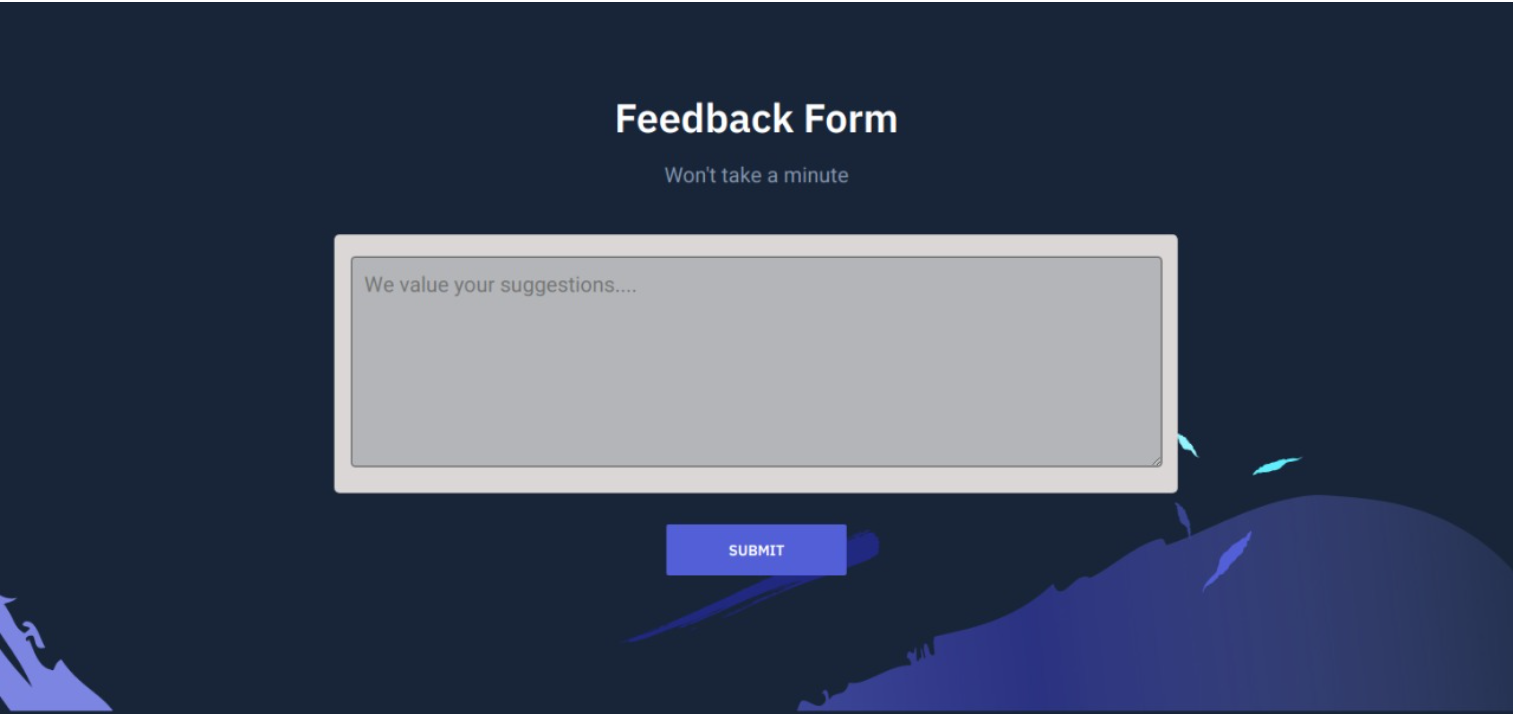


Fig.4.13 Feedback Page

After visiting **http://srmvideoconfrence.herokuapp.com/**, you are directed to video conferencing web application, then you are asked to enter your name and start video conferencing with other participants having the link

The Real Time Communication modules are designed to allow the users to create and join a room, share video and audio streams, and communicate with each other.

With our app, users can create and share their presentations and videos with other users. They can also communicate with each other through public chat.

**CODING**

**users.js :**

**const users = [];**

**// Join user to chat**

**function userJoin(id, username, room) {**

**const user = { id, username, room };**

**users.push(user);**

**return user;**

**}**

**// Get current user**

**function getCurrentUser(id) {**

**return users.find(user => user.id === id);**

**}**

**// User leaves chat**

**function userLeave(id) {**

**const index = users.findIndex(user => user.id === id);**

**if (index !== -1) {**

**return users.splice(index, 1)[0];**

**}**

**}**

**// Get room users**

**function getUsers(room) {**

**return users.filter(user => user.room === room);**

**}**

**module.exports = {**

**userJoin,**

**getCurrentUser,**

**userLeave,**

**getUsers**

**};**

**feedback.html :**

**<!DOCTYPE html>**

**<html lang="en" class="no-js">**

**<head>**

**<meta charset="utf-8">**

**<meta http-equiv="X-UA-Compatible" content="IE=edge">**

**<meta name="viewport" content="width=device-width, initial-scale=1">**

**<title>All In One</title>**

**<link href="https://fonts.googleapis.com/css?family=Heebo:400,700|IBM+Plex+Sans:600" rel="stylesheet">**

**<link rel="stylesheet" href="dist/css/style.css">**

**<script src="https://unpkg.com/scrollreveal@4.0.0/dist/scrollreveal.min.js"></script>**

**<style>**

**\* {**

**box-sizing: border-box;**

**}**

**#subject{**

**background-color:rgb(179, 181, 185);**

**}**

**input[type=text], select, textarea {**

**width: 100%;**

**padding: 0px;**

**border: 1px solid rgb(70, 68, 68);**

**border-radius: 4px;**

**resize: vertical;**

**}**

**input[type=email], select, textarea {**

**width: 100%;**

**padding: 12px;**

**border: 1px solid rgb(70, 68, 68);**

**border-radius: 4px;**

**resize: vertical;**

**}**

**label {**

**padding: 12px 12px 12px 0;**

**display: inline-block;**

**}**

**input[type=submit] {**

**background-color: rgb(37, 116, 161);**

**color: white;**

**padding: 12px 20px;**

**border: none;**

**border-radius: 4px;**

**cursor: pointer;**

**float: right;**

**}**

**input[type=submit]:hover {**

**background-color: #45a049;**

**}**

**.container1 {**

**border-radius: 5px;**

**background-color: rgb(219, 215, 215);**

**padding: 15px;**

**}**

**.col-25 {**

**float: left;**

**width: 25%;**

**margin-top: 6px;**

**}**

**.col-75 {**

**float: left;**

**width: 75%;**

**margin-top: 6px;**

**}**

**.col-75-1 {**

**float: left;**

**width: 100%;**

**margin-top: 6px;**

**}**

**/\* Clear floats after the columns \*/**

**.row:after {**

**content: "";**

**display: table;**

**clear: both;**

**}**

**h2{**

**text-align:center;**

**}**

**</style>**

**</head>**

**<body class="is-boxed has-animations">**

**<div class="body-wrap boxed-container">**

**<main>**

**<section class="cta section">**

**<div class="container-sm">**

**<div class="cta-inner section-inner">**

**<div class="cta-header text-center">**

**<h2 class="section-title mt-0">Feedback Form</h2>**

**<p class="section-paragraph">Won't take a minute</p>**

**<input id="lights-toggle" type="checkbox" name="lights-toggle" class="switch" >**

**<div class="container1">**

**<form>**

**<div class="row">**

**<div class="col-75-1">**

**<textarea id="subject" name="subject" placeholder="We value your suggestions...." style="height:200px"></textarea>**

**</div>**

**</div>**

**</form>**

**</div>**

**<br>**

**<div class="cta-cta">**

**<a class="button button-primary" href="mailto:eshakaushik2001@gmail.com" >Submit</a>**

**</div>**

**</div>**

**</div>**

**</div>**

**</section>**

**</main>**

**</div>**

**<script src="dist/js/main.min.js"></script>**

**</body>**

**</html>**

**index.html :**

**<!DOCTYPE html>**

**<html lang="en" class="no-js">**

**<head>**

**<meta charset="utf-8">**

**<meta http-equiv="X-UA-Compatible" content="IE=edge">**

**<meta name="viewport" content="width=device-width, initial-scale=1">**

**<title>All In One</title>**

**<link href="https://fonts.googleapis.com/css?family=Heebo:400,700|IBM+Plex+Sans:600" rel="stylesheet">**

**<link rel="stylesheet" href="dist/css/style.css">**

**<script src="https://unpkg.com/scrollreveal@4.0.0/dist/scrollreveal.min.js"></script>**

**</head>**

**<!-- This is the first landing page of the website -->**

**<body class="is-boxed has-animations">**

**<div class="body-wrap boxed-container">**

**<main>**

**<section class="hero">**

**<div class="container">**

**<div class="hero-inner">**

**<div class="hero-copy">**

**<h1 class="hero-title mt-0" style="font-size:50px;">Keeping you connected wherever you are!!</h1>**

**<div class="hero-cta">**

**<br><br><br><br>**

**<a class="button button-primary" href="./direct">Create a team </a>**

**<div class="lights-toggle">**

**<input id="lights-toggle" type="checkbox" name="lights-toggle" class="switch" >**

**</div>**

**</div>**

**<br><br>**

**<!--Directed to the github repository-->**

**<p class="hero-paragraph"><a href="https://github.com/EshaKaushik/Video-Conferencing-App#readme">Want to explore more....</a></p>**

**</div>**

**<div class="hero-media">**

**<div class="header-illustration">**

**<img class="header-illustration-image asset-light" src="dist/images/header-illustration-light.svg" alt="Header illustration">**

**<img class="header-illustration-image asset-dark" src="dist/images/header-illustration-dark.svg" alt="Header illustration">**

**</div>**

**<div class="hero-media-illustration">**

**<img class="hero-media-illustration-image asset-light" src="dist/images/hero-media-illustration-light.svg" alt="Hero media illustration">**

**<img class="hero-media-illustration-image asset-dark" src="dist/images/hero-media-illustration-dark.svg" alt="Hero media illustration">**

**</div>**

**<div class="hero-media-container">**

**<img class="hero-media-image asset-light" src="dist/images/design1.png" width="500px" alt="Hero media">**

**<img class="hero-media-image asset-dark" src="dist/images/design1.png" width="500px" alt="Hero media">**

**</div>**

**</div>**

**</div>**

**</div>**

**</section>**

**</div>**

**<script src="dist/js/main.min.js"></script>**

**</body>**

**</html>**

**room.ejs :**

**<!DOCTYPE html>**

**<html lang="en">**

**<head>**

**<meta charset="UTF-8" />**

**<meta name="viewport" content="width=device-width, initial-scale=1.0" />**

**<title>All In One</title>**

**<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/font-awesome/4.7.0/css/font-awesome.min.css"/>**

**<link rel="stylesheet" href="style.css" />**

**<script src="https://kit.fontawesome.com/c939d0e917.js"></script>**

**<script src="https://unpkg.com/peerjs@1.3.1/dist/peerjs.min.js"></script>**

**<script src="/socket.io/socket.io.js"></script>**

**<script src="https://cdnjs.cloudflare.com/ajax/libs/moment.js/2.18.1/moment.min.js"></script>**

**<script>**

**const ROOM\_ID = "<%= roomId %>";**

**console.log(ROOM\_ID);**

**</script>**

**</head>**

**<body>**

**<div class="main">**

**<div class="main\_\_left">**

**<div class="main\_\_videos">**

**<div id="video-grid"></div>**

**</div>**

**<div class="main\_\_controls">**

**<div class="main\_\_controls\_block">**

**<div class="main\_\_controls\_button" id="muteButton" onclick="muteUnmute()">**

**<i class="fa fa-microphone-slash"></i>**

**<span>Mute</span>**

**</div>**

**<div class="main\_\_controls\_button" id="playPauseVideo" onclick="playStop()">**

**<i class="fa fa-video-camera"></i>**

**<span>Pause Video</span>**

**</div>**

**</div>**

**<div class="main\_\_controls\_block">**

**<div class="main\_\_controls\_button" id="recording">**

**<i class="fa fa-record-vinyl" id="record-icon" style='font-size:26px'></i>**

**<span id="record">Start Recording</span>**

**<select id="codecPreferences" style="display: none;" ></select>**

**</div>**

**<select id="codecPreferences" style="display: none;" ></select>**

**<div class="main\_\_controls\_button">**

**<i class="fa fa-desktop"></i>**

**<span id="screen" >Share Screen</span>**

**</div>**

**<div class="main\_\_controls\_button" id="inviteButton" >**

**<i class="fa fa-user-plus"></i>**

**<span>Invite</span>**

**</div>**

**<div class="main\_\_controls\_button" id="showChat" >**

**<i class="fa fa-comment"></i>**

**<span>Chat</span>**

**</div>**

**</div>**

**<div class="main\_\_controls\_block">**

**<div class="main\_\_controls\_button leaveMeeting" id="discontinue">**

**<i class="fa fa-sign-out-alt"></i>**

**<span >Discontinue</span>**

**</div>**

**</div>**

**</div>**

**</div>**

**<div class="main\_\_right">**

**<div class="chat-container">**

**<header class="chat-header">**

**<div>**

**<img src="/images/logo1.png" height="90px" width="250px">**

**</div>**

**<div class="main\_\_controls\_block">**

**<div id="join-video" class="main\_\_controls\_button ">**

**<i class="fa fa-handshake"></i>**

**<span id="join-video">Join Meeting</span>**

**</div>**

**<div class="main\_\_controls\_button" id="inviteButton\_chat" >**

**<i class="fa fa-user-plus"></i>**

**<span>Invite</span>**

**</div>**

**<div class="main\_\_controls\_block">**

**<div class="main\_\_controls\_button leaveMeeting" id="leave-meeting" onclick="leave()">**

**<i class="fa fa-sign-out-alt"></i>**

**<span >Leave Meeting</span>**

**</div>**

**</div>**

**</header>**

**<main class="chat-main">**

**<div class="chat-sidebar">**

**<h3><i class="fas fa-users"></i> Team Members</h3>**

**<div id="users"></div>**

**</div>**

**<div class="main\_\_chat\_\_window" id="main\_\_chat\_\_window">**

**<div class="messages" id="all\_messages">**

**</div>**

**<div class="main\_\_message\_container">**

**<input type="text" id="chat\_message" placeholder="Type message here.."/>**

**<div id="send" class="main\_\_controls\_button send">**

**<i class="fa fa-paper-plane" ></i>**

**</div>**

**</div>**

**</div>**

**</main>**

**</div>**

**</div>**

**<script src="script.js"></script>**

**</body>**

**</html>**

**server.js :**

**const express = require("express");**

**var nodemailer = require('nodemailer');**

**const app = express();**

**const server = require("http").Server(app);**

**const { v4: uuidv4 } = require("uuid");**

**const io = require("socket.io")(server,{**

**cors: {**

**origin: '\*'**

**}**

**});**

**const { userJoin,getCurrentUser,userLeave,getUsers} = require('./utils/users');**

**const { ExpressPeerServer } = require("peer");**

**const { Console } = require("console");**

**const peerServer = ExpressPeerServer(server, {**

**debug: true,**

**});**

**// Function to send an invitation mail to team members**

**function sent\_mail(mail\_id,link){**

**var transporter = nodemailer.createTransport({**

**service: 'gmail',**

**auth: {**

**user: '\*\*username\*\*',**

**pass: '\*\*password\*\*'**

**}**

**});**

**var mailOptions = {**

**from: '\*\*user\_mailID\*\*',**

**to: mail\_id,**

**subject: 'Invitation link to join the team !!',**

**text:link**

**};**

**transporter.sendMail(mailOptions, function(error, info){**

**if (error) {**

**console.log(error);**

**} else {**

**console.log('Email sent: ' + info.response);**

**}**

**});**

**};**

**app.set("view engine", "ejs");**

**app.use(express.static("public"));**

**app.use("/peerjs", peerServer);**

**app.engine('html', require('ejs').renderFile);**

**// Render the first landing page**

**app.get('/', function(req, res) {**

**res.render("index.html");**

**});**

**// Render the feedback form after leaving the call**

**app.get('/feedback', function(req, res) {**

**res.render("feedback.html");**

**});**

**// Creating a unique ID for new room**

**app.get("/direct", (req, rsp) => {**

**rsp.redirect(`/${uuidv4()}`);**

**});**

**// Joining the room generated via unique Id**

**app.get("/:room", (req, res) => {**

**roomId=`/${uuidv4()}`;**

**console.log(roomId);**

**res.render("room", { roomId: req.params.room });**

**});**

**// When connection happens**

**io.on("connection", (socket) => {**

**socket.on("join-room", (roomId, userId,userName) => {**

**const user = userJoin(socket.id, userName, roomId);**

**socket.join(roomId);**

**socket.to(roomId).broadcast.emit("user-connected", userId);**

**io.to(roomId).emit('Users',{**

**users:getUsers(roomId)**

**});**

**socket.on("message", (message,time) => {**

**io.to(roomId).emit("createMessage", message,userName,time);**

**});**

**/\*socket.on("disconnect", () => {**

**const user = userLeave(socket.id);**

**io.to(roomId).emit('userLeft', userName);**

**io.to(roomId).emit('Users',{**

**users:getUsers(roomId)**

**});**

**});\*/**

**socket.on("initiate", () => {**

**io.to(roomId).emit("share-screen");**

**});**

**socket.on("mail\_sent",(mail\_id,link) =>{**

**sent\_mail(mail\_id,link);**

**io.to(roomId).emit("success");**

**});**

**});**

**});**

**server.listen(process.env.PORT || 80);**

**script.js :**

**const socket = io("/");**

**const all\_messages = document.getElementById("all\_messages");**

**const main\_\_chat\_\_window = document.getElementById("main\_\_chat\_\_window");**

**const videoGrid = document.getElementById("video-grid");**

**const myVideo = document.createElement("video");**

**myVideo.muted = true;**

**var myUserId="";**

**var peers={};**

**const user = prompt("Enter your name");**

**if(user.trim().length ==0){**

**document.write("Enter the username is mandatory to create a room ");**

**}**

**/// Declaring the peers**

**var peer = new Peer(undefined, {**

**path: "/peerjs",**

**host: "video-conferencing-webapp.herokuapp.com",**

**port: 443,**

**secure: true,**

**});**

**let myVideoStream;**

**var getUserMedia = navigator.getUserMedia || navigator.webkitGetUserMedia || navigator.mozGetUserMedia;**

**navigator.mediaDevices.getUserMedia({**

**video: true,**

**audio: true,**

**})**

**.then((stream) => {**

**myVideoStream = stream;**

**myVideoStream.getAudioTracks()[0].enabled = false;**

**addVideoStream(myVideo, stream);**

**peer.on("call", (call) => {**

**call.answer(stream);**

**const video = document.createElement("video");**

**call.on("stream", (userVideoStream) => {**

**addVideoStream(video, userVideoStream);**

**});**

**peer.peerConnection.onconnectionstatechange = function (event) {**

**if (event.currentTarget.connectionState === 'disconnected') {**

**peer.close();**

**}**

**};**

**});**

**socket.on("user-connected", (userId) => {**

**myUserId=userId;**

**connectToNewUser(userId, stream);**

**});**

**});**

**////// Getting connected team members list**

**const userList = document.getElementById('users');**

**socket.on('Users', ({ users }) => {**

**//outputRoomName(room);**

**outputUsers(users);**

**});**

**function outputUsers(users) {**

**userList.innerHTML = '';**

**users.forEach((user) => {**

**const li = document.createElement('li');**

**li.innerText = user.username;**

**userList.appendChild(li);**

**});**

**}**

**// When new user is connecting**

**const connectToNewUser = (userId, streams) => {**

**var call = peer.call(userId, streams);**

**console.log(call);**

**var video = document.createElement("video");**

**call.on("stream", (userVideoStream) => {**

**addVideoStream(video, userVideoStream);**

**});**

**peers[userId] = call;**

**};**

**// Adding a new video-stream to the video-grid**

**const addVideoStream = (videoEl, stream) => {**

**videoEl.srcObject = stream;**

**videoEl.addEventListener("loadedmetadata", () => {**

**videoEl.play();**

**});**

**videoGrid.append(videoEl);**

**let totalUsers = document.getElementsByTagName("video").length;**

**if (totalUsers > 1) {**

**for (let index = 0; index < totalUsers; index++) {**

**document.getElementsByTagName("video")[index].style.width =**

**100 / totalUsers + "%";**

**}**

**}**

**};**

**/// Join the video conference meeting by clicking on Join button in Main page**

**const join\_meet=document.getElementById("join-video");**

**join\_meet.addEventListener("click", (e) =>{**

**document.querySelector(".main\_\_right").classList.toggle("click");**

**document.querySelector(".main\_\_left").classList.toggle("click");**

**});**

**socket.on("user-disconnected", (userId) => {**

**if (peers[userId]) peers[userId].close();**

**});**

**////////////// CHAT BOX FUNCTIONALITY**

**let chatInputBox = document.querySelector("#chat\_message");**

**let send = document.getElementById("send");**

**let messages = document.querySelector(".messages");**

**//**

**send.addEventListener("click", (e) => {**

**var time=moment().format('h:mm a');**

**if (chatInputBox.value.length !== 0) {**

**socket.emit("message", chatInputBox.value,time);**

**chatInputBox.value = "";**

**}**

**});**

**chatInputBox.addEventListener("keydown", (e) => {**

**var time=moment().format(' MMMM Do YYYY, h:mm a');**

**if (e.key === "Enter" && chatInputBox.value != "") {**

**socket.emit("message", chatInputBox.value,time);**

**chatInputBox.value = "";**

**}**

**});**

**socket.on("createMessage", (message, userName,time) => {**

**const receivedMsg = `**

**<div class="message\_recieve" id ="msg">**

**<b><i class="fa fa-user-circle"></i> <span> ${**

**userName**

**}</span> </b> &nbsp &nbsp <span class="time">${time}</span><br>**

**<span class="message">&nbsp &nbsp&nbsp&nbsp${message}</span>**

**</div>`;**

**const myMsg = `**

**<div class="message\_sent" id ="msg">**

**<b><i class="fa fa-user-circle"></i> <span> ${**

**"me"**

**}</span></b> &nbsp &nbsp<span class="time">${time}</span><br>**

**<span >&nbsp &nbsp&nbsp&nbsp${message}</span>**

**</div>`;**

**messages.innerHTML =messages.innerHTML + (user === userName ? myMsg : receivedMsg);**

**});**

**//// When chat button is clicked it shows the history of chat**

**var buttons = document.getElementById("showChat");**

**var discontinue = document.getElementById("discontinue");**

**buttons.addEventListener("click", (e) => {**

**document.querySelector(".main\_\_right").classList.toggle("click");**

**document.querySelector(".main\_\_left").classList.toggle("click");**

**});**

**discontinue.addEventListener("click", (e) => {**

**document.querySelector(".main\_\_right").classList.toggle("click");**

**document.querySelector(".main\_\_left").classList.toggle("click");**

**});**

**peer.on("call", function (call) {**

**getUserMedia(**

**{ video: true, audio: true },**

**function (stream) {**

**call.answer(stream); // Answer the call with an A/V stream.**

**const video = document.createElement("video");**

**call.on("stream", function (remoteStream) {**

**addVideoStream(video, remoteStream);**

**});**

**},**

**function (err) {**

**console.log("Failed to get local stream", err);**

**}**

**);**

**});**

**peer.on("open", (id) => {**

**socket.emit("join-room", ROOM\_ID, id,user);**

**});**

**// Called when screen is shared in the meeting**

**const connectToscreen = (userId, streams) => {**

**var call = peer.call(userId, streams);**

**var video = document.createElement("video");**

**call.on("stream", (screenTrack) => {**

**console.log(screenTrack);**

**});**

**call.on("close", () => {**

**video.remove();**

**});**

**peers[userId] = call;**

**};**

**/// Share screen functionality**

**let share = document.getElementById("screen");**

**function shareScreen() {**

**navigator.mediaDevices.getDisplayMedia({ cursor: "true" ,video:true }).then(stream => {**

**screenTrack = stream;**

**var video = document.createElement("video");**

**addVideoStream(video, screenTrack);**

**socket.emit("initiate");**

**stream.getVideoTracks()[0].onended = function () {**

**share.textContent = 'Share Screen';**

**};**

**})**

**}**

**share.addEventListener("click", (e) =>{**

**if (share.textContent === 'Share Screen') {**

**share.textContent = 'Sharing';**

**shareScreen();**

**}**

**});**

**socket.on("share-screen",() =>{**

**console.log("Sharing screen ");**

**console.log(screenTrack);**

**connectToscreen(myUserId,screenTrack);**

**console.log("Sharing screen ");**

**})**

**/// This section deals with the toggling of audio and video track**

**const playStop = () => {**

**let enabled = myVideoStream.getVideoTracks()[0].enabled;**

**if (enabled) {**

**myVideoStream.getVideoTracks()[0].enabled = false;**

**setPlayVideo();**

**} else {**

**setStopVideo();**

**myVideoStream.getVideoTracks()[0].enabled = true;**

**}**

**};**

**const muteUnmute = () => {**

**const enabled = myVideoStream.getAudioTracks()[0].enabled;**

**if (enabled) {**

**myVideoStream.getAudioTracks()[0].enabled = false;**

**setUnmuteButton();**

**} else {**

**setMuteButton();**

**myVideoStream.getAudioTracks()[0].enabled = true;**

**}**

**};**

**const setPlayVideo = () => {**

**const html = `<i class="unmute fa fa-pause-circle"></i>**

**<span class="unmute">Resume Video</span>`;**

**document.getElementById("playPauseVideo").innerHTML = html;**

**};**

**const setStopVideo = () => {**

**const html = `<i class=" fa fa-video-camera"></i>**

**<span class="">Pause Video</span>`;**

**document.getElementById("playPauseVideo").innerHTML = html;**

**};**

**const setUnmuteButton = () => {**

**const html = `<i class="unmute fa fa-microphone-slash"></i>**

**<span class="unmute">Unmute</span>`;**

**document.getElementById("muteButton").innerHTML = html;**

**};**

**const setMuteButton = () => {**

**const html = `<i class="fa fa-microphone"></i>**

**<span>Mute</span>`;**

**document.getElementById("muteButton").innerHTML = html;**

**};**

**// Invite a guest in your meeting**

**const inviteButton = document.querySelector("#inviteButton");**

**inviteButton.addEventListener("click", (e) => {**

**var mail\_id=prompt("Please enter mail id of guest that you want to invite","");**

**var link=window.location.href ;**

**socket.emit("mail\_sent",mail\_id, link);**

**});**

**// Invite a new member to your team**

**const inviteButton2 = document.querySelector("#inviteButton\_chat");**

**inviteButton2.addEventListener("click", (e) => {**

**var mail\_id=prompt("Please enter mail id of team member that you want to invite","");**

**var link=window.location.href ;**

**socket.emit("mail\_sent",mail\_id, link);**

**});**

**socket.on("success",()=>{**

**console.log("success");**

**setTimeout(() => { console.log("World!");alert("message successfully sent!!"); }, 3000);**

**});**

**//// SCREEN RECORDING FUNCTIONALITY**

**'use strict';**

**let mediaRecorder;**

**let recordedBlobs;**

**const codecPreferences = document.querySelector('#codecPreferences');**

**const errorMsgElement = document.querySelector('span#errorMsg');**

**const recordedVideo = document.querySelector('video#recorded');**

**const recordButton = document.querySelector('#record');**

**const downloadButton = document.querySelector('button#download');**

**recordButton.addEventListener('click', () => {**

**console.log("record start");**

**recordButton.disabled = false;**

**console.log('getUserMedia() got stream:', myVideoStream);**

**window.stream = myVideoStream;**

**const gumVideo = document.querySelector('#video-grid');**

**gumVideo.srcObject = myVideoStream;**

**getSupportedMimeTypes().forEach(mimeType => {**

**const option = document.createElement('option');**

**option.value = mimeType;**

**option.innerText = option.value;**

**codecPreferences.appendChild(option);**

**});**

**codecPreferences.disabled = false;**

**if (recordButton.textContent === 'Start Recording') {**

**startRecording();**

**document.getElementById("record-icon").style.color = "red";**

**} else {**

**stopRecording();**

**recordButton.textContent = 'Start Recording';**

**document.getElementById("record-icon").style.color = "white";**

**codecPreferences.disabled = false;**

**}**

**});**

**function download(){**

**const blob = new Blob(recordedBlobs, {type: 'video/webm'});**

**const url = window.URL.createObjectURL(blob);**

**const a = document.createElement('a');**

**a.style.display = 'none';**

**a.href = url;**

**a.download = 'test.webm';**

**document.body.appendChild(a);**

**a.click();**

**setTimeout(() => {**

**document.body.removeChild(a);**

**window.URL.revokeObjectURL(url);**

**}, 100);**

**}**

**function handleDataAvailable(event) {**

**console.log('handleDataAvailable', event);**

**if (event.data && event.data.size > 0) {**

**recordedBlobs.push(event.data);**

**}**

**}**

**function getSupportedMimeTypes() {**

**const possibleTypes = [**

**'video/webm;codecs=vp9,opus',**

**'video/webm;codecs=vp8,opus',**

**'video/webm;codecs=h264,opus',**

**'video/mp4;codecs=h264,aac',**

**];**

**return possibleTypes.filter(mimeType => {**

**return MediaRecorder.isTypeSupported(mimeType);**

**});**

**}**

**function startRecording() {**

**recordedBlobs = [];**

**const mimeType = codecPreferences.options[codecPreferences.selectedIndex].value;**

**const options = {mimeType};**

**try {**

**mediaRecorder = new MediaRecorder(window.stream, options);**

**} catch (e) {**

**console.error('Exception while creating MediaRecorder:', e);**

**errorMsgElement.innerHTML = `Exception while creating MediaRecorder: ${JSON.stringify(e)}`;**

**return;**

**}**

**console.log('Created MediaRecorder', mediaRecorder, 'with options', options);**

**recordButton.textContent = 'Stop Recording';**

**// downloadButton.disabled = true;**

**codecPreferences.disabled = true;**

**mediaRecorder.onstop = (event) => {**

**console.log('Recorder stopped: ', event);**

**console.log('Recorded Blobs: ', recordedBlobs);**

**download();**

**};**

**mediaRecorder.ondataavailable = handleDataAvailable;**

**mediaRecorder.start();**

**console.log('MediaRecorder started', mediaRecorder);**

**}**

**function stopRecording() {**

**mediaRecorder.stop();**

**}**

**//// When someone leaves that room**

**const leave = () => {**

**let want\_to\_leave = confirm("Are you sure you want to leave the team?");**

**if(want\_to\_leave)**

**{**

**socket.emit("disconnect");**

**window.location.replace("/feedback");**

**}**

**};**

**socket.on('userLeft', (userName) =>{**

**console.log("disconnect client");**

**alert(userName+' has left the meeting');**

**});**

**Chapter 5**

**Conclusion and Future Work**

5.1 **Conclusion**

Video conferencing platforms are very popular among various industries due to their ease of use and ability to accommodate various communication styles. The emergence of virtual platforms such as the lockdown has changed the way people communicate. Although it can be very beneficial for some, it can also be very risky for others. With this application, we have been able to solve the problems that traditional meeting and interviews face. It is beneficial in offices and companies that have good internet access and as such desire a quick and effective way of conducting meetings and even interviews. The application is able to achieve its purpose.

5.2 **Future Work**

* Make our project mobile friendly
* Introducing file sharing system
* Improvement in user experience
* Adding admin rights
* Introducing more tools for participants interaction

**Chapter 6**

**Reference**

[1] Wesley Chai, Irwin Lazar-”Video Conferencing”,

https://searchunifiedcommunications.techtarget.com/definition/video-conference (accessed 22/09/21)

[2] Ing. Slavomír Petrík, prof. Ing. Václav Skala, CSc- ”Report on Video Conferencing system”, December 2006, Available-<https://www.kiv.zcu.cz/site/documents/verejne/vyzkum/publikace/technicke-zpravy/2006/tr-2006-04.pdf>

[3] K.V. Rop and N.K. Bett- “Video Conferencing and its application in distance learning”, June 2012, Available- <https://www.researchgate.net/publication/251237239_VIDEO_CONFERENCING_AND_ITS_APPLICATION_IN_DISTANCE_LEARNING>

[4] Sanabil A Mahmood, Ergun Erceblebe, Sanabil Mahmood Hadeed, “Deveopment of videoconferencing platform based on web RTC”, [Online]08/2018, <https://www.researchgate.net/publication/337006847_DEVELOPMENT_OF_VIDEO_CONFERENCE_PLATFORM_BASED_ON_WEB_RTC>

[5] Andreas Hallberg, “A protocol for decentralized video conferencing with WebRTC’’,[Online] Sweden 2016,https://kth.diva-portal.org/smash/get/diva2:1081457/FULLTEXT01

[6] Angie Del Rio-Chillcce, Luis Jara-Monge , Laberiano Andrade, ‘‘Analysis of the Use of Videoconferencing in the Learning Process During the Pandemic’’, [Online], Nov,05,2021,Arenas<https://thesai.org/Downloads/Volume12No5/Paper_102-Analysis_of_the_Use_of_Videoconferencing_in_the_Learning_Process.pdf>

[7] https://socket.io (accessed 24/09/21)

[8] www.w3school.com (accessed 24/09/21)

[9] https://webrtc.org (accessed 25/09/21)

[10] https://comparisons.financesonline.com/cisco-webex-vs-skype (accessed 26/09/21)

[11] Gadjo Sevilla, “Zoom vs. Microsoft Teams vs. Google Meet: A Videoconferencing Face Off”, https://in.pcmag.com/how-to-work-from-home/135851/zoom-vs-microsoft- teams-vs-google-meet-a-videoconferencing-face-off, April 16, 2020, [Online], (accessed 1/10/21)

[12] https://www.communicloud.com/blog/10-business-benefits-of-video-conferencing (accessed 5/10/21)

[13] Mr. Omkar Damle, Mr. Rishabh Verma, Mr. Yash Trivedi, “Real Time Communication with WebRTC”, https://github.com/omkar13/Video-conferencing-web-application/blob/master/project\_report.pdf, (accessed 11/10/21)

[14] Lucio Grandinetti, Seyedeh Leili Mirtaheri, Reza Shahbazian (Eds.), *High- Performance Computing and Big Data Analysis*, Second International Congress, TopHPC 2019 Tehran, Iran, April 23–25, 2019 Revised Selected Papers, Springer.

[15] Bandhakavi, A., Wiratunga, N., & Massie, S. (2018). Emotion‐aware polarity lexicons for Twitter sentiment analysis. *Expert Systems*, e12332.