VoicyApp - A Real Time Voice Chat Application

Project Report Submitted in Partial Fulfilment of the Requirements for the Degree of

Bachelor of Engineering
in
Information Technology

Submitted by

Aman Goyal : Roll No. 19UITE9001

Avinash Gautam : Roll No. 19UITE9005

Under the Mentorship of **&** ***Under the Guidance of***

Dr. Shrawan Ram	Dr. N C Barwar
Associate Professor	Professor & HOD

Department of Computer Science and Engineering
MBM University, Jodhpur
June, 2022

VoicyApp - A Real Time Voice Chat Application

Project Report Submitted in Partial Fulfilment of the Requirements for the Degree of

Bachelor of Engineering
in
Information Technology

Submitted by

Aman Goyal : Roll No. 19UITE9001

Avinash Gautam : Roll No. 19UITE9005

Under the Mentorship of **&** ***Under the Guidance of***

Dr. Shrawan Ram	Dr. N C Barwar
Associate Professor	Professor & HOD

Department of Computer Science and Engineering
MBM University, Jodhpur
June, 2022

Department of Computer Science & Engineering

M.B.M. University, Jodhpur

CERTIFICATE

This is to certify that the work contained in this report entitled **"VoicyApp - A Real Time Voice Chat Application "** is submitted by the group members Mr. Aman Goyal (Roll. No: 19UITE9001) and Mr. Avinash Gautam (Roll No: 19UITE9005) to the Department of Computer Science & Engineering, M.B.M. University, Jodhpur, for the partial fulfilment of the requirements for the degree of **Bachelor of Engineering** in **Information Technology**.

They have carried out their work under my supervision. This work has not been submitted elsewhere for the award of any other degree or diploma.

The project work in our opinion, has reached the standard fulfilling of the requirements for the degree of Bachelor of Engineering in Information Technology in accordance with the regulations of the Institute.

Dr. Shrawan Ram
Associate Professor
(Mentor)
Dept. of Computer Science & Engg.
MBM University, Jodhpur

Dr. N C Barwar
(Head & Guide)
Dept. of Computer Science & Engg.
MBM University, Jodhpur

This page was intentionally left blank.

DECLARATION

We, ***Aman Goyal and Avinash Gautam,*** hereby declare that this project titled **"VoicyApp - A Real Time Voice Chat Application "** is a record of original work done by us under the supervision and guidance of ***Dr. N C Barwar***.

I further certify that this work has not formed the basis for the award of the Degree/Diploma/Associateship/Fellowship or similar recognition to any candidate of any university and no part of this report is reproduced as it is from any other source without appropriate reference and permission.

Aman Goyal	**Avinash Gautam**
8th Semester, IT	8th Semester, IT
Roll No. - 19UITE9001	Roll No. - 19UITE9005

This page was intentionally left blank.

ACKNOWLEDGEMENT

We take the opportunity to express our gratitude to all who have provided their immense support and aspect their valuable time in guiding us. It is due to their support, Guidance, Supervision and Encouragement that We have successfully completed our Project Report on **“VoicyApp - A Real Time Voice Chat”**. We are highly indebted to our guide Professor N C Barwar Sir for his guidance and constant supervision as well as providing necessary information regarding the Project.

This page was intentionally left blank.

ABSTRACT

This project Real-time voice chat using WebRTC allows users to have voice chat using browsers without installing any other plugins or downloading native apps. It creates voice channels for users for their personal and public chats/talks with the minimal use of the internet (with lower bandwidth).

Real-time voice chat app is a web application for those who are tired of their bandwidth issues and want to work and voice chat with their respective concerned person. Here in this application we have a user login portal, user profile portal, homepage with voice channel options where multiple kinds of channel options are available.

Users need to register themselves in this application. Registration can be done using Phone number or Email id. Then users can choose the customised user name and Avatar. After successfully login, the user will see a homepage with multiple channel options. Users can select Public channel, Social Channel or private channel.
On public channels users can create a public group which will be public for all and anyone can join and have voice chat over there. On Social groups only the owners or speakers of the group can speak and everyone else will be on mute by default, it will be a podcast kind of channel. In a private channel, the owner/creator of the group can share the link of the group and that group can be joined by using that link only. Homepage also contains the search bar where you can search the group by its name if it is available in public or Social category.

The latest technology WebRTC will be used here as it is very powerful to build voice and video communication solutions and supports all modern browsers. WebRTC (Web Real-Time Communication) is a free and open-source project providing web browsers and mobile applications with real-time communication(RTC) via application programming interfaces(APIs). It allows audio and video communication to work inside web pages by allowing direct peer-to-peer communication, eliminating the need to install plugins or download native apps. The WebRTC project is supported by Apple, Google, Microsoft and Mozilla etc.

Contents

This page was intentionally left blank.

List of Figures

This page was intentionally left blank.

Chapter 1

INTRODUCTION

VoicyApp is a social-networking tool that leverages on technology advancement thereby allowing its users to communicate. It offers a wonderful one stop shop experience for keeping in touch with people you know. It can be used for placing voice messages and making voice calls.

1.1 **Motivation and Problem Statement**

The evolution of internet technologies has benefited people to access the web easily. More and more services are provided by this internet. All of this can be virtualized thanks to the technologies. Communication between people using the internet becomes part of their daily life. People used to communicate with each other using the online voice chat system to transmit their messages. Traditionally, when people need to communicate with others they will have a face to face conversation to deliver the message, same goes to the education field.

1.2 **Existing System**

The existing communication system is complex. Everybody needs to install plugins or applications. To make this complex communication job simple and allow the users to participate in live communication and save unproductive time it is to be built as a software application.

Each and every user or employee of an organization has to register, get into his inbox and check for his mail which doesn’t provide live communication resemblance to the user. This facility does not categorize the users depending on their interests. This type of communication channel fails in providing effective user friendly communication between the users. If this channel grows up to some extent then it will be harder to place some restrictions on the users. As a result, ineffective communication wastes the user's time.

1.3 Proposed System

The first step of the analysis process involves the identification of need. The success of a system depends largely on how accurately a problem is defined, thoroughly investigated and satisfying the customer needs by providing a user friendly environment.

This system has been developed in order to overcome the difficulties encountered while using voice chatting. Providing a user friendly communication channel, live communication facility, categorizing the users, logging the communication transaction, sending public & private messages, sending instant messages are motivating factors for the development of this system.

1.4 Project Scope

This project will be developed web based. The project is planned to introduce an online web chat system solution for users. The project included 3 types of communication channels i.e. public, private and social. Furthermore, a real time communication chat system will be included as the feature in the project to make a face to face communication channel between the users.

Chapter 2

Technology Used

2.1 **System requirements**

Hardware Requirements:

- Processor: Intel ® Core i3
- Installed Memory: 2.00 GB
- System Type: 64-bit Operating System
- 512 KB Cache Memory
- Hard Disk 10 GB

Software Requirements:

- Operating System: MS Windows 10
- Compiler: Visual Studio Code Editor
- Programming Languages: HTML, CSS, JavaScript, NodeJS, ExpressJS
- Database: MongoDB

2.2 Technology Used

We used following technologies to build this app:

HTML5:

- HTML stands for Hyper Text Markup Language

- HTML is the standard markup language for creating Web pages
- HTML describes the structure of a Web page
- HTML consists of a series of elements
- HTML elements tell the browser how to display the content
- HTML elements label pieces of content such as "this is a heading", "this is a paragraph", "this is a link", etc.

Features of HTML5 :

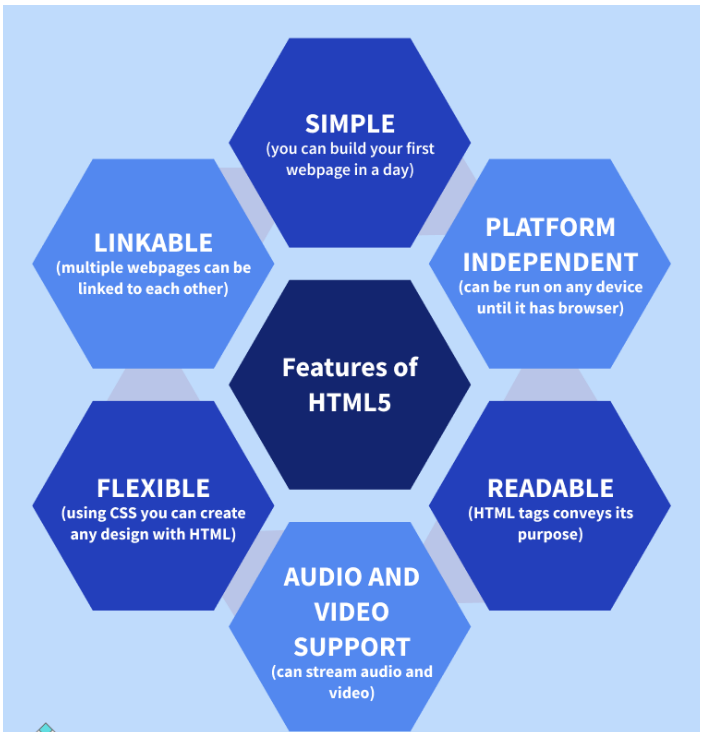

SIMPLE
(you can build your first webpage in a day)
LINKABLE
(multiple webpages can be linked to each other)
PLATFORM INDEPENDENT
(can be run on any device until it has browser)
Features of HTML5
FLEXIBLE
(using CSS you can create any design with HTML)
READABLE
(HTML tags conveys its purpose)
AUDIO AND VIDEO SUPPORT
(can stream audio and video)

Fig 2.1 Features of HTML5

<u>CSS3:</u>

- CSS stands for Cascading Style Sheets
- CSS describes how HTML elements are to be displayed on screen, paper, or in other media
- CSS saves a lot of work. It can control the layout of multiple web pages all at once

Features of CSS3:

- Advanced animations
- Multiple backgrounds and gradient
- Multiple column layouts
- Opacity
- Rounded corner
- Selectors

JavaScript:

JavaScript is a multi-paradigm, dynamic language with types and operators, standard built-in objects, and methods. Its syntax is based on the Java and C languages — many structures from those languages apply to JavaScript as well. JavaScript supports object-oriented programming with object prototypes, instead of classes.

Features of JavaScript:

- Validating User's Input
- Simple Client Side Calculations
- Greater Control
- Platform Independent
- Handling dates and time
- Generating HTML Content
- Detecting the user's browser and OS

JavaScript has become amusing as well as a really exciting space to work in recent years. JavaScript has accomplished a well-organized, sustainable code base to language that gives great productivity, readability and accessibility which is also pretty fun to work with.

NodeJS:

Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient, perfect for data-intensive real-time applications that run across distributed devices. Node.js is an open source, cross-platform runtime environment for developing server-side and networking applications. Node.js applications are written in JavaScript, and can be run within the Node.js runtime on OS X, Microsoft Windows, and Linux. Node.js also provides a rich library of various JavaScript modules which simplifies the development of web applications using Node.js to a great extent.

Features of NodeJS:

Following are some of the important features that make Node.js the first choice of software architects.

- Asynchronous and Event Driven − All APIs of Node.js library are asynchronous, that is, non-blocking. It essentially means a Node.js based server never waits for an API to return data. The server moves to the next API after calling it and a notification mechanism of Events of Node.js helps the server to get a response from the previous API call.
- Very Fast − Being built on Google Chrome's V8 JavaScript Engine, Node.js library is very fast in code execution.
- Single Threaded but Highly Scalable − Node.js uses a single threaded model with event looping. Event mechanism helps the server to respond in a non-blocking way and makes the server highly scalable as opposed to traditional servers which create limited threads to handle requests. Node.js uses a single threaded program and the same program can provide service to a much larger number of requests than traditional servers like Apache HTTP Server.
- No Buffering − Node.js applications never buffer any data. These applications simply output the data in chunks.

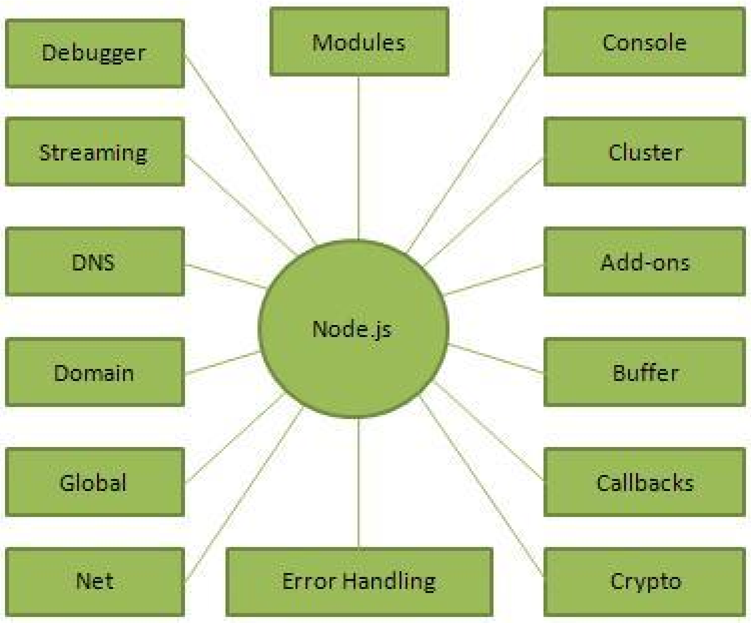


Fig 2.2 Features of NodeJS

<u>ExpressJS:</u>

Express provides a minimal interface to build our applications. It provides us with the tools that are required to build our app. It is flexible as there are numerous modules available on npm, which can be directly plugged into Express.

Express was developed by TJ Holowaychuk and is maintained by the Node.js foundation and numerous open source contributors.

Features of ExpressJS:

- Faster server side development.
- Middleware
- Routing
- Templating
- Debugging

MongoDB:

MongoDB, the most popular NoSQL database, is an open-source document-oriented database. The term ‘NoSQL’ means ‘non-relational’. It means that MongoDB isn’t based on the table-like relational database structure but provides an altogether different mechanism for storage and retrieval of data. This format of storage is called BSON (similar to JSON format).

RDBMS vs MongoDB:

- RDBMS has a typical schema design that shows the number of tables and the relationship between these tables whereas MongoDB is document-oriented. There is no concept of schema or relationship.
- Complex transactions are not supported in MongoDB because complex join operations are not available.
- MongoDB allows a highly flexible and scalable document structure. For example, one data document of a collection in MongoDB can have two fields whereas the other document in the same collection can have four.
- MongoDB is faster as compared to RDBMS due to efficient indexing and storage techniques.
- There are a few terms that are related in both databases. What’s called Table in RDBMS is called a Collection in MongoDB. Similarly, a Tuple is called a Document and A Column is called a Field. MongoDB provides a default ‘_id’ (if not provided explicitly) which is a 12-byte hexadecimal number that assures the uniqueness of every document. It is similar to the Primary key in RDBMS.

Features of MongoDB:

- **Document Oriented**: MongoDB stores the main subject in the minimal number of documents and not by breaking it up into multiple relational structures like RDBMS. For example, it stores all the information of a computer in a single document called Computer and not in distinct relational structures like CPU, RAM, Hard disk, etc.
- **Indexing**: Without indexing, a database would have to scan every document of a collection to select those that match the query which would be inefficient. So, for efficient searching Indexing is a must and MongoDB uses it to process huge volumes of data in very less time.
- **Scalability**: MongoDB scales horizontally using sharding (partitioning data across various servers). Data is partitioned into data chunks using the shard key, and these data chunks are evenly distributed across shards that reside across many physical servers. Also, new machines can be added to a running database.
- **Replication and High Availability**: MongoDB increases the data availability with multiple copies of data on different servers. By providing redundancy, it protects the database from hardware failures. If one server goes down, the data can be retrieved easily from other active servers which also had the data stored on them.
- **Aggregation**: Aggregation operations process data records and return the computed results. It is similar to the GROUPBY clause in SQL. A few aggregation expressions are sum, avg, min, max, etc.

WebRTC:

WebRTC (Web Real-Time Communications) is an open source project that enables real-time voice, text and video communications capabilities between web browsers and devices. WebRTC provides software developers with application programming interfaces (APIs) written in JavaScript.

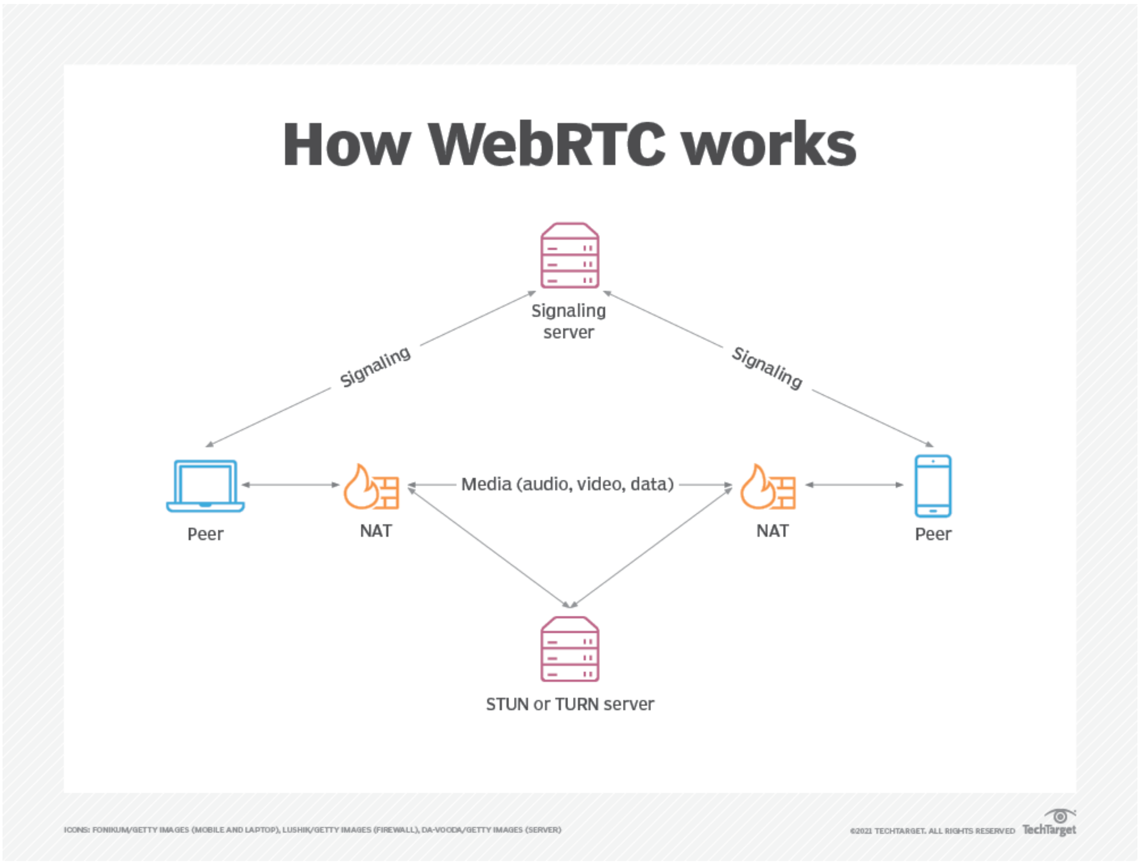
Developers use these APIs to create peer-to-peer (P2P) communications between internet web browsers and mobile applications without worrying about compatibility and support for audio-, video- or text-based content.
With WebRTC, data transfer occurs in real time without the need for custom interfaces, extra plugins or special software for browser integration. WebRTC enables real-time audio and video communication simply by opening a webpage.

How does WebRTC work?

WebRTC uses JavaScript, APIs and Hypertext Markup Language to embed communications technologies within web browsers. It is designed to make audio, video and data communication between browsers user-friendly and easy to implement. WebRTC works with most major web browsers.
WebRTC APIs perform several key functions, including accessing and recording video-, audio- and text-based data from devices to initiating, monitoring and ending P2P connections between devices via browsers and facilitating bidirectional data transfer over multiple data channels.

In most cases, WebRTC connects users by transferring real-time audio, video and data from device to device using P2P communications. In situations where users are on different Internet Protocol (IP) networks that have Network Address Translation (NAT) firewalls that prevent RTC, WebRTC can be used in conjunction with Session Traversal Utilities for NAT (STUN) servers. This enables a given IP address to be translated into a public internet address so peer connections can be established.

But there are also networks that are so restrictive that even a STUN server cannot be used to translate IP addresses. In these cases, WebRTC is used with a Traversal Using Relays around NAT (TURN) server, which relays traffic between users, enabling them to connect. The Interactive Connectivity Establishment protocol is used to find the best connection.


How WebRTC works
Signaling server
Signaling
Signaling
Peer
NAT
Media (audio, video, data)
NAT
Peer
STUN or TURN server
ICONS: FONIKUM/GETTY IMAGES (MOBILE AND LAPTOP), LUSHIK/GETTY IMAGES (FIREWALL), DA-VOODA/GETTY IMAGES (SERVER)
©2021 TECHTARGET. ALL RIGHTS RESERVED TechTarget

What is WebRTC used for?

The goal of WebRTC is to facilitate real-time P2P communications over the internet. There are several use cases for WebRTC, including the following:

- WebRTC is used for video chats and meetings on video calling platforms, such as Zoom, Microsoft Teams, Slack or Google Meet.
- Industries, including healthcare, surveillance and monitoring, and internet of things, use WebRTC. For example, WebRTC use in Telehealth enables doctors to conduct virtual office visits with a patient over a web browser.
- In the field of home and business security and surveillance, WebRTC is used as a connecting agent between browsers and security cameras.
- WebRTC is heavily used for real-time media.
- WebRTC provides the underlying connection between instructors and students for online education.

What are the pros and cons of WebRTC?

WebRTC presents opportunities and challenges to organizations.

The advantages of WebRTC include the following:

- eliminates much of the in-house manual integration work required of IT;
- can adjust communication quality, bandwidth and traffic flow whenever network conditions change;
- is supported by most major web browsers, including Google Chrome for desktop and Android, Mozilla Firefox for desktop and Android, and Safari;
- works on any operating system as long as the browser supports WebRTC;
- does not require third-party components or plugins; and
- is free as open source software.

Disadvantages of WebRTC include the following:

- Each user must establish a P2P browser connection, making bandwidth an issue.
- Maintenance costs can be high because WebRTC requires powerful servers.
- Security and privacy standards are still unclear, leaving it up to IT departments to ensure that corporate security and privacy standards can be met.
- There are no definitive quality of service standards, which means that quality of video or audio over the internet may be inconsistent.

Is WebRTC secure?

Every WebRTC software component is encrypted, and every WebRTC API requires secure origins via Hypertext Transfer Protocol Secure (HTTPS) or localhost.

Nevertheless, there are still open security questions that developers using WebRTC must consider. Signaling processing methods, or the methods used to exchange metadata, are not specified for WebRTC signaling. This means that developers must decide which security protocols to use and ensure that the protocols they select can be maintained with WebRTC.

This page was intentionally left blank.

Chapter 3

Work Done

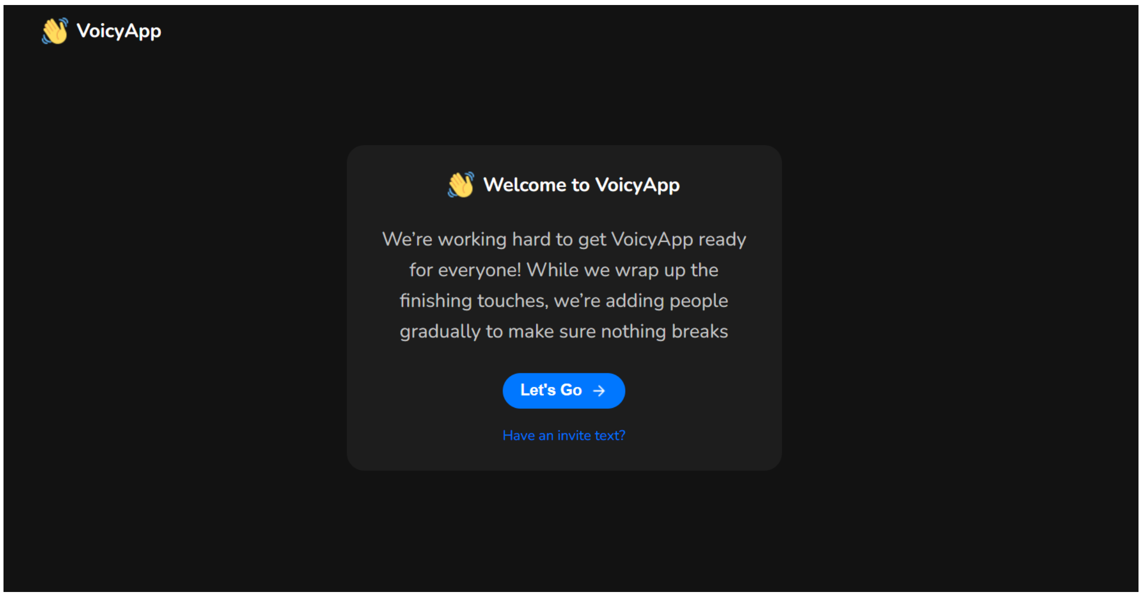
3.1 Source Code

Frontend : Here is some example code of frontend:

Code for Homepage:

```
import React from 'react';
import styles from './Home.module.css';
import { Link, useHistory } from 'react-router-dom';
import Card from '../../components/shared/Card/Card';
import Button from '../../components/shared/Button/Button';
const Home = () => {
  const signInLinkStyle = {
    color: '#0077ff',
    fontWeight: 'bold',
    textDecoration: 'none',
    marginLeft: '10px',
  };
  const history = useHistory();
  function startRegister() {
    history.push('/authenticate');
    }
    return (
      <div className={styles.cardWrapper}>
        <Card title="Welcome to VoicyApp" icon="logo">
          <p className={styles.text}>
            We’re working hard to get VoicyApp ready for everyone!
                While we wrap up the finishing touches, we’re adding
people
            gradually to make sure nothing breaks
          </p>
          <div>
            <Button onClick={startRegister} text="Let's Go" />
          </div>
          <div className={styles.signinWrapper}>
            <span className={styles.hasInvite}>
              Have an invite text?
            </span>
          </div>
        </Card>
      </div>
    );
  };

  export default Home;
```

and this is how it will look like:



Welcome to VoicyApp

We're working hard to get VoicyApp ready for everyone! While we wrap up the finishing touches, we're adding people gradually to make sure nothing breaks

Fig 3.1 : Homepage

Backend: Here is example code of backend

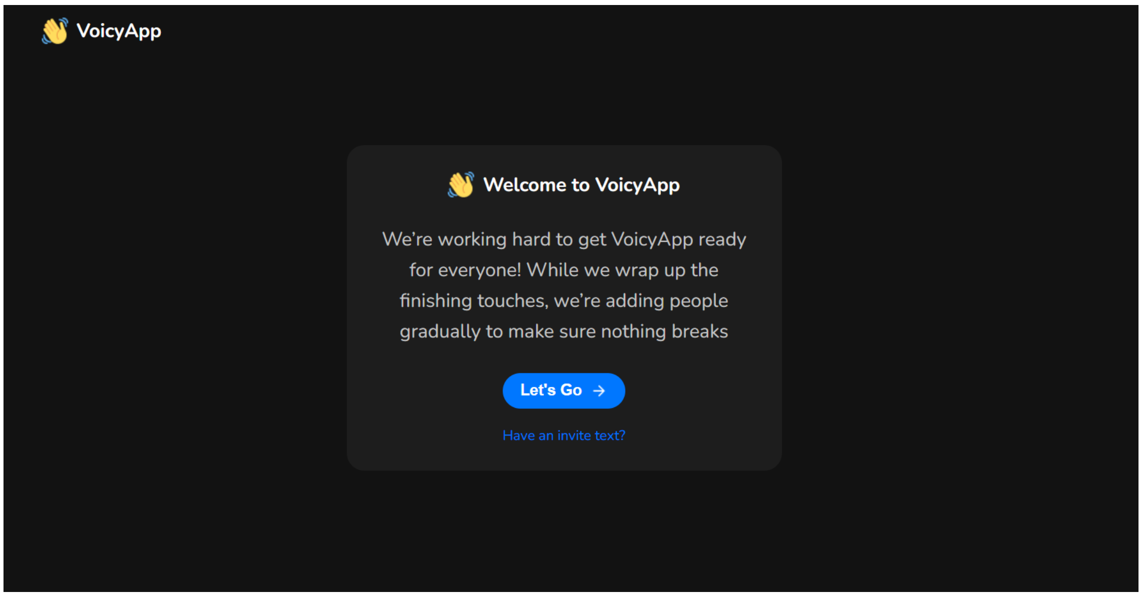
Code for otp-services:

```
const crypto = require('crypto');

const hashService = require('./hash-service');


const smsSid = process.env.SMS_SID;

const smsAuthToken = process.env.SMS_AUTH_TOKEN;

const twilio = require('twilio')(smsSid, smsAuthToken, {

   lazyLoading: true,

});
```

```
class OtpService {
  async generateOtp() {
    const otp = crypto.randomInt(1000, 9999);
    return otp;
  }

  async sendBySms(phone, otp) {
    return await twilio.messages.create({
      to: phone,
      from: process.env.SMS_FROM_NUMBER,
      body: `Your VoicyApp OTP is ${otp}`,
    });
  }

  verifyOtp(hashedOtp, data) {
    let computedHash = hashService.hashOtp(data);
    return computedHash === hashedOtp;
  }
}

module.exports = new OtpService();
```

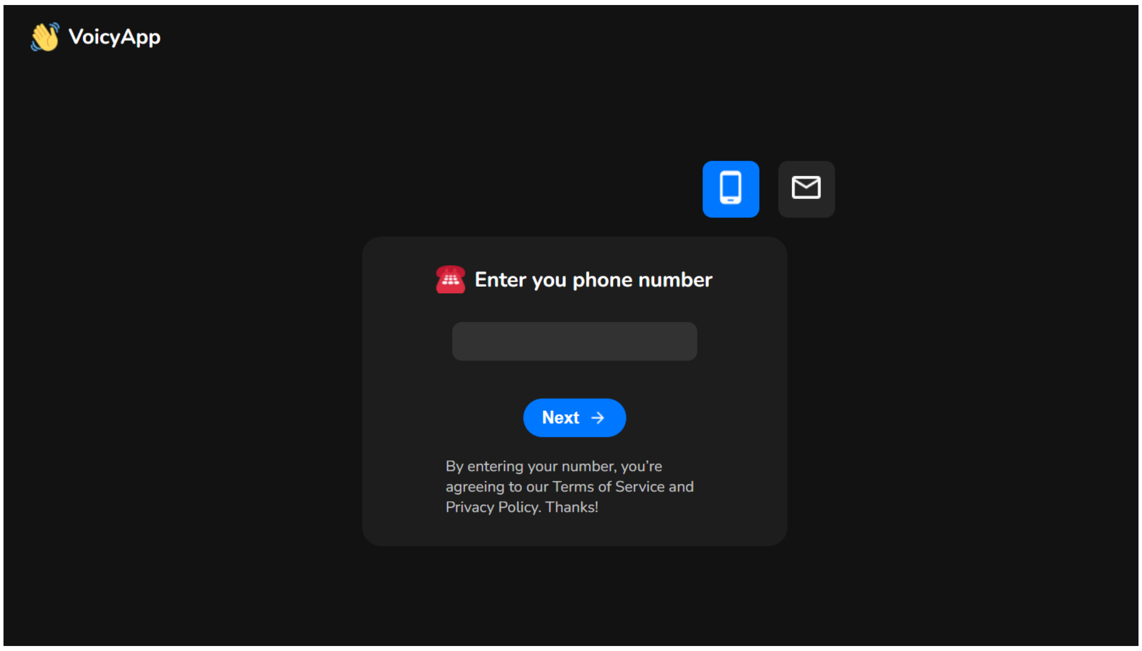
Chapter 4
Results

Here are some Screenshots from the resulting UI .



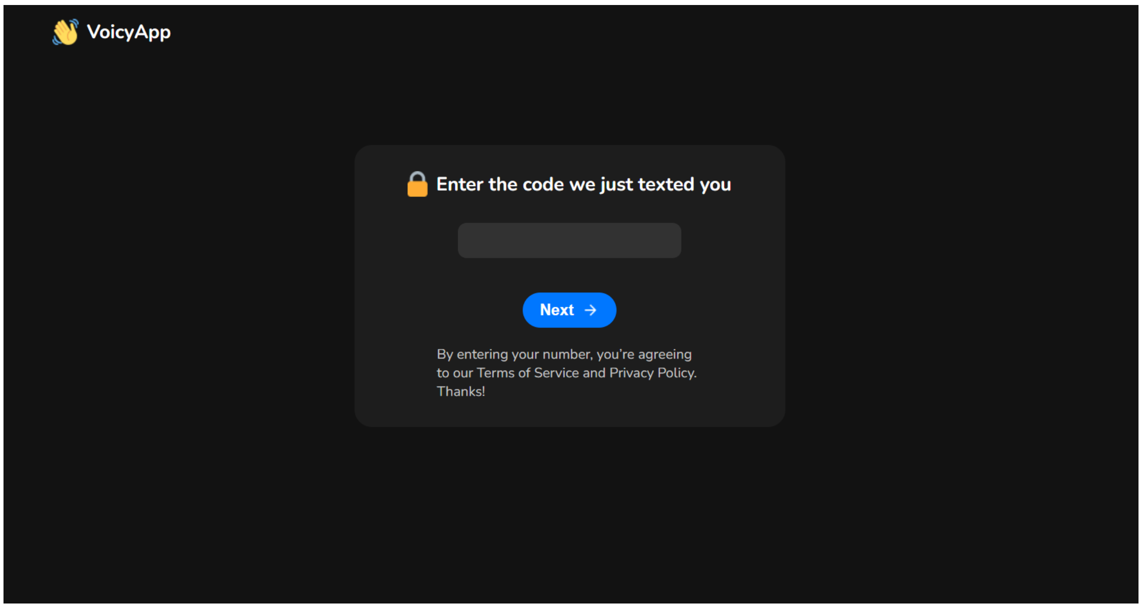
We're working hard to get VoicyApp ready for everyone! While we wrap up the finishing touches, we're adding people gradually to make sure nothing breaks

FIg 4.1 Welcome Page



By entering your number, you're agreeing to our Terms of Service and Privacy Policy. Thanks!

Fig 4.2 : Phone Number for OTP



By entering your number, you're agreeing to our Terms of Service and Privacy Policy. Thanks!

Fig 4.3 : OTP Page

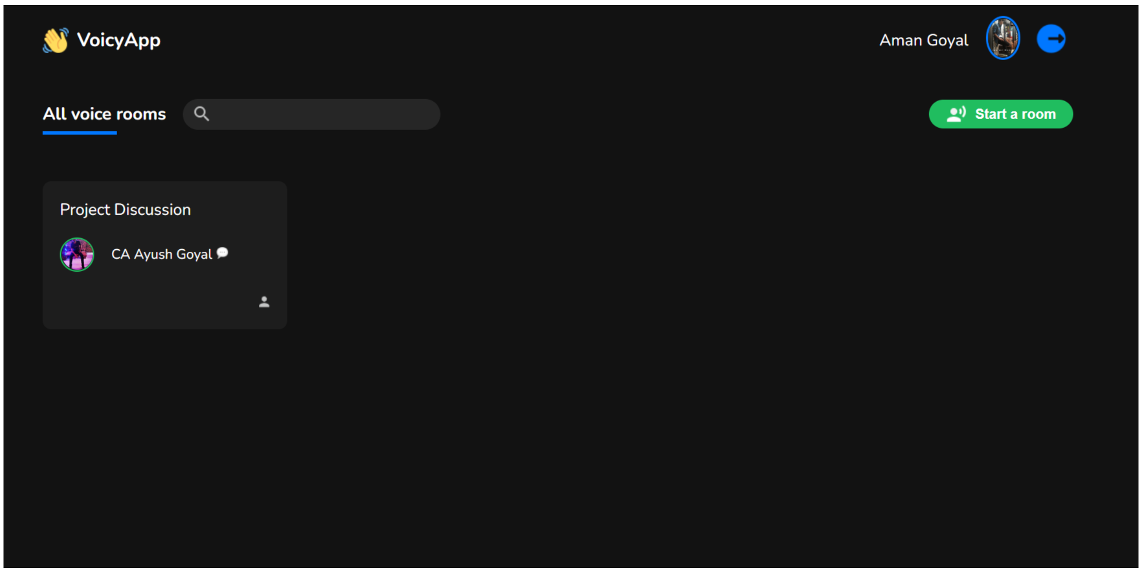


Fig 4.4 : Home Page After login

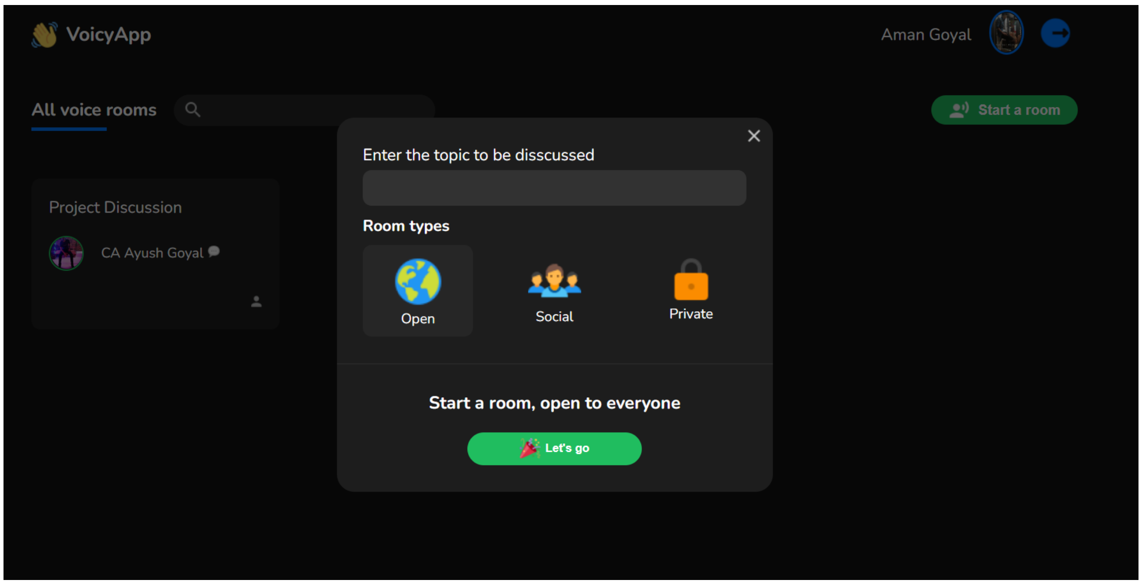


Fig 4.5 : Start New Room

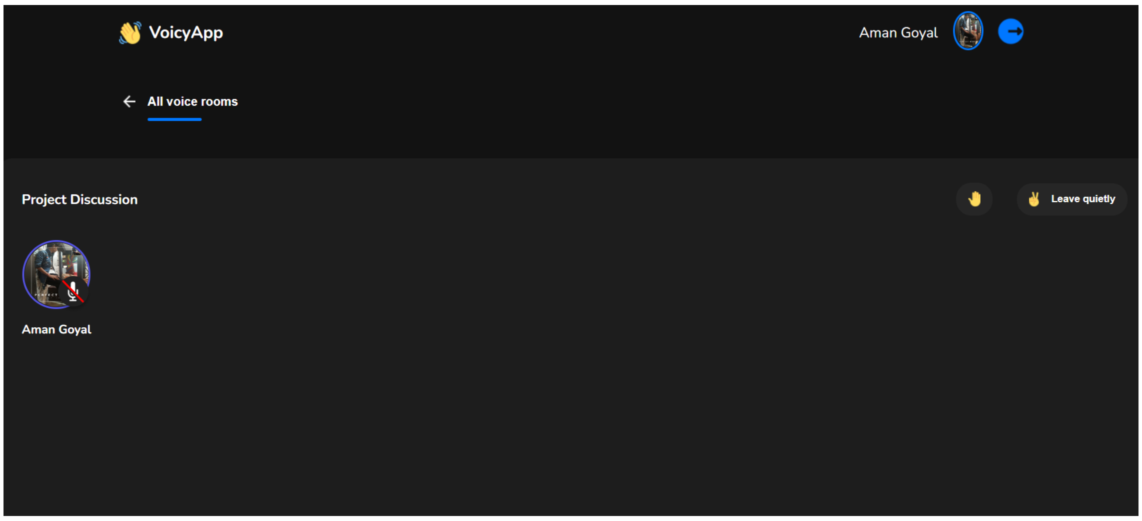


Fig 4.6 : Inside Room

Chapter 5

Conclusion & Future Work

An audio-conferencing system using WebRTC technology was created . In this project we had used WebRTC technology for its ease of use and did not need any plugins or applications to install, it just needed a browser of any mobile multimedia devices like android phone or personal computers .

The audio conferencing system is designed as web based which will be able to be used on various kinds of operating systems. The aim of this project is to reduce the effort and difficulty of mobility to communicate and to create an audio conference and mentioned goals are achieved yet.

As an improvement , we can improve it by modifying it for a video conferencing web application where we can do video and voice chat both. Moreover, we can record the call/meeting with a dummy face and provide it with minimum bandwidth.

This page was intentionally left blank.

References

[1] HTML Study: https://www.w3schools.com/html/

[2] CSS Study: https://www.w3schools.com/css/

[3] JavaScript Study: https://www.w3schools.com/js/

[4] REACT Study: https://reactjs.org/tutorial/tutorial.html

[5] NodeJS Study: https://nodejs.dev/learn

[6] ExpressJS Study: https://devdocs.io/express/

[7] MongoDB Study: https://www.mongodb.com/docs/

[8] WebRTC Overview: https://webrtc.org/getting-started/overview