|  |  |
| --- | --- |
| SOFTWARE REQUIREMENT SPECIFICATION |  |

**SOFTWARE REQUIREMENT SPECIFICATION**

Version No.:

Date: 29-August-2016

Project Name: Web Talk

Project Code:

This document contains proprietary information of NEC Technologies India Pvt. Ltd**.** Unauthorized access, copying and replication are prohibited. This document must not be copied in whole or part by any means, without the written authorization of NEC Technologies India Pvt. Ltd, Noida, India.

Table of Contents

[**1 Introduction** 7](#_Toc460584281)

[**1.1 Purpose** 7](#_Toc460584282)

[**1.2 Scope** 7](#_Toc460584283)

[2 **Overall System Description** 8](#_Toc460584284)

[**2.1 Product Perspective** 8](#_Toc460584285)

[2.1.1 Deployment perspective 9](#_Toc460584286)

[**2.2 User Characteristics** 11](#_Toc460584287)

[**2.3 Product Functions / Operational concepts and Scenarios** 11](#_Toc460584288)

[2.3.1 Operating Modes 12](#_Toc460584289)

[2.3.2 Operations 12](#_Toc460584290)

[2.3.3 Boundary Handling 12](#_Toc460584291)

[**2.4 Standards, Protocol and Conformance Requirements** 12](#_Toc460584292)

[**2.5 Constraints** 14](#_Toc460584293)

[2.5.1 Memory (& other system) Constraints 14](#_Toc460584294)

[2.5.2 Constraint Types 14](#_Toc460584295)

[**2.6 Technical Assumptions and Dependencies** 14](#_Toc460584298)

[**3** **Requirements Specifications** 15](#_Toc460584299)

[**3.1 Functional Requirements / System Features Specifications** 15](#_Toc460584300)

3.1.1 Stimulus/Response Sequence……………………………………………………………........14

[3.1.2 Functional Requirement 16](#_Toc460584301)

3.1.3 Use Cases of Features/Functionalities………………………………………………………..15

3.1.3.1 Features/Functionalities of Use Cases………………………………………………15

3.1.3.1.1 Use Case Diagram…………………………………………………….........16

3.1.3.1.2. Use Case………………………………………………………………........16

[**3.2 External Interface Requirements Specification**](#_Toc460584302) 17

[3.2.1 User Interface Specifications 17](#_Toc460584303)

3.2.1.1. GUI Specification……………………………………………………………………….17

[3.2.2 Hardware Interface Specifications 17](#_Toc460584305)

[3.2.3 Software Interface Specifications 18](#_Toc460584306)

[3.2.4 Communication Interface Specifications 18](#_Toc460584307)

[**3.3 Non Functional Requirements** 18](#_Toc460584309)

3.3.1. Performance Requirements…………………………………………………………………...18

[3.3.2 Exception Handling 18](#_Toc460584310)

[3.3.3 Logging / Debugging requirements 18](#_Toc460584311)

[3.3.4 Software System Attributes 19](#_Toc460584312)

[**3.4 Testing Requirements**](#_Toc460584313) 20

[**4 Requirements Handling** 21](#_Toc460584315)

[4.1 **Apportioning of Requirements** 21](#_Toc460584316)

[**4.2 Requirement Validation Methodology** 21](#_Toc460584317)

5 References …………………………………………………………………………………………….22

**Definitions, Acronyms, and Abbreviations:**

|  |  |
| --- | --- |
| ***Web Application*** | The component that runs on end-user’s web Browsers to serve request from internet clients. |
|  |  |
| ***Channel*** | Channels are created by a broadcaster which contains only Their own audio stream. Users can join a channel to view that broadcaster’s audio stream, in which they become a peer for that stream, in which they become a peer for that Specific channel. |
| ***End-user*** | A person who uses the web application, to broadcast audio Streams |
| ***Google Chrome*** | A web browser developed by Google, the  Primary target of our web application. |
| ***ICE Framework*** | ICE is a framework used to connect users. First tries UDP, then TCP with HTTP, then TCP with HTTPS, then TURN servers. |
| ***ICE Candidate*** | ICE candidate provides the information about the IPad dress and port from where the data is going to be exchanged. |
| ***NAT*** | A network protocol used in IPv4 networks that allows multiple devices to connect to a public network using the same public IPv4 address. |
| ***Peer-to-peer*** | A method of communication, where data |
|  | Is transmitted between end-users instead of centralized servers. |
| ***SDP*** | Session Description Protocol (SDP) is a set of rules that describe media communication sessions. |
| ***Signaling*** | A process to exchange data and Coordinate communication between two end users. |
| ***SRS*** | Software Requirements Specification |
| ***Stream*** | A sequence of data. |
| ***STUN*** | Session Traversal Utilities for NAT (STUN) is a protocol that uses a third-party STUN server to allow peers to discover each other’s IP Address and helps to identify each users and find a good connection between them. |
| ***TURN*** | Traversal Using Relays around NAT (TURN) is a protocol. |
| ***UDP*** | The User Datagram Protocol (**UDP**) is a transport layer protocol defined for use with the IP network layer protocol. |
| ***Web Browser*** | A software application for retrieving, presenting and traversing information resources Over |
| ***WebRTC*** | A WebRTC is a technology based on Real-Time Communications (RTC) that provides peer to peer connection without using any plugin and third party software. |



# **1 Introduction**

## **1.1 Purpose**

The purpose of this document is to present a detailed description of the peer-to-peer audio and text broadcast which specify how to deal with real time communication (voice, data) from within a browser by using simple APIs.

## **1.2 Scope**

This Document contains requirements for Web Talk which is a web based Real time Audio and Text Communication Application using WebRTC technology. It also contains working principle, requirements for Web Talk Application and use case analysis of the application

This application is a browser-based method of broadcasting media using peer-to-peer technologies. .Its purpose is to allow users to stream audio, text, files easily, quickly, and free of cost

# 2 **Overall System Description**

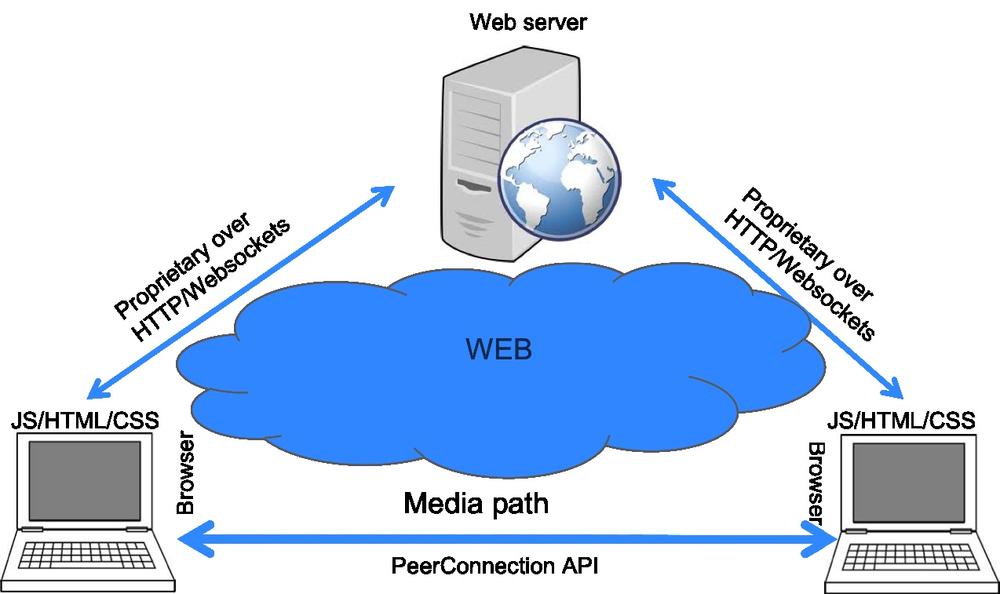
## **2.1 Product Perspective**

There are currently no easy or decentralized way to do audio, text or video broadcasting. Existing Solution require operating system centric program or to install browser plugins. Moreover the existing technology require the use of centralized server resources. So broadcasting of any multimedia whether that is audio, video or text is neither cheap or nor convenient for users to decide a live stream or broadcast. Popular approaches to live streaming are Skype, twitch.tv etc. but all of these need the installation of the application in both the client side. Twitch.tv is centralized and requires the third party program to stream.

WebRTC enables any Web server to deliver a unique real-time communications experience, with simplicity and reliability, without dependence on service providers or other services. Moreover it enables users to participate in a communications experience as delivered by any website without downloads, registration or general cost. It also delivers simplicity, and through that reliability and availability in two ways.

First, by putting the elements of communications into the browser in an open standard it eliminates the complexity of developing separate soft clients for each device.

Second, The WebRTC client is stateless and uses stimulus input through the graphic side of the browser to the server to initiate state change.

**

### 2.1.1 Deployment perspective

#### 2.1.1.1 Hardware Requirements

|  |  |  |
| --- | --- | --- |
| **Hardware** | **Specifications** | **Description** |
| RAM | <2 GB RAM will be required with at least 200 MB free space.> | Normal RAM in which OS and other software can be installed. |
| Processor | <1 gigahertz (GHz) or faster 32-bit (x86) or 64-bit (x64) processor will be required to run the application.> | Any processor which is capable of running web applications and basic software. |
| Hard Disk | <150 MB free space to install the application> | Normal Hard Disk to store the OS programs. |
| Microphone |  |  |

#### 2.1.1.2 Software Requirements

|  |  |  |
| --- | --- | --- |
| **Software interfaces** | **Specification** | **Description** |
| Operating System | Linux(UBUNTU or DEBIAN) | OS on which application will be installed. |
| Tools | Chromium Depot Tools. | It is a package of scripts that is used to manage checkouts and code reviews. |

#### 2.1.1.3 Communication Interfaces (Standards & Protocols)

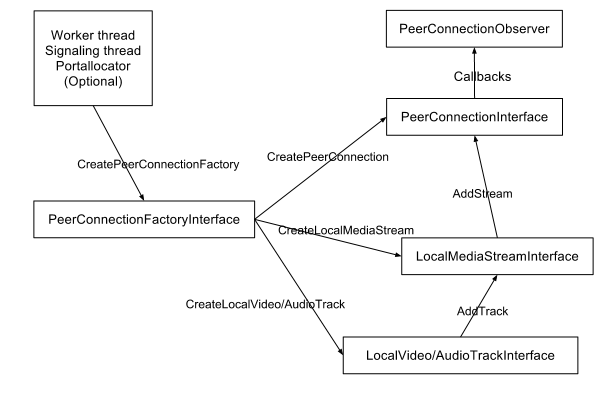
Webrtc have mainly 3 parts.

* Media stream
* RTC peer connections
* RTC data channel

Media Stream () API is used to receive data from a user's microphone. Media Stream () API abstracts the microphone complexities from the user.

Communication interfaces will be abstracted by the Webrtc API mainly by the RTCPeerConnection () which is created by RTCPeerConnectionFactory () interface. This allows peer to peer communication between two users with a high level approach without worrying about the lower level implementation. With RTCPeerConnection () users exchange SDP (Session Protocol Description) packets which contain many of the network and user information i.e. ip address, port number, transport layer protocol used etc. RTCPeerConnection () also contain method for the encryption of the data to be sent and it also maintain the signaling mechanism.

Webrtc also focuses on the ICE framework for connecting two peers directly. ICE framework first try to connect the peers using UDP protocol with the most minimum latency. If UDP fails the ICE framework try to connect the peers with TCP with HTTP protocol and later with TCP with HTTPS. Finally when a direct connection does not work ICE will try to use TURN server.



So the list of interfaces are:

* [**RTCPeerConnection**](https://developer.mozilla.org/en-US/docs/Web/API/RTCPeerConnection)**:** Represents a WebRTC connection between the local computer and a remote peer.
* [**RTCSessionDescription**](https://developer.mozilla.org/en-US/docs/Web/API/RTCSessionDescription): Represents the parameters of a session. Each RTCSessionDescription consists of a description [type](https://developer.mozilla.org/en-US/docs/Web/API/RTCSessionDescription/type) indicating which part of the offer/answer negotiation process it describes and of the [SDP](https://developer.mozilla.org/en-US/docs/Glossary/SDP) descriptor of the session.
* [**RTCIceCandidate**](https://developer.mozilla.org/en-US/docs/Web/API/RTCIceCandidate): Represents a candidate internet connectivity establishment (ICE) server for establishing an  [RTCPeerConnection](https://developer.mozilla.org/en-US/docs/Web/API/RTCPeerConnection).
* [**RTCPeerConnectionIceEvent**](https://developer.mozilla.org/en-US/docs/Web/API/RTCPeerConnectionIceEvent): Represents events that occurs in relation to ICE candidates with the target, usually an [RTCPeerConnection](https://developer.mozilla.org/en-US/docs/Web/API/RTCPeerConnection).
* [**RTCRtpReceiver**](https://developer.mozilla.org/en-US/docs/Web/API/RTCRtpReceiver): Manages the reception and decoding of data through a [MediaStreamTrack](https://developer.mozilla.org/en-US/docs/Web/API/MediaStreamTrack) for a [RTCPeerConnection](https://developer.mozilla.org/en-US/docs/Web/API/RTCPeerConnection).
* [**RTCRtpSender**](https://developer.mozilla.org/en-US/docs/Web/API/RTCRtpSender): Manages the encoding and transmission of data through a [MediaStreamTrack](https://developer.mozilla.org/en-US/docs/Web/API/MediaStreamTrack) for a [RTCPeerConnection](https://developer.mozilla.org/en-US/docs/Web/API/RTCPeerConnection).
* [**RTCDataChannel**](https://developer.mozilla.org/en-US/docs/Web/API/RTCDataChannel): Represents a bi-directional data channel between two peers of a connection.
* [**RTCDataChannelEvent**](https://developer.mozilla.org/en-US/docs/Web/API/RTCDataChannelEvent): Represents events that occur while attaching a [RTCDataChannel](https://developer.mozilla.org/en-US/docs/Web/API/RTCDataChannel) to a [RTCPeerConnection](https://developer.mozilla.org/en-US/docs/Web/API/RTCPeerConnection). The only event sent with this interface is [data channel](https://developer.mozilla.org/en-US/docs/Web/Events/datachannel).

## **2.2 User Characteristics**

* **Academic Purpose*:***

i> Professors who want to live broadcast their lectures to students.

ii>Student also rebroadcast to other student perhaps a fellow classmate fell ill and unable to attend.

iii>Scholars can use this to stream academia

a>Online tutorials

b>TED talks

* **News Purpose:**

i>Civilians can use this to stream a live news

a>Riots

b>Car accident

c>Political debates

* **Others:**

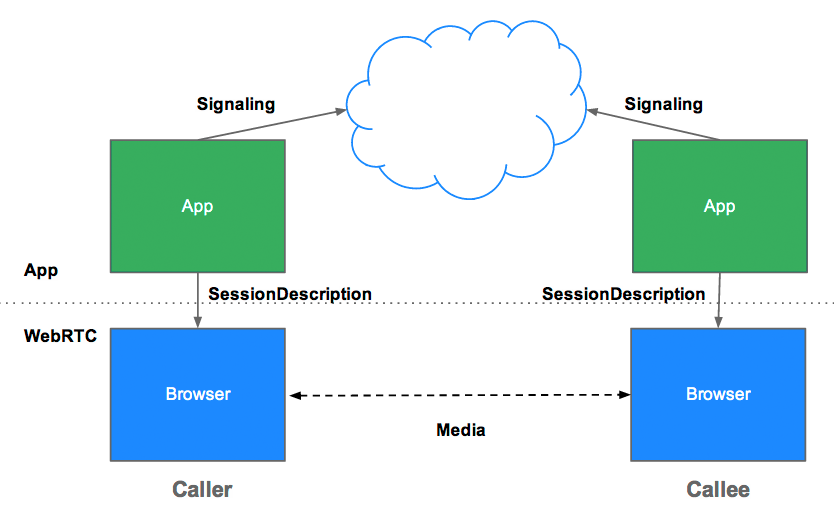
Broadcasters can use this to share various entertainment content

a>Audio Concerts

b>Live events

## **2.3 Product Functions / Operational concepts and Scenarios**

* A broadcaster can go to a website and stream audio to other users via a broadcaster created channel.
* Two users can talk with each other in through real time communication via browser and no need to install plugins.
* A user friendly GUI to call or chat.
* User can at a time call other user and also able to send text to that user using the chat box.



### 2.3.1 Operating Modes

There are two operating modes available in this requirement namely

* Real time chat
* Audio call

### 2.3.2 Operations

* **Real time chat:**

In this mode the registered user is applicable to send or receive instant messaging. There is a login page after which the user gets a random generated number and a URL corresponding to that generated number. On clicking the URL other user which is active on that time will be connected. After connection is established the data is transferred with the help of RTC- Data channel.

* **Audio call:**

In this mode users are allowed to communicate with each other using TCP UDP. In this application two users on separate devices to communicate using WebRTC audio streams. Our application will have two pages. One for login and the other for making an audio call to another user. Here we use SDP handshaking.

### 2.3.3 Boundary Handling

* **Browser Incompatibility:** Webrtc application is applicable only for those browser which supports webrtc. So if the browser does not support webrtc then the program will not run and nothing will be displayed on the browser.
* **Hardware Not Found:** If the microphone is not plug in then user will get a message that “audio device is not found”.
* **Connection Problem:** If the system is not connected with internet then the user will not able to open the web browser to run the application.
* **Poor Connection:** If suddenly the connection become slow then it will try to reconnect the peer to peer connection for a particular time period.

## **2.4 Standards, Protocol and Conformance Requirements**

Following are the standards and protocols that are used in webrtc:

* **i>UDP/TCP:** UDP is the less reliable transport layer protocol. In UDP the following things are not sure:

i> the order of the frames that are received

ii>the delivery status of the frame that is whether it is received or notified

iii>the state of every single data packet.

UDP is used in webrtc as I real time scenarios packet loss is not the main concerned.

On the other hand TCP is reliable transport layer protocol as it is a connection oriented protocol. It always guarantees the following:

i>any data send marked as received

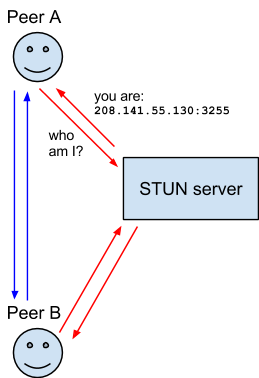
ii>any packet that are lost will be resend and sending of other data will be temporarily terminated

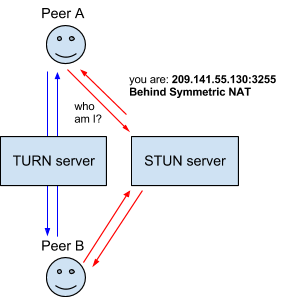
iii>every packet is unique (no duplicate)

* **ii>Session Description Protocol (SDP):** SDP is an important part of webrtc. It is a protocol that describe media communication sessions. It peers of various audio and video codecs, network topologies, and other device information.
* **iii>Stream Control Transmission Protocol (SCTP):** The SCTP protocol is used today to send blob data on top of our currently setup peer connection when using the RTCDataChannel object. SCTP protocol is built on the top of the DTLS that is Datagram Transport layer security protocol which is built on the top of the user datagram protocol.
* **iv>Interactive Connectivity Establishment (ICE):** ICE is a framework to allow the web browser to connect with peers. Sometimes a straight up connection between two pairs simply won’t work. In those cases ICE bypass firewalls that would prevent opening connections, give a unique address if like most situations device doesn’t have a public IP address, and relay data through a server if the router doesn’t allow to directly connect with peer. ICE uses STUN or TURN to achieve this.
* **v>TURN/STUN:** In order to find a clear route to another user STUN (Session traversal utilities for NAT) or TURN (Traversal using relay around NAT) technology is used. They are generally used to know the user about its public IP address from the private IP of that user.

In STUN standard, first of all the user makes a request to a server, enabled with the STUN protocol. Then the server sends back the IP address of the client.

Sometimes there is a firewall not allowing any STUN-based traffic to the other user in that case TURN is used.





## **2.5 Constraints**

The main constraints occurs between system and user will come across includes incompatibility of WebRTC with their preferred browser.

* The latest Chrome version the browser supports.
* Bandwidth reduction/limitation on peers with limited internet capabilities.
* A reconnection system as a safety net in the case of a sudden broadcaster disconnection.
* TCP based vs. UDP based on
* Reliability vs. Speed and minimizing bandwidth usage.

### 2.5.1 Memory (& other system) Constraints

This project is basically a web based project so there is no need of a lot of memory space. Basically the memory space is needed only for the installation of the Linux operating system and for the other software like git, chrome depot tools etc.

### 2.5.2 Constraint Types

### Browser Constraint: Peer to peer communication through webrtc is browser based application which may not be supported in different web browser like internet explorer, safari etc.

### Hardware Constraint: In our project microphone is the only external hardware that are needed to get the audio from the user.

### Bandwidth Constraint: For establishment of the peer to peer connection a standard bandwidth is needed

## **2.6 Technical Assumptions and Dependencies**

* User should have a stable modern internet connection.
* User should have the latest version of Google Chrome, Mozilla Firefox and Opera that supported the Webrtc.
* User computer should have enough power to rebroadcast.
* Broadcaster should have either an internal or external microphone.
* The service is used preferably on a desktop or laptop.

# **Requirements Specifications**

## **3.1 Functional Requirements / System Features Specifications**

1) **Internet:** WebRTC uses Internet for communication to the clients through web browsers.

**2) Browsers:** Web Browser is an application which acts as an interface for WEBRTC.

**3) Microphone.** It is an external Hardware used for Audio Communication.

### Stimulus/Response Sequences

Both the sender and the receiving side will contain this four basic functionalities. So these four parts will reside in a single application in both the sender and receiver side.

**1) Call Initialization**

Sender initialize the call by using navigator.usermedia () method and sends the output to RTC peer Connection method.

**2) Offer creation**

Then a method RTCPeerConnection.CreateOffer() is called which create a SDP offer and send this offer via signaling.

**3) Offer received && Answer Creation:**

Receiver receives the offer from the Sender and use navigator.getUSerMedia method to create a stream and adds this stream to RTCPeer Connection.

RTC Session Description () object is created using the SDP offer of the sender and set up as the Remote Description by calling RTCPeerConnection.SetRemoteDescription () Method and use create Answer () Method to create an answer and send to the server.

**4) Answer Received:**

Answer is received by the sender and once that happens call setRemoteDescription() Method to set the Response as the Remote end of the connection.

### 3.1.2 Functional Requirements

1) **Operating System**: Linux, Windows.

2) **Browser**: Google Chrome, Mozilla Firefox.

3**) Hardware**: Microphone.

### 3.1.3 Use Cases of Features/Functionalities

#### 3.1.3.1 Features/Functionalities of Use Case

1 **Sender**: Sender sign in with the help of login page of the application.

2 **Receiver**: Receiver also sign in on the login page and ready to communicate with the sender.

3. **Browser**. Browser is used by the sender and receiver for communication.

4 **Homepage**: Sender will be redirected to this page which contains option for audio and text chat respectively.

5 **Homepage1**: Sender will be redirected to this page on choosing the appropriate audio or text chat option.

6**. Homepage2**: This is the page opened on the receiver browser by entering the login name of sender and connection is established.

7 Login ID: Login ID is required by both sender and receiver which is provided at login time that is exchanged for the communication.

##### 3.1.3.1.1 Use Case Diagram

##### 3.1.3.1.2 Use Case

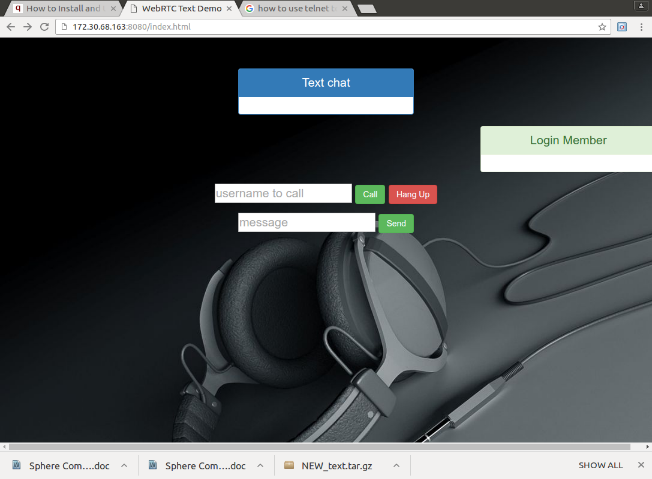
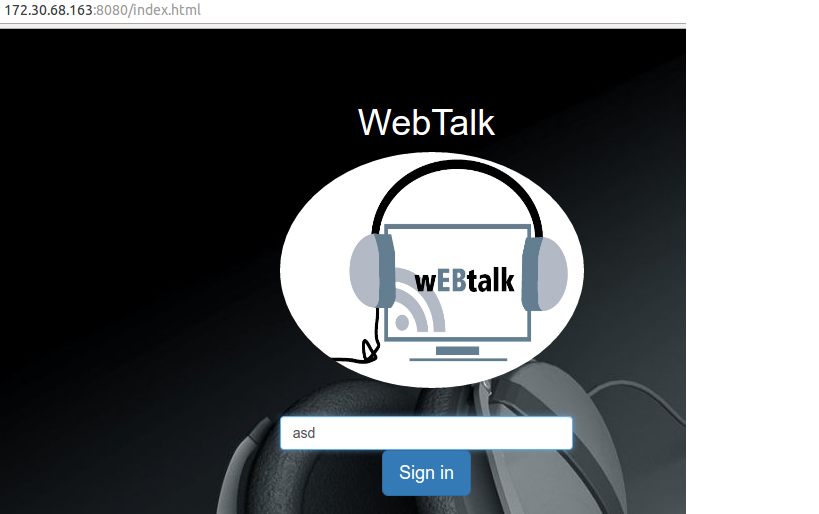
|  |  |  |
| --- | --- | --- |
| **Use Case Name** | WEBRTC |  |
| **Use Case ID** | UC\_XX\_001 |  |
| **Overview** | Establish a two way real time communication on the web. |  |
|  |  |  |
|  |  |  |
| **Actors/ Use Case Diagram** | C:\Users\rehan.raza\Desktop\uml.png  Actor Name: Sender , Receiver |  |
| **Preconditions** | 1. Web Browser should be installed, Microphone should be plugged in. |  |
| **Outcome** | Receiver will receive a call. |  |
| **Exception(s)** | Browser Incompatible, Hard Ware Not found. |  |
| **Notes** | - |  |
| |  | | --- | | Steps | | **Sender:**   * Open the application in the browser. * Click on sign in button which will open a new window which contain the call, chat button. * Select the audio or text button for communicating with the receiver. * Pass the login id l to the intended recipient to establish the connection. | | **Receiver:**   * Uses the login ID received from sender to establish connection. * Click on the join button to establish the two way communication. | | | |

## **3.2 External Interface Requirements Specification**

### 3.2.1 User Interface Specifications

**3.2.1.1 GUI Specification**

The user interface will be designed which consist of real time chatting and audio communication options. It consist of a web page which generates a unique number and generates a URL by concatenating the current URL and the unique number which is used by the recipient for connection. It also consist of separate chatting and audio button which is used for communication.



### 3.2.2 Hardware Interface Specifications

We will be using the WebRTC API, specifically Media Stream (), to receive data from a user's microphone. Media Stream () abstracts the microphone complexities from the user. Hardware interfaces include desktops, laptops, which will be abstracted by the user's browser and operating system.

### 3.2.3 Software Interface Specifications

WebRTC API

### 3.2.4 Communication Interface Specifications

1. **GetUserMedia:** It allows a web browser to access the camera and microphone and to capture media.
2. **RTCPeerConnection:** It sets up audio/video calls.
3. **RTCDataChannel**: it allow browsers to share data via peer-to-peer.

## **3.3 Non Functional Requirements**

#### 3.3.1.1 Performance Requirements:

1. Browser should be compatible between sender and recipient for better performance.

2. Internet speed should be good for better communication.

3. Hardware used should be properly working for continuous communication.

### 3.3.2 Exception Handling

1. Hardware Issue Microphone is not used by any user.

2 Browser Issues User using Browser not supported by webRTC.

3. Handling the situation when the receiver does not receive or disconnect

The call.

4. Time Out Exception when the receiver does not receive the call within the

Set time frame.

### 3.3.3 Logging / Debugging requirements

1. Working with a WebRTC API.

2. Debugging with Browser.

3 Debugging with Networking.

### 3.3.4 Software System Attributes

#### 3.3.4.1 Reliability

The server should make sure a peer is always having a connection to another peer in the network.

#### 3.3.4.2 Availability

1. Any user may access the service to broadcast or view broadcasts via the website.
2. Server-side software will not have any unusual requirements allowing for deployment in many environments.

#### 3.3.4.3 Security

1. Security can be achieved by the encryption followed by decryption between sender and receiver.
2. Communications between peers is encrypted with DTLS-SRTP as defined in WebRTC.

#### 3.3.4.4 Maintainability

1. The host shall be competent in the administration of common web application software and techniques
2. The host must make sure that web application is accessible to end-users
3. The end-user will not need to download or install any software to make use of this application.

#### 3.3.4.5 Portability

The service should be able to run on any latest version of Google Chrome browser with no additional plugins or software on a desktop or laptop.

#### 3.3.4.6 Install ability

There is no requirement of installing any software or plugins to use this

Application.

## **3.4 Testing Requirements**

Unit testing is used throughout the development of this application in which each and every modules is tested at their respective stages.

# **4 Requirements Handling**

## 4.1 **Apportioning of Requirements**

* **Connectivity:**

For the deployment of this project the user must have an internet connection to interact with other client. For data transfer there must be a good connectivity is required.

* **Operating Environment:**

Windows 7, 8 and 10(32 or 64 bit), Mac OS X 10.8+ and Linux

* **Browser(s)**

Google Chrome v32+, Mozilla Firefox and Opera.

* **WebRTC API**

Media Stream API (getUserMedia)

RTCPeerConnection API

RTCDataChannel API

ICE Framework

* **Languages and Tools:**

HTML5, Node js, java script.

* **Hardware:**

Microphone for communication between the users

## **4.2 Requirement Validation Methodology**

Through our project user can send audio and chat to different user via peer to peer real time connection which is made by using webrtc. This requirement is validated by the following steps:

* Requirement per-reviews
* Requirement reviews
* Testing based requirement validation
* Viewpoint-oriented requirement validation

**5. References**

* <http://www.tutorialspoint.com/webrtc/webrtc_browser_support.htm>
* <https://webrtc.org/native-code/development/prerequisite-sw/>
* <https://webrtc.org/>
* <https://en.wikipedia.org/wiki/WebRTC>