DESARROLLO DE UN WEBPHONE CON TECNOLOGÍA WEBRTC PARA UNA EMPRESA DE SERVICIOS VOIP

DAVINSON ANDRES CAÑAVERAL GRAJALES

INSTITUCIÓN UNIVERSITARIA PASCUAL BRAVO
FACULTAD DE INGENIERÍA
INGENIERIA DE SOFTWARE
MEDELLÍN
2023

DESARROLLO DE UN WEBPHONE CON TECNOLOGÍA WEBRTC PARA UNA EMPRESA DE SERVICIOS VOIP

DAVINSON ANDRES CAÑAVERAL GRAJALES

Trabajo de grado para optar al título de Ingeniería de Software

Asesor

Carlos Alberto Monsalve Jaramillo

Especialista en gerencia y mantenimiento

Paola Andrea Buitrago Cadavid

Magister en Modelación y Ciencia Computacional

INSTITUCIÓN UNIVERSITARIA PASCUAL BRAVO

FACULTAD DE INGENIERÍA

INGENIERÍA DE SOFTWARE

MEDELLÍN

2023

Dedicatoria

Este trabajo de grado se dedica con profundo agradecimiento. A Dios, por ser mi guía y fortaleza en cada paso de este arduo camino académico. A mis padres, por su amor incondicional, apoyo inquebrantable y sacrificio constante, sin los cuales no hubiera sido posible llegar hasta aquí. A mi amada tierra, por ser mi fuente de inspiración y por brindarme las raíces que me han dado identidad y propósito. A la alcaldía de Medellín, por su generosidad y visión al ofrecer oportunidades de estudio gratuitas, permitiéndome acceder a una educación de calidad. Esta tesis es el resultado de sus bendiciones y contribuciones, y es un testimonio de mi compromiso de hacer un impacto positivo en mi comunidad y en el mundo.

Agradecimientos

Quiero expresar mi profundo agradecimiento a la Institución Universitaria Pascual Bravo por brindarme la oportunidad de realizar este proyecto de grado y por su invaluable contribución a mi formación académica.

Agradezco a las autoridades de la institución por su visión y compromiso con la educación de calidad. Su liderazgo ha permitido que estudiantes como yo puedan acceder a una formación integral y especializada, impulsando nuestro desarrollo personal y profesional.

Agradezco también a mis profesores y docentes, quienes con su experiencia y dedicación han guiado mi aprendizaje y me han brindado las herramientas necesarias para abordar esta investigación. Su compromiso con la excelencia académica ha sido una inspiración constante.

No puedo dejar de mencionar al personal administrativo y de apoyo de la institución. Su labor incansable ha facilitado mi trayectoria académica, desde la inscripción hasta la gestión de recursos y servicios. Su amabilidad y disposición han hecho de mi experiencia en la institución una experiencia enriquecedora.

Agradezco también a mis compañeros de clase, con quienes he compartido momentos de aprendizaje, reflexión y colaboración. Su apoyo mutuo y espíritu de camaradería han sido fundamentales en mi desarrollo académico.

A la Institución Universitaria Pascual Bravo y a todos sus miembros, mi más sincero agradecimiento por ser una parte fundamental de mi formación académica. Su compromiso con la educación y el desarrollo de sus estudiantes ha dejado una huella indeleble en mi trayectoria, y espero poder honrar su legado a través de mi contribución a la sociedad.

Contenido

Lista de figuras

Lista de tablas

Resumen

DESARROLLO DE UN WEBPHONE CON TECNOLOGÍA WEBRTC PARA UNA EMPRESA DE SERVICIOS VOIP

DAVINSON ANDRES CAÑAVERAL GRAJALES

En el marco de este proyecto, se abordó el problema de la falta de un sistema de comunicación eficiente y moderno en una empresa proveedora de servicios de voz basados en VoIP en la Ciudad de Medellín. Para solucionar esta problemática, se llevó a cabo una intervención consistente en el desarrollo e implementación de un WebPhone utilizando la tecnología WebRTC.

La intervención se centró en alcanzar los siguientes objetivos: comprender los conceptos y tecnologías clave de WebRTC y su integración con el protocolo SIP, implementar el WebPhone utilizando tecnologías modernas como React JS y la librería JsSip, y realizar pruebas exhaustivas para evaluar el funcionamiento y la calidad del WebPhone desarrollado.

La implementación del WebPhone con tecnología WebRTC ha brindado numerosas ventajas a la empresa proveedora de servicios de voz. En primer lugar, se ha logrado establecer comunicación de SIP a WebRTC y viceversa, permitiendo una integración fluida con las tecnologías y sistemas existentes. Además, el WebPhone desarrollado ofrece a los usuarios la capacidad de realizar y recibir llamadas de voz en tiempo real, mejorando significativamente la experiencia de comunicación.

Los resultados obtenidos han sido satisfactorios. El WebPhone ha demostrado ser efectivo y funcional, cumpliendo con los requisitos y objetivos establecidos. Las pruebas de funcionamiento y calidad realizadas han arrojado resultados positivos, validando la efectividad del uso de la tecnología WebRTC en la empresa proveedora de servicios de voz.

Palabras claves: comunicación, implementación, tecnología, WebPhone, WebRTC.

Abstract

DEVELOPMENT OF A WEBPHONE WITH WEBRTC TECHNOLOGY FOR A VOIP SERVICES COMPANY

DAVINSON ANDRES CAÑAVERAL GRAJALES

Within the framework of this project, the problem of the lack of an efficient and modern communication system in a provider of voice services based on VoIP in the City of Medellín was addressed. To solve this problem, an intervention consisting of the development and implementation of a WebPhone using WebRTC technology was carried out.

The intervention focused on achieving the following objectives: understanding the key concepts and technologies of WebRTC and its integration with the SIP protocol, implementing the WebPhone using modern technologies such as React JS and the JsSip library, and conducting extensive tests to assess the functionality and reliability of the WebRTC. quality of the developed WebPhone.

The implementation of the WebPhone with WebRTC technology has brought numerous benefits to the voice service provider company. First of all, it has been possible to establish communication from SIP to WebRTC and vice versa, allowing a smooth integration with existing technologies and systems. In addition, the developed WebPhone offers users the ability to make and receive voice calls in real time, significantly improving the communication experience.

The results obtained have been satisfactory. The WebPhone has proven to be effective and functional, meeting the established requirements and objectives. The performance and quality tests carried out have yielded positive results, validating the effectiveness of the use of WebRTC technology in the voice service provider company.

Keywords: communication, implementation, technology, WebPhone, WebRTC

Glosario

Gateway: un Gateway es un dispositivo o software que actúa como intermediario entre dos redes o sistemas diferentes. Permite la transferencia de datos y la comunicación entre estos sistemas, traduciendo los protocolos y formatos de datos necesarios para asegurar una operabilidad adecuada.

NAT trasversal: es el proceso de superar las limitaciones de las redes que utilizan Network Address Translation (NAT). Consiste en establecer conexiones entre dispositivos ubicados detrás de diferentes routers NAT, permitiendo la comunicación directa entre ellos.

Protocolo de transporte: un protocolo de transporte define las reglas y formatos utilizados para transmitir datos entre dos dispositivos en una red. Establece la forma en que se envían, reciben y confirman los datos, garantizando la entrega correcta y segura de la información.

PSTN: la PSTN (Public Switched Telephone Network) se refiere a la red de telefonía convencional basada en conmutación de circuitos. Es la infraestructura tradicional que permite realizar llamadas telefónicas a través de líneas telefónicas físicas.

SIP (Session Initiation Protocol): es un protocolo de señalización utilizado para establecer, modificar y finalizar sesiones de comunicación en tiempo real, como llamadas de voz y video a través de IP. Es ampliamente utilizado en sistemas de comunicaciones basados en VoIP.

SDP (Session Description Protocol): es un protocolo utilizado para describir y negociar parámetros de sesión en una comunicación en tiempo real, como la configuración de codecs, resolución de video, entre otros.

WebRTC (Web Real-Time Communication): es un conjunto de tecnologías y estándares web que permiten la comunicación en tiempo real directamente a través de navegadores web, sin necesidad de plugins o software adicional. Facilita la transmisión de audio, video y datos entre los usuarios de la web.

Websockets: es un protocolo de comunicación bidireccional que permite una conexión persistente y en tiempo real entre un servidor y un cliente web. es útil para aplicaciones que requieren una comunicación continua y actualizaciones en tiempo real.

Introducción

El tema del trabajo se centra en el desarrollo de un WebPhone utilizando la tecnología WebRTC, para realizar y recibir llamadas en una empresa proveedora de servicios de voz con tecnología basada en VoIP de la ciudad de Medellín.

El objetivo principal de este proyecto es presentar una solución innovadora y moderna para mejorar la comunicación empresarial utilizando tecnología de última generación. Para lograr este propósito, se han planteado objetivos específicos que incluyen analizar la tecnología WebRTC y los protocolos utilizados en la aplicación, comprender la señalización de red y desarrollar el WebPhone con tecnologías modernas.

El método empleado en este proyecto se basa en una investigación rigurosa y un enfoque práctico. Se han utilizado diversas herramientas y tecnologías para alcanzar los objetivos planteados.

El alcance del trabajo incluye la construcción de un WebPhone que permita realizar y recibir llamadas de voz en tiempo real, utilizando la tecnología WebRTC y la librería JsSip. Además, se pretende desplegar la aplicación en el servidor de la empresa para su uso cotidiano.

En conclusión, este proyecto de grado presenta una solución innovadora para mejorar la comunicación empresarial utilizando tecnología de última generación. A través de la investigación y el desarrollo de un WebPhone utilizando la tecnología WebRTC, se espera ofrecer una herramienta moderna y efectiva para el sector empresarial.

1. Planteamiento del problema

1.1 Descripción

La comunicación es el proceso de intercambio de información entre dos o más personas o sistemas. Este proceso implica la transmisión de un mensaje por parte de un emisor a un receptor a través de un medio de comunicación, con el objetivo de establecer una comprensión compartida entre las partes involucradas. Por eso hay conjunto de reglas y especificaciones técnicas que definen como los dispositivos y sistemas de comunicación intercambian información y como se comunican entre sí, conocido como estándares de comunicación, el objetivo de este establecer una comprensión compartida entre las partes involucradas y lograr que el mensaje sea entendido o respondido de manera adecuada.

La tecnología es el conjunto de todas estas herramientas y procesos que el sistema utiliza para operar y mantener el servicio, se utiliza hoy en día en diferentes aspectos de la vida desde la comunicación y la industria hasta la medicina y comunicación. Los Protocolos son esenciales para garantizar la compatibilidad entre los diferentes dispositivos y sistemas de comunicación, ya que posee una serie de reglas y normas que establecen como los dispositivos intercambian información y se comunican entre sí.

La arquitectura del sistema de telefonía por internet influye en la importancia de que todos los componentes de un sistema se integren y funcionen correctamente, como los servidores, los clientes y dispositivos de los usuarios y como se comunican entre sí. Esto es fundamental para brindar la calidad y confiablidad de un sistema de telefonía por internet aplica a la tecnología WebRTC.

El Desarrollo de un WebPhone con tecnología WebRTC para una empresa de servicios basada en tecnología VoIP se refiere a la necesidad de desarrollar un sistema de telefonía por Internet (VoIP) que permita a los usuarios realizar y recibir llamadas telefónicas a través de un navegador web. Este sistema debe estar basado en la tecnología WebRTC, que permite la transmisión de datos de audio y video en tiempo real en la web, sin la necesidad de descargar o

instalar software adicional. Según el autor David D. Clark, "Los protocolos y estándares para la comunicación en red son como lenguajes que permiten que los diferentes dispositivos se comuniquen de manera efectiva, independientemente de su origen o función específica" (Clark). En este caso, la tecnología WebRTC proporciona un estándar de comunicación para la transmisión de datos de audio y video en tiempo real a través de la web, lo que permite una comunicación más eficiente y efectiva entre los usuarios.

La empresa es un proveedor de servicios de telefonía por internet, basada en el protocolo VoIP. Esta requiere una solución de telefonía web que ofrezca una experiencia de usuario sencilla y eficiente, y que permita a sus clientes realizar llamadas desde cualquier lugar y dispositivo con acceso a Internet. Además, es importante que el sistema cumpla con los estándares de calidad y seguridad exigidos por la industria VoIP.

El desarrollo de un WebPhone con tecnología WebRTC para la empresa proveedora de servicios de voz con tecnología basada en VoIP es un desafío técnico que requiere una combinación de habilidades en programación web, ingeniería de voz sobre IP y seguridad de la información. Como menciona Tanenbaum y Wetherall en su libro "Redes de Computadoras", para implementar soluciones de VoIP es necesario contar con conocimientos en programación de aplicaciones web, protocolos de transporte de voz, como el protocolo RTP, y protocolos de señalización de sesión, como el protocolo SIP. Además, se necesitan habilidades en seguridad de la información para garantizar la confidencialidad y la integridad de las comunicaciones. (A. S & D, 2012)

Sin embargo, una vez implementado, este sistema permitirá a la empresa ofrecer un servicio de telefonía de alta calidad a sus clientes y mejorar su posición en el mercado de servicios VoIP.

La tecnología WebRTC se encuentra en constante evolución y desarrollo, y es considerada una de las tecnologías más importantes para la comunicación en tiempo real en la web. La implementación de un WebPhone con esta tecnología permitirá a la empresa proveedora de servicios de voz con tecnología basada en VoIP ofrecer un servicio de alta calidad a sus clientes y mantenerse a la vanguardia en cuanto a tecnología de comunicación se refiere. La

popularidad de la comunicación por Internet ha aumentado en los últimos años, y se espera que continúe creciendo en el futuro. Según un estudio realizado por el Centro de Investigación y Desarrollo de Tecnologías de la Información, el número de usuarios de servicios VoIP se espera que alcance los 3.8 mil millones en 2025. (CloudTalk, 2023). Este aumento en la demanda de servicios de comunicación por Internet justifica la relevancia práctica de este proyecto.

1.2 Formulación

¿Cómo desarrollar un WebPhone con tecnología WebRTC para la empresa proveedora de servicios de voz con tecnología basada en VoIP que cumpla con los requisitos de calidad y seguridad en la comunicación en tiempo real y sea atractivo para los usuarios?

2. Justificación

La tecnología WebRTC ha revolucionado la forma en que las personas se comunican a través de Internet, permitiendo realizar llamadas y videoconferencias en tiempo real directamente desde un navegador web sin la necesidad de descargar y configurar software adicional. La implementación exitosa de la tecnología WebRTC tendría un gran impacto social y tecnológico. En lo social se estima una mayor accesibilidad y democratización de la comunicación, la tecnología WebRTC puede permitir que cualquier persona con una conexión a Internet tenga acceso a la comunicación en tiempo real sin tener que pagar por costosas soluciones de telecomunicaciones. Esto puede ser especialmente importante para comunidades marginadas o personas en países en desarrollo que no tienen acceso a tecnologías avanzadas de comunicación. Mayor seguridad y privacidad, la tecnología WebRTC utiliza encriptación y otras medidas de seguridad para proteger la privacidad de los usuarios y las comunicaciones. En lo tecnológico, el desarrollo de nuevas aplicaciones y servicios. La tecnología WebRTC puede permitir el desarrollo de nuevas aplicaciones y servicios en línea que aprovechen las capacidades de la comunicación en tiempo real. Por ejemplo, se pueden desarrollar aplicaciones de juegos y de colaboración en línea. Desafío para los proveedores de telefonía existentes. WebRTC puede presentar un desafío para los proveedores de telefonía existentes al permitir que los usuarios realicen llamadas y videoconferencias sin tener que pagar por costosos planes de telefonía. En lo Ambiental. La reducción de la huella de carbono. WebRTC puede permitir una mayor colaboración en línea y teletrabajo, lo que puede reducir la necesidad de viajar y tener oficinas físicas. Esto puede tener un impacto positivo en la huella de carbono de las empresas, también tiene un impacto negativo en lo ambiental, Aumento del consumo de energía: la comunicación en tiempo real requiere una gran cantidad de energía. Si WebRTC se convierte en la principal forma de comunicación, podría aumentar el consumo de energía en los centros de datos y en los dispositivos que se utilizan para la comunicación. Esto podría tener un impacto negativo en el medio ambiente.

La importancia del desarrollo de un WebPhone con tecnología WebRTC se basa en las oportunidades que brinda a la empresa para ofrecer un servicio de calidad y mejorar la experiencia del usuario. En primer lugar, el uso de esta tecnología permite una comunicación de alta calidad y con una latencia mínima, lo que se traduce en una experiencia de usuario

satisfactoria. Además, la flexibilidad que ofrece el WebPhone permite realizar llamadas desde cualquier lugar y en cualquier momento, lo que resulta esencial en la comunicación de negocios.

La adopción de un WebPhone también permite a la empresa mejorar su oferta de servicios, aumentar la satisfacción de los clientes y mejorar su competitividad en el mercado. Los clientes pueden acceder a los servicios de la empresa de manera más fácil y cómoda, lo que se traduce en una mayor retención de clientes y en la posibilidad de atraer a nuevos usuarios.

3. Objetivos

3.1 Objetivo general

Desarrollar un WebPhone para realizar y recibir llamadas utilizando la tecnología WebRTC. En una empresa proveedora de servicios de voz con tecnología basada en VoIP de la Ciudad de Medellín.

3.2 Objetivos específicos

Comprender los conceptos y tecnologías clave de la tecnología WebRTC y su integración con el protocolo SIP, con el fin de establecer comunicación de SIP a WebRTC y viceversa.

Implementar el WebPhone con tecnología WebRTC, utilizando tecnologías modernas como React JS y la librería JsSip, para que los usuarios puedan realizar y recibir llamadas de voz en tiempo real.

Aplicar pruebas de funcionamiento y calidad del WebPhone desarrollado, para determinar la efectividad del uso de la tecnología WebRTC en la empresa proveedora de servicios de voz.

4. Marco teórico

WebRTC es el segundo protocolo de video más popular después del patentado por Zoom. WebRTC supera todos los demás protocolos estándar (H.323 y SIP) y patentados (Microsoft Teams y Cisco Webex). La tecnología WebRTC ha tenido un impacto profundo en el mercado de las videoconferencias. Tras el lanzamiento de los primeros navegadores con WebRTC en 2013, el número potencial de puntos finales de videoconferencia aumentó en mil millones de dispositivos en todo el mundo. De hecho, cada navegador es ahora un punto final de videoconferencia con capacidades básicas. (TrueConf, 2023)

WebRTC es una tecnología de comunicación en tiempo real basada en estándares web que permite la transmisión de audio, video y datos entre navegadores web sin la necesidad de plugins o aplicaciones externas. Fue desarrollada por Google y posteriormente fue adoptada por la World Wide Web Consortium (W3C) como un estándar web. (3CX, s.f.)

¿Por qué usar WebRTC?

WebRTC ofrece una solución para la comunicación en tiempo real que es fácil de usar y de implementar, lo que la hace atractiva para diversas aplicaciones, incluyendo videoconferencias, juegos multijugador, colaboración en línea, atención médica a distancia, entre otras.

4.1 ¿Cómo funciona WebRTC?

WebRTC utiliza tecnologías web existentes, como HTML, CSS y JavaScript, junto con protocolos como Real-Time Protocol (RTP) y Real-Time Control Protocol (RTCP) para permitir la comunicación en tiempo real entre navegadores web. También utiliza el protocolo Secure Real-time Transport Protocol (SRTP) para cifrar los datos transmitidos.

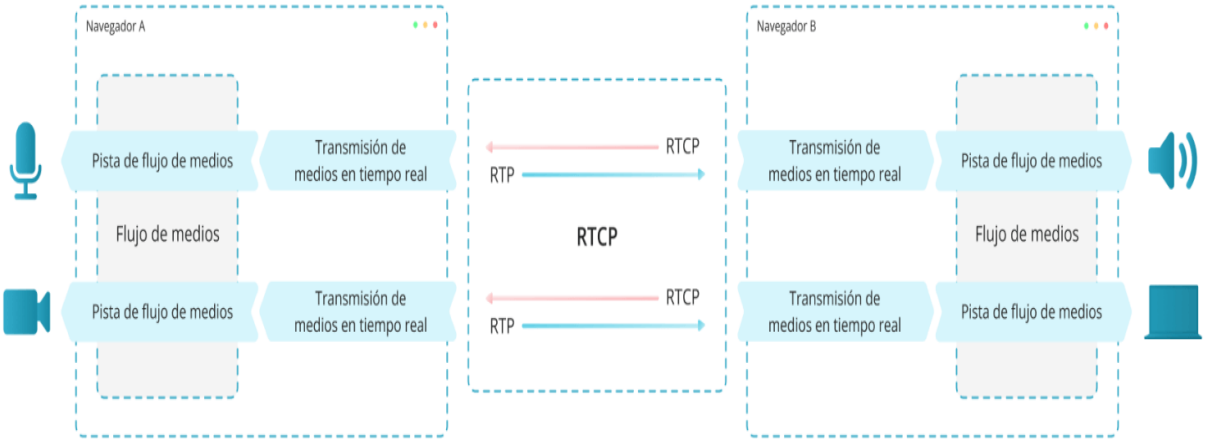

Figura 1. Funcionamiento WebRTC entre Navegadores
Fuente: extraído de https://trueconf.com/
Nota. La figura representa la funcionalidad técnica del WebRTC entre 2 navegadores

4.1.1 ¿Cuándo usar WebRTC? WebRTC se puede utilizar en cualquier situación en la que se requiera la comunicación en tiempo real entre navegadores web. Algunos ejemplos incluyen:

Videoconferencias: WebRTC se utiliza comúnmente para la realización de videoconferencias entre usuarios en diferentes ubicaciones geográficas.

Juegos multijugador: WebRTC permite la creación de juegos multijugador en tiempo real en los que los jugadores pueden interactuar entre sí.

Colaboración en línea: WebRTC se utiliza para permitir la colaboración en tiempo real entre usuarios que trabajan en documentos o proyectos en línea.

Atención médica a distancia: WebRTC se utiliza para permitir la atención médica a distancia en la que los pacientes pueden interactuar con los profesionales de la salud a través de videoconferencias. (TrueConf, 2023)

4.2 Estudios realizados en materia de esta tecnología

Desde su lanzamiento en 2011, WebRTC ha sido objeto de numerosos estudios y análisis en la industria tecnológica. Los estudios se han centrado en áreas como la seguridad, la interoperabilidad, la escalabilidad y el rendimiento. Además, se han desarrollado herramientas y bibliotecas para facilitar la implementación de WebRTC en aplicaciones web.

4.2.1 WebRTC: APIs and RTCWEB Protocols of the HTML5 Real-Time Web: Este libro proporciona una descripción detallada de la tecnología WebRTC, su arquitectura, APIs y protocolos. También cubre los desafíos de seguridad y privacidad asociados con WebRTC. (B. Johnston & C. Burnett, 2012)

4.2.2 WebRTC Security: Este artículo revisa los riesgos de seguridad asociados con WebRTC, incluyendo vulnerabilidades de enrutamiento, ataques de inundación, ataques de intermediario, ataques de secuestro de sesión y violaciones de privacidad. También describe las soluciones de seguridad y las mejores prácticas que pueden ayudar a mitigar estos riesgos. (Rescorla, 2021)

4.2.3 WebRTC: A Hands-On Guide: Este libro proporciona una guía práctica para desarrollar aplicaciones WebRTC. Cubre temas como la configuración del entorno de desarrollo, la creación de una aplicación de chat de video básica y la integración de WebRTC con otras tecnologías web. (Sergiienko, 2015)

4.2.4 WebRTC Media Quality Benchmark: Este documento describe una metodología para medir la calidad de audio y video en las aplicaciones WebRTC. También proporciona una comparación de la calidad de audio y video en diferentes navegadores web y plataformas. (Google, 2021)

4.3 Casos exitosos nacionales e internacionales

WebRTC se ha utilizado en diversas aplicaciones exitosas en todo el mundo. Algunos ejemplos incluyen:

Facebook Messenger: Facebook utiliza WebRTC para la función de videollamada en su aplicación de mensajería instantánea.

Google Meet: Google Meet utiliza WebRTC para la realización de videoconferencias en línea.

Spotify: Spotify utiliza WebRTC para su función de colaboración en línea, permitiendo a los usuarios escuchar música juntos en tiempo real.

Tpaga: es una plataforma de pagos móviles en Colombia que utiliza WebRTC para realizar verificaciones de identidad en línea. La compañía utiliza una cámara web y un micrófono para capturar imágenes y sonidos en tiempo real del usuario durante el proceso de verificación de identidad. La tecnología WebRTC permite una comunicación segura y en tiempo real para garantizar la autenticidad del usuario.

Banco Davivienda: El Banco Davivienda en Colombia utiliza WebRTC en su plataforma de banca en línea para permitir a los usuarios comunicarse con los representantes de servicio al cliente en tiempo real. Los usuarios pueden realizar videoconferencias con los representantes de servicio al cliente para resolver sus dudas y consultas.

Telefónica: La compañía de telecomunicaciones Telefónica ha utilizado WebRTC para desarrollar aplicaciones de atención al cliente a través de videoconferencia. (quobis.com, 2021)

Antecedentes Históricos

“Descripción general de la tecnología WebRTC y diseño e implementación de soluciones de señalización”. Este estudio tiene como objetivo describir la tecnología WebRTC y su implementación tanto en el cliente como en el servidor, así como en la señalización. Se detallan y explican las partes más importantes de la API de WebRTC, y se ha diseñado e implementado un mecanismo de señalización innovador, ya que la señalización no está regulada por los estándares de WebRTC. La secuencia de mensajes correspondiente a la comunicación de WebRTC se muestra en el estudio, lo que permite entender cómo fluye la comunicación entre los pares y el servidor. En este caso, el servidor se ha implementado como un servidor WebSocket. La aplicación cliente que se presenta en el estudio ejemplifica el uso de la API de WebRTC para lograr comunicaciones en tiempo real. Por último, se discuten los avances y las posibilidades futuras para el desarrollo de la tecnología WebRTC. (Sredojev, Samardzija, & Posarac, 2015)

“WebRTC - Una nueva tecnología web al servicio de la educación”, Describe el desarrollo, de un proyecto en la Universidad Manuela Beltrán para resolver problemas en la herramienta de chat que requería plugins y barreras tecnológicas. Se optó por utilizar nuevos estándares informáticos de comunicación para permitir el uso de texto y vídeo, con el objetivo de ahorrar recursos tecnológicos. La aplicación se basó en la "Web en tiempo real" utilizando Node.js, WebSockets y WebRTC para crear un sistema de conversación con notificaciones en tiempo real y potenciar la comunicación a través del video. Los resultados muestran que la eliminación de complementos facilitó el uso de la herramienta y la comunicación entre todos los usuarios mejoró el aprendizaje y la interacción entre estudiantes. Desde la perspectiva docente, se observó mayor efectividad en la atención de inquietudes y participación estudiantil. El marco teórico de la investigación proporciona una base sólida para entender el problema y su solución a través de teorías, conceptos y antecedentes relevantes. (Rubiano, Mena, & Hernández, 2014)

“Desarrollar un Prototipo de Sistema Call Center Utilizando la Tecnología WEB-RTC con ELASTIX en la Plataforma Web de la Empresa COMWARE S.A”. En este proyecto cuentan con un estudio y presentación de una nueva tendencia que se está presentando en el mercado, conocida como WebRTC, el proyecto se encarga de mejorar la atención al cliente mediante

una plataforma web ya existente en la empresa COMWARE S.A añadiendo a este como complemento al sitio Web como una herramienta de comunicación, se describe todos las tecnologías usadas y herramientas usadas en la implementación de esta herramienta con WebRTC. (Cobos, Pauta Cuji, & Stalyn, 2016)

Bases teóricas

4.4 Navegador

Un Navegador es un programa informático que cuenta con una interfaz visual accesible al usuario que nos permite acceder a diferentes sitios web e interactuar con ellos, bien sea investigando, compartiendo información, multimedia, etc. Los navegadores web leen e interpretan códigos HTML, CSS y JavaScript, que son los lenguajes utilizados para crear páginas web, y muestran el contenido y la estructura de la página en una ventana en el ordenador o dispositivo móvil. A estos se accede mediante un protocolo (conjunto de reglas y normas que permiten la intercomunicación de dispositivos) conocido como HTTP (Hypertext Transfer Protocol, o Protocolo de transferencia de hipertexto).

Figura 2. Componentes de los navegadores de internet
Fuente: Extraído de www.mediasource.mx

Por último, es importante tener un navegador compatible con WebRTC para que los usuarios puedan realizar llamadas y videoconferencias de manera fluida y sin interrupciones. La mayoría de los navegadores modernos, incluyendo Google Chrome, Mozilla Firefox, Safari y Microsoft Edge, son compatibles con WebRTC. (Álvarez, 2020)

4.5 Protocolo SIP usado en voz IP

4.5.1 Session Initiation Protocol (SIP o Protocolo de Inicio de Sesiones): Es un protocolo desarrollado por el grupo de trabajo MMUSIC del IETF con la intención de ser el estándar para la iniciación, modificación y finalización de sesiones interactivas de usuario donde intervienen elementos multimedia como el video, voz, mensajería instantánea, juegos en línea y realidad virtual. (NFON, 2021)

Para la comunicación a través de SIP, se necesitan al menos dos clientes SIP. Un cliente SIP puede ser un teléfono físico SIP, un SoftPhone, WebPhone, un cliente móvil o un producto Axis compatible con Axis.

Cada cliente SIP tiene asignada su propia dirección SIP. Una dirección SIP es similar a una dirección de correo, pero con el prefijo “sip:”. (NFON, 2021)

Por ejemplo, sip:bob@axis.com **[sip:<usuario@]<proveedor>.** Este identificador puede utilizarse en diferentes dispositivos y es similar a un número de teléfono vinculado a una tarjeta SIM, que puede usarse en varios dispositivos. (NFON, 2021)

4.6 Arquitectura SIP

El propósito de SIP es la comunicación entre dispositivos multimedia. SIP hace posible esta comunicación gracias a dos protocolos que son RTP/RTCP y SDP. El protocolo RTP se usa para transportar los datos de voz en tiempo real (igual que para el protocolo H.323, mientras que el protocolo SDP se usa para la negociación de las capacidades de los participantes, tipo de codificación, etc.) (Matango, 2016)

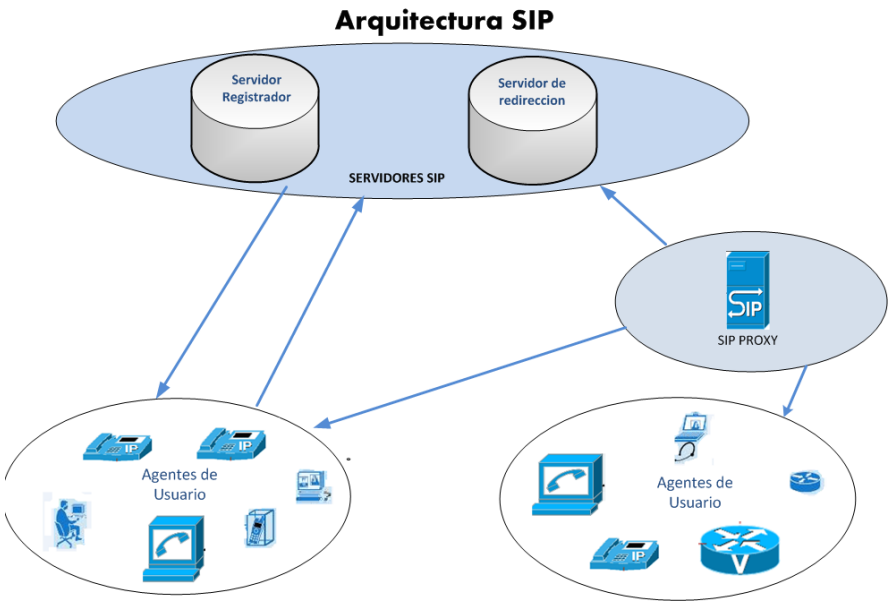

Figura 3. Arquitectura SIP
Fuente: extraído de https://docplayer.es/43780624-Paul-andres-arias-vallejo.html

4.7 VoIP

La VoIP, o Voz por el Protocolo de Internet (Voice over Internet Protocol), es un método para tomar señales de audio análogas, como las que se escuchan en una llamada telefónica, convertidas en data digital para que pueda ser transmitida por la Internet. Esto significa que VoIP puede cambiar una conexión de internet de manera que esta pueda hacer llamadas telefónicas gratis. El resultado práctico de esto es que, al usar los programas de VoIP para hacer llamadas por internet, se elimina a la compañía telefónica (y sus cuotas) completamente.

La VoIP es una gran mejora con respecto al sistema telefónico actual en su eficiencia, el costo y la flexibilidad. Como cualquier tecnología emergente, la VoIP tiene algunos desafíos que superar, pero es claro que los desarrolladores mantener refinado esta tecnología hasta que ésta se sustituye el actual sistema telefónico. (SOTO, MORENO, & DIAZ, 2009)

Toda la comunicación VoIP se envía como paquetes (segmentos de datos) a través de una

LAN (Red de Área Local) y WAN (Red de Área Amplia), en lugar del tradicional cableado de cobre. Usando el modelo de VoIP, en lugar de tener a un proveedor tradicional de servicio análogo o digital, se requiere un proveedor de VoIP. El proveedor de VoIP asigna direcciones IP estáticas a los dispositivos, lo cual permite que los dispositivos sean identificados cada vez por el mismo número cuando son llamados. Estas direcciones IP estáticas conectan sus dispositivos con el mundo exterior y los hacen accesibles a los demás.

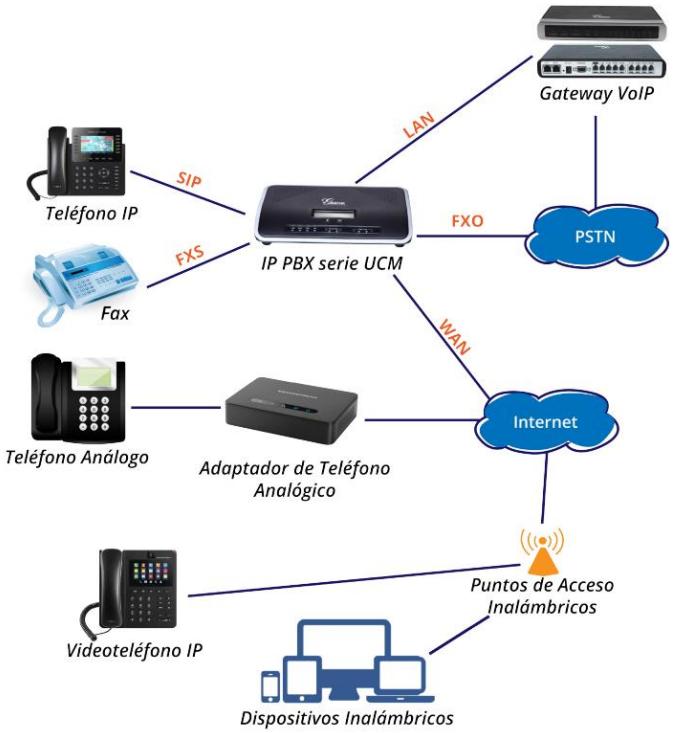

Figura 4. Diagrama de una Red VoIP
Fuente: Extraído de Grandstream.com

4.8 Tecnología WebRTC

La WebRTC, también conocida como Web Real Time Communication, es un software de código abierto que permite realizar llamadas de voz, videollamadas y compartir archivos en línea y en tiempo real entre navegadores. Las diferentes aplicaciones que ofrecen servicios de comunicación a distancia emplean diverso tipo de software como es el micrófono, auriculares, cámara, etc. Es precisamente la que permite que el navegador desde el que se quiera realizar la comunicación bien sea una llamada, videollamada, etc. pueda acceder a

dicho software. Es decir, la tecnología WebRTC hace posible que el navegador web tenga acceso al software y hardware necesario para poder crear aplicaciones web que permitan la establecer la comunicación a distancia. (MasIP, 2019)

4.9 Desarrollo de la WebRTC

La tecnología WebRTC no ha experimentado todo el crecimiento y desarrollo esperado debido a varios factores que lo han retrasado. Entre estos factores, cabe destacar la falta de soporte de algunos navegadores web.

Los navegadores Chrome, Firefox y Opera soportan la tecnología WebRTC desde hace mucho tiempo, tanto en el caso de sus versiones de escritorio como en el caso de sus versiones para Android. En la actualidad, casi todos los navegadores soportan la tecnología WebRTC, por lo que podrás tener acceso a esta tecnología en cualquier dispositivo que cuente con alguno de ellos.

Esto implica grandes ventajas como son la flexibilidad y la libertad de movimiento, ya que hace posible enviar y recibir llamadas de voz, video y cualquier otro tipo de datos en tiempo real desde donde nos encontremos, siendo únicamente necesario contar con un dispositivo conectado a Internet con un navegador compatible. (TrueConf, 2023)

4.10 Funcionamiento de la WebRTC

- El usuario abre una página de WebRTC.
- El navegador solicita acceso a la cámara y micrófono del ordenador y el usuario debe otorgarlo (a uno de los dos o ambos según se desee).
- El SDP (Protocolo de Descripción de Sesión) se genera en el navegador que inicia la conexión.
- Un iniciador de conexión transfiere los datos de conexión a otros usuarios participantes. Generalmente, se utiliza un servidor de señalización y un protocolo WebSocket para el propósito.

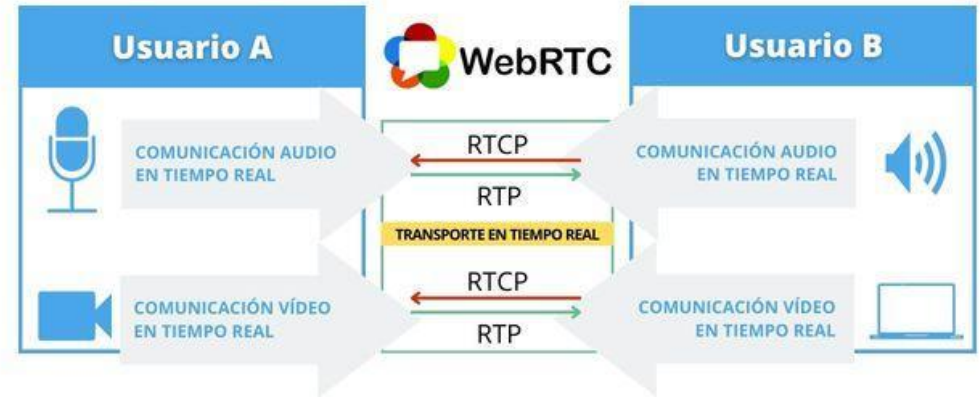

Figura 5. Funcionamiento WebRTC
Fuente: extraído de https://gesditel.es/webrtc/

- Del lado del usuario receptor de la comunicación, un navegador recibe un paquete SDP y luego genera otro similar que toma los datos también desde el primer paquete. El segundo paquete se envía de vuelta al lado inicial.
- Dependiendo de su implementación, el estado de conexión de la red en analizada a la par que los pasos anteriores. Los usuarios reciben una dirección de servidor STUN que se utiliza para conocer la dirección IP externa del dispositivo.

A continuación, se compara con la dirección IP interna para determinar si se está utilizando NAT con la conexión y, de ser así, cómo se enrutan los paquetes UDP.

En algunos casos más complejos, como puede ser, cuando se usa un doble NAT, los desarrolladores utilizan servidores TURN. Básicamente son repetidores capaces de convertir una red de pares (P2P) en una cliente-servidor-cliente.

- Finalmente, después de completar con éxito los pasos anteriores, la conexión quedará establecida. (Gesditel, 2020)

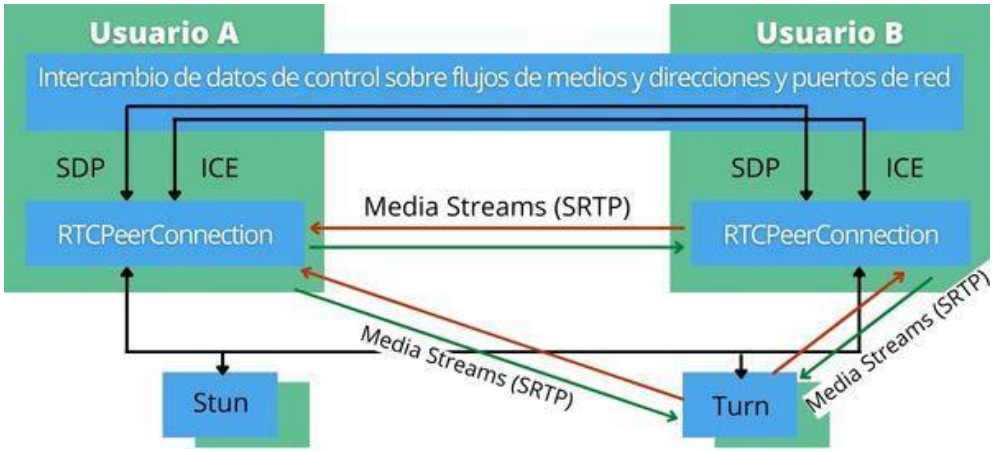

Figura 6. Media Streams de WebRTC
Fuente: extraído de https://gesditel.es/webrtc/

4.11 Configuración WebRTC

Para realizar la configuración de un sistema de comunicación basado en WebRTC, son necesarios tres componentes principalmente:

4.11.1 Servidor de señalización WebRTC. Para establecer conexiones con este sistema, los pares deben contactar con un servidor de señalización. Este servidor será el encargado más tarde de proporcionar la información de la dirección necesaria para establecer una conexión de igual a igual. Algunos ejemplos de servidores de señalización son:

- **Signalmaster:** servidor ligero basado en JavaScript
- **NextRTC**: servidor basado en Java
- **Kurento:** marco integral de WebRTC
- **Janus:** WebRTC Gateway de propósito general

4.11.2 Aplicación de cliente WebRTC. El cliente accede a WebRTC de un navegador a través de una API de JavaScript o usa una biblioteca WebRTC como parte de una aplicación de escritorio o móvil. Para conectar con un par, el primer cliente debe conectarse al servidor de señalización.

4.12 Servidor STUN / TURN

Las Utilidades de Trasversal de sesión para NAT (STUN) permiten a los clientes intercambiar información de dirección incluso si están detrás de los enrutadores que emplean la Traducción de dirección de red (NAT).

Si las restricciones de red impiden que los interlocutores se comuniquen directamente, el tráfico se enrutará a través de un Servidor de uso de retransmisiones alrededor del servidor NAT (TURN). (Gesditel, 2020)

4.13 WebSockets

El protocolo WebSocket permitió por primera vez acceder a una web de forma dinámica en tiempo real. Con este protocolo, basta con que el cliente establezca una conexión con el servidor, que se confirma mediante el llamado apretón de manos o WebSocket Protocol Handshake. Con él, el cliente envía al servidor todos los datos de identificación necesarios para el intercambio de información. El protocolo basado en TCP establece como deben intercambiarse datos entre redes. Estables conexiones entre dos puntos finales de comunicación, llamados sockets. De esta manera, el intercambio de datos puede producirse en las dos direcciones. (IONOS, 2020)

Los WebSockets establecen una conexión bidireccional persistente entre el cliente y el servidor. A través de esta conexión, el cliente puede enviar datos al servidor en cualquier momento, y el servidor puede enviar datos al cliente por iniciativa propia en cualquier momento.

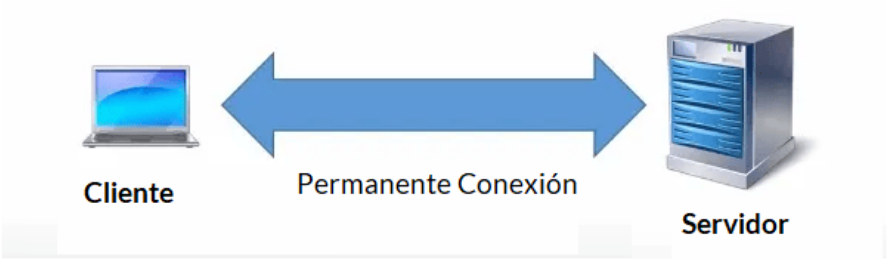

Figura 7. Conexión Permanente entre cliente y servidor
Fuente: Extraído de https://ccnadesdecero.es/que-es-websocket/

Los WebSockets, así como el HTTP, funcionan en los puertos 80 y 443 si se utiliza la encriptación. Para los web sockets hay un prefijo especial en la URL ws/ que significa web sockets o wss/ si los sockets se utilizan junto con SSL TLS para el cifrado. La URL de los web sockets es la siguiente: ws://www/ccnadesdecero.es/chat).

4.14 JsSIP

JsSIP es una biblioteca para el lenguaje de programación JavaScript. Aprovecha SIP y WebRTC para proporcionar un punto final SIP con todas las funciones en cualquier sitio web. JsSIP permite que cualquier sitio web obtenga funciones de comunicación en tiempo real utilizando audio y video. Permite crear agentes de usuario SIP que envían y reciben llamadas de audio y video, así como mensajes de texto. (hmong.es, n.d.)

4.15 React JS

React ayuda a crear interfaces de usuario interactivas de forma sencilla. Diseña vistas simples para cada estado en la aplicación, React se encargará de actualizar y renderizar de manera eficiente los componentes correctos cuando los datos cambien. React es una biblioteca o librería de código abierto que está escrita en JavaScript. Fue desarrollada por Facebook en el 2013 con la finalidad de facilitar la creación de componentes reutilizables e interactivos para las interfaces de usuario.

ReactJS se integra perfectamente con WebRTC, permitiendo a los desarrolladores

construir aplicaciones de comunicación en tiempo real con una experiencia de usuario suave y reactiva. ReactJS proporciona una gran cantidad de componentes y herramientas que simplifican el desarrollo de aplicaciones web complejas, y su capacidad para manejar estados y renderizados dinámicos es ideal para aplicaciones de comunicación en tiempo real.

Al utilizar ReactJS con WebRTC, los desarrolladores pueden crear aplicaciones con una experiencia de usuario fluida y una interacción en tiempo real entre los usuarios. Además, ReactJS permite la creación de componentes reutilizables que se pueden compartir y utilizar en múltiples aplicaciones, lo que agiliza el desarrollo y reduce el tiempo y los costos de desarrollo. (Next U, 2020)

4.16 Leyes y estándares para VoIP

Para la prestación de servicios de telecomunicaciones se requiriere la respectiva concesión, expedida por el Gobierno Nacional. Las tecnologías en sí mismas son neutrales y pueden ser aptas para la configuración de las redes y la prestación de los servicios de telecomunicaciones. La transmisión de voz por medio del protocolo IP (VoIP) se considera dentro del marco legal y es cobijada por la licencia para la prestación de servicios de Valor Agregado y Telemáticos. Para prestar los servicios de Valor Agregado y Telemáticos se debe solicitar el Título Habilitante Convergente, otorgado mediante el Decreto 2870 del 31 de Julio de 2007, esta denominación comprende, entre otros servicios, los servicios de Valor Agregado y Telemáticos.

Asimismo, la transmisión de voz por medio del protocolo IP (VoIP) se considera legal, siempre y cuando en todos los extremos de la red existan interfaces para terminales de valor agregado, y tal red no sea abonada, pues si en algún momento dicho servicio, abandona la red de valor agregado, será considerado clandestino en la medida que carezca de la habilitación como operador telefónico o en la medida en que carezca de un acuerdo comercial con un operador legalmente habilitado para tal fin, por lo tanto, solo los operadores con licencia para prestar servicios de Valor Agregado y Telemáticos se encuentran autorizados para utilizar sus redes en la transmisión de voz por IP y las comunicaciones exclusivas de voz desde y hacia el territorio nacional, solo podrán ser cursadas por los operadores de telefonía Publica Básica Conmutada de Larga Distancia TPBCLD autorizados

5. Metodología

5.1 Tipo de proyecto

Tipo aplicativo: se escogió por el método investigativo y desarrollo que se necesita para alcanzar los objetivos.

5.2 Método

Con el método aplicativo se busca determinar el desarrollo del WebPhone. Este método se ajusta al contexto del proyecto y permite evaluar si el desarrollo resultante es más intuitivo que los servicios separados que utilizan WebRTC. Al desarrollar el WebPhone con tecnología WebRTC, se busca ofrecer un servicio de llamadas eficiente tanto de entrada como de salida, elevando la calidad de estas.

5.3 Instrumentos de recolección de información

Instrumento de recolección de información fue sacado de la empresa para cual trabajo actualmente, se analizó un nuevo requisito de desarrollar un nuevo servicio para los clientes, surgió la necesidad de diseñar un WebPhone con WebRTC con el objetivo de ofrecer una opción más económica a aquellos clientes que buscan un servicio más asequible en comparación con opciones más costosas.

La población y muestra van como objeto a los clientes de la empresa, a los cuales está dirigido el proyecto, puesto que estos serán los beneficiarios, por medio de ellos se obtendrá el conocimiento e información sobre los requerimientos que debe cumplir el desarrollo.

5.3.1 Fuentes primarias.

La fuente Primario de información son las librerías Oficiales de las tecnologías que se usaran que son el sector Tecnológico Principal WebRTC, la fuente principal la librería JsSip que es el que contiene todas las funciones y conexiones que se requieren para usar WebRTC.

5.3.2 Fuentes secundarias.

La fuente Secundarias son búsquedas en Google académico, archivos PDF de trabajos investigativos, Tesis, y documentación de desarrollo de Google, también el soporte TI de la empresa que se encarga de brinda la información necesaria de conexiones SIP.

6. Resultados del proyecto

Cuando se le asignó la tarea al equipo de trabajo de construir un WebPhone utilizando WebRTC, no se contaba con conocimientos previos sobre esta tecnología. Por lo tanto, tuvieron que dedicar tiempo a investigar y comprender qué era WebRTC y cómo se podía utilizar para construir un WebPhone.

Con el objetivo de obtener una mejor comprensión de los conceptos, el equipo comenzó describiendo qué es un WebPhone.

6.1 ¿Qué es un WebPhone?

Un WebPhone es un SoftPhone (Software para llamadas) con tecnología WebRTC que funciona a través de cualquier navegador WEB con soporte html5, para hacer y recibir llamadas directamente desde tu ordenador o tablets, ya comprendido esto, se hayo también funcionalidades y características que debe tener disponibles un WebPhone para su completo funcionamiento.

6.2 ¿Qué funcionalidades y/o características tiene el WebPhone?

Cada funcionalidad hace que el WebPhone se parezca a un teléfono físico y la idea es hacerlo lo mejor parecido en funcionalidad. Las siguientes características son las que tiene disponible un WebPhone:

- Realizar y recibir llamadas desde tu navegador web Chrome o Firefox, etc.
- Histórico de llamadas realizadas, recibidas y perdidas.
- Buzón de voz (solo para extensiones).
- Transferencia de llamadas (directa, ciega o atendida).

- Marcación directa desde el registro de llamadas.
- Llamada en espera.
- Silenciar el micrófono.
- Contador de segundos y Minutos.

Se investigó cómo utilizar WebRTC para construir un WebPhone. Se descubrió que construir un WebPhone utilizando WebRTC requería habilidades en programación web, como HTML, CSS y JavaScript, así como una comprensión de la arquitectura de redes y los protocolos de comunicación. Por consiguiente, era necesario invertir tiempo en adquirir estas habilidades antes de poder dar inicio a la construcción del WebPhone.

6.3 WebRTC: Funcionamiento de captura de Audio y video

WebRTC está diseñado para conexiones peer-to-peer, lo que significa que datos como flujos de video o audio pueden fluir directamente entre dos pares a través de un servidor, Configurar una conexión peer-to-peer de este tipo no es intrínsecamente difícil, pero el estándar permite configurar muchas opciones. Esta innovación revolucionaria permite la transmisión fluida de audio y video a través de la web, sin requerir la descarga de aplicaciones externas. En esencia, brinda la capacidad de entablar conversaciones y conexiones en tiempo real con otras personas en línea, simplemente utilizando un navegador web.

Pero realmente cómo funciona esto internamente, bien sabemos que el navegador transmite el audio y video, pero como lo hace, WebRTC hace este trabajo en realidad, captura los dispositivos de entrada de audio y video del dispositivo (como el micrófono y cámara) para capturar el audio y video que se transmitirá. El estándar de WebRTC proporciona una API a fin de acceder a cámaras y micrófonos conectados a la computadora o smartphone, por lo general WebRTC los llama como “dispositivos multimedia” y se acceden a ellos con JavaScript a través del objeto *navigator.mediaDevices.*

Desde este objeto, podemos enumerar todos los dispositivos conectados, detectar los cambios en ellos (cuando se conecta o desconecta) y abrir uno para recuperar una transmisión multimedia.

La forma más común como WebRTC utiliza o captura los dispositivos es a través de su función *getUserMedia()*, a continuación se demuestra la llamada de esta función.

```
const constraints = {
    'video': true,
    'audio': true
}
navigator.mediaDevices.getUserMedia(constraints)
    .then(stream => {
        console.log('Got MediaStream:', stream);
    })
    .catch(error => {
        console.error('Error accessing media devices.', error);
    });
```

Figura 8. Código para capturar medios
Fuente: extraído de https://webrtc.org/getting-started/media-devices?hl=es-419#using-promises

La llamada a *getUserMedia()* activará una solicitud de permisos. Si el usuario acepta el permiso, la promesa se resuelve con un objeto *MediaStream* que contiene un video y una pista de audio. Si se niega el permiso, se genera una *PermissionDeniedError "Permisos denegados"*. En caso de que no haya dispositivos coincidentes conectados, se arrojará una *NotFoundError "No se encuentra"*.

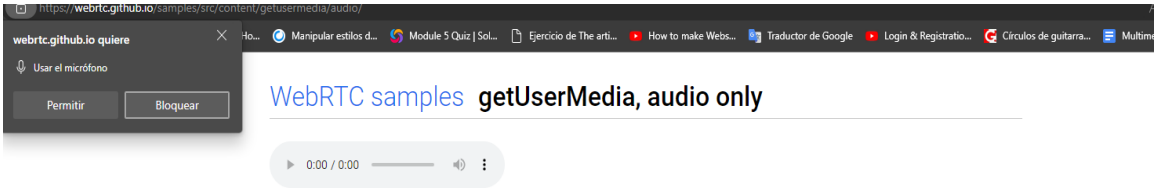

Figura 9. Demostración de captura de medios Audio
Fuente: extraído de https://webrtc.github.io/samples/src/content/getusermedia/audio/

A medida que se profundizaba en la investigación, se descubrió que WebRTC ofrece herramientas y protocolos para la comunicación en tiempo real, la codificación y decodificación

de datos, y la administración de conexiones de red. Esta tecnología utiliza protocolos de red como UDP (User Datagram Protocol) y TCP (Transmission Control Protocol) para garantizar una transmisión confiable y estable.

6.4 WebRTC: Uso de los Protocolo UDP y TCP

El protocolo UDP se utilizó para la transmisión de audio y video en tiempo real, ya que es un protocolo más rápido y eficiente que TCP. UDP transmite los datos en paquetes pequeños y simples, lo que permite una transmisión más rápida a través de la red. Sin embargo, como UDP no tiene mecanismos de control de errores ni de retransmisión de paquetes, existe la posibilidad de que algunos paquetes se pierdan durante la transmisión.

Para solucionar este problema, WebRTC también utilizó el protocolo TCP para la transmisión de datos críticos, como los datos de señalización y control de sesión. TCP es un protocolo más lento que UDP, pero es más confiable ya que garantiza que todos los paquetes se entreguen en el orden correcto y sin errores.

Además de UDP y TCP, WebRTC también utilizó otros protocolos de red para realizar tareas específicas, como el protocolo ICE (Interactive Connectivity Establishment) para la gestión de conexiones de red y el protocolo SRTP (Secure Real-time Transport Protocol) para la seguridad de la transmisión de audio y video.

6.5 WebRTC: Uso de los Real-Time Protocol (RTP)

se utilizó en WebRTC para transmitir audio y video en tiempo real. Este protocolo permitió la entrega eficiente y sincronizada de los flujos de datos multimedia entre los participantes de una comunicación. Dividió los datos en paquetes y los envió a través de la red, asegurando una reproducción coherente y manteniendo la calidad de la transmisión. RTP también facilitó la negociación de códecs para adaptarse a diferentes capacidades de red y dispositivos.

6.6 WebRTC: Uso de los ICE (Interactive Connectivity Establishment)

ICE se utilizó para establecer conexiones de comunicación en tiempo real entre dos extremos en una red. Permitió la detección y el establecimiento de una ruta de comunicación eficiente entre los participantes, esto nos ayuda a establecer conexiones de comunicación entre dos extremos a través de redes complejas, como NAT (Network Address Translation) o cortafuegos (firewall).

Cuando un cliente WebRTC deseaba establecer una comunicación directa con otro cliente, utilizaba ICE para descubrir las direcciones IP y los puertos disponibles para recibir conexiones. Esto se hizo a través de un proceso llamado "recolección de candidatos". Durante esta etapa, el cliente recopiló información sobre sus interfaces de red, direcciones IP y puertos disponibles.

Una vez que se recopilaron los candidatos, el cliente utilizó ICE para determinar la mejor manera de establecer la conexión.

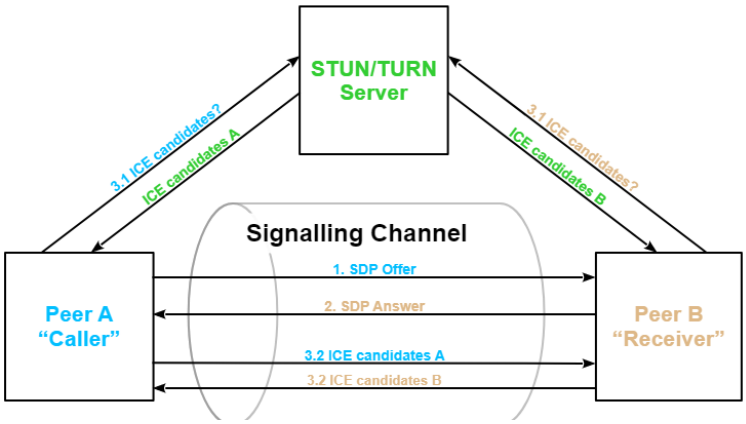

Figura 10. Demostración de funcionamiento de ICE
Fuente: extraído de https://thalerit.net/software%20development/webrtc/understanding_webrtc/

La configuración de una conexión WebRTC peer-to-peer se puede dividir aproximadamente en tres pasos:

- Creación y envío de una oferta SDP
- Creación y envío de una respuesta SDP
- Reunir y enviar candidatos de ICE

La *figura 10* representa esquemáticamente estos cuatro pasos.

6.6.1 Canal de señalización. El canal de señalización es una componente esencial en el proceso de establecimiento de una conexión peer-to-peer en WebRTC. A través de este canal, los participantes pueden intercambiar ofertas de SDP, respuestas de SDP y candidatos de ICE entre sí. Es mediante el canal de señalización que los pares se comunican y coordinan para establecer una conexión exitosa.

6.6.2 Crear y enviar una oferta de SDP. En primer lugar, cuando el par A, conocido como "llamante", desea establecer una conexión WebRTC con el par B, debe crear una oferta de Protocolo de Descripción de Sesión (SDP). Esta oferta contiene información relevante sobre la sesión multimedia que se desea establecer, como los tipos de medios (audio, video, datos), los formatos de medios y los protocolos de transporte que se utilizarán.

Una vez que el SDP de la oferta ha sido creado, el par A envía esta oferta a través del canal de señalización al par B. El canal de señalización actúa como un medio de comunicación entre los pares, permitiéndoles intercambiar información necesaria para establecer la conexión.

Una vez que el par B recibe la oferta de SDP, examina su contenido y puede decidir si aceptarla o no. Si decide aceptarla, el par B crea una respuesta de SDP que tiene una estructura similar a la oferta de SDP. La respuesta de SDP también contiene información sobre los tipos de medios, los formatos de medios y los protocolos de transporte que el par B utilizará en la sesión.

Finalmente, el par B envía la respuesta de SDP a través del canal de señalización de vuelta al par A. Con la oferta y la respuesta de SDP intercambiadas, ambos pares tienen la información necesaria para establecer los parámetros de la conexión y comenzar a comunicarse mediante WebRTC. (Thaler, 2021)

6.6.3 Reunir un envió de candidatos Ice. Luego, tanto el par A como el par B solicitan candidatos de Establecimiento de Conexión a Internet (ICE) a uno o varios servidores de

utilidades transversales de sesión para NAT (STUN). Estos candidatos ICE contienen información sobre los puntos finales de conexión a Internet públicos de cada par, lo que permite establecer una conexión punto a punto entre ellos.

6.7 WebRTC: uso de Secure Real-Time Protocol (SRTP)

Secure Real-Time Protocol (SRTP): Es una extensión del RTP que proporciona seguridad y confidencialidad para los flujos de datos multimedia. SRTP se utiliza para cifrar y autenticar los datos transmitidos en WebRTC, asegurando la privacidad de la comunicación.

6.8 Integración SIP con WebRTC

El Protocolo de Inicio de Sesión (SIP) ha estado presente desde los años 90 y se utiliza para administrar sesiones de medios entre dos puntos finales conectados a IP. Con frecuencia, se emplea en comunicaciones de voz sobre IP (VoIP), lo que ha ampliado su alcance de manera significativa a lo largo del tiempo. Aunque ha evolucionado para incluir numerosas especificaciones, propuestas y extensiones, los conceptos fundamentales de SIP son bastante simples.

WebRTC está muy naturalmente relacionado con todo esto. Al igual que SIP, está destinado a admitir la creación de sesiones de medios entre dos puntos finales conectados a IP. Al igual que SIP, las conexiones utilizan el Protocolo de transporte en tiempo real (RTP) para los paquetes en el plano de los medios una vez que se completa la señalización. Al igual que SIP, utiliza SDP para describirse a sí mismo. (Venema, 2022)

La integración de WebRTC con SIP fue un enfoque poderoso que combinó la capacidad de comunicación en tiempo real de WebRTC con la infraestructura de señalización y control de sesiones de SIP. Esta combinación permitió una comunicación multimedia segura y confiable a través de aplicaciones web y dispositivos compatibles con SIP.

Para lograr la integración de WebRTC con SIP, se requería una pasarela o un servidor de sesión de medios que actuara como intermediario entre los protocolos. Este servidor podía

convertir las señales y mensajes de señalización de SIP en formatos comprensibles para WebRTC y viceversa, lo que permitía la interoperabilidad entre los dos sistemas.

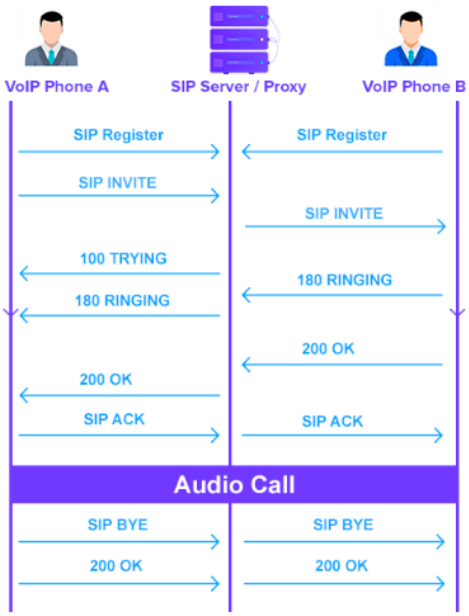

Figura 11. Demostración de funcionamiento flujo llamada entre SIP y WebRTC
Fuente: extraído de https://thalerit.net/software%20development/webrtc/understanding_webrtc/

Por lo general como se ve en la figura 6 el usuario A debe registrarse al servidor SIP, lo mismo para el usuario B o los usuarios que en efecto usen una infraestructura SIP, ya cuando los usuarios estén registrados, pueden hacer invitación que quiere decir que se está enviando los requerimientos para establecer una conexión, todo este proceso es conocido como flujo de llamadas SDP, y esto se encarga el servidor SIP, de hacer esa transcodificación de WebRTC a SIP, de esta forma se combinan las 2 tecnologías, dando una integración entre ellas.

Para finalizar SIP se utiliza como protocolo de señalización para establecer y controlar las sesiones, mientras que WebRTC se encarga del transporte de los datos multimedia (audio y video). Los mensajes SIP se utilizan para la negociación de parámetros de comunicación, como la dirección IP y los puertos utilizados por los participantes. Una vez que se han intercambiado los mensajes SIP necesarios para establecer la llamada, WebRTC se utiliza para transmitir el audio y el video directamente entre los participantes, utilizando los protocolos RTP y SRTP.

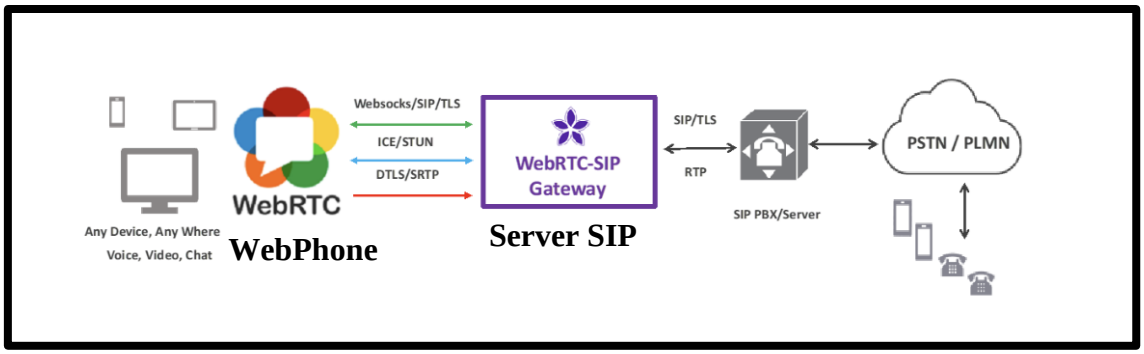

Figura 12. Arquitectura Integración SIP con WebRTC
Fuente: Propia

La arquitectura implementada para proporcionar conectividad al WebPhone se basa en una combinación de SIP y WebRTC, lo que permite la realización de llamadas entre dispositivos de diferentes tipos, incluyendo llamadas desde WebRTC a través de la web, desde WebRTC a PSTN (red telefónica pública conmutada) y entre dispositivos que utilizan exclusivamente SIP.

En esta arquitectura, el WebPhone es accesible a través de una aplicación web basada en WebRTC, que utiliza esta tecnología para capturar y transmitir audio en tiempo real desde los dispositivos de los usuarios.

Para establecer las sesiones de comunicación, la aplicación web realiza solicitudes SIP al servidor correspondiente. Estas solicitudes SIP, como INVITE, se utilizan para negociar los parámetros de la llamada, tales como las direcciones IP, los puertos y los códecs de audio y video.

Una vez que se han establecido los parámetros de la sesión, la aplicación web utiliza WebRTC para establecer una conexión directa entre los dispositivos involucrados en la llamada. Durante este proceso, se utiliza el protocolo ICE para sortear obstáculos como las restricciones de NAT y los cortafuegos, permitiendo así establecer una conexión punto a punto confiable y segura. La arquitectura implementada para el WebPhone combina las capacidades de SIP y WebRTC, lo que facilita la realización de llamadas entre diferentes tipos de dispositivos. Proporciona una solución versátil y escalable para la comunicación en tiempo real, tanto a través de la web como a través de redes telefónicas convencionales lo cual cumple con los objetivos de la empresa de mejorar sus servicios y ampliar su alcance.

6.9 Implementación WebPhone

Durante la implementación de un WebPhone utilizando tecnología WebRTC, seguí una metodología de desarrollo web que me permitió organizar y llevar a cabo el proyecto de manera eficiente.

6.9.1 Modelado de la Aplicación Web casos de uso. El modelado de una aplicación web es un proceso esencial que permite diseñar y representar la estructura y las interacciones de la aplicación de manera visual. Al utilizar herramientas y técnicas de modelado, se pueden identificar y comprender de manera más clara los componentes, las relaciones y el flujo de datos de la aplicación, lo que facilita su desarrollo y mantenimiento.

6.9.1.1 Actores. Los actores representan un tipo de usuario en el sistema que realiza un rol. El usuario final es quien utiliza el WebPhone para realizar y recibir llamadas de voz en tiempo real. Puede ser un empleado de una empresa, un cliente o cualquier persona que utilice la aplicación, por lo que el WebPhone únicamente tiene como actor al cliente.

6.9.1.2 Diagramas de uso WebPhone

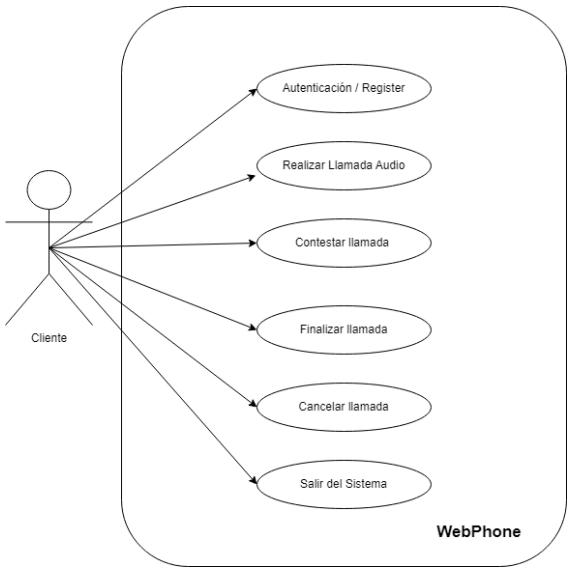

Figura 13. Diagrama de uso WebPhone
Fuente: Propia

6.9.1.3 Detalle de los casos de uso

Tabla 1
Caso de uso Autenticación

CASO DE USO: Autenticación / Register	**Código:** CU-001
Actores: Cliente	
Descripción: El cliente ingresa los parámetros necesarios para registrarse al servidor	
Activación: clic en el botón de Loguearse	
Curso Normal	**Alternativas**
1. Ingresar dirección SIP 2. Ingresar contraseña 3. Clic en el botón Loguearse 4. el estado de conexión pasara "Conectado" y se activa el botón para llamar.	En el caso de que se ingrese mal los parámetros se mostrara un mensaje de error. Si el usuario no posee acceso debe solicitarlo a la administración de las cuentas
Precondiciones: El navegador web debe mostrar la página de inicio del sitio web, mientras que los servidores WebSocket y SIP deben estar en funcionamiento para proporcionar los servicios.	
Postcondiciones: Cuando se ingresa correctamente, en el webPhone debe estar el estado de "Conectado" o "Conected".	
Observaciones y Datos: Los parámetros necesarios para la autenticación son: dirección SIP del servidor, dirección del WebSockets y contraseña.	

Tabla 2
Casos de uso Realizar llamada

CASO DE USO: Realizar la llamada	**Código:** CU-002
Actores: Cliente	
Descripción: El cliente ingresa el número de celular o extensión en el WebPhone para realizar la llamada.	
Activación: Ingresa el numero o extensión y dar en el icono de llamada para llamar.	
Curso Normal	**Alternativas**
1. Ingresar número o extensión 2. se abre una ventana de la llamada 3. se presenta un mensaje de espera 4. si el usuario responde se establece la llamada	Si el usuario de destino no contesta u ocurre algún error da un aviso y se cierra la ventana de llamada.
Precondiciones: los usuarios que deseen hacer una llamada o recibir una llamada deben estar conectados al sistema.	
Postcondiciones:	
Observaciones y Datos: Durante la llamada no se puede realizar o recibir una nueva, se debe esperar a finalizar.	

Tabla 3
Casos de uso Contestar llamada

CASO DE USO: Contestar la llamada	**Código:** CU-003
Actores: Cliente	
Descripción: El cliente al estar conectado se le abrirá una ventana de llamada entrante y le aparece el botón e información del llamante para contestar la llamada	
Activación: Pulsar icono del teléfono para contestar la llamada	
Curso Normal	**Alternativas**
1. Estar conectado en el sistema 2. se abre una ventana de la llamada entrante con la información de quien llama. 3. en la ventana abra 2 botones uno para contestar y otro para rechazar. 4. si el usuario responde se establece la llamada	Si el usuario contesta u ocurre algún error da un aviso y se cierra la ventana de llamada.
Precondiciones: los usuarios que deseen hacer una llamada o recibir una llamada deben estar conectados al sistema.	
Postcondiciones:	
Observaciones y Datos: Durante la llamada no se puede realizar o recibir una nueva, se debe esperar a finalizar.	

Tabla 4
Casos de uso Finalizar llamada

CASO DE USO: Finalizar llamada	**Código:** CU-004
Actores: Cliente	
Descripción: Cualquiera de los 2 clientes puede finalizar la llamada	
Activación: Pulsar icono del teléfono botón rojo para finalizar la llamada	
Curso Normal	**Alternativas**
1. Estar conectado en el sistema 2. Estar en una llamada activa 3. Con el botón rojo de la ventana se finaliza la llamada.	
Precondiciones:	
Postcondiciones: Mientras el usuario no cierre la página web seguirá disponible para recibir llamadas.	
Observaciones y Datos:	

Tabla 5
Casos de uso Cancelar llamada

CASO DE USO: Cancelar la llamada	**Código:** CU-005
Actores: Cliente	
Descripción: El cliente puede rechazar la llamada si lo desea	
Activación: Pulsar icono del teléfono botón rojo para finalizar la llamada	
Curso Normal	**Alternativas**
1. Estar conectado en el sistema 2. Se visualiza la ventana cuando entra una llamada. 3. Con el botón rojo de la ventana se cancela la llamada.	
Precondiciones:	
Postcondiciones: Mientras el usuario no cierre la página web seguirá disponible para recibir llamadas.	
Observaciones y Datos:	

Tabla 6
Casos de uso Salir del sistema

CASO DE USO: Salir del sistema	**Código:** CU-006
Actores: Cliente	
Descripción: Al cerrar la página principal el cliente sale del sistema	
Activación: cierra conexión del webPhone, cierra la página, salir del navegador	
Curso Normal	**Alternativas**
1. Estar conectado en el sistema 2. Estar en una llamada activa 3. Al Cerrar la pestaña, cerrar conexión, o salir de navegador se sale del WebPhone. 4. si el usuario responde se establece la llamada	
Precondiciones: El cliente debe estar conectado al sistema	
Postcondiciones:	
Observaciones y Datos:	

6.10 Definición de requisitos funcionales y no funcionales

Fue fundamental definir los requisitos funcionales y no funcionales para el desarrollo de un WebPhone con WebRTC, asegurando que cumpliera con las expectativas de los usuarios y los

estándares de seguridad, calidad y rendimiento. A continuación, se presentaron los requisitos que se tuvieron en cuenta durante el proceso de diseño e implementación de dicho WebPhone.

6.10.1 Requisitos funcionales

- Inicio de sesión: Proporcionar un mecanismo seguro para que los usuarios inicien sesión en el WebPhone.
- Llamadas de voz: Permitir a los usuarios realizar y recibir llamadas de voz a través del WebPhone
- Transferencia de llamadas: Permitir a los usuarios transferir llamadas a otros usuarios o números telefónicos.
- Mute y remove mute: Permitir a los usuarios silenciar o reactivar el micrófono durante una llamada.
- Cerrar conexión Segura: permitir a los usuarios un mecanismo de poder cerrar conexión en el WebPhone

6.10.2 Requisitos no funcionales

- Seguridad: Garantizar la seguridad de las comunicaciones a través de encriptación de extremo a extremo y autenticación segura de usuarios.
- Escalabilidad: Diseñar el sistema para que sea escalable y capaz de manejar un alto volumen de usuarios y llamadas simultáneas.
- Rendimiento: Asegurarse de que el sistema tenga un rendimiento óptimo al manejar las llamadas de forma eficiente y minimizando la latencia.
- Compatibilidad con navegadores: Asegurarse de que el WebPhone sea compatible con los principales navegadores web, como Google Chrome, Mozilla Firefox y Safari, para garantizar una amplia accesibilidad
- Interfaz de usuario intuitiva: Diseñar una interfaz de usuario fácil de usar e intuitiva que permita a los usuarios realizar y gestionar llamadas de manera eficiente.
- Disponibilidad: Garantizar la disponibilidad del servicio WebPhone para que los usuarios puedan realizar y recibir llamadas en cualquier momento.
- Tolerancia a fallos: Implementar mecanismos de respaldo y recuperación ante posibles fallos del sistema para minimizar la interrupción del servicio.

6.11 Diagrama de Actividades

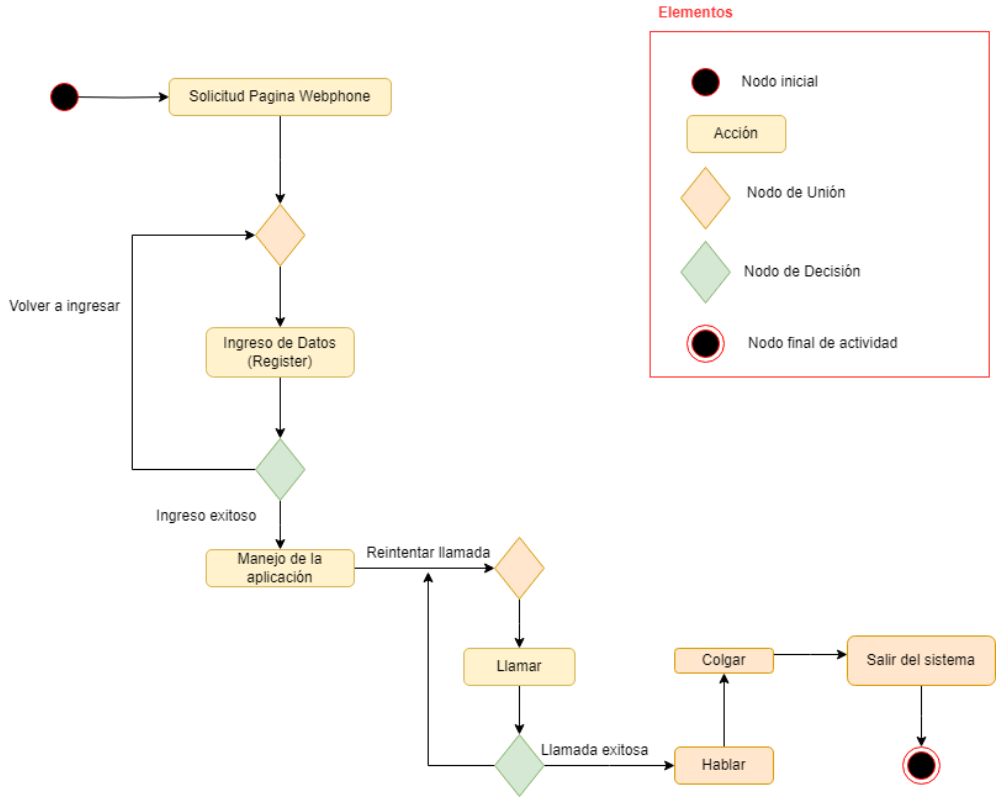

Figura 14. Diagrama de Actividades Funcionamiento
Fuente: propia

6.11 Diseño gráfico y de interfaces

Utilizar wireframes durante la etapa de diseño gráfico del WebPhone fue importante porque permitió la organización y estructuración efectiva de la interfaz, facilitó la iteración y revisión, promovió la comunicación y alineación con los interesados, y enfocó en la usabilidad de la interfaz. Esto condujo a un diseño gráfico sólido y una interfaz bien diseñada para el WebPhone. Es importante aclarar que este no es el diseño oficial propuesto para la empresa ya que por seguridad y privacidad se usó otro con el fin de demostrarlo en este trabajo.

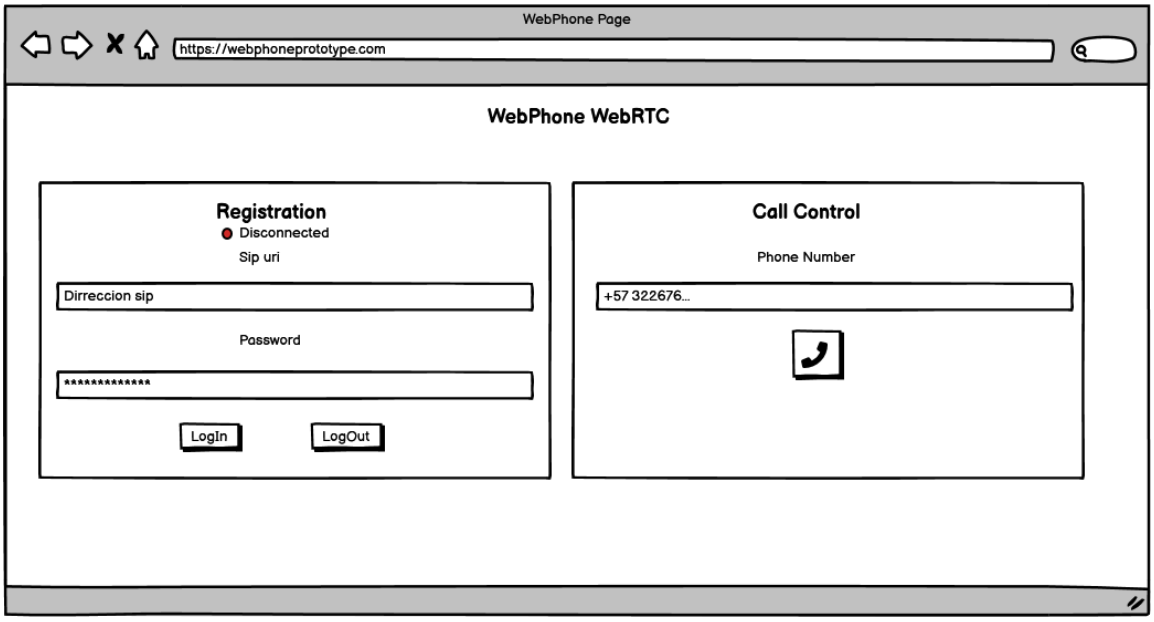

Figura 15. Interfaz Formulario de ingreso y de llamada
Fuente: Diseño propio hecho en Balsamiq Wireframes

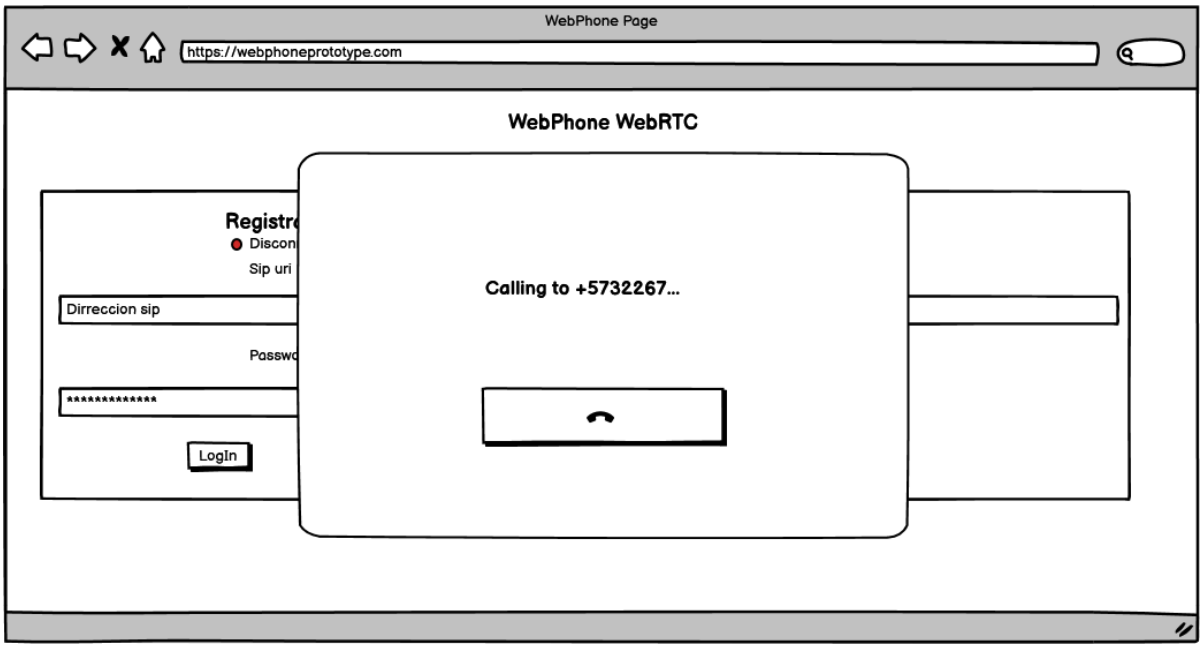

Figura 16. Interfaz de llamando
Fuente: Diseño propio hecho en Balsamiq Wireframes

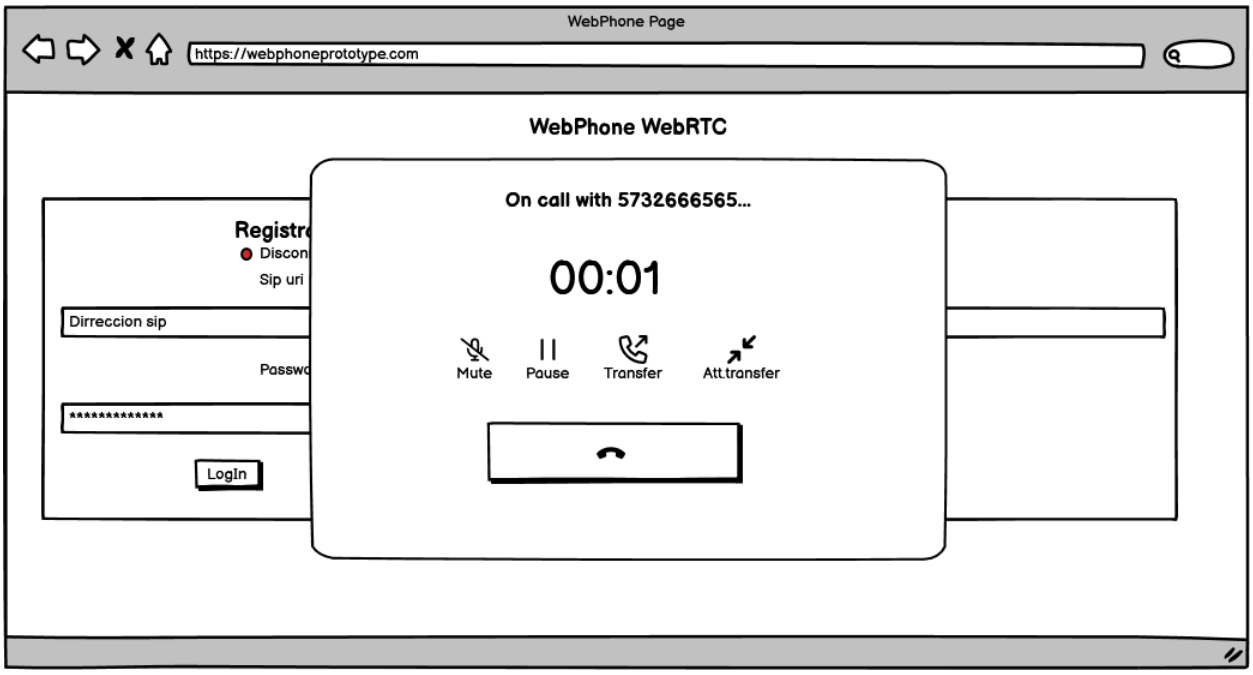

Figura 17. Interfaz de en llamada
Fuente: Diseño propio hecho en Balsamiq Wireframes

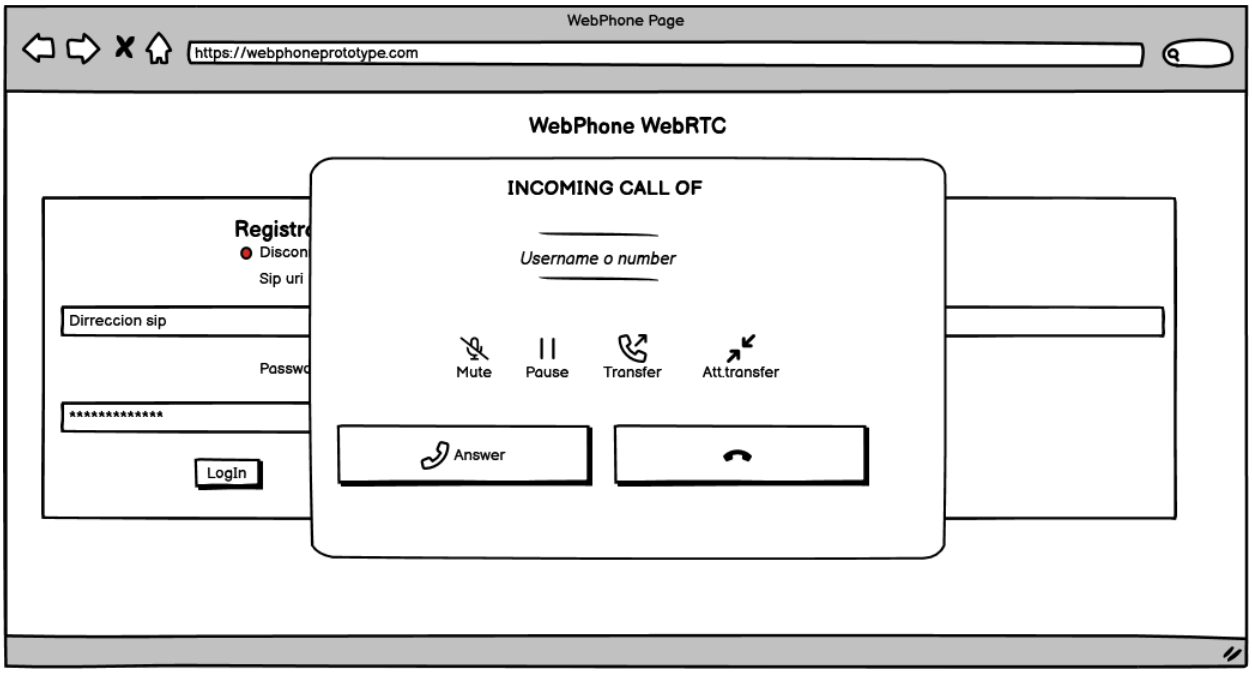

Figura 18. Interfaz de llamada entrante
Fuente: Diseño propio hecho en Balsamiq Wireframes

6.12 Componentes de Software y Hardware

Tabla 7
Componente Cliente

Cliente	
Navegador Web:	Google Chrome, Firefox, Edge...
Lenguaje:	Hmtl5, CSS, JavaScript
Framework:	JsSIP, React
Sistema Operativo:	Linux, Windows

Tabla 8
Componente servidores

Servidores	**Aplicaciones**
Web Server:	Apache
SIP Router:	Opensips
IP PBX:	Asterisk

Tabla 9
Componente Hardware
Fuente: Propia

Hardware	**Equipo**
Equipo de Red:	Switch
Clientes:	Laptops, PC,
Servidores:	Pc o Latop

6.12.1 Descripción de cada componente. El esquema propuesto se componía de tres host. Cada uno de ellos ejecutaba diferentes aplicaciones como servicios. Uno del host se desempeñaba como servidor de la aplicación WebRTC, otro actuaba como un Router SIP y Gateway SIP UDP-SIP WebSockets, y, por último, había una central IP PBX capaz de gestionar SIP sobre UDP y admitir conexiones a la PSTN tradicional, estos servidores y su configuración está fuera del alcance del proyecto, ya que estos componentes son proporcionados por la empresa. El enfoque del proyecto se centra en otras áreas

específicas, sin involucrar la configuración y gestión del host y las aplicaciones mencionadas.

6.12.1.1 Web Server. El servidor web desempeña un papel fundamental al servir la aplicación WebRTC, permitiendo a los usuarios acceder a ella y utilizarla a través de sus navegadores web. En el host GNU/Linux Debian, se ejecuta el proceso Apache como aplicación, lo cual le brinda al host la capacidad de interpretar las solicitudes HTTP y proporcionar contenido web. De esta manera, el servidor web facilita el acceso y la interacción con la aplicación WebRTC a través de los navegadores web de los usuarios.

6.12.1.2 SIP Router. El SIP Router desempeña una doble función dentro del sistema. En primer lugar, actúa como servidor, proporcionando todo el soporte necesario para las funcionalidades SIP, como el registro y la localización de usuarios, así como el funcionamiento del SIP Proxy. De esta manera, facilita las transacciones entre los usuarios del sistema WebRTC que utilizan SIP sobre WebSockets.

Además, este servidor tiene la capacidad de admitir diferentes protocolos de transporte SIP, incluyendo SIP sobre UDP, SIP sobre TCP y SIP sobre TLS. Esto lo convierte en un Gateway que permite a los usuarios del sistema WebRTC interactuar con la IP PBX Asterisk a través de SIP sobre UDP. Así, se establece una conexión entre los usuarios WebRTC y la IP PBX, permitiendo la comunicación fluida y eficiente entre ambos mediante los protocolos de transporte SIP adecuados.

6.12.1.3 IP PBX. Es la central telefónica principal en el esquema propuesto. Funciona como una central de conmutación para los usuarios cuyos dispositivos de comunicación utilizan SIP sobre UDP para establecer diálogos. Además, en este esquema, la IP PBX está conectada a la red de telefonía pública conmutada (PSTN), lo que permite a los usuarios de WebRTC interactuar con los suscriptores de la PSTN de forma bidireccional. Esta conexión a la PSTN tradicional

amplía la funcionalidad de la IP PBX y facilita la comunicación entre los usuarios de WebRTC y los abonados de la PSTN.

6.12 Implementación del WebPhone

La implementación del WebPhone se realizó utilizando la biblioteca JsSIP, seleccionada por su habilidad para simplificar el uso de las APIs proporcionadas por WebRTC. JsSIP fue elegida debido a su capacidad para combinar las características de SIP y WebRTC, lo que permitió aprovechar todas las funcionalidades de comunicación en tiempo real de audio y video necesarias para el desarrollo del Webphone. Esta elección se basó en la capacidad de JsSIP para unificar las funcionalidades de ambas tecnologías y aprovechar al máximo las capacidades ofrecidas.

La biblioteca JsSIP proporcionó una interfaz amigable y una amplia gama de métodos y eventos que facilitaron la implementación de las funcionalidades requeridas para el WebPhone.

Para la parte Fronted, se utilizó el framework React para desarrollar la interfaz de usuario del Webphone. La elección de React se basó en su popularidad, modularidad y capacidad para crear interfaces de usuario interactivas y dinámicas. Al utilizar React, se pudo dividir la interfaz en componentes reutilizables y aplicar un enfoque basado en componentes para el desarrollo.

A Continuación, se presentará la metodología de Desarrollo que se usó utilizando JsSIP y React para construcción del WebPhone.

6.12.1 Configuración del entorno de desarrollo

Esta metodología fue hecha en base a las necesidades del equipo de desarrollo y los requisitos específicos que conllevaba crear el WebPhone del lado del Fronted.

6.12.1.1 Requisitos compatibles para JsSIP: JsSIP utiliza unos requisitos previos para poder ser usado en el ámbito cliente, por eso la librería declara los siguientes requisitos

- Navegador web compatible con WebRTC: JsSIP se basa en WebRTC para proporcionar funcionalidades de comunicación en tiempo real, como audio y video. Por lo tanto, es necesario utilizar un navegador web compatible con WebRTC, como Google Chrome, Mozilla Firefox, Microsoft Edge o Safari.
- Servidor SIP: utiliza el transporte SIP sobre WebSocket para enviar solicitudes y respuestas SIP de recepción y, por lo tanto, requiere un proxy/servidor SIP compatible con WebSocket.

6.12.1.2 Instalación de herramientas y dependencias: Se configuro el entorno de desarrollo de React, node y npm (Node Package Manager), el gestor de código usado fue el visual studio Code.

Configuración de Node y npm: En primer lugar, es necesario descargar el archivo de instalación de Windows Installer (.msi) del sitio web oficial de Node.js, Es de destacar que el instalador también lleva el gestor de paquetes Node.js (npm) dentro de él. Esto significa que no necesitas instalar el npm por separado. Al descargarlo, selecciona la versión correcta según tu sistema operativo, por lo general como el sistema fue x64 se eligió de esa forma, por consiguiente, se elige la versión reciente. A continuación, se muestra en la figura 19 el ejemplo de la página de descarga.

Figura 19. Descarga de Node Js
Fuente: Extraído de https://nodejs.org/es

Luego se comienza el proceso de instalación del node, se establecen algunos parámetros antes de ejecutar el proceso de instalación. Node Js pide aceptar el acuerdo de licencia, y luego solo es dar siguiente y seguir los pasos de la instalación que entre ellos está el lugar de instalación.

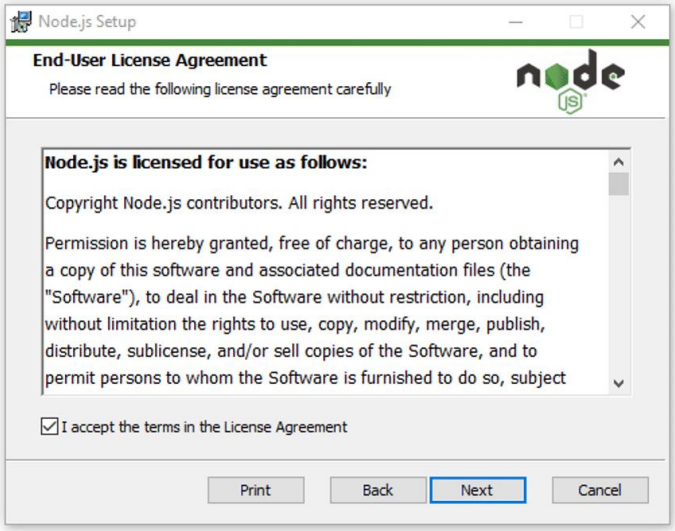

Figura 20. Ejemplo ventana proceso instalación
Fuente: Propia.

Después de la instalación se verifica la instalación del Node Js**.** Para verificar la instalación y confirmar si se ha instalado la versión correcta, se abre la línea de comandos de Windows y se introdujo el siguiente comando: *Node –versión.* Y para comprobar la versión de npm, se ejecutó este comando: *npm –version.*

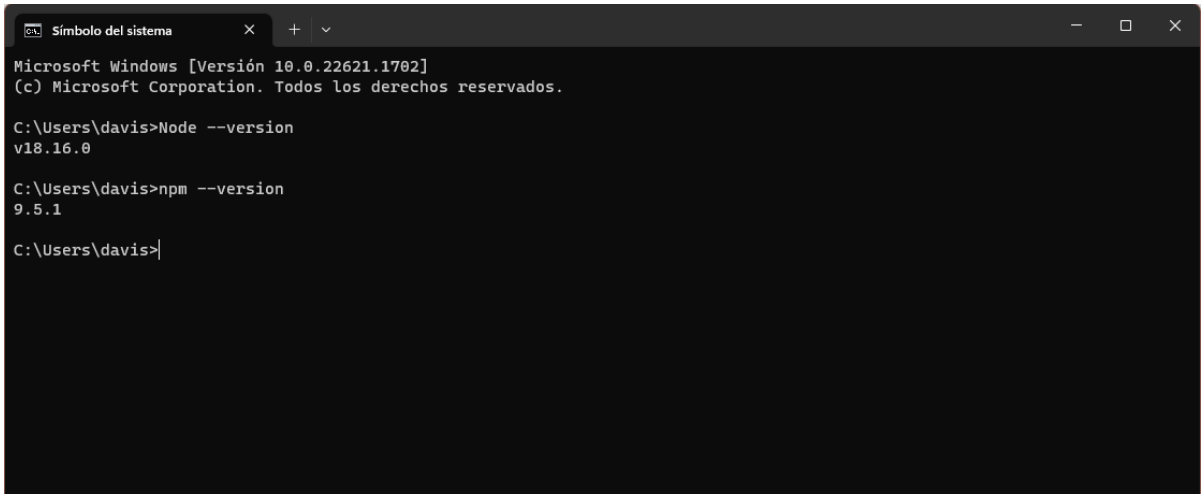

Figura 21. Verificación de instalación de Node y npm
Fuente: propia

Configuración de React js: fue importante instalar primero Node Js ya que este posee lo paquetes necesarios de npm para instalar React.

Primero se creó un nuevo proyecto en React con *create React app* que es una herramienta que configura automáticamente un entorno desarrollo en React para facilitar la configuración inicial y eliminar la necesidad de configuraciones complicadas.

Para crear un nuevo proyecto React se utiliza el comando *npx create-react-app* ***nombre-del-proyecto,*** esto inicio la instalación y configuración del proyecto. A continuación, se mostrará el proceso gráficamente.

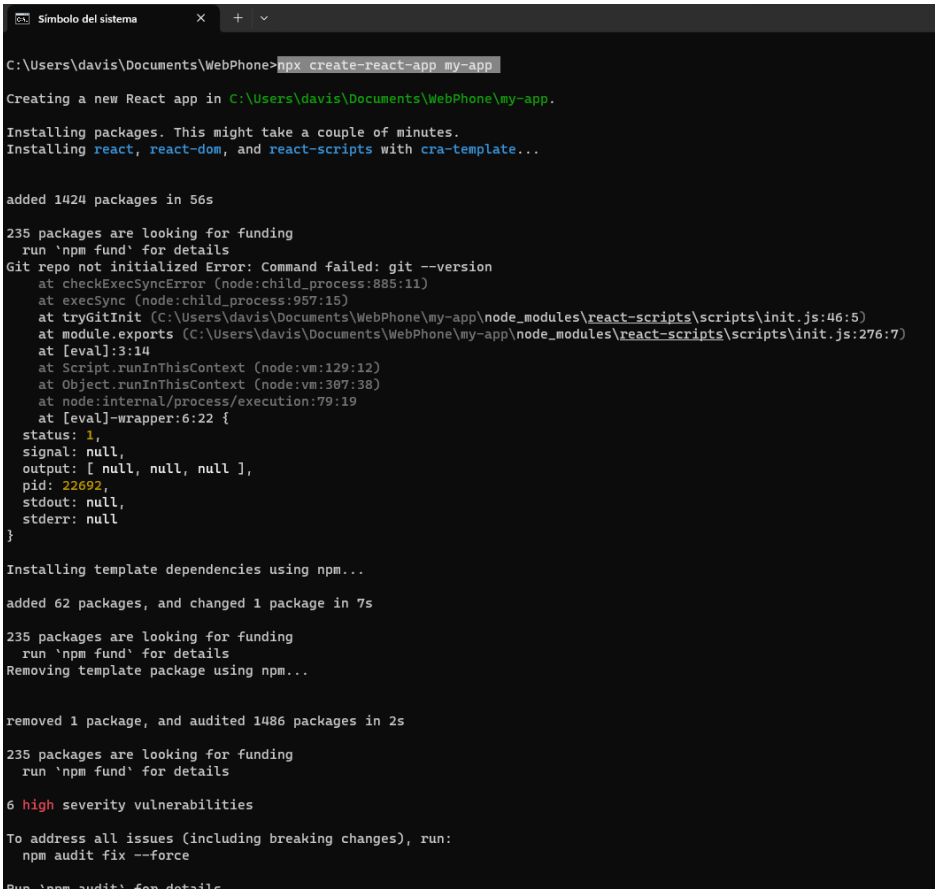

Figura 22. Creación de un nuevo proyecto en React
Fuente: propia

En última instancia, se procedió a acceder a la carpeta del proyecto de React que había sido creado mediante el uso de Create React App. Esto se logró mediante el comando *cd nombredelproyecto*. Una vez en el directorio del proyecto, se inició el servidor de desarrollo de React ejecutando el comando *npm start*. Al ejecutar este comando, automáticamente se abrió la aplicación de React en el navegador predeterminado, y se accedió a ella a través de la dirección *http://localhost:3000*. Esta acción permitió visualizar y probar la aplicación en tiempo real, proporcionando un entorno de desarrollo óptimo.

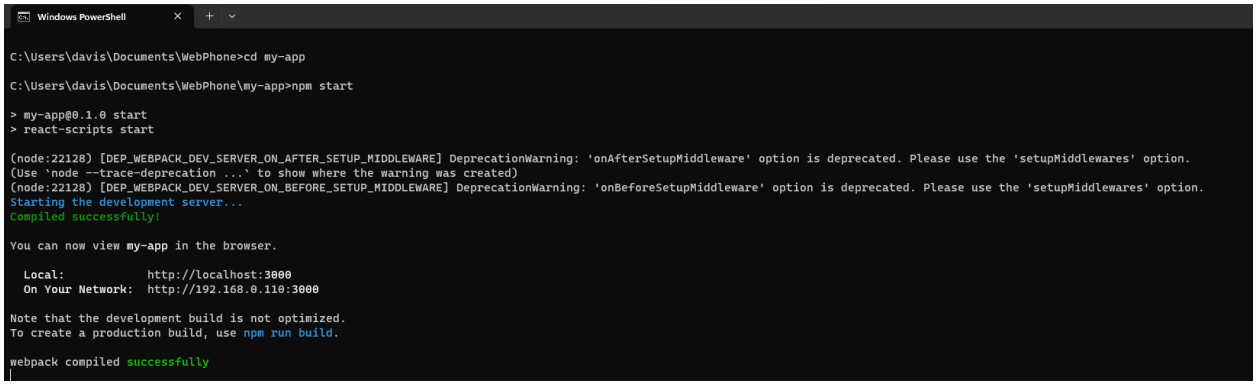

Figura 23. Iniciando el servidor para entorno de desarrollo
Fuente: propia

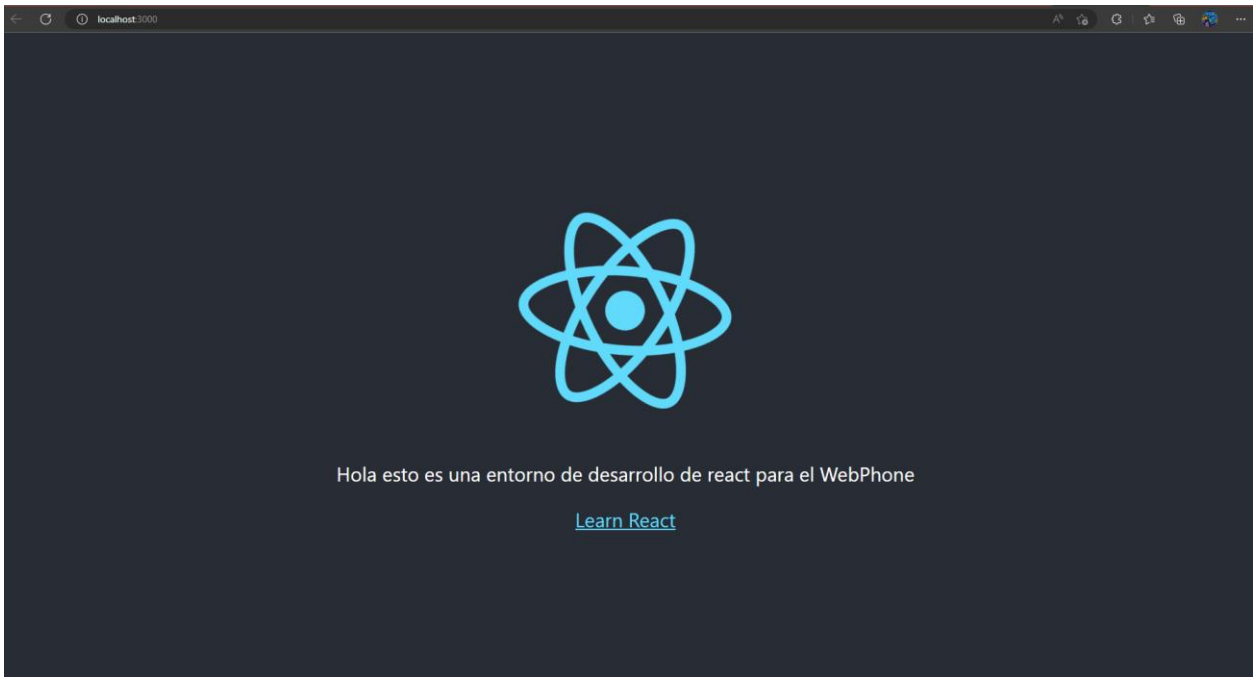

Figura 24. Ejecución de la aplicación
Fuente: propia

Instalación de dependencias adicionales: se instaló paquetes adicionales, para mejorar la calidad del desarrollo en el Fronted, entre las principales JsSIP para instalar esta librería se usó el comando *npm install jssip –save.* Luego de la instalación de la librería se importa en el componente con la siguiente línea de código *import JsSIP from 'jssip', de* esta forma podemos empezar a usar las clases y atributos que posee esta biblioteca.

Para manejar estilos rápidos e interfaces responsivas se utilizó Bootstrap que se instaló con el siguiente comando *npm install Bootstrap.* y se importa los elementos necesarios para la utilización de sus componentes, como botones o formularios.

Con estas acciones, se logró integrar la biblioteca JsSIP al proyecto de React, proporcionando las herramientas necesarias para establecer y gestionar llamadas de voz de manera eficiente.

6.12.2 Funciones para realizar y recibir llamadas de voz en JsSIP.

La utilización de JsSIP en el desarrollo del Webphone permitió aprovechar las características y beneficios de WebRTC, como la transmisión de audio tiempo real, la seguridad integrada y la compatibilidad con múltiples navegadores web. La biblioteca JsSIP proporcionó una interfaz fácil de usar y una amplia gama de métodos y eventos que facilitaron la implementación de las funcionalidades requeridas para el Webphone.

A continuación, se mostrará las funciones y eventos necesarios que se usaron el desarrollo del WebPhone.

Inicio de sesión de usuario: Se implementó un formulario de inicio de sesión que solicitaba al usuario su nombre de usuario y contraseña.

Se utilizó la función login de JsSIP para autenticar al usuario y establecer la conexión con el servidor SIP, esto es como conocido como “register”.

```
var socket = new JsSIP.WebSocketInterface('wss://sip.myhost.com');
var configuration = {
  sockets  : [ socket ],
  uri      : 'sip:alice@example.com',
  password : 'superpassword'
};
```

Figura 25. Agente de usuario para registrarse
Fuente: propia
Nota: por privacidad en la empresa no se muestra los parámetros de configuración reales

El agente de usuario JsSIP desempeña un papel fundamental en el framework de JsSIP. Este agente representa al cliente SIP que está vinculado a una cuenta SIP específica. En concreto, se encuentra definido en la clase JsSIP.UA como se ve en la *figura 25*.

En la primera línea, se crea una instancia de la interfaz WebSocketInterface de JsSIP. Se proporciona como argumento la URL del servidor SIP al que se desea conectar, en este caso, 'wss://sip.myhost.com'. Esto indica que se utilizará una conexión segura WebSocket (wss) para comunicarse con el servidor.

El agente de usuario JsSIP requiere un objeto de configuración para su inicialización. Hay algunos parámetros de configuración obligatorios y muchos opcionales. En este objeto, se especifica la configuración necesaria para establecer la conexión SIP.

La propiedad "sockets" es un arreglo que contiene el objeto de interfaz WebSocket creado anteriormente. Indica los tipos de conexiones que se utilizarán para comunicarse con el servidor. En este caso, solo se proporciona la conexión WebSocket.

- La propiedad "uri" define la dirección SIP del cliente. En este ejemplo, se establece como 'sip:alice@example.com', lo que significa que el cliente SIP se identifica con el nombre de usuario "alice" en el dominio "example.com".

- La propiedad "password" especifica la contraseña asociada a la cuenta SIP del cliente. En este caso, se utiliza 'superpassword' como ejemplo de contraseña.

Luego se crea una instancia del agente de usuario JsSIP utilizando la configuración previamente definida.

Figura 26. Agente de usuario
Fuente: propia

La variable *"userAgent"* representa el agente de usuario JsSIP y se crea utilizando la clase JsSIP.UA. Se pasa como argumento el objeto de configuración *"configuration"* que se definió anteriormente.

Una vez creada la instancia del agente de usuario, se utiliza el método *"start()"* para iniciar la funcionalidad del agente. Esto significa que se establecerá la conexión SIP con el servidor y se iniciará el proceso de registro con la cuenta SIP especificada en la configuración.

Marcación y realización de llamadas: Se creó una interfaz de marcación que permitía al usuario ingresar un número de teléfono para realizar una llamada.
Se utilizó la función call de JsSIP para iniciar una llamada saliente hacia el número especificado.

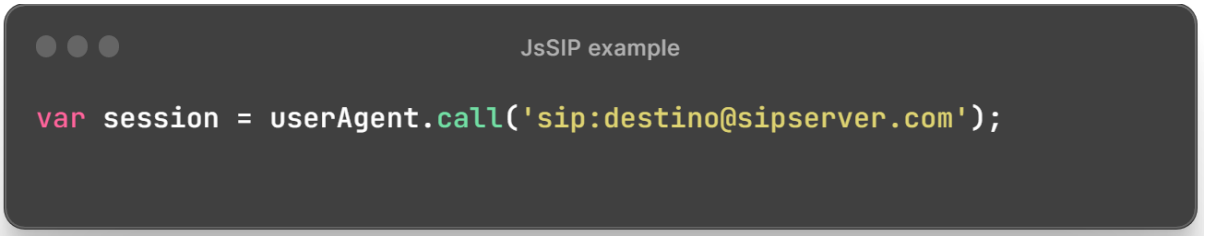

Figura 27. Marcación y realización de llamadas
Fuente: propia

La variable *"session"* representa la sesión de llamada y se crea utilizando el método "call()" del objeto *"userAgent"*. Se pasa como argumento la dirección SIP de destino a la que se desea llamar, en este caso, 'sip:destino@sipserver.com'.

Al llamar al método *"call()"*, se establecerá una conexión de llamada entre el agente de usuario y el destino especificado. Esto iniciará el proceso de establecimiento de la llamada SIP, enviando las señales adecuadas al servidor SIP.

Recepción de llamadas: Se configuró un manejador de eventos para detectar llamadas entrantes. Se utilizó la función answer de JsSIP para responder a la llamada entrante.

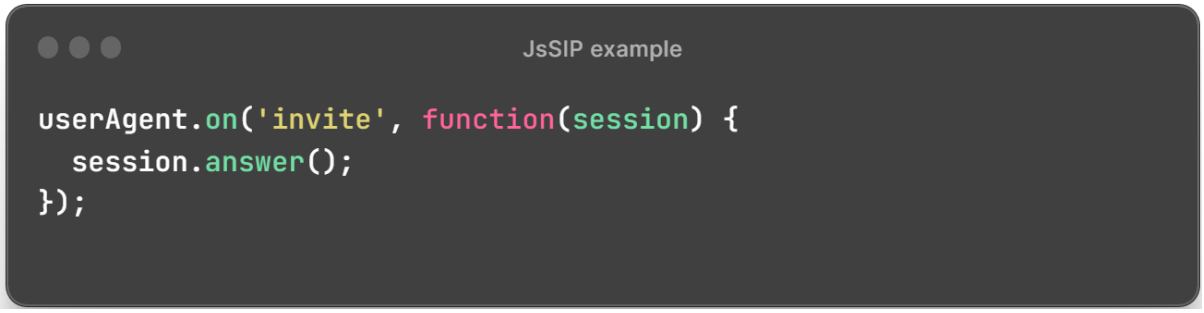

Figura 28. Recepción de llamadas
Fuente: propia

se configura un controlador de eventos para el evento *"invite"* del agente de usuario JsSIP. La función *on()* se utiliza para suscribirse a un evento específico. En este caso, se utiliza para escuchar el evento "invite", que ocurre cuando se recibe una invitación de llamada entrante.

Dentro de la función, se define una acción a realizar cuando se recibe una invitación de llamada. En este caso, se llama al método *answer()* de la sesión recibida como parámetro en la función. Esto significa que cuando se reciba una invitación de llamada, se responderá automáticamente a la llamada.

Silenciar y activar sonido: Se agregaron botones en la interfaz para permitir al usuario silenciar o activar el micrófono durante una llamada. Se utilizó la función mute de JsSIP para

silenciar el micrófono del usuario.

Figura 29. Silenciar Audio
Fuente: propia

Se utiliza el método mute() de la sesión de llamada para silenciar el audio. El objeto *session* representa la sesión de llamada en curso. Al llamar al método *mute()* y pasar como argumento la cadena 'audio', se silencia el audio de la llamada. Esto significa que el sonido del micrófono del lado local no se transmitirá al extremo remoto de la llamada.

Eventos manejados para flujo del WebPhone.

- **Evento 'connected':** Este evento se dispara cuando el usuario se ha conectado con éxito al servidor SIP. Se utilizó para notificar al usuario que la conexión se estableció correctamente.
- **Evento 'disconnected':** Este evento se dispara cuando se pierde la conexión con el servidor SIP. Se utilizó para manejar situaciones en las que se produce una desconexión inesperada.
- **Evento 'newRTCSession'**: Este evento se dispara cuando se inicia una nueva sesión de RTC (Real-Time Communication). Se utilizó para manejar llamadas entrantes y salientes.

Figura 30. Eventos JsSIP
Fuente: propia

6.12.3 Diseño aplicación cliente:

Toda la aplicación fue hecha en React, se estructuró la aplicación utilizando componentes, lo cual facilitó la organización y reutilización del código, todo el diseño de interfaz y responsive fue hecha en Bootstrap. También se hizo uso de los estados de React para gestionar y controlar la información dinámica en los componentes.

Se presenta a continuación el diseño de la aplicación, siguiendo la maquetación definida en los wireframes:

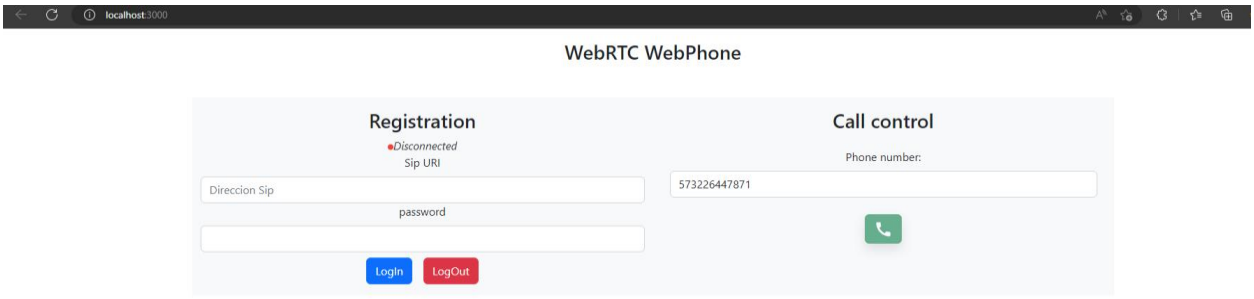

Figura 31. Aplicación Cliente
Fuente: propia

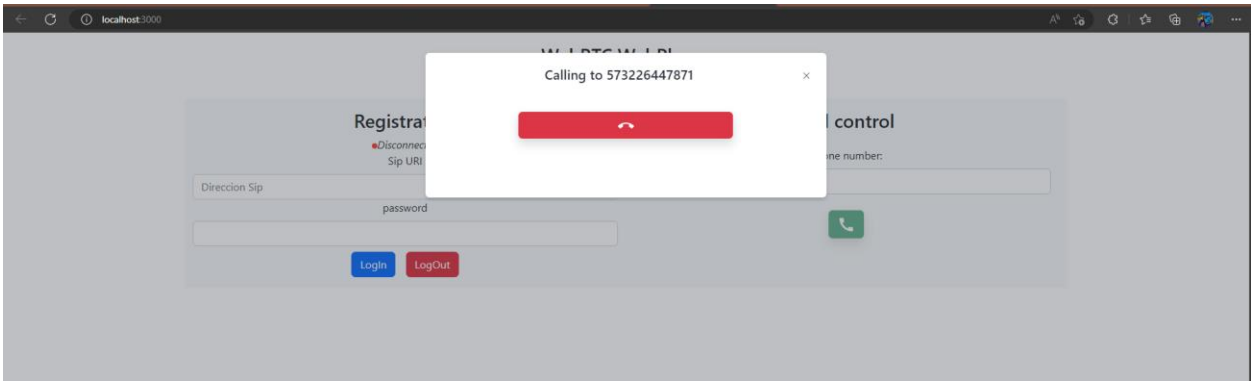

Figura 32. Interfaz de estado llamando
Fuente: propia

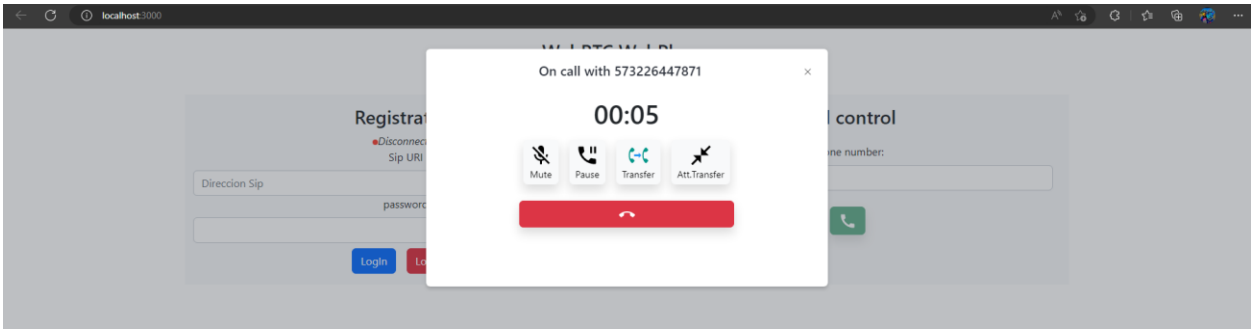

Figura 33. Interfaz llamada conectada
Fuente: propia

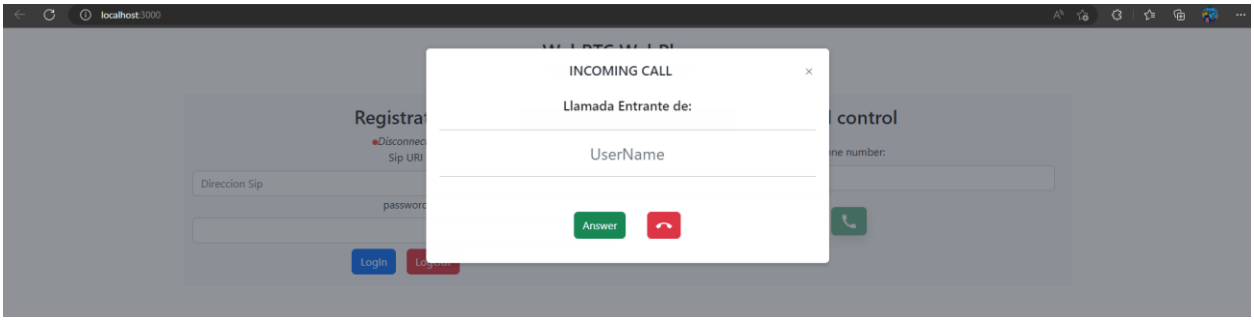

Figura 34. Interfaz Llamada entrante
Fuente: propia

6.13 Pruebas y Resultados del funcionamiento

Para lograr la integración exitosa de los elementos que conforman el prototipo de comunicación, se realizaron varias pruebas para asegurar que el sistema pudiera realizar las siguientes funciones:

Registro de usuarios: Se verificó que los usuarios pudieran registrarse correctamente en el sistema utilizando sus credenciales, como dirección SIP y contraseña.

Iniciación, recepción y cancelación de llamadas: Se comprobó que los usuarios pudieran iniciar llamadas hacia otros usuarios, recibir llamadas entrantes y cancelar llamadas en curso cuando fuera necesario.

Llamadas de WebRTC a WebRTC: Se comprobó la capacidad del sistema para establecer llamadas directas de audio y video entre dos clientes WebRTC. Se verificó que la comunicación fuera bidireccional y de buena calidad.

Llamadas de WebRTC a SIP: Se probó la funcionalidad de realizar llamadas desde un cliente WebRTC a un cliente SIP tradicional. Esto implicó verificar la interoperabilidad entre los dos protocolos y garantizar que la comunicación se estableciera correctamente.

Llamadas de WebRTC a PSTN: Se evaluó la capacidad del sistema para realizar llamadas desde un cliente WebRTC a números de teléfono convencionales en la red telefónica pública (PSTN). Se verificó que se estableciera una conexión adecuada y que las llamadas salientes fueran enrutadas correctamente hacia los números de teléfono especificados.

Registro de usuarios

Se inició conexión con el WebSocket y registro del usuario al servidor SIP con los parámetros de autenticación, si todo está correcto el estado de conexión pasara a “Connected”. La *figura 35* proporciona una representación visual de este estado de conexión del WebPhone, lo que permite al cliente visualmente confirmar que la conexión y el registro en el servidor SIP se han realizado correctamente.

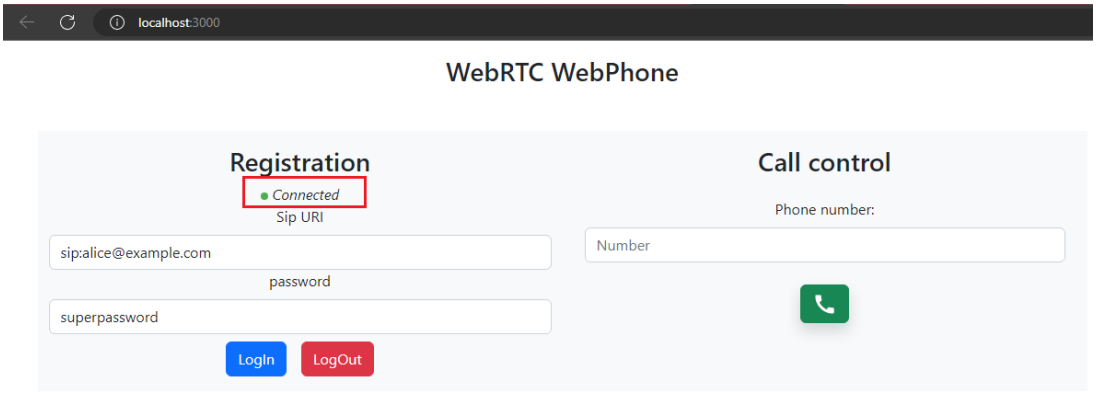

Figura 35. Conexión exitosa
Fuente: propia

Al activar el depurador de JsSIP en la consola del navegador, se puede observar el registro de eventos relacionados con la conexión del WebSocket. En la consola, se mostrará información sobre la conexión exitosa del WebSocket y confirmará que todo está en orden en términos de la conexión establecida con el WebSocket. Esto proporciona una verificación adicional de que la conexión del WebSocket se ha realizado correctamente, ver la figura 36.

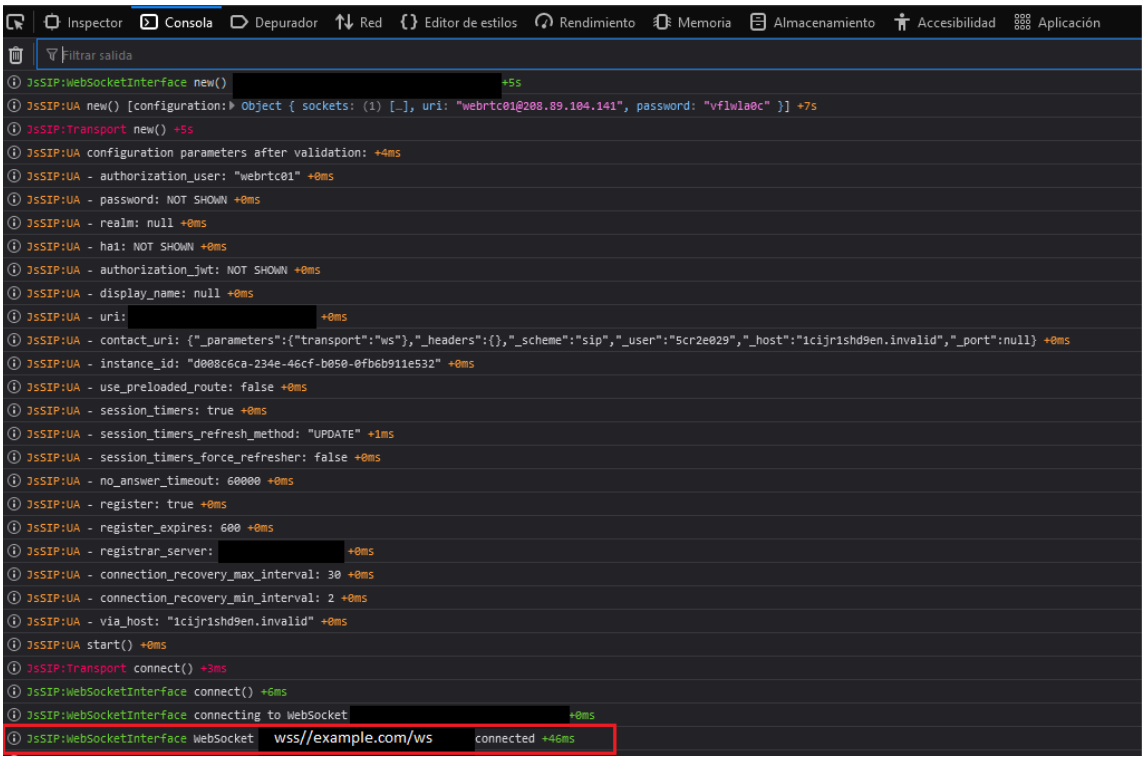

Figura 36. Conexión con el WebSocket
Fuente: propia

Después de que el usuario se registró exitosamente del lado del servidor SIP, se pudo confirmar que el registro se realizó de manera exitosa. Esto indicó que el usuario estaba listo y que la conexión e integración con SIP habían sido exitosas para realizar llamadas. Como se ve en el flujo de la *figura 37.*

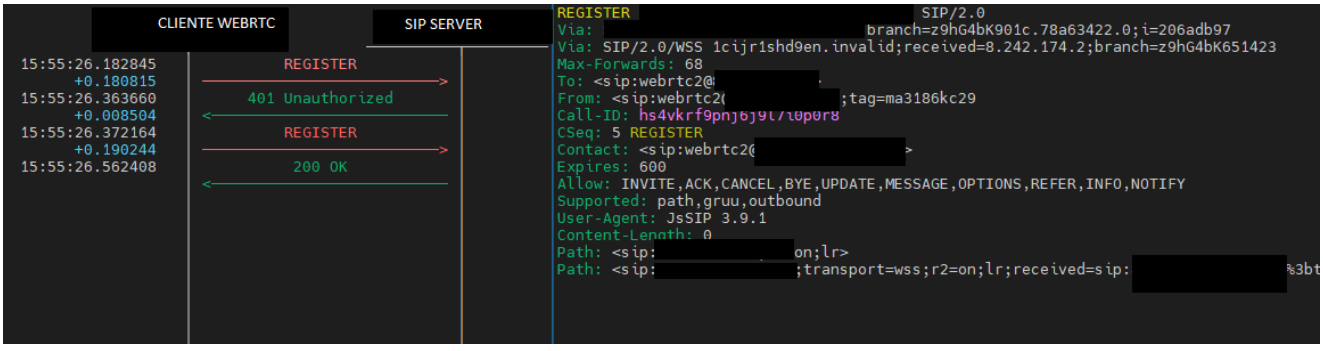

Figura 37. Register en el servidor SIP
Fuente: propia

Al verificar que el usuario se hubiera registrado correctamente del lado del servidor SIP, se confirmó que la autenticación y configuración de la cuenta SIP eran válidas. Esto aseguró que el usuario contara con los permisos y credenciales adecuadas para realizar y recibir llamadas a través del sistema de comunicación.

Esta confirmación del registro exitoso del usuario en el servidor SIP fue un paso importante para garantizar que el WebPhone estuviera preparado para establecer comunicaciones y realizar llamadas con otros usuarios dentro del entorno de SIP.

Iniciación, recepción y cancelación de llamadas

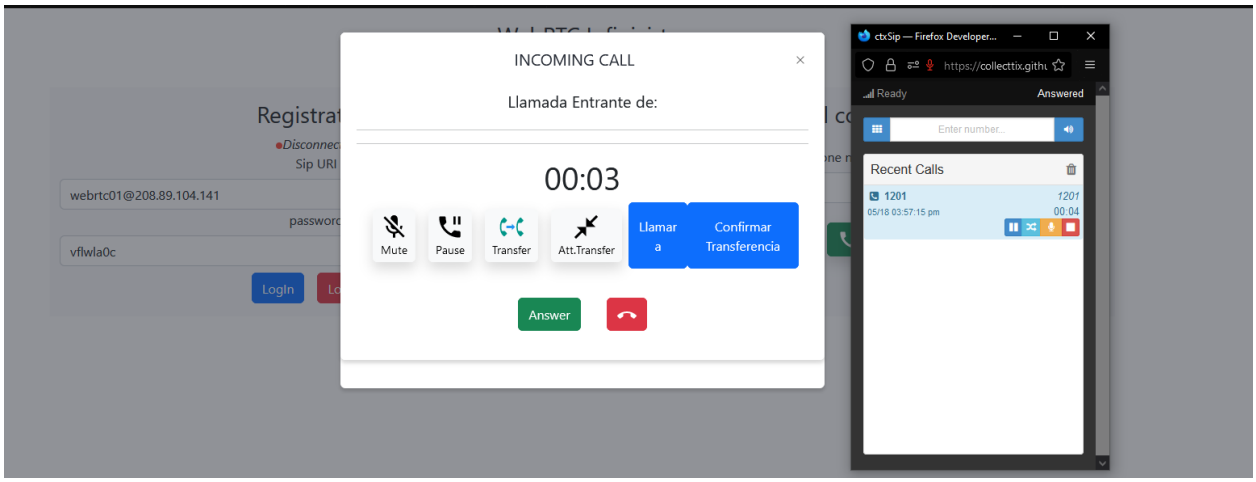

Figura 38. Iniciación de llamada.
Fuente: propia

En la *figura 38*, se puede observar cómo se lleva a cabo la iniciación de la llamada desde el cliente. Esto implica que, después de seleccionar el contacto o marcar el número de destino, se establece una conexión con éxito y la llamada se envía correctamente hacia su destino previsto también en la *figura 38.*

Esto demuestra la eficacia de la funcionalidad de iniciación de llamadas y la capacidad del sistema para establecer conexiones de voz adecuadas.

Durante el proceso de recepción de llamadas, se identificó un problema inicial en el cual las llamadas de PSTN a WebRTC no llegaban correctamente. La razón detrás de este problema fue que la extensión donde se registraba el SIP no contenía un número identificador único, similar a un número de celular. Para solucionar este problema, se realizó una configuración específica en el switch o central telefónica (configuración que hace la empresa). Esta configuración permitió que las llamadas destinadas al número identificador único asociado a la extensión SIP se desbordaran correctamente hacia el WebRTC.

Al configurar el switch para que las llamadas dirigidas a ese número identificador específico fueran enrutadas hacia el WebRTC, se logró solucionar el problema y permitir que las llamadas de PSTN se recibieran correctamente en el WebRTC.

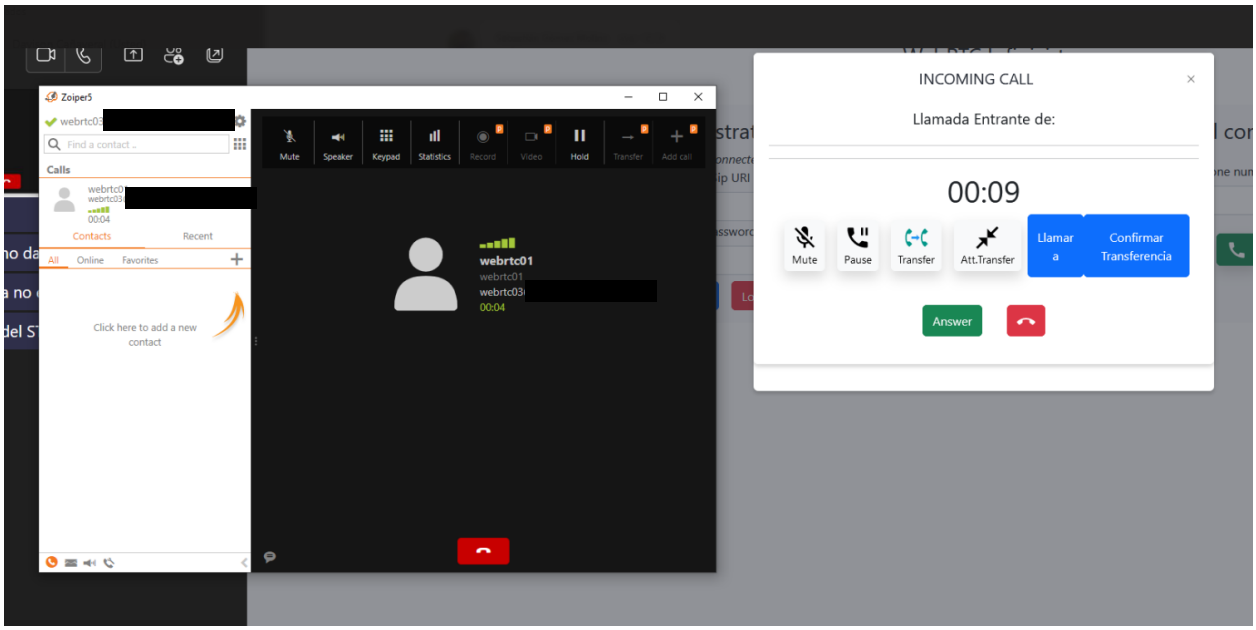

Figura 39. Recepción de llamadas
Fuente: propia

En la *figura 39* se muestra el escenario en el cual se realiza una llamada desde otro softphone hacia WebRTC con el objetivo de verificar el correcto funcionamiento de las llamadas entrantes. Se puede observar que esta prueba resultó exitosa, ya que la llamada llega correctamente al WebRTC y se puede escuchar audio en ambos lados de la comunicación.

Este paso exitoso en las pruebas de verificación asegura que los usuarios del WebRTC puedan recibir y participar activamente en llamadas entrantes desde otros softphones, facilitando así la comunicación bidireccional entre diferentes plataformas o dispositivos.

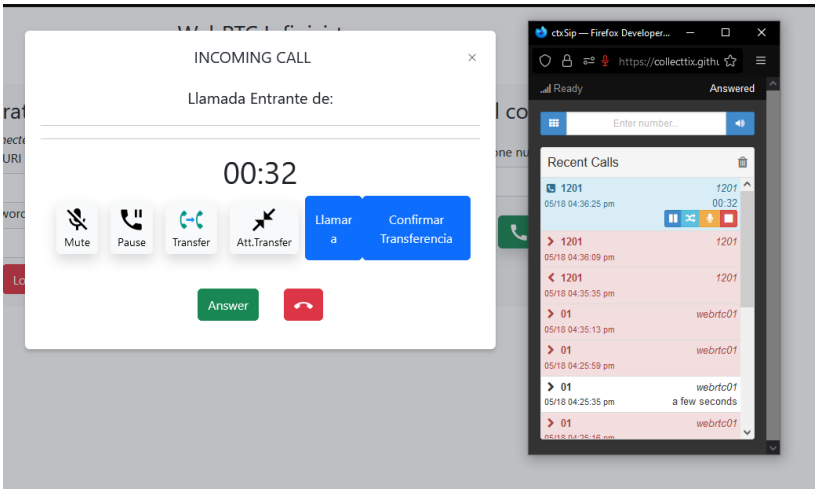

Figura 40. Llamadas entre 2 WebPhone
Fuente: propia

En la figura 40 se representa el escenario en el cual se realiza una llamada entre dos WebPhone, es decir, una llamada de WebRTC a WebRTC. Esta figura sugiere que se estableció una comunicación exitosa entre dos clientes WebRTC, lo que indica que la llamada se realizó con éxito y que ambos extremos pueden intercambiar audio en tiempo real.

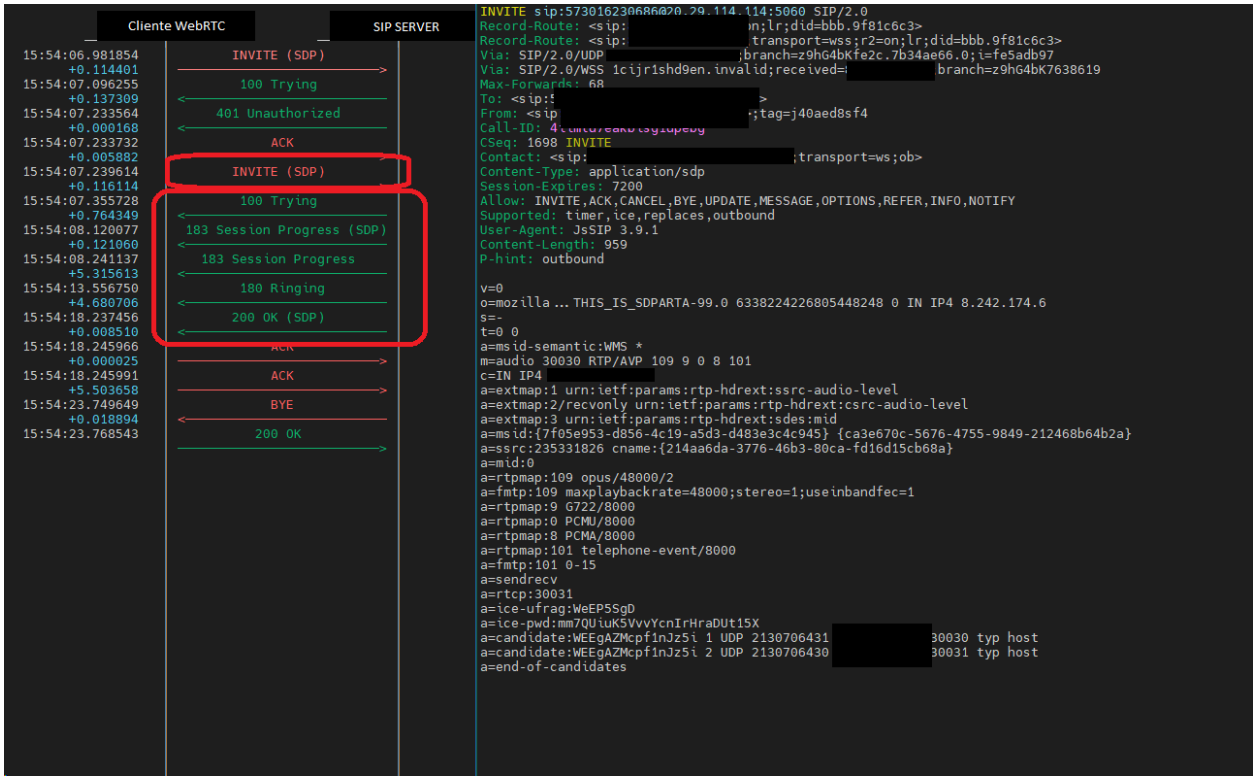

Figura 41. Flujo SDP llamada PSTN
Fuente: propia

En el flujo de la llamada, se pudo evidenciar que el cliente WebRTC enviaba un mensaje INVITE al servidor SIP. Este mensaje contenía información como la dirección del destino al que se quería llamar, los codecs de audio utilizados y otros parámetros relevantes para la llamada.

Una vez que el servidor SIP recibió el mensaje INVITE, respondió al cliente con un código de respuesta provisional "100 Trying". Este código indicaba que el servidor había recibido la solicitud y la estaba procesando. Si todo iba bien y la llamada podía establecerse, el servidor SIP enviaba un mensaje de respuesta final "200 OK" al cliente WebRTC. Este código confirmaba que la llamada se había conectado correctamente y que se había establecido una sesión en progreso. Al ver el "200 OK", se podía confirmar que la llamada estaba conectada debidamente y que la sesión se encontraba en progreso.

7. Conclusiones

Durante el transcurso de este proyecto de construcción de un webPhone utilizando WebRTC, se han identificado varias conclusiones importantes. Estas conclusiones se basan en los aspectos sobresalientes y los aportes generados a nivel local, nacional e internacional, a partir de la investigación realizada.

En primer lugar, se destaca el potencial de WebRTC como una tecnología que permite la transmisión de audio y video en tiempo real a través de la web. Su capacidad para comunicarse directamente desde el navegador web, sin necesidad de descargar aplicaciones externas, lo convierte en una herramienta accesible y conveniente para los usuarios.

Además, se ha constatado que WebRTC ofrece numerosas posibilidades de aplicación, no solo en el ámbito de las llamadas y videollamadas, sino también en la transmisión de contenido multimedia en tiempo real. Esto abre nuevas oportunidades tanto en el campo de las comunicaciones personales como en el ámbito empresarial, donde las conferencias virtuales y la colaboración en tiempo real son cada vez más relevantes.

A nivel local, se puede destacar el impacto que la implementación de un webPhone basado en WebRTC puede tener en las empresas y organizaciones. Al proporcionar una solución de comunicación versátil y accesible, puede mejorar la productividad y la eficiencia de las comunicaciones internas y externas.

A nivel nacional e internacional, se observa que WebRTC es una tecnología ampliamente adoptada y compatible con múltiples navegadores web y sistemas operativos. Esto implica que el alcance y la aplicabilidad del proyecto no se limitan a un entorno específico, sino que pueden tener un impacto a nivel global.

8. Recomendaciones

Basado en los aspectos evidenciados durante el desarrollo del proyecto y considerando oportunidades de mejora se pueden hacer las siguientes recomendaciones:

8.1 Explorar la integración con otras tecnologías: Aunque WebRTC en sí mismo es una tecnología poderosa, se pueden investigar y explorar oportunidades de integración con otras tecnologías emergentes. Esto podría incluir la integración con inteligencia artificial, realidad virtual o aumentada, para mejorar la experiencia de usuario y expandir las funcionalidades del webPhone.

8.2 Mejorar la seguridad: Dado que la comunicación en tiempo real a través de la web implica el intercambio de datos sensibles, se recomienda enfocarse en fortalecer las medidas de seguridad. Esto puede incluir la implementación de cifrado de extremo a extremo, autenticación de usuarios y protección contra ataques cibernéticos.

8.3 Optimizar el rendimiento: A medida que el proyecto se expanda y se utilice en escenarios de mayor escala, es importante optimizar el rendimiento del webPhone. Esto implica realizar pruebas exhaustivas de carga, monitorear el consumo de recursos y realizar ajustes en la infraestructura para garantizar una experiencia fluida y de alta calidad.

8.4 Investigar sobre casos de uso específicos: Aprovechar el potencial de WebRTC implica investigar y comprender las necesidades específicas de diferentes sectores o industrias. Se pueden explorar casos de uso particulares, como telemedicina, educación en línea, servicios de atención al cliente, entre otros, para adaptar el webPhone y desarrollar soluciones especializadas.

8.5 Fomentar la colaboración y el código abierto: Considerando que WebRTC es una tecnología de código abierto, se recomienda fomentar la colaboración con la comunidad de desarrolladores. Participar en proyectos de código abierto relacionados con WebRTC, contribuir con mejoras y compartir conocimientos puede impulsar el avance de la tecnología y generar oportunidades de aprendizaje y networking. (Google, 2023)

9. Referencias bibliográficas

3CX. (s.f.). *3CX*. Obtenido de https://www.3cx.es/voip-sip/que-es-webrtc/

A. S, T., & D, W. (2012). *Redes de computadoras. Pearson Educación.*

Álvarez, F. (18 de 05 de 2020). *Media Source*. Obtenido de https://www.mediasource.mx/blog/navegadores-de-internet

B. Johnston, A., & C. Burnett, D. (2012). *WebRTC: APIs and RTCWEB Protocols of the HTML5 Real-Time Web.*

Clark, D. D. (s.f.). Laboratorio de informática de MIT. 50.

CloudTalk. (2023). *CloudTalk*. Obtenido de https://www.cloudtalk.io/es/blog/como-conseguir-un-numero-voip-guia-completa-2023/

Cobos, E., Pauta Cuji, E. I., & Stalyn, A. (3 de 12 de 2016). *repositorio.ug.edu.ec.* Obtenido de http://repositorio.ug.edu.ec/handle/redug/17945

Gesditel. (10 de 2020). *Gesditel*. Obtenido de https://gesditel.es/webrtc/

Gobierno de Colombia. (2021). *Ministerio de tecnologías de la Información y Comunicaciones*.

Google. (2021). *Google Academy.*

hmong.es. (s.f.). Obtenido de https://hmong.es/wiki/JsSIP

IONOS. (07 de 08 de 2020). *IONOS*. Obtenido de https://www.ionos.es

MasIP. (2019). *masip.es*. Obtenido de https://www.masip.es/

Matango, F. (02 de 09 de 2016). *ServerVoip*. Obtenido de http://www.servervoip.com/blog/tag/protocolo-sip-pdf/

Next U. (2020). *Next U*. Obtenido de https://www.nextu.com/blog/que-es-react-js-como-funciona-rc22/#:~:text=React%20es%20una%20biblioteca%20o,para%20las%20interfaces%20de%20usuario.

NFON. (2021). *NFON*. Obtenido de Protocolo SIP usado en voz IP: https://blog.nfon.com/es/sip-que-es#:~:text=SIP%20(Session%20Initiation%20Protocol%20o,manera%20muy%20sencilla%20y%20consistente.

quobis.com. (2021). *quobis*. Obtenido de https://quobis.com/

Rescorla, E. (2021). WebRTC Security.

Rubiano, J. H., Mena, A. F., & Hernández, J. (2014). WebRTC Una nueva tecnología web al servicio de la educación.

Sergiienko, A. (2015). *WebRTC: A Hands-On Guide.*

SOTO, D., MORENO, J., & DIAZ, M. (20 de 04 de 2009). *urbe.edu.* Obtenido de https://www.urbe.edu/info-consultas/web-profesor/12697883/articulos/ensayos/TELEFONIA%20VoIP.pdf

Sredojev, B., Samardzija, D., & Posarac, D. (16 de 05 de 2015). *IEEE Xplore.* Obtenido de https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7160422&isnumber=7160221

Tamayo, M. (2016). *Universidad CLEA.* Obtenido de https://clea.edu.mx/biblioteca/Tamayo%20Mario%20-%20El%20Proceso%20De%20La%20Investigacion%20Cientifica.pdf

Thaler, S. (10 de 2021). *thaler-it.net*. Obtenido de https://thaler-it.net/software%20development/webrtc/understanding_webrtc/#step-1---creating-and-sending-an-sdp-offer

TrueConf. (12 de 02 de 2023). *TrueConf*. Obtenido de https://trueconf.com/es/webrtc.html

Venema, A. (Marzo de 13 de 2022). *liveswitch*. Obtenido de https://www.liveswitch.io/blog/how-can-i-use-sip-with-webrtc

10. Bibliografía

JsSip. (05 de 11 de 2022). *JsSip*. Obtenido de https://jssip.net/

Google. (2023). Obtenido de https://webrtc.org/?hl=es-419

Fabián, P., & Cioccatto, M. (2015). *repo.iua.edu.ar.* Obtenido de https://repo.iua.edu.ar/bitstream/123456789/1129/1/TFG%20-%20Cioccatto%20-%20Pignataro%20-%202015.pdf

Ayala, C. P. (2014). *bibdigital.* Obtenido de https://bibdigital.epn.edu.ec/bitstream/15000/8628/4/CD-5801.pdf (Meta, 2023)

Meta. (2023). *React*. Obtenido de https://react.dev/