

Trabajo de Fin de Grado

Grado en Ingeniería Informática

Aplicación segura para asistencia sanitaria con streaming

Secure application for healthcare with streaming

David Pérez Rivero

La Laguna, 5 de septiembre de 2017

D. **María Candelaria Hernández**, con N.I.F. 45.441.714-Q profesor Titular de Universidad adscrito al Departamento de Ingeniería Informática y de Sistemas de la Universidad de La Laguna, como tutor

D. **Alexandra Rivero García**, con N.I.F. 78.646.309-V personal Docente Investigador de Universidad adscrito al Departamento de Ingeniería Informática y de Sistemas de la Universidad de La Laguna, como cotutor

CERTIFICA (N)

Que la presente memoria titulada:

“Aplicación segura para asistencia sanitaria con streaming.”

ha sido realizada bajo su dirección por D. **David Pérez Rivero**, con N.I.F. 54.116 134-D.

Y para que así conste, en cumplimiento de la legislación vigente y a los efectos oportunos firman la presente en La Laguna a 5 de septiembre de 2017.

Agradecimientos

A mi tutora Dña. María Candelaria Hernández Goya, y a mi cotutora, Dña. Alexandra Rivero García, por su dedicación, apoyo, motivación e interés durante la realización de este proyecto y por que saliera hacia adelante.

A mi familia y amigos por haberme apoyado en cada momento.

Licencia



© Esta obra está bajo una licencia de Creative Commons Reconocimiento-NoComercial-CompartirIgual 4.0 Internacional.

Resumen

Hoy en día, la tecnología puede ayudar a socorrer víctimas en casos de accidentes o situaciones de emergencias. Por ello, el objetivo de este trabajo está centrado en la asistencia sanitaria de pacientes en situaciones que requieran asistencia mediante video en streaming con un personal médico en el otro lado.

Los avances tecnológicos de los últimos años, como los protocolos para el vídeo en streaming en tiempo real, los protocolos de seguridad, entre otros, han sido enfocados, principalmente, al mundo de las redes sociales en lugar de darle un uso más útil y necesario para el día a día, como puede ser el mundo del m-health.

Es por ello, por lo que se ha realizado un sistema compuesto por tres componentes principales: la aplicación móvil, el servidor web y el puesto web.

La aplicación móvil está desarrollada para Android y su función principal será la comunicación bidireccional entre el usuario (persona que vaya a socorrer) y el personal médico mediante video y audio en streaming. Mediante la respuesta por parte del usuario a una serie de preguntas que le hará el personal sanitario, llamada triage, se conocerá el estado de salud del paciente y las maniobras a aplicar en cada momento por el usuario de la aplicación, mediante imágenes o videos explicativos que recibirá.

El servidor web se encargará de dar soporte al puesto web y ha sido desarrollado en NodeJS. Contiene, además, la capa de seguridad de la aplicación. Este componente también contendrá el servidor que da soporte al streaming.

El puesto web será utilizado por el personal médico que asistirá e indicará las maniobras que ha de aplicarse al accidentado. Las maniobras para aplicar serán enviadas según las respuestas que vaya dando el usuario a una serie de preguntas que irá trasladando el personal médico. Además, al estar en contacto mediante video y audio, el personal médico tiene una visión más real del paciente.

Todas las comunicaciones se harán bajo una capa de seguridad HTTPS, ya que se trata de información sensible y necesita ser protegida debido a las normativas de seguridad actuales.

Palabras clave: m-Health, asistencia sanitaria, Android, NodeJS, HTTPS, streaming

Abstract

Today, technology can help victims in accidents or emergencies. Therefore, the objective of this work is focused on the assistance of patients in situations that require aid with video streaming with a medical staff on the other side.

The technological advances of recent years, such as protocols for video in real time streaming, security protocols, among others, have been mainly focused on the scenario of social networks instead of giving a more useful and necessary everyday use, as the world of m-health can be.

For this reason, the proposed system has been made up of three main components: a mobile application, a web server and a web site.

The mobile application is developed for Android and its main function will be the bidirectional communication between the user (person to be assisted) and the medical staff through video and audio streaming. Through the response by the user to a series of questions that will be asked by the health staff (triage) the patient's health status and the maneuvers to be applied at any time by the user of the application will be determined, through images or explanatory videos received.

The web server will be in charge of supporting the web post and has been developed in NodeJS. It also contains the security layer of the application. This component will also contain the server that supports streaming.

The web site will be used by the medical staff who will attend and indicate the maneuvers to be applied to the injured. The maneuvers to apply will be sent according to the answers that the user is giving to a series of questions that will be transferred by the medical staff. In addition, being in contact through video and audio, the medical staff has a real view of the patient.

All communications will be made under an HTTPS security layer, as this is sensitive information and needs to be protected due to current security regulations.

Keywords: *m-Health, healthcare, Android, NodeJS, HTTPS, streaming*

Índice

Capítulo 1. Introducción	1
1.1 Motivación	1
1.2 Estado del arte	1
1.3 Conceptualización de la propuesta	4
1.4 Estructura de la memoria	5
Capítulo 2. Tecnologías	7
2.1 Aplicación móvil	7
2.2 Servidor web	9
2.3 Aplicación web	9
2.4 Comunicaciones	10
2.5 Base de datos	12
Capítulo 3. Servicio web	14
3.1 Características generales	14
3.2 Estructuras del servicio	14
3.2.1 Base de datos	15
3.2.2 Rutas	18
3.3 Implementación de las vistas	19
3.3.1 Index.ejs	20
3.3.2 Signup.ejs	20
3.3.3 Patients.ejs	21
3.3.4 Main.ejs	23
3.3.5 Notfound.ejs	25
Capítulo 4. Aplicación móvil	26
4.1 Características generales	26
4.2 Servicios integrados	26
4.2.1 Servicio de localización de Google	27
4.2.2 Firebase API	27
4.3 Implementación de las actividades	28
4.3.1 Login Activity	28
4.3.2 Signin Activity	29

4.3.3	Start Activity.....	29
4.3.4	Main Activity	29
Capítulo 5. Sistema global		33
5.1	Esquema general.....	33
5.2	Seguridad	34
5.2.1	HTTPS.....	34
5.2.2	Firebase.....	35
5.2.3	Streaming.....	35
5.2.4	Tokens.....	36
5.3	Comunicaciones	37
5.3.1	HTTPS.....	38
5.3.2	Streaming.....	38
5.3.3	Wowza	38
5.3.4	FFMPEG.....	38
5.3.5	Firebase Messaging	39
Capítulo 6. Presupuesto		40
6.1	Costes de personal	40
6.2	Costes de software.....	41
6.3	Coste total.....	41
Capítulo 7. Conclusiones y trabajos futuros		42
7.1	Conclusiones	42
7.2	Trabajos futuros	42
Capítulo 8. Conclusions and future works		44
8.1	Conclusions	44
8.2	Future works	44
Bibliografía		46

Índice de figuras

Figura 1. Esquema del triage utilizado en el sistema	2
Figura 2. Esquema del sistema	4
Figura 3. Firebase Cloud Messaging	7
Figura 4. Firebase Authentication	7
Figura 5. Firebase Storage	8
Figura 6. Firebase Database	8
Figura 7. Servicio localización Google	8
Figura 8. Express.js y NodeJS	9
Figura 9. Firebase	9
Figura 10. Google Maps API	10
Figura 11. VideoJS	10
Figura 12. Wowza Streaming Engine	11
Figura 13. HTTPS	11
Figura 14. FFMPEG	12
Figura 15. Estructura de la estructura de datos	16
Figura 16. Vista de index.ejs	20
Figura 17. Vista de signup.ejs	21
Figura 18. Vista responsiva de patients.ejs	21
Figura 19. Vista de patients.ejs	22
Figura 20. Vista de main.ejs	23
Figura 21. Vista main.ejs mostrando la ubicación del paciente	23
Figura 22. Vista main.ejs responsiva	24
Figura 23. Vista notfound.ejs	25
Figura 24. Activity login	29
Figura 25. Activity Main	¡Error! Marcador no definido.
Figura 26. Activity main mostrando maniobra	¡Error! Marcador no definido.
Figura 27. Activity main mostrando estado final	¡Error! Marcador no definido.
Figura 28. Esquema general del sistema	33

Índice de tablas

Tabla 1. Costes de personal.....	40
Tabla 2. Costes de software	41
Tabla 3. Coste total	41

Capítulo 1.

Introducción

1.1 Motivación

Las principales motivaciones que han impulsado este proyecto han sido: innovar y aplicar tecnologías ya existentes en otros campos al ámbito de la m-Health, y, principalmente, facilitar la vida a las personas en situaciones de emergencia.

Como se explica en el resumen inicial de esta memoria, las tecnologías relacionadas con el streaming de video han sido principalmente empleadas para la seguridad, como cámaras IP de vigilancia; y para las redes sociales. Ya que éste es un campo en constante auge y que reúne a una gran cantidad de usuarios, se innova y actualmente se permite a los usuarios de las redes sociales emitir videos en directo y que puedan ser seguidos por otros usuarios de todo el mundo.

El trabajo que aquí se presenta, muestra una novedad respecto a la utilización de las nuevas tecnologías referentes a la transmisión de video en streaming en los últimos años, aplicándose al mundo del m-Health, en concreto, a la asistencia sanitaria en casos de accidentes o situaciones de emergencia. De esta manera, el video en streaming se aplica para salvar vidas o auxiliar a una persona en situación de riesgo, y deja de ser solamente utilizado para compartir contenido con otros usuarios o vigilar una propiedad.

Además de extender estas tecnologías a otros campos, la principal motivación de este proyecto es crear un sistema capaz de hacer más sencilla la vida a las personas, ya que facilitaría la reacción a ciertas situaciones que requieren de una asistencia médica en un determinado momento.

1.2 Estado del arte

En la actualidad, existen algunas aplicaciones de asistencia médica, pero son en mayor medida de ámbito didáctico. La principal función que éstas desempeñan es la de enseñar al usuario a responder a distintas situaciones en las que debe socorrer o asistir a una persona accidentada, mediante juegos o imágenes y videos explicativos. Además de enseñar, las más avanzadas de las aplicaciones testeadas incluyen los números de teléfono y la ubicación de los hospitales y servicios de urgencias más cercanos.

A continuación, se detallan un par de aplicaciones que se asemejan con el proyecto que aquí se presenta, junto con sus principales características y funciones:

- Primeros Auxilios – Cruz Roja (American Red Cross): Esta aplicación está desarrollada para la Cruz Roja Americana y es de las más completas. Proporciona cuestionarios interactivos para aprender y saber manejar diferentes casos de primeros auxilios que se nos puedan presentar. Además, incorpora una lista de hospitales cercanos, así como la posibilidad de llamar al servicio médico de emergencia.
- Emergency First Aid/Treatment (Phoneflips): La aplicación desarrollada por Phoneflips, es bastante útil ya que basta con buscar el síntoma o la alergia que presente el paciente y nos indica los pasos a seguir para asistirlo.

El sistema de este proyecto propone la mejora de los sistemas y aplicaciones actuales, ya que no tendrá que ser el usuario de la aplicación quien conozca o busque las maniobras necesarias a aplicar en cada situación, sino que será un personal médico capacitado quien envíe al usuario una imagen o un vídeo con las acciones a realizar o se las comunique mediante lenguaje oral. Al estar conectado mediante video en streaming, el médico podrá corregir o dar información extra de la que aparecerá en las imágenes de la aplicación, así como percibir de una forma más clara el estado del paciente.

El estado del paciente se obtiene a partir de un triage. El triage utilizado en este sistema es el triage START [1] (Simple Triage And Rapid Treatment) o triage simple y tratamiento rápido, y es uno de los métodos de clasificación de heridos más habituales en los servicios de emergencias tanto españoles como internacionales.

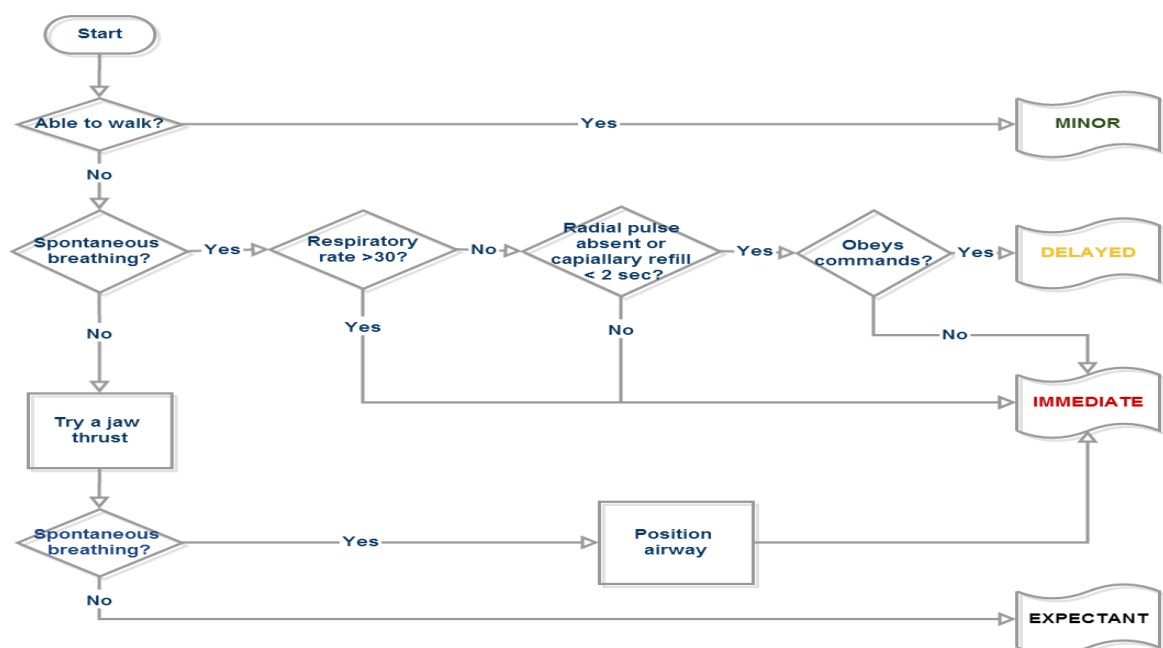


Figura 1. Esquema del triage utilizado en el sistema

Tras la realización del triage, mediante la respuesta a una serie de preguntas como podemos observar en la *Figura 1*, da como resultado el nivel de prioridad que agrupa a los pacientes por criterio de gravedad. Los estados posibles son:

- Prioridad 1 / Color Rojo: Gravedad extrema. Requiere tratamiento y estabilización inmediata.
- Prioridad 2 / Color Amarillo: Sin riesgo vital inmediato. El paciente debe ser atendido en, como máximo, una hora.
- Prioridad 3 / Color Verde: Puede esperar sin riesgo vital. La asistencia puede demorarse hasta una duración entre cuatro y seis horas.
- Prioridad 4 / Color Negro: El usuario ha fallecido.

La prioridad de la asistencia se asigna en función del estado de funciones vitales básicas, valorando cuatro aspectos: la marcha, la respiración, la circulación (mediante el pulso radial o relleno capilar) y el estado mental.

Otras aplicaciones pensadas para la asistencia, pero, esta vez, de personas que realizan actividades en montañas como escalada, esquí... son las siguientes:

- Help me – SOS International
- Alpify

La principal función de estas aplicaciones es el envío de las coordenadas del usuario que está en una situación de emergencia, como puede ser una avalancha, un accidente, o simplemente que se ha perdido para que pueda ser asistido o rescatado.

Por ello, siguiendo la línea de estas aplicaciones, la aplicación Android desarrollada enviará la ubicación para, en los casos más graves, el personal sanitario pueda enviar un servicio de emergencias al lugar donde se encuentre el paciente.

1.3 Conceptualización de la propuesta

El sistema desarrollado, como podemos ver en el esquema que aparece debajo (Figura 2), se divide en cuatro componentes principales que se resumen a continuación y se detallarán en los siguientes capítulos de esta memoria:

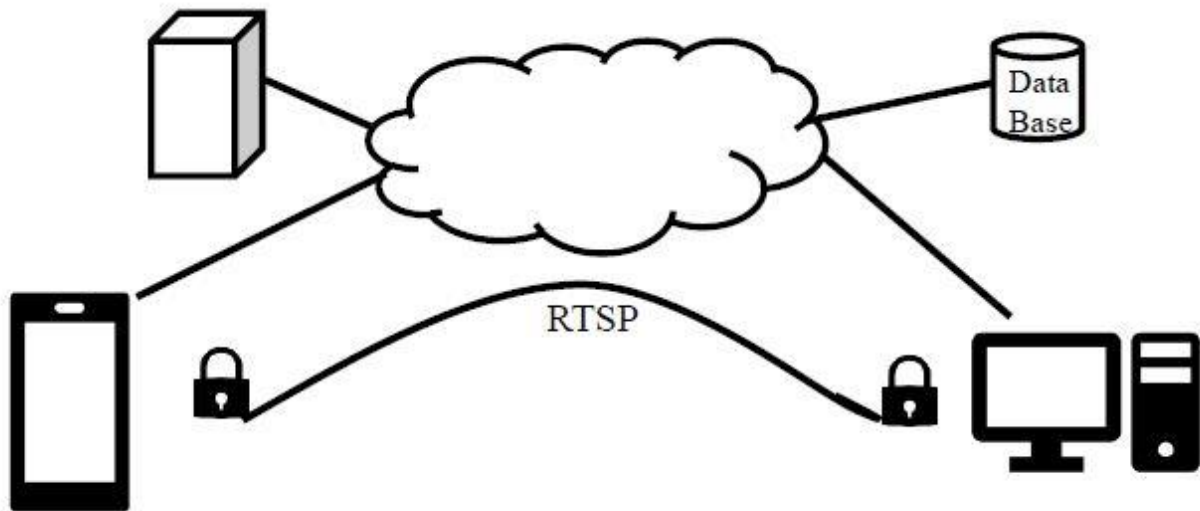


Figura 2. Esquema del sistema

Siguiendo un orden de derecha a izquierda y de abajo hacia arriba, en primer lugar, se encuentra la aplicación desarrollada para Android [2], que será utilizada por el usuario que desea asistir a un accidentado o a él mismo. El diseño es usable y fluido, con distintos botones y campos de textos que informan sobre el uso de la misma. Se ha evitado sobrecargar la aplicación de elementos que la hagan engorrosa y/o confundan al usuario sobre las acciones a realizar. Asimismo, en la actividad principal y más importante de la aplicación, el usuario prácticamente no hace uso de la pantalla, solamente para comprobar las respuestas que está dando al personal médico al triage que está realizando, para ver las maniobras que tiene que aplicar al paciente, así como el estado final del asistido al concluir el triage. Para el uso de la aplicación es necesario tener el token de sesión que se obtendrá al hacer el login mediante un usuario y contraseña que previamente se ha registrado en la base de datos.

En segundo lugar, la aplicación web (desarrollada con JavaScript, CSS y HTML) ha sido diseñada con el principal objetivo de ofrecer la asistencia a situaciones de urgencias de los usuarios de la aplicación móvil por parte del personal médico, realizando una serie de preguntas (triage) para conocer el estado del paciente. El personal médico, además de poder ver en tiempo real al paciente gracias al video en streaming y poder comunicarse con el usuario de la aplicación a través de audio, tiene la opción de conocer la ubicación del éste para, en los casos más graves, mandar un

servicio médico de urgencias que pueda asistirlo de manera física. La conexión con el servidor, del que hablaremos a continuación, se realiza bajo una capa intermedia de seguridad gracias al protocolo HTTPS [3]. De esta manera, la información confidencial y crítica de este sistema de m-health queda protegida.

Como ya se ha comentado, a estos dos primeros componentes del sistema los une la conexión de video y audio en streaming bajo el protocolo RTSP [4] de forma segura.

El tercer componente es la nube, en la que se encuentra tanto el servidor como la base de datos. Siguiendo el orden anterior, en primer lugar, tenemos el servidor. El servidor está desarrollado en NodeJS [5] y da soporte a la aplicación web desde la que se conectará el personal médico, bajo el protocolo HTTPS. Contiene las distintas vistas y librerías a utilizar por la aplicación web. A continuación, también en la nube, nos encontramos con la base de datos del sistema, alojada en la nube haciendo uso de Firebase. Es una base de datos en tiempo real, que permite el manejo de información, usuarios y almacenamiento. Además, tiene APIs tanto para Android como para el desarrollo web. Es una base de datos no relacional y aporta escalabilidad, flexibilidad y una mayor velocidad al sistema.

Los distintos componentes en los que se divide el sistema serán descritos con mayor detalle en los siguientes capítulos.

1.4 Estructura de la memoria

Este documento ha sido estructurado en ocho capítulos diferenciados.

En primer lugar, se realiza una introducción al sistema donde se define las funciones del mismo, el estado del arte del ámbito tratado y la conceptualización de éste, donde se explican los componentes que lo forman. Seguidamente, se exponen las tecnologías que se han utilizado para el desarrollo de cada una de las distintas partes, así como sus principales implicaciones o funcionalidades.

Puesto que este proyecto engloba distintos tipos de sistemas, a continuación, se definen y detallan cada uno de ellos, sus funciones u objetivos principales y cómo se ha abordado su implementación. De este modo, se incluyen los siguientes capítulos: servicio web (capítulo 3) y aplicación móvil (capítulo 4).

Una vez definidos cada uno de los sistemas implicados, el capítulo 5 detalla cómo se integran los servicios y tecnologías utilizadas, así como la manera de implementar la capa de seguridad añadida para mantener el sistema seguro; y las comunicaciones implementadas de las que hace uso tanto la aplicación móvil como el servicio web.

Como todo proyecto tiene unos costes, el capítulo 6 hace referencia a éste, y se detalla el presupuesto estimado de todo el proceso de desarrollo.

Posteriormente, se exponen las conclusiones de este proyecto, así como futuras implementaciones interesantes a desarrollar (capítulo 7). A continuación, en el capítulo 8, se encuentra la versión en inglés del capítulo 7.

Finalmente, se incluirá la biografía con enlaces y recursos de referencia que se han utilizado durante el proceso de desarrollo del proyecto.

Capítulo 2.

Tecnologías

Para abordar el aspecto tecnológico de este proyecto debemos distinguir los distintos componentes que se mencionaron en el capítulo 1: aplicación, puesto web, servidor, base de datos y comunicaciones.

2.1 Aplicación móvil

La principal tecnología que ha sido utilizada para este proyecto ha sido la API de Firebase [6] para Android. Firebase permite el almacenamiento de archivos como imágenes, el envío y recepción de notificaciones, el control y gestión de usuarios, el uso de bases de datos... Para cada una de las acciones anteriores, existe una biblioteca distinta. A continuación, se detalla cada una de las usadas y su uso:

- La tecnología que permite el intercambio de información entre el servidor y la aplicación móvil es el servicio de notificaciones Push de Firebase (Cloud Messaging). Este servicio está incluido en la API de Firebase (en la biblioteca Firebase-Messaging [7]) y permite la recepción



Figura 3. Firebase Cloud Messaging

tanto de mensajes como de alertas y notificaciones desde un servidor web o desde la propia consola de Firebase a una aplicación móvil. En este caso, se reciben notificaciones Push enviadas desde la aplicación web, que contienen las

respuestas del triage y las maniobras a aplicar al paciente.

- Otra tecnología utilizada en la aplicación móvil es el servicio de



Figura 4. Firebase Authentication

autenticación que presta Firebase en su biblioteca Firebase-Auth [8]. Esta biblioteca permite la gestión y control de usuarios de la

aplicación, tanto la creación, el login, como escuchas para cambios de estado del usuario actual.

- La API de Firebase también es utilizada para la descarga de imágenes en



la nube (Firebase-Storage [9]). Es utilizada para descargar las imágenes de las maniobras a aplicar al paciente y los estados finales tras el triage.

Figura 5. Firebase Storage

- Por último, la biblioteca de la base de datos (Firebase-Database [10]) permite el control de la base de datos que ofrece Firebase. Como principal



característica es que no es relacional, y permite el anidamiento de datos en forma de árbol. Es utilizada para almacenar la información de los

Figura 6. Firebase Database

usuarios en espera, junto a su información como el token (para saber a dónde enviar las notificaciones), la ubicación, y si está siendo atendido o está en espera aún.

Otra tecnología utilizada en la aplicación móvil es el servicio de Google de geolocalización [11] mediante la red móvil. Gracias a este servicio, se obtiene la localización de la forma más precisa posible para saber la ubicación del paciente, necesaria en los casos más graves.



Figura 7. Servicio localización Google

Por último, la tecnología principal del proyecto es la de la librería libstreaming [12]. Es la tecnología que permite enviar el video en streaming desde la aplicación móvil. Esta tecnología será descrita más adelante en el apartado de tecnologías de las comunicaciones (Capítulo 2, Apartado 4).

2.2 Servidor web

Entre las tecnologías para el desarrollo del servidor destacan NodeJS y Express en el proceso de desarrollo de la arquitectura cliente-servidor.

La primera, NodeJS, constituye un entorno de ejecución para JavaScript multiplataforma de código abierto, destacando como pilar de programación ECMAScript asíncrono. Destaca por su utilidad en el campo del desarrollo de aplicaciones escalables, como, por ejemplo, en este caso, los servidores web.

Express [13], por su parte, actúa como la infraestructura de las aplicaciones web en NodeJS, proporcionando recursos y características para el desarrollo web. Estas dos tecnologías han participado en la creación de la estructura de archivos y directorios de nuestra aplicación, en la configuración de las rutas de acceso, vistas de usuario, ...

Para las vistas de usuario, se ha utilizado el lenguaje de vistas EJS. Este lenguaje utiliza la misma sintaxis que HTML y es bastante potente y rápido en la ejecución.



Figura 8. Express.js y NodeJS

2.3 Aplicación web



Figura 9. Firebase

La tecnología que permite el intercambio de información entre el servidor y la aplicación móvil es el servicio de notificaciones Push de Firebase (Cloud Messaging). Este servicio permite el envío de mensajes desde un servidor web o desde la propia consola de Firebase a una aplicación móvil, así como alertas y notificaciones. En este caso, se envían notificaciones Push desde la aplicación web, haciendo uso de la API de Firebase para web. Esta API permite indicar que datos enviar, así como el dispositivo que deseamos que reciba esa información. De esta manera, no todos los usuarios reciben la información, que es altamente delicada.

Otro uso de Firebase en la aplicación web es para el control y gestión de los usuarios. Nos permite acciones como: registrar nuevos usuarios, iniciar sesión, obtener información sobre el usuario actual, detectar cuando el usuario se desconecta...

También es utilizada la API de Firebase para acceder a la base de datos en tiempo real. En la aplicación web se usa tanto para actualizar datos del paciente, principalmente su estado (atendiendo o sin atender), como para leer la base de datos de los pacientes (para cargar los usuarios que están esperando a ser atendidos), como para borrar (una vez los pacientes han sido atendidos y ha finalizado la asistencia, se eliminan de la lista de espera). Gracias a esta base de datos en tiempo real, al personal médico le aparecen al momento (sin necesidad de actualizar la página) los nuevos pacientes que debe atender.



Figura 10. Google Maps API

La tecnología que permite visualizar la ubicación del usuario al que está atendiendo el personal médico es la que ofrece Google en su Google Maps API JavaScript [14]. Gracias a ella, podemos ver la ubicación del paciente, marcada con una chincheta, así como movernos alrededor, alejándonos y acercándonos.

Por último, otra librería utilizada en este proyecto, y, en concreto, en la aplicación web, es la librería videojs [15]. Es una librería cuya principal función en este sistema es la de ser capaz de reproducir videos en streaming, cuyo protocolo (RTSP) no es aceptado por el reproductor de HTML.



Figura 11. VideoJS

2.4 Comunicaciones

En cuanto a las comunicaciones, existen cuatro tecnologías principales que han sido utilizadas y que son claves, ya que en ellas se sustenta este proyecto: libstreaming, Wowza Streaming Engine [16][17], el protocolo HTTPS y FFmpeg [18][19][20].

En primer lugar, como ya se comentó brevemente en el apartado de las tecnologías de la aplicación móvil (2.1), se ha hecho uso de la librería libstreaming.

Esta librería ha sido añadida como un módulo a la aplicación móvil y su principal función es la de permitir el envío de video en streaming desde la aplicación móvil al personal médico. Además, permite visualizar lo que estamos enviando. Esta librería permite el uso tanto de la cámara trasera del teléfono móvil como de la cámara frontal. De la misma manera que permite enviar solamente video, solamente audio, o ambas (como ocurre en este proyecto), también permite ajustar la calidad tanto del audio como del video a enviar y el protocolo.

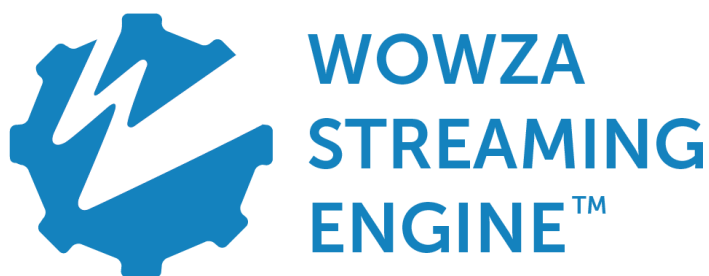


Figura 12. Wowza Streaming Engine

En segundo lugar, se encuentra Wowza Streaming Engine. Wowza es un software que hace las funciones de servidor de media en streaming, es decir, es el servidor que se encarga de servir tanto los videos como los audios. Una de las principales características es que permite la

seguridad del envío a través de contraseña, por lo tanto, el video del paciente está protegido hasta que es recibido por el personal médico. En este proyecto el servidor Wowza es utilizado tanto para servir los videos de los pacientes, como para servir el audio del equipo del personal médico.



Figura 14. HTTPS



Figura 13. OpenSSL

En tercer lugar, tenemos el protocolo HTTPS. HTTPS es un protocolo destinado a la transferencia segura de datos de hipertexto, basado en el protocolo HTTP, es decir, es la versión segura de HTTP. Este protocolo encripta la información a enviar entre la capa de aplicación y la capa de transmisión, es decir, entre HTTP y TCP, introduciendo una capa llamada SSL de cifrado. De esta manera, la información se cifra previamente al envío y se descifra una vez recibido por el destinatario. Para la creación de las claves necesarias se ha utilizado OpenSSL. Gracias al

protocolo HTTPS, la información delicada, como las credenciales del usuario, pueden ser transmitidas sin problemas de ser vistas por equipos intermedios.



Figura 15. FFMPEG

Por último, FFMPEG. Es un software libre que permite grabar, convertir y hacer streaming de audio y video. En este proyecto, es utilizado en el equipo del personal médico para

hacer streaming de audio con el paciente que está atendiendo. Dicho streaming es servido, mediante el servidor de Wowza anteriormente comentado, al usuario de la aplicación que está recibiendo la asistencia.

2.5 Base de datos

En cuanto a la base de datos utilizada en el sistema, la elegida ha sido Firebase. La elección de ésta es debido, además de todas las ventajas ya comentadas anteriormente, como las APIs tanto como para Android como para web, al amplio abanico de posibilidades y ventajas que ofrece respecto a otras. Estas ventajas son: las distintas bibliotecas y utilidades que proporciona para la gestión de usuarios, de archivos almacenados y de la base de datos no relacional, las notificaciones Push que son vitales para el funcionamiento del sistema, la escalabilidad y rapidez que proporciona, etc.

Como ya se ha comentado anteriormente, las distintas bibliotecas son bastante potentes. La encargada de la gestión de usuarios permite tanto crear usuarios, como iniciar sesión, comprobar el estado de la sesión, rescatar información del usuario, ... Además de dar la posibilidad de registrarse con diversos proveedores como Google, Facebook, o simplemente el correo electrónico y la contraseña. Además, permite aplicar restricciones como: definir los dominios autorizados, establecer cuantas cuentas se pueden registrar con el mismo correo, definir plantillas de verificación de la dirección de correo electrónico, de cambio de contraseña por correo electrónico y por SMS, ... Un sinfín de posibilidades que vienen implementadas ya en esta útil herramienta y que de manera sencilla podemos hacer uso de ellas. La base de datos en tiempo real presenta grandes ventajas al ser de tipo no relacional. Estas ventajas son las siguientes: escalabilidad y carácter descentralizado, soportando estructuras distribuidas; permiten adaptarse a distintos proyectos de manera más fácil que las bases de datos relacionales, siendo más flexibles; permite hacer cambios en el esquema sin tener que parar la base de datos... Otro de los aspectos destacados de Firebase es la base de datos en tiempo real, muy útil para este proyecto, ya que se actualiza de manera automática cuando un nuevo paciente está en espera. Las notificaciones Push de Firebase son el modo en el que este sistema intercambia

información, aprovechando la manera en la que se indica al usuario que se están recibiendo sus respuestas.

Además, Firebase contempla la atomicidad de las instrucciones, y, al estar desarrollada en JavaScript, proporciona una mayor rapidez de operación junto a NodeJS, y, por si fuera poco, es muy útil que ésta esté en la nube.

Capítulo 3.

Servicio web

3.1 Características generales

El servicio web está basado en una aplicación cliente-servidor. Las tecnologías principales para su desarrollo han sido las siguientes: NodeJS, Express, EJS [21][22] y Firebase. Además, se han utilizado otras librerías y APIs como videojs y Google Maps API JavaScript.

Las principales funciones de este servicio web son los siguientes:

- Recepción del vídeo en streaming transmitido desde la aplicación móvil.
- Realización del triage, es decir, responder a una serie de preguntas para conocer el estado del paciente.
- Enviar notificaciones con las distintas respuestas que va dando el usuario al dispositivo móvil del paciente, así como las maniobras oportunas a aplicarle a éste.
- Conocer la ubicación del paciente.

Una de las particularidades del sistema es el uso de la base de datos en tiempo real que proporciona la API de Firebase para el desarrollo web. De esta manera, el personal médico dispondrá del número de pacientes y la información relativa a éste de manera actualizada sin necesidad de actualizar la página. Esta información del paciente es la siguiente: el número en la lista de espera que tiene, el correo asociado al usuario de la aplicación y el estado en el que se encuentra su asistencia, es decir, si ya está siendo asistido (color verde) o si está en espera (color rojo).

3.2 Estructuras del servicio

En este apartado se expone a nivel técnico la estructura que sostiene el servicio web y permite la gestión y administración de los datos y recursos. Se comentará y analizará la base de datos Firebase, así como las rutas configuradas en el servidor.

3.2.1 Base de datos

Como ya se ha comentado durante este documento, Firebase hace uso de distintas bibliotecas para gestionar la distinta información que se puede almacenar en este servicio como: los usuarios, los archivos (tales como imágenes, videos o documentos), y los datos de la propia base de datos. Por ello, se detallará cada estructura de manera aislada.

En primer lugar, tenemos la estructura que contiene la información relativa a los usuarios. Ésta contiene los siguientes campos, que pueden ser consultados desde la consola de Firebase:

- **Identificador:** En este caso, será el correo electrónico del propio usuario. Otra ventaja más que ofrece Firebase es que, al crear un nuevo usuario, comprueba que se ha introducido una dirección de correo electrónico correcta. De esta manera, nos ahorramos implementar la gestión de errores que puedan ocurrir durante la creación de un nuevo usuario.
- **Proveedor:** Como ya se ha comentado, Firebase permite diversos proveedores para crear usuarios e iniciar sesión. Los proveedores que permite son: correo electrónico/contraseña, teléfono, Google, Facebook, Twitter, Github y anónimo. En el sistema desarrollado hasta el momento, se ha implementado la opción de correo electrónico/contraseña, mientras que la de Google se ha probado. Añadir el resto de proveedores es una de las líneas futuras de trabajo, para ahorrar tiempo al usuario.
- **Fecha de creación:** Este campo indica la fecha de creación del usuario.
- **Inicio de sesión:** Hace referencia a la fecha del último inicio de sesión del usuario en el sistema.
- **UID de usuario:** Para cada usuario del sistema, se crea aleatoriamente un UID de usuario. Es una combinación de caracteres alfanuméricos de longitud variable única para cada usuario.

Además de estos campos que pueden ser consultados fácilmente desde la consola de Firebase, existen otros campos que se almacenan en la base de datos utilizada para la gestión de usuarios a la que accedemos mediante funciones y métodos de la biblioteca de Firebase encargada de estas tareas. Estos campos son: el nombre para mostrar (distinto al correo electrónico), la contraseña y la foto del usuario.

En segundo lugar, nos encontramos la estructura del almacenamiento de archivos. Esta base de datos es utilizada en el sistema para descargar, desde la aplicación móvil, las imágenes tanto de las maniobras a aplicar, como de los estados finales posibles. De esta manera, cuando la cantidad de maniobras posibles a aplicar

crezca, o se incluyan videos explicativos, la aplicación no ocupará más. La consola de Firebase permite subir archivos directamente, crear carpetas, descargar los archivos, establecer reglas de restricción tanto para subir como para descargar archivos... Los campos que contiene esta base de datos y se pueden consultar desde la consola de Firebase son:

- Nombre: Este campo hace referencia al nombre del archivo contenido en la base de datos.
- Tamaño: Indica el espacio que ocupa dicho archivo.
- Tipo: Este atributo señala el tipo de archivo del que se trata. De esta manera, se muestra si es una imagen, un vídeo, un documento... así como la extensión. En este caso, las imágenes almacenadas tienen extensión jpeg, por lo que se muestra image/jpeg.
- Última modificación.

Por último, a continuación, se explica la propia base de datos en tiempo real, su estructura, sus características y las funcionalidades dentro del sistema. Se trata de una base de datos no relacional, que permite almacenar información en estructura de árbol. Básicamente se comporta como un fichero JSON, por lo que es muy potente y útil para trabajar en el desarrollo web, como es el caso. Desde la consola de Firebase podemos tanto añadir, eliminar, actualizar y modificar datos a mano, como mediante ficheros JSON; así como establecer reglas para controlar quién puede leer y escribir en nuestra base de datos; además de poder consultar el uso de la base de datos. En este proyecto se utiliza la siguiente estructura:

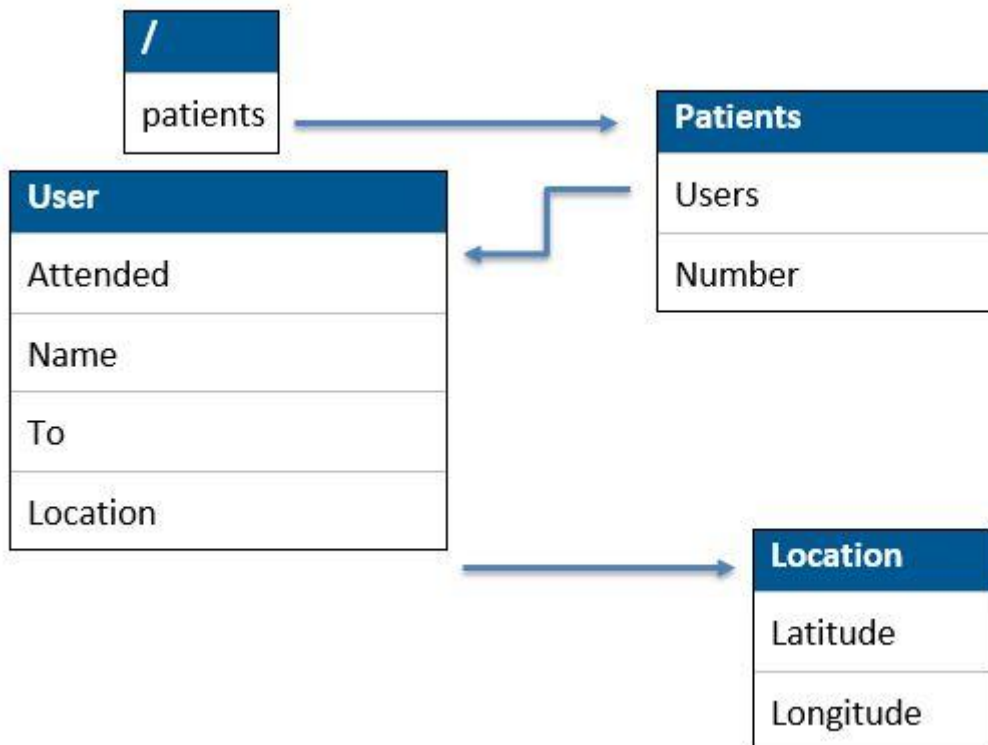


Figura 16. Estructura de la estructura de datos

La estructura usada es muy sencilla a la par que útil. Realmente no se representaría mediante tablas, ya que no se trata de una base de datos relacional, pero ayuda a representar mejor como está organizada la idea.

En primer lugar, se encuentra el objeto patients, que se encuentra en la raíz de la base de datos. Este objeto, a su vez, contiene al menos un objeto (el número de usuarios en espera), más tantos objetos como usuarios haya. Es decir, si hay tres usuarios en espera o siendo atendidos, el objeto patients contendrá cuatro objetos. Los objetos referentes a los pacientes son de tipo Hash, es decir, combinación de pares clave-valor, donde la raíz es el correo electrónico.

A continuación, por cada usuario en la base de datos, se crea un objeto con la siguiente estructura:

- **Attended:** Muestra el estado de atención del paciente. Toma el valor "Yes" si el usuario está siendo atendido, o el valor "No" si el usuario no ha sido atendido aún por ningún personal médico. De esta manera, al mostrar los usuarios en espera, (se muestran tanto los que están siendo atendidos como los que aún no) el personal médico puede comprobar fácilmente el estado del paciente, ya que se muestra una casilla de color verde al lado del nombre de este si está siendo atendido, o, de lo contrario, la casilla se pinta de color rojo.
- **Name:** Hace referencia al correo electrónico del usuario. Es utilizado para mostrar el nombre de usuario en la lista de pacientes.
- **To:** Es el campo más importante de la base de datos y hace referencia al token de sesión único que recibe el usuario por parte de Firebase al iniciar sesión. Es el más importante porque, gracias a él, se puede hacer uso de las notificaciones Push para el intercambio de información de forma segura entre el personal médico y el usuario de la aplicación, sin que pueda ser vista por ningún otro usuario del sistema.
- **Location:** Este valor del objeto de usuario es, a su vez, otro objeto de tipo Hash, en el que se almacena las coordenadas de la ubicación del usuario. Los pares clave-valor que contiene son los siguientes:
- **Latitude:** Este campo hace referencia al atributo de latitud de la ubicación del usuario obtenido por la aplicación móvil.
- **Longitude:** Este campo indica el atributo de longitud de la ubicación del usuario obtenido por la aplicación móvil.

3.2.2 Rutas

Las rutas configuradas en el servidor son las siguientes:

- Ruta `/`: Se dirige a la vista principal `'index.ejs'`. Esta vista contiene el formulario de inicio de sesión o login, así como la opción de acceder a la vista `'signup.ejs'` para registrar un nuevo usuario. En esta vista se ha implementado un control en los campos del formulario de inicio de sesión que comprueba la longitud de la cadena y el formato correcto de la dirección de correo electrónico, mostrando una alerta si se detecta algún error antes de enviar la información a Firebase. Además, este servicio de control de usuarios mejora la gestión de errores, mostrando los fallos en el inicio de sesión. Si el usuario tiene una sesión activa en el sistema, se redirige a la vista `'patients.ejs'`.
- Ruta `'/signup'`: Se dirige a la vista de registro `'signup.ejs'`. Esta vista contiene el formulario de registro de un nuevo usuario del sistema. Para esta vista, al igual que en la vista de inicio de sesión, se ha implementado un control para comprobar que los campos han sido rellenados correctamente, incluido el formato del correo electrónico y la longitud de la contraseña, que alerta de cualquier problema antes de enviar la información a Firebase. Gracias a este servicio, el control es mayor, ya que alerta de posibles errores al crear un nuevo usuario, por ejemplo, si ya existe un usuario con esa dirección de correo electrónico. De la misma forma que en la vista anterior, se comprueba si el usuario tiene una sesión activa, si es así, se le redirige a la vista `'patients.ejs'`, si no, se queda en esta vista. Además del formulario, contiene un enlace a la vista de inicio de sesión por si el usuario tiene una cuenta ya creada.
- Ruta `'/patients'`: Se dirige a `'patients.ejs'`, la lista principal del servicio web a la que se dirige al usuario una vez ha iniciado sesión en el sistema. Contiene un control de inicio de sesión de usuario que redirige a todo usuario que no se haya logueado antes en el sistema. Esta vista es la antesala a la vista de mayor funcionalidad, donde se asiste al paciente. En esta se carga una tabla con, tanto los usuarios que están esperando a ser atendidos como aquellos que están siendo asistidos, junto a la información relativa a estos: número que tiene en la "cola" para ser atendido, el nombre del paciente y el estado (atendiendo o por atender). Además, permite al asistente médico cerrar su sesión en el sistema.

- Ruta 'main/:user': Se trata de la vista con mayor funcionalidad del servicio web. La página que se carga es la misma para todos los usuarios a atender, variando lo que está a continuación de la barra, el nombre identificativo del usuario. De esta manera, se carga la información relativa al usuario, buscando en la base de datos el token, la ubicación... Estamos ante la vista de mayor funcionalidad, ya que es donde se asiste a los pacientes, mostrando el video en streaming recibido de la aplicación móvil, realizando el triage y teniendo la opción de consultar la ubicación del usuario en un mapa interactivo.
- Ruta '*': Esta ruta dirige a todas las direcciones distintas a las anteriores a una vista creada para avisar al usuario que esa ruta no está en el sistema. Dicha vista se llama 'notfound.ejs'. En ella se nos ofrece, además de indicarnos que la página no se ha encontrado, un enlace para dirigirnos a la página de inicio de sesión. En caso de que el usuario ya tenga una sesión abierta en el sistema, como ya se comentó en la ruta '/', se le dirigirá automáticamente a la vista 'patients.ejs'.

3.3 Implementación de las vistas

A continuación, se detalla el aspecto, la implementación, las funciones, y la forma en la que se utiliza cada una de las vistas mencionadas en el subapartado anterior.

3.3.1 Index.ejs

Esta vista es la primera página que se encontrará el usuario. En ella se encontrará con el logo del sistema, un formulario de inicio de sesión, para aquellos usuarios que ya estén registrados, y un enlace a la vista que se describirá a continuación para que los usuarios de nuevo ingreso se creen una cuenta en el sistema.



Figura 17. Vista de index.ejs

En primer lugar, al entrar en esta vista, se inicializa la API de Firebase y se comprueba si existe una sesión de usuario abierta. En caso afirmativo, se redirige automáticamente al usuario a la vista donde aparecerán los pacientes a atender (patients.ejs). En caso negativo, se queda a la espera de que se inicie sesión o cree una nueva cuenta de usuario.

Un aspecto importante de esta vista es la comprobación de los campos. En primer lugar, es necesario rellenar los dos campos (correo electrónico y contraseña), teniendo el primero de ellos un patrón para detectar aquellos emails mal escritos o incorrectos; y el segundo, una longitud mínima de al menos seis caracteres, dadas las restricciones de Firebase. En el caso de que no se cumplan alguno de los requisitos anteriores (campos sin rellenar, dirección de correo incorrecta o contraseña con longitud inferior a seis caracteres) se mostrará una alerta. A continuación, se envían tanto el email como la contraseña al servicio de gestión de usuarios (Firebase), y comprueba si los datos coinciden con alguno de los usuarios registrados. Si encuentra al usuario, dirige al personal médico a la vista 'patients.ejs', si no, muestra un mensaje de error de fallo en la autenticación.

3.3.2 Signup.ejs

La vista de registro de usuarios en el sistema es bastante similar tanto en la apariencia como en la funcionalidad a la vista anterior ('index.ejs'). En este caso, se

muestran tres campos a rellenar, los dos anteriores (correo electrónico y contraseña) más un campo referente al nombre de usuario.



Figura 18. Vista de signup.ejs

Al igual que en 'index.ejs', se inicializa la API de Firebase y se comprueba si existe una sesión abierta, y, en caso afirmativo, se redirige a 'patients.ejs'. Como ocurre en la vista anterior, se comprueban todos los campos antes de ser enviados a Firebase para crear el nuevo usuario. En caso de que ocurra algún error antes del envío, se notificará mediante una alerta. Una vez pasado el control implementado en el sistema, se envían los datos al gestor de usuarios para la creación de un nuevo usuario. En ese momento, si hubiera algún error, se notificará de igual manera.

3.3.3 Patients.ejs

Numero de Espera	Nombre del Paciente	Estado del Paciente
0	luis@yahoo.es	Verde
1	alu0100826166@ull.edu.es	Verde
2	jose@gmail.com	Rojo
3	juan@ull.edu.es	Rojo
4	maria@hotmail.com	Rojo
5	miguel@dominio.es	Rojo

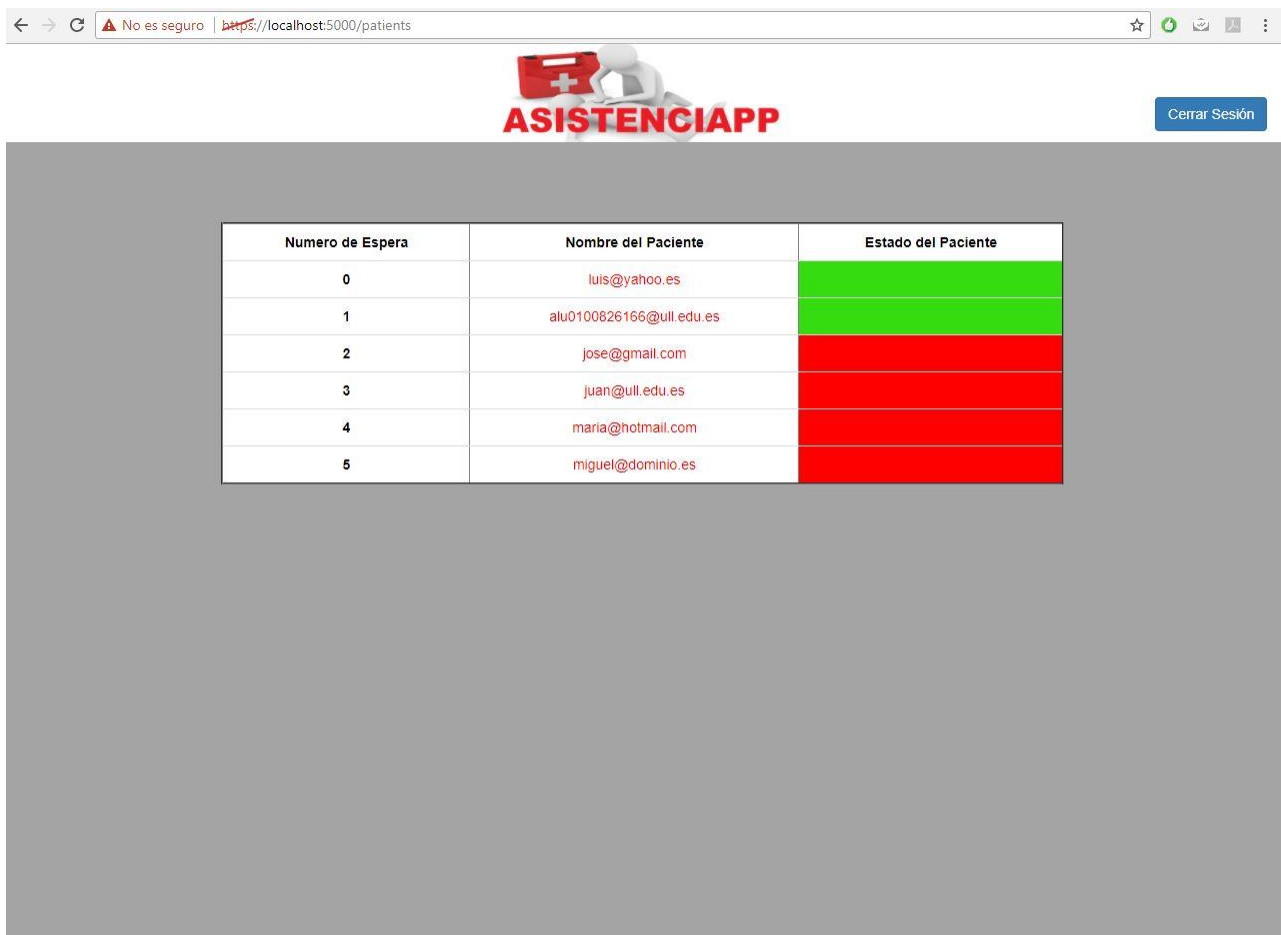
La vista de 'patients.ejs' es utilizada para mostrar los usuarios que están siendo o esperan ser atendidos, y consta de un encabezado donde se muestra el logo de la aplicación y un botón para cerrar la sesión del usuario actual, y debajo, un mensaje mientras carga los posibles usuarios. En caso de que existan usuarios en espera, se mostrarán en una tabla con tres columnas (número de espera del paciente, nombre y estado). En caso de que no existan usuarios en espera, se notificará que no existen usuarios esperando a ser atendidos.

Figura 19. Vista responsiva de patients.ejs

Al igual que ocurriese en las dos vistas anteriores, al cargar la página se inicializa la API de Firebase y se comprueba el estado de la autenticación, redirigiéndose a 'index.ejs' en caso de no existir una sesión abierta.

Una vez cargada la página, se hace una consulta a la base de datos en busca de pacientes en espera, creándose una tabla con tres columnas con la información obtenida: una para el número de espera según el orden en el que solicitaron la asistencia, otra columna para el correo electrónico del paciente, y la última columna que será pintada de: rojo, si el usuario sigue esperando a ser atendido; o verde, en el caso de que el usuario esté siendo atendido por un personal médico.

La ventaja que presenta la API de Firebase y la base de datos en tiempo real que ofrece es que no es necesario que el usuario del servicio web esté continuamente actualizando la vista para comprobar si un nuevo paciente requiere asistencia, sino que aparecerá de manera automática, estando actualizado continuamente.



Numero de Espera	Nombre del Paciente	Estado del Paciente
0	luis@yahoo.es	Verde
1	alu0100826166@ull.edu.es	Verde
2	jose@gmail.com	Rojo
3	juan@ull.edu.es	Rojo
4	maria@hotmail.com	Rojo
5	miguel@dominio.es	Rojo

Figura 20. Vista de patients.ejs

3.3.4 Main.ejs

La vista 'main.ejs' es la que mayor funcionalidad e implementación conlleva. Se accede a ella con el identificador del usuario, como se puede comprobar en el subapartado correspondiente a las rutas ('main/:identificador').

En cuanto a la apariencia de esta vista, tenemos un encabezado con el logo de la aplicación, justo encima del reproductor de video en el que se visualiza el streaming recibido desde el usuario que está siendo atendido, y del triage a realizar a éste. También se muestran dos botones: uno para concluir la asistencia; y otro, para mostrar u ocultar el mapa en el que aparece la ubicación del usuario marcada.

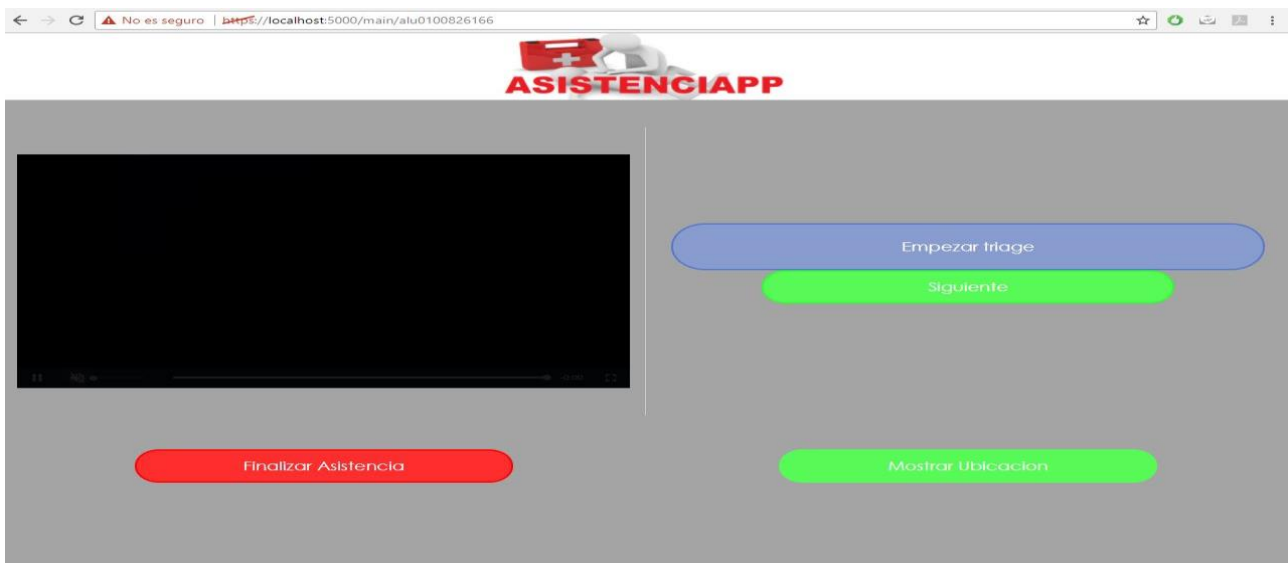


Figura 21. Vista de main.ejs

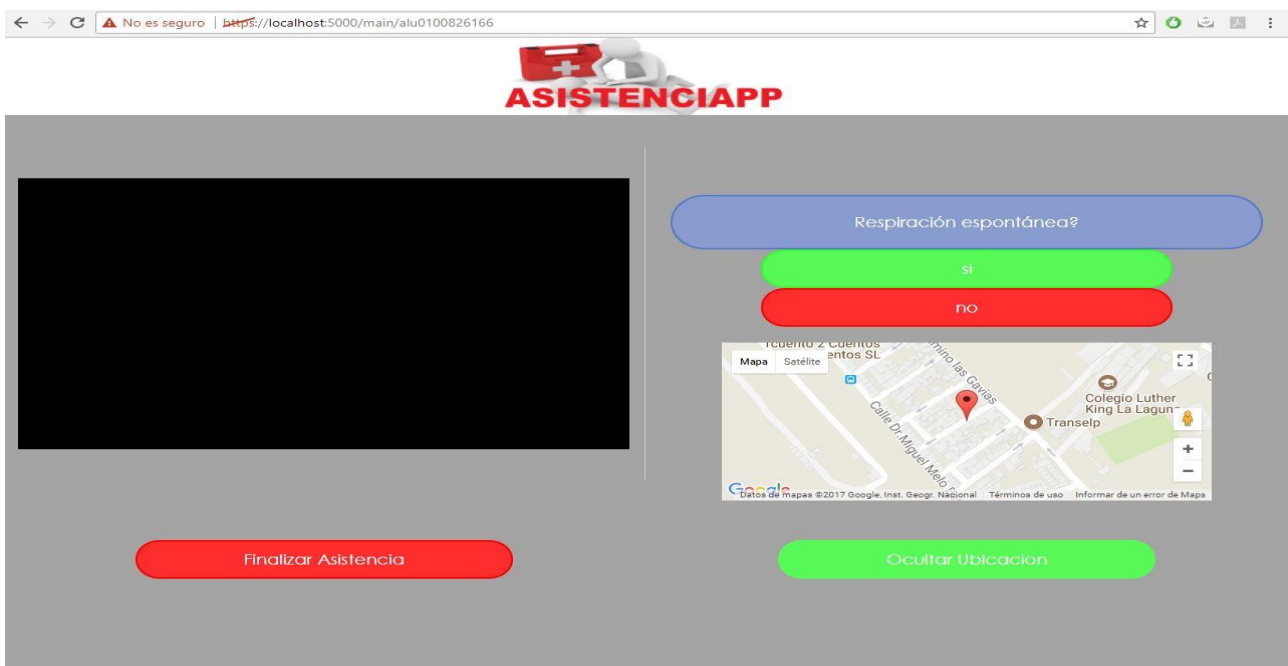


Figura 22. Vista main.ejs mostrando la ubicación del paciente

Como ya ocurriera en las vistas anteriores, se comprueba si existe una sesión activa, para evitar que pueda acceder cualquiera, redirigiéndose a la página inicial ('index.ejs') en caso de no existir. En la situación en la que sí exista una sesión iniciada, se comienza a cargar la información del usuario: el video en streaming, inicializa el mapa con la localización del usuario... Otro aspecto importante es que, nada más cargar la vista, cambia el estado del paciente a verde, para que otro personal médico no entre a asistir a un usuario que ya está ocupado.

El video en streaming se comienza a reproducir de manera autónoma, para facilidad del usuario del servicio web. El reproductor utilizado, videojs, permite mostrar los controles para subir y bajar el volumen del audio del video, pausarlo, ir hacia atrás en la reproducción...

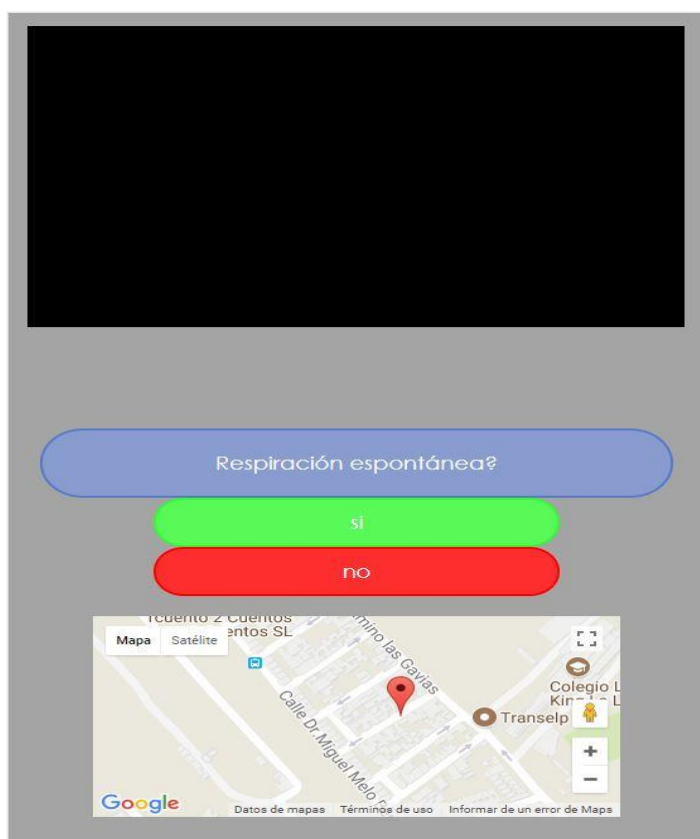


Figura 23. Vista main.ejs responsiva

El triage está implementado de manera que sea usable e intuitivo para el personal médico. Comenzará el triage cuando lo indique éste, responderá “si” o “no” a las preguntas que le irán apareciendo, y “siguiente” a las indicaciones de las maniobras a aplicar. Para facilitar y no confundir al personal médico, se ocultarán aquellas respuestas incoherentes. Al concluir el triage, verá el estado en el que se encuentra el paciente, pudiendo repetir el triage si no está conforme con el resultado o percibe que la situación ha cambiado.

A medida que el personal médico va respondiendo a las preguntas del triage según vea al paciente o escuche las respuestas del usuario, se activa la “maquinaria” de las notificaciones Push de Firebase

incluida en la API de Firebase. El mecanismo es sencillo y ajeno al usuario del servicio web, ya que él se limita a responder a las preguntas y las notificaciones se envían automáticamente. Las notificaciones se envían únicamente al usuario que está recibiendo la asistencia gracias al token almacenado en la base de datos, y, como contenido, transporta desde el servicio web a la aplicación móvil, o bien la pregunta que ha sido contestada y la respuesta dada; o el título de la imagen de la maniobra que debe ser aplicada al paciente en aquellas situaciones en las que se requiera.

Otro detalle importante y útil para el personal médico es la posibilidad de conocer la ubicación del paciente, marcada en un mapa interactivo donde puede moverse, alejarse o acercarse para tener mejores referencias acerca de la localización del asistido. Para la implementación de este mapa, se ha hecho uso de la Google Maps API JavaScript, que permite bastantes opciones usadas como: centrar el mapa en una ubicación dada por una latitud y una longitud, (en este caso, obtenidas de la base de datos) así como marcar dicha ubicación en el mapa y alejar o acercar mediante zoom al inicializar la vista del mapa. Debido a que el mapa puede molestar al personal médico durante la realización del triage, permanecerá oculto en el inicio, pudiéndose mostrar y ocultar al antojo del usuario del servicio web mediante el accionamiento de un botón. Al finalizar el triage, y tras obtener el estado final del paciente, se mostrará el mapa con la ubicación del usuario de manera automática, para que, si el personal médico lo cree oportuno, emitir un servicio de emergencias al lugar apuntado por la ubicación establecida en el mapa.

3.3.5 Notfound.ejs

La vista 'notfound.ejs' es utilizada para indicar al usuario que intenta acceder a una página inexistente en el sistema. De esta manera, se maneja los posibles errores al tratar de acceder a una página que no está configurada en el servidor.

En cuanto a la apariencia, muestra el mensaje de error "Página no encontrada" seguida de un enlace a la vista 'index.ejs'.



Figura 24. Vista notfound.ejs

Capítulo 4.

Aplicación móvil

4.1 Características generales

Para la realización de la aplicación móvil se ha utilizado el entorno de desarrollo integrado oficial para la plataforma Android, es decir, Android Studio (información acerca de Android Studio [23] y enlace para descargar la última versión [24]), en su versión 2.3.1. La versión Android [25] mínima requerida por la aplicación es la Android 4.0, 4.0.1 y 4.0.2 (API 14 [26]), aunque el nivel de API al cual va dirigida la aplicación es 23 [27] (Android 6.0).

La aplicación móvil desarrollada en Android tiene por objetivo fundamental conectar con el servicio web para asistir a una persona en estado de emergencia o que ha sufrido un accidente.

Esta aplicación cuenta con un sistema de inicio de sesión y de registro similar al implementado en la aplicación web, también gestionado por Firebase.

Las principales características de la aplicación son las siguientes:

- Emitir las imágenes capturadas por la cámara y el audio captado por el micrófono mediante streaming.
- Conectarse con la base de datos en tiempo real para almacenar la información necesaria para la asistencia y la comunicación.
- Recibir las notificaciones Push de Firebase que contiene las respuestas dadas a las preguntas del triage.
- Descargar y mostrar las imágenes de las maniobras a aplicar al paciente para asistirlo y las imágenes de los distintos posibles estados finales.
- Recibir y reproducir el audio procedente del personal médico para recibir instrucciones y, principalmente, comunicarse con éste para mejorar las acciones realizadas.

Por lo tanto, una de las particularidades de esta aplicación es la sincronización continua con el servicio web durante la asistencia médica.

4.2 Servicios integrados

Los principales servicios integrados en la aplicación móvil son los que se describen en los siguientes apartados.

4.2.1 Servicio de localización de Google

El servicio de geolocalización que permite obtener la ubicación mediante las redes telefónicas o mediante Wifi. Para ello, se ha hecho uso del servicio de localización que ofrece Google ('play-services-location'). Además, ya que la ubicación es de vital importancia para los casos más graves, la aplicación móvil desarrollada solicita permiso para acceder a datos de localización con una precisión alta. Se ha hecho de esta forma ya que la precisión baja daba resultados bastante pobres.

4.2.2 Firebase API

La API de Firebase [28] es otro servicio utilizado en la aplicación, ya que permite el manejo y gestión de la base de datos de manera sencilla y potente. Como se ha comentado durante el presente documento, Firebase trata de forma distinta los usuarios, los archivos almacenados y la base de datos. Además, también se ha utilizado esta API para poder recibir las notificaciones Push de Firebase. Para cada uno de éstos, se ha utilizado un servicio distinto. A continuación, se detallan los servicios utilizados, así como las principales características y las funcionalidades que desempeñan en la aplicación:

- En primer lugar, el servicio para la gestión de usuarios (Firebase-auth), es utilizado en cada una de las distintas "activities" para comprobar el estado de la sesión del usuario y redirigirlo a la "activity" correspondiente. Siguiendo con este servicio, también es usado en la actividad de login para el inicio de sesión del usuario, en la de registro para crear un nuevo usuario y en la actividad principal para obtener información acerca del usuario y escribirla en la base de datos.
- Otro servicio que sirve esta API es el de almacenamiento (Firebase-storage). Gracias a éste, la descarga de las imágenes de las maniobras y de los estados finales se realiza de forma sencilla y rápida.
- A continuación, el servicio de la base de datos en tiempo real (Firebase-database) nos permite la escritura, lectura, actualización y borrado de datos en tiempo real de forma sencilla y rápida. Es utilizada para almacenar la información en la base de datos relativa al usuario (nombre, estado de la asistencia, token para el envío de las notificaciones Push y ubicación del usuario) que será leída por el servicio web para comunicarse con el paciente.
- Por último, la API de Firebase es utilizada para la recepción de notificaciones Push mediante el servicio Firebase-messaging. De esta manera, se crea un token único para cada usuario registrado para que

pueda recibir solamente él, y de manera segura, la notificación enviada por el servicio web que contiene información privada y delicada.

- Otro servicio utilizado es la librería libstreaming que permite una de las funciones principales de la aplicación, es decir, emitir video en streaming desde la propia aplicación hacia el servidor Wowza para que pueda ser visto por el personal médico que atenderá al paciente. Esta librería permite configurar distintas opciones como: establecer la calidad del audio y del video a reproducir, definir tanto el codificador de audio (en este caso AAC) como el codificador de video, introducir las credenciales para un transporte seguro, así como la ruta del servidor al que va dirigido el streaming. Además, esta librería permite cambiar la emisión entre la cámara trasera y la cámara frontal, una utilidad bastante interesante en este proyecto. Otras posibilidades que ofrece libstreaming es definir que se quiere retransmitir, ya sea solamente las imágenes obtenidas por la cámara, solamente el audio captado por el micrófono o ambos (como se ha implementado en este proyecto). El estándar H.264 es el elegido para codificar el vídeo a emitir. Las ventajas que proporciona, y por las que ha sido elegido este codificador entre las distintos que implementa libstreaming son las siguientes: permite ofrecer vídeo de alta calidad a una gran variedad de dispositivos, desde teléfonos móviles con pocas prestaciones hasta los dispositivos Blu-ray dotadas de las máximas prestaciones, por lo que este estándar está llamado a sustituir a los formatos habituales hoy en día; además, el estándar H.264 reduce la cantidad de información necesaria para reproducir un vídeo, por lo que ofrece unas mejores sustanciales en cuanto al rendimiento en comparación con sus predecesores.

4.3 Implementación de las actividades

En este apartado se nombran las distintas actividades que componen la aplicación móvil desarrollada, así como las características y funcionamiento de cada una de ellas.

4.3.1 Login Activity

Esta es la actividad en la que se realiza la acción de inicio de sesión del usuario que ya tiene una cuenta creada. Contiene los campos destinados a introducir el correo



Figura 25. Activity Login

electrónico y la contraseña, además de un enlace a la actividad donde se crea una nueva cuenta de usuario (Signin Activity).

Esta actividad tiene implementado un control de estado de sesión de usuario, apoyado en el servicio Firebase auth que ofrece la API de Firebase, que comprueba si el usuario de la aplicación tiene una sesión activa. Si es así, se redirige al usuario a la actividad Start Activity.

Además, el servicio Firebase auth ayuda a gestionar mejor los errores al iniciar sesión en el sistema. De esta manera, en caso de ocurrir un error, se muestra en la pantalla.

4.3.2 Signin Activity

Esta actividad es la encargada de registrar un nuevo usuario en el sistema. Al igual que en la actividad anterior, se presentan dos campos de texto para registrar un nuevo usuario en el sistema, además de un enlace, por si el usuario ya tiene una cuenta creada, que le dirige a la actividad previa (Login Activity).

Signin Activity, al igual que la actividad de inicio de sesión, hace uso del servicio de Firebase auth para gestionar el control de usuarios. Las funciones que desempeña este servicio en la actividad actual son: comprobar el estado de la sesión de usuario, redirigiéndolo en caso de estar activa; y crear un nuevo usuario con los datos introducidos, gestionando y mostrando los posibles errores que puedan ocurrir.

4.3.3 Start Activity

Esta actividad está desarrollada en modo demostración, ya que el sistema no posee una dirección IP fija. Los componentes de esta aplicación son sencillos, consta de un campo de texto para introducir la dirección IP del servidor, y un botón para dirigir al usuario a la actividad principal y comenzar la asistencia al paciente.

4.3.4 Main Activity

Main Activity es la actividad principal de la aplicación móvil desarrollada para este sistema de asistencia sanitaria, ya que concentra el grueso de funcionalidades implementadas.

Mientras el usuario espera a que el personal médico se conecte con él, se muestra un mensaje en pantalla indicando que espere, así como información sobre el funcionamiento y dónde se mostrarán los distintos elementos, además de un botón que muestra los posibles estados del usuario tras la asistencia.



Figura 27. Activity Main mientras espera a ser atendido

Durante la carga de la actividad, se crea una nueva entrada en la base de datos relativa a la información del usuario de la aplicación. Esta información comprende el nombre del paciente, el estado de no atendido, el token asignado por Firebase para la recepción de notificaciones Push y la latitud y longitud de la ubicación del usuario. Sólo esta acción de escribir los datos en la base de datos requiere la intervención de los servicios de database, messaging y auth ofrecidos por la API de Firebase, así como el servicio de geolocalización mediante las redes móviles que ofrece Google.

Al igual que en las actividades de inicio de sesión y de registro de usuario, se ha implementado un control de autenticación, ayudado del servicio de Firebase, para evitar que usuarios sin una sesión activa puedan hacer uso de esta actividad.

Una vez el personal médico inicia la asistencia y, por tanto, la comunicación con el usuario, éste verá en la esquina inferior derecha las imágenes que está emitiendo al servidor Wowza mediante la librería libstreaming, y comenzará a escuchar al personal médico. En este momento, comienza el procedimiento de asistencia.

A medida que se van respondiendo las respuestas del triage, ya sea por lo que perciba el personal médico mediante las imágenes que estará viendo o por las respuestas que vaya recibiendo éste a las preguntas planteadas, el usuario de la aplicación comenzará a recibir notificaciones indicando que se ha respondido a una pregunta (haciendo uso del servicio de notificaciones Push de Firebase). Las notificaciones que recibe la aplicación por parte del servicio web pueden ser clasificadas en cuatro tipos a razón de las acciones que conllevan, y son las siguientes:

- Notificación de respuesta. Es la notificación que contiene tanto la pregunta como la respuesta del triage. Una vez recibida, se escribe en

la lista de respuestas la pregunta con un fondo de color verde si la respuesta ha sido positiva (“si”), o con fondo rojo si la respuesta ha sido “no”.

- Notificación de maniobra. Esta notificación es recibida cuando el paciente requiere la aplicación de una maniobra. En este caso, se muestra en la lista de respuestas del triage, el nombre de la maniobra a aplicar con un fondo de color amarillo, para diferenciar entre las respuestas a las preguntas y las maniobras a aplicar. A continuación, se obtiene el nombre de la imagen de la maniobra (que está incluida en la notificación) y se descarga mediante el servicio database de Firebase, mostrándola en el lado izquierdo de la pantalla.



Figura 28. Activity Main mostrando una maniobra

- Notificación de estado final. Se recibe la notificación que, al igual que en el caso de la notificación de maniobra, contiene el nombre de la imagen del estado final. A continuación, se descarga y se muestra en el lado izquierdo de la pantalla. La diferencia con la notificación anterior es que no se añade ningún elemento a la lista de respuestas de triage.



Figura 29. Activity Main mostrando el estado final

- Notificación de reseteo. Esta notificación es recibida cuando el personal médico considera que el estado final obtenido a partir del triage no se corresponde con el estado del paciente, o la situación del paciente ha cambiado, por ejemplo. En este caso, se indica al usuario en el texto a mostrar de la notificación que se ha reseteado el triage. La acción que se realiza al recibir la notificación Push es “limpiar” tanto la lista de respuestas como el visualizador de imágenes.

Una vez concluida la asistencia, el personal médico finalizará la comunicación con el paciente desde el servicio web mediante el botón destinado a ello, lo que provocará que se elimine al usuario de la lista de pacientes en espera en la base de datos. Además, otra acción que se lleva a cabo al finalizar la comunicación es dirigir al usuario a la actividad Start Activity, ya que la asistencia se da por terminada.

Capítulo 5.

Sistema global

En este capítulo se describirá: en primer lugar, el esquema general del proyecto desarrollado; a continuación, los métodos de seguridad implantados que hacen que el sistema sea seguro; y, por último, las herramientas relativas a las comunicaciones entre el servicio web y la aplicación móvil.

5.1 Esquema general

El sistema desarrollado se divide en distintos subsistemas de la siguiente manera:

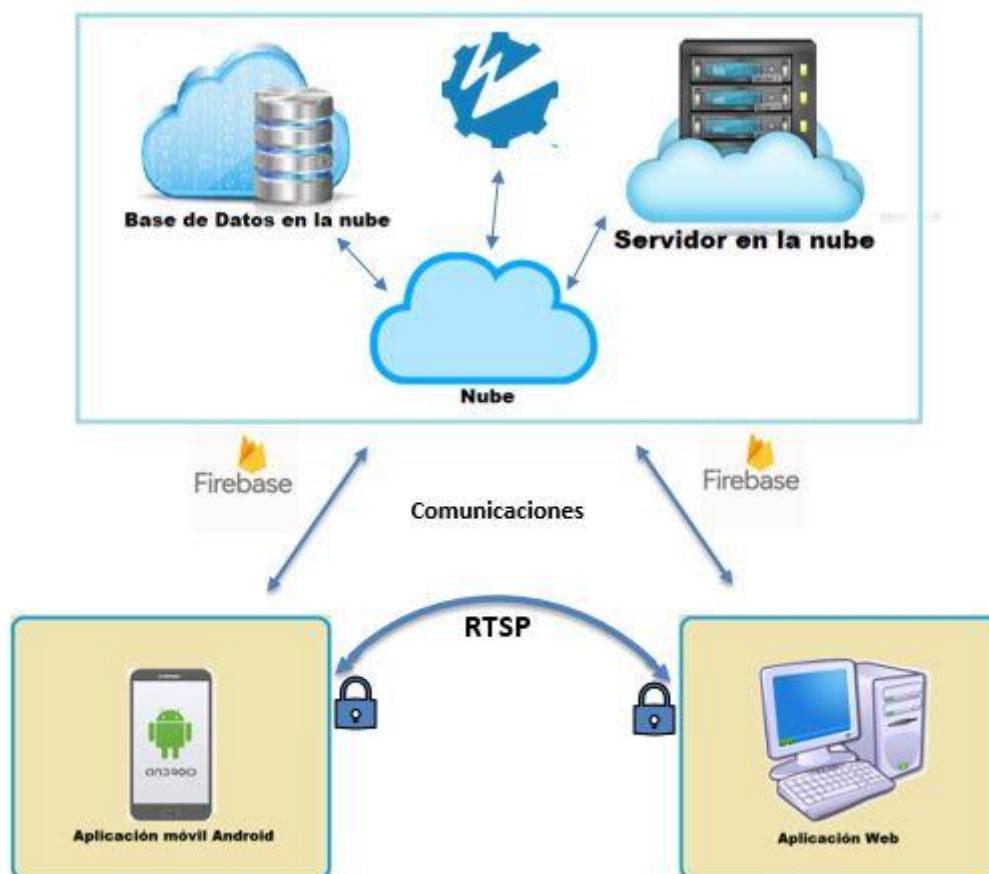


Figura 30. Esquema general del sistema

Tal y como puede observarse en la figura que hace referencia al esquema general del sistema, se aprecian los distintos componentes del propio sistema.

Por un lado, se encuentra el componente de la nube, donde está alojado el motor del sistema, ya que contiene tanto la base de datos, como el servidor de la aplicación web y el servidor de streaming de Wowza.

Por otro lado, la aplicación móvil desarrollada para dispositivos móviles Android, que se comunica con la nube mediante el servicio Firebase. A su vez, también existe otra comunicación, pero esta con la aplicación web mediante el protocolo RTSP, de manera segura, para transmitir el video en streaming y recibir el audio.

Por último, se observa la aplicación web, destinada al personal médico. Al igual que la aplicación móvil, se comunica con la nube mediante el servicio Firebase y con la propia aplicación móvil mediante el protocolo RTSP de manera segura.

5.2 Seguridad

A continuación, se procede a explicar los distintos mecanismos de seguridad que proporciona el sistema. Es un aspecto muy importante, ya que para cumplir la normativa actual existente para las aplicaciones médicas o m-Health, la información debe estar protegida y el sistema debe seguir unos protocolos de seguridad.

5.2.1 HTTPS

La primera medida de seguridad que proporciona el sistema desarrollado es el uso del protocolo HTTPS, el modo seguro del protocolo HTTP. El sistema HTTPS permite cifrar la información que se intercambian el cliente y el servidor, de esta manera, las claves de usuario y otros datos de éste, no pueden ser obtenidas por un atacante que intercepte los paquetes, ya que verá esta información cifrada.

Para la implementación del protocolo HTTPS en el sistema se ha utilizado la herramienta OpenSSL [29][30]. OpenSSL es un paquete de herramientas de administración y bibliotecas relacionadas con la criptografía. Se ha utilizado para implementar la capa intermedia entre la de transporte y la de aplicación, llamada SSL. Haciendo uso de ella, se han creado los ficheros que contienen las claves necesarias para hacer segura la conexión entre el servidor y el cliente.

El protocolo HTTPS opera entre la capa de transporte (TCP) y la capa de aplicación (HTTP), en una nueva capa creada a nivel de transporte denominada SSL/TLS. En ella es donde se cifra el mensaje, y no es descifrado hasta que es recibido por el correcto destinatario, que contendrá la clave para hacerlo.

El funcionamiento de HTTPS se basa en el uso de las claves públicas y privadas, que son un par de claves relacionadas, ya que un mensaje cifrado con una clave solo puede ser descifrado por su par correspondiente. Para ello, en el momento de la

conexión entre el servidor y el cliente, el navegador cifra una pre-clave generada en el momento con la clave pública del propio servidor, que descifra la pre-clave con su clave privada. De esta manera, se comparte la clave entre el cliente y el servidor sin que nadie pueda interceptarla y conocerla. A partir de este momento, comienzan a cifrarse los mensajes antes de enviarlos y se descifran con esa clave una vez recibidos.

5.2.2 Firebase

Otra medida de seguridad implementada en el sistema se apoya en la API de Firebase. Por una parte, permite la autenticación segura de los usuarios, y por otra, el envío seguro de las notificaciones Push al destinatario correcto.

El sistema permite la autenticación de manera segura gracias al uso de la API de Firebase. Las características principales que hace que sea más seguro son las siguientes:

- Controla el acceso de usuarios a las rutas (en el caso del servicio web) y/o actividades (en la aplicación móvil) para que no puedan acceder usuarios sin una sesión activa.
- Garantiza el uso de contraseñas con una longitud mínima de seis caracteres, por lo que éstas pasan a ser menos débiles.
- Usa HTTPS para el transporte de las credenciales.

El otro punto interesante del uso de Firebase es el relativo a las notificaciones Push. En la aplicación móvil, se asigna un token aleatorio y único a cada usuario al iniciar la sesión, de esta manera, Firebase permite enviar notificaciones Push únicamente a un usuario concreto. Esta funcionalidad es bastante útil en este proyecto, ya que, mientras se asiste a un paciente, los demás usuarios que puedan estar en espera no tienen porqué (ni deben) recibir la información de la asistencia que se está realizando. Además de ir dirigida la notificación Push de Firebase a un usuario en concreto, se realiza de manera segura gracias al uso de los mencionados tokens.

5.2.3 Streaming

El intercambio de datos de tipo media que se produce en este sistema está bajo seguridad, ya que las herramientas utilizadas permiten proteger la comunicación mediante contraseña.

Una de las propiedades más importantes que presenta el protocolo de transmisión en tiempo real (RTSP) utilizado en el sistema para el video en streaming es la seguridad que proporciona. Éste reutiliza mecanismos de seguridad web como protocolos de transporte (TLS [31]), además de poder aplicarse todas las formas de autenticación de HTTP.

5.2.4 Tokens

El estándar JSON Web Token (JWT) [32] es un estándar abierto basado en el lenguaje JSON para crear un token que sirva para enviar datos entre aplicaciones o servicios y garantizar que sean válidos y seguros. Los casos más comunes de uso de los tokens JWT son: la autenticación (al iniciar sesión, el servidor genera el JWT y se lo manda al cliente, que con cada petición es enviada y utilizado por el servidor para verificar quién es el usuario); y la transferencia de datos (se envía el JWT junto a los datos a transferir, de esta manera se asegura que el contenido no es alterado). Este estándar es el utilizado por Firebase. La estructura del JWT consiste en tres partes diferenciadas por puntos que son las siguientes: cabecera, carga útil o “payload” (contenido a transferir) y la firma. Un ejemplo de cada uno ellos se muestran a continuación:

```
{
  "alg": "HS256",
  "typ": "JWT"
}
```

Figura 31. Ejemplo de cabecera

```
{
  "sub": "1234567890",
  "name": "John Doe",
  "admin": true
}
```

Figura 33. Ejemplo de carga útil

```
HMACSHA256(
  base64UrlEncode(header) + "." +
  base64UrlEncode(payload),
  secret)
```

Figura 32. Ejemplo de firma

A continuación, se muestra tanto la cabecera, la carga útil y la firma anteriores codificados, haciendo uso del algoritmo HS256 que aparece indicado en la cabecera:

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4gRG9lIiwiaXN0b2NpYWwiOiOnRydWV9.4pcPyMD09o1PSyXnrXCjTwXyr4BsezDI1AVTmud2fU4
```

Figura 34. Ejemplo de JSON Web Token (JWT)

Además, los tokens mencionados en el apartado de Firebase (5.2.2) son distintos a los planteados en el apartado de futuros trabajos. A continuación, se procede a comentar la diferencia:

- Los tokens de Firebase son conocidos como “Instance ID”. Éstos se generan de manera única por cada instancia de aplicación. Los principales usos que tiene son los siguientes: verificar la autenticidad de la aplicación (permite verificar el nombre del paquete de la aplicación y comprobar si tiene una firma válida), confirmar que el dispositivo de la aplicación está activo (ofrece la posibilidad de saber cuándo se utilizó por última vez el dispositivo), identificar y realizar el seguimiento de aplicaciones. Este token está respaldado por un par de claves pública/privada, estando la clave privada almacenada en el dispositivo local y la pública en el servicio de ID de instancias. Además, puede renovarse el token mediante el correspondiente método. Al ser un token único para cada instancia de aplicación, es utilizado para el envío y recepción de notificaciones, ya que nos aseguramos de que el receptor es el correcto y está bajo cifrado de clave pública/privada.
- Los tokens que se comentan como futuro trabajo para mejorar la seguridad del sistema son relativos a la sesión del usuario. Hasta la creación de esta tecnología, la sesión se almacenaba en una base de datos, que registraba cada inicio de sesión. A partir del uso de tokens, cuando el usuario se autentica en un sistema, este recibe un token que es generado en el servidor y es almacenado en el lado del cliente. De esta manera, cada petición HTTP que haga el cliente, llevará en la cabecera el token, que no es más que una firma cifrada que permite al sistema identificar al usuario. Asimismo, son realmente potentes porque de esta manera, el sistema es totalmente escalable, ya que no almacena información de estado.

Por lo tanto, los primeros tokens (ID Instance) se generan por cada instancia de aplicación y son únicos para cada dispositivo. Los segundos tokens, que se plantean como una posible línea futura de desarrollo, son claves únicas para la sesión del usuario.

5.3 Comunicaciones

En este apartado se describen las distintas herramientas utilizadas para la comunicación dentro del sistema, así como el protocolo de seguridad aplicado. Estas herramientas son utilizadas para la transmisión de video en streaming desde la aplicación móvil, para servir ese flujo de datos, para la transmisión de audio en

streaming desde el servicio web y para la comunicación entre la aplicación web y la aplicación móvil.

5.3.1 HTTPS

Una característica importante de las comunicaciones es el uso del protocolo seguro de transporte de hipertexto o HTTPS. Gracias a este sistema, las comunicaciones entre el servicio web se producen de manera segura.

5.3.2 Streaming

El contenido de este apartado hace referencia solamente al streaming de video procedente de la aplicación móvil, es decir, las imágenes y audio del paciente.

La comunicación se establece entre la aplicación y el servidor Wowza, que será el encargado de recibir el flujo de datos y, posteriormente, proceder a servirlo al servicio web.

El uso del protocolo RTSP para el transporte del video, y la codificación H.264, dan la sensación al usuario que la comunicación se realiza de manera directa entre el servicio web y la aplicación móvil. Debido a esto último, en el esquema general de este capítulo, se muestra la comunicación directa entre ambas aplicaciones. Una característica importante es el intercambio seguro, ya que se realiza haciendo uso de una contraseña.

5.3.3 Wowza

La herramienta Wowza es un software utilizado para crear el servidor de media en streaming.

La primera función de Wowza en este sistema es recibir el video en streaming procedente de la aplicación móvil del paciente para, luego, servirla al servicio web para que pueda ser visualizado por el personal médico.

La segunda función de Wowza es la de recibir el audio en streaming procedente de la herramienta FFMPEG para servirla al paciente que va a recibir la asistencia médica.

Las comunicaciones se realizan de manera segura, ya que se establece una contraseña para cada uno de los flujos.

5.3.4 FFMPEG

FFMPEG es una herramienta que permite diversas funcionalidades relacionadas con el audio y el video como convertir entre distintos formatos un archivo o el video/audio capturado por los dispositivos del equipo.

En este sistema FFMPEG juega un papel importante, ya que permite obtener el audio del personal médico capturado por el micrófono del equipo en el que se encuentre y enviarlo al servidor Wowza para que lo transmita al usuario que atiende en cada momento. Asimismo, ofrece la posibilidad de establecer diversos parámetros como: el bitrate o tasa de bits variables, que es la relación de bits por segundo que consume un archivo; el codificador utilizado para convertir el audio; el tamaño del buffer; el dispositivo o dispositivos del que queremos obtener la entrada; y la salida, pudiendo ser un archivo o una URL.

5.3.5 Firebase Messaging

Las comunicaciones entre el servicio web y la aplicación móvil en el momento de la asistencia hacen uso de Firebase Messaging. Este servicio permite el envío y la recepción de notificaciones Push de Firebase para el intercambio de información, utilizado, en este sistema, para enviar las respuestas del triage, las maniobras a aplicar, el estado final del paciente tras la asistencia y reiniciar el proceso de reiniciar el procedimiento de asistencia. Gracias al token de seguridad que asigna Firebase, la comunicación es segura.

Capítulo 6.

Presupuesto

En este capítulo se incluyen los costes estimados del proyecto, distinguiéndose en los recursos que han sido necesarios a nivel de software y a nivel de personal.

6.1 Costes de personal

A continuación, se exponen los costes de los recursos humanos necesarios para el desarrollo del sistema. Tomando como referencias que se han empleado aproximadamente cuatro horas como jornada laboral durante cinco días a la semana, siendo el precio de trabajo por hora del trabajador quince euros.

Tarea	Jornadas	Total
Análisis, investigación y documentación de las soluciones actuales de streaming	6	360 €
Estudio y selección del software a utilizar	5	300 €
Estudio y selección de las tecnologías de desarrollo e implementación	5	300 €
Análisis, investigación y selección de las tecnologías de comunicación	5	300 €
Implementación del servidor	15	900 €
Implementación de la aplicación web	15	900 €
Implementación de la aplicación móvil	21	1260 €
Combinación y sincronización de sistemas	16	960 €
Añadir capa de seguridad al sistema	12	720 €
TOTAL	100	6000 €

Tabla 1. Costes de personal

6.2 Costes de software

Software	Coste
Android Studio IDE	0 €
Licencia de desarrollador Android	22,15 €
Wowza Media Server	0 €
Firebase	0 €
FFMPEG	0 €
TOTAL	22,15 €

Tabla 2. Costes de software

6.3 Coste total

Tipo de Coste	Total
Costes de personal	22,15 €
Costes de software	6000 €
TOTAL	6022,15 €

Tabla 3. Coste total

Capítulo 7.

Conclusiones y trabajos futuros

En este capítulo se concretan las conclusiones del proyecto, así como las líneas futuras por las que debe encaminarse el desarrollo y los trabajos futuros para ampliar o mejorar el sistema implementado.

7.1 Conclusiones

El sistema descrito en este trabajo de final de grado se ha desarrollado teniendo en cuenta tecnologías y herramientas novedosas relacionadas con la retransmisión de video y audio en streaming. Concretamente, se ha resuelto múltiples retos trabajando con los distintos protocolos de transmisión, a pesar de haberse decidido utilizar RTSP, las distintas codificaciones de audio y video, y la interacción entre los distintos subsistemas que, a su vez, utilizaban e integran distintas tecnologías como Android, NodeJS...

Dada la importancia de garantizar la confidencialidad y la autenticidad de la información transferida, se ha dotado al sistema con servicios de seguridad tales como el uso del protocolo HTTPS y el uso de la API de Firebase.

Por último, gracias al tema del que trata este proyecto, ha sido posible formar una idea de la importancia de enfocar y aplicar las nuevas tecnologías al ámbito de la medicina y de la asistencia sanitaria, en lugar de ser utilizadas para otros campos en los que son utilizadas para ocio o diversión, como en las redes sociales y servicios de retransmisión de videos.

7.2 Trabajos futuros

De cara a mejorar el sistema desarrollado y de añadir funcionalidades, a continuación, se enumeran algunas tareas a realizar a corto plazo:

- En vista de mejorar y agilizar el proceso tanto de registro como de inicio de sesión en el sistema, podría aprovecharse las distintas funcionalidades que permite Firebase para loguearse y crear nuevos usuarios haciendo uso de los distintos proveedores como Facebook, Twitter, Github...

- Para mejorar la comunicación con el paciente, podría añadirse la posibilidad de que el personal médico decida si desea emitir también las imágenes captada por la cámara de su equipo.
- Otro ámbito en el que podría centrarse las siguientes líneas de desarrollo sería la seguridad, ya que en los tiempos que corren nunca es suficiente. De este modo, se podría plantear nuevas técnicas, como el uso de tokens de sesión.
- Para mejorar la rapidez de asistencia en los casos más graves en los que es necesario que se traslade un servicio de emergencias como una ambulancia, se podría desarrollar una herramienta que enviara los síntomas del paciente y la ubicación de éste a los servicios encargados de ello.
- Por último, se podría llevar a cabo la integración real de este sistema en algún organismo para verificar el comportamiento en una situación real en distintos simulacros.

Capítulo 8.

Conclusions and future works

In this chapter the conclusions of the project are concretized, as well as the future lines for development and future work to expand or improve the implemented system.

8.1 Conclusions

The system described in this final grade work has been developed taking into account new technologies and tools related to the retransmission of video and audio streaming. Specifically, several challenges have been solved by working with the different transmission protocols, in spite of having decided to use RTSP, the different audio and video codings, and the interaction between the different subsystems that, in turn, used and integrate different technologies as Android, NodeJS...

Given the importance of ensuring the confidentiality and authenticity of the information transferred, the system has been provided with security services such as the use of the HTTPS protocol and the use of the Firebase API.

Finally, thanks to the theme of this project, it has been possible to form an idea of the importance of focusing and applying new technologies to the field of medicine and health care, instead of being used for other fields in which are used for leisure or entertainment, such as in social networks and video broadcasting services.

8.2 Future works

In order to improve the developed system and add functionalities, here are some tasks to do in the short term:

In order to improve and streamline both registration and logon processes in the system, you can take advantage of the different functionalities that Firebase allows to log in and create new users using different providers such as Facebook, Twitter, Github...

- In order to improve communication with the patient, the possibility of medical personnel deciding whether or not to transmit the images captured by the camera of their equipment could also be added.

- Another area that could focus on the following lines of development would be security, since in these times it is never enough. In this way, new techniques could be introduced, such as the use of session tokens.
- To improve the speed of care in the most serious cases, where an emergency service such as an ambulance needs to be moved, a tool could be developed that would send the patient's symptoms and the location of the patient to the services responsible for it.
- Finally, the real integration of this system in some organism could be carried out to verify the behavior in a real situation.

Bibliografía

- [1] Explicación del Triage utilizado:
<http://soportevital112.blogspot.com.es/2015/03/triage-start.html>
- [2] Android: <https://es.wikipedia.org/wiki/Android>
- [3] Protocolo HTTPS:
https://es.wikipedia.org/wiki/Protocolo_seguro_de_transferencia_de_hipertexto
- [4] Protocolo RTSP:
https://es.wikipedia.org/wiki/Protocolo_de_transmisi%C3%B3n_en_tiempo_real
- [5] NodeJS: <https://nodejs.org/es/>
- [6] Firebase: <https://firebase.google.com/?hl=es-419>
- [7] Servicio Firebase-Cloud-Messaging: <https://firebase.google.com/products/cloud-messaging/?hl=es-419>
- [8] Servicio Firebase-Auth: <https://firebase.google.com/products/auth/?hl=es-419>
- [9] Servicio Firebase-Storage: <https://firebase.google.com/products/storage/?hl=es-419>
- [10] Servicio Firebase-Database:
<https://firebase.google.com/products/database/?hl=es-419>
- [11] Servicio de geolocalización de Google para Android:
<https://developers.google.com/maps/documentation/android-api/location?hl=es-419>
- [12] Libstreaming: <https://www.androidhive.info/2014/06/android-streaming-live-camera-video-to-web-page/>
- [13] Express JS: <http://expressjs.com/es/>
- [14] Google Maps API JavaScript:
<https://developers.google.com/maps/documentation/javascript/?hl=es-419>
- [15] VideoJS: <http://videojs.com/>
- [16] Sitio oficial de Wowza y descarga de Wowza:
<https://www.wowza.com/es/products/streaming-engine>
- [17] Wikipedia sobre el servidor Wowza:
https://en.wikipedia.org/wiki/Wowza_Streaming_Engine

- [18] Sitio oficial de FFMPEG donde poder descargar la herramienta:
<https://www.ffmpeg.org/>
- [19] Enlace útil sobre FFMPEG: <https://trac.ffmpeg.org/wiki/StreamingGuide>
- [20] Wikipedia sobre FFMPEG: <https://es.wikipedia.org/wiki/FFmpeg>
- [21] Sitio oficial del motor EJS: <http://ejs.co/>
- [22] Instalación de EJS: <https://www.npmjs.com/package/ejs>
- [23] Wikipedia sobre el IDE Android Studio:
https://es.wikipedia.org/wiki/Android_Studio
- [24] Sitio oficial donde poder descargar el entorno de desarrollo Android Studio:
<https://developer.android.com/studio/index.html?hl=es-419>
- [25] Información sobre las versiones API de Android:
<https://developer.android.com/guide/topics/manifest/uses-sdk-element.html?hl=es-419>
- [26] Documentación oficial de la API 14 de Android:
<https://developer.android.com/about/versions/android-4.0.html?hl=es-419>
- [27] Documentación oficial de la API 23 de Android:
<https://developer.android.com/about/versions/marshmallow/android-6.0.html?hl=es-419>
- [28] Documentación oficial de la API de Firebase:
<https://firebase.google.com/docs/reference/>
- [29] Wikipedia sobre la herramienta OpenSSL: <https://es.wikipedia.org/wiki/OpenSSL>
- [30] Sitio oficial de OpenSSL: <https://www.openssl.org/>
- [31] Información sobre el protocolo TLS:
https://es.wikipedia.org/wiki/Transport_Layer_Security
- [32] Sitio oficial de JSON Web Token (JWT): <https://jwt.io/>
- [33] Free, C., Phillips, G., Felix, L., Galli, L., Patel, V., & Edwards, P. (2010). The effectiveness of M-health technologies for improving health and health services: a systematic review protocol. *BMC research notes*, 3(1), 250.
- [34] Santos-González, I., Rivero-García, A., González-Barroso, T., Molina-Gil, J., & Caballero-Gil, P. (2016). Real-Time Streaming: A Comparative Study Between RTSP and WebRTC. In *Ubiquitous Computing and Ambient Intelligence: 10th International Conference, UCAmI 2016, San Bartolomé de Tirajana, Gran Canaria, Spain, November 29–December 2, 2016, Part II 10* (pp. 313-325). Springer International Publishing.