Diseño de un prototipo de plataforma basada en IoT para la detección, registro y alerta de crisis convulsivas en pacientes epilépticos.

Javier Andrés Carrillo Infante y Kevin Santiago Gutiérrez Méndez

Trabajo de Grado para Optar al Título de Ingeniero de Sistemas

Director

Jose Geralbert Rubiano

Msc. En TIC para la educación

Codirector

Iván Mauricio Peña Castellanos

Médico Especialista en neurología

Universidad Industrial de Santander

Facultad de Ingenierías Físico-mecánicas

Escuela de Ingeniería de Sistemas e Informática

Programa Académico

Bucaramanga

2024

Dedicado a

A mi mamá, por ser mi principal apoyo y compañía durante toda esta etapa, y por toda la paciencia, dedicación y amor que siempre me ha brindado durante todos mis años de vida. Infinitas gracias por hacer de mí la persona que soy hoy y la que seré mañana. Este trabajo es todo tuyo; aunque se refleja mi esfuerzo, también veo inevitablemente reflejado el tuyo. Por años, superaste todas las adversidades que la vida te puso en el camino para poder tenerme aquí hoy. Eres mi mayor orgullo. Te amo infinitamente, má.

A mi tía María Elisa, por apoyarme desde el inicio y brindarme herramientas para llegar a este punto. Nunca olvidaré toda la ayuda que me brindaste desde el primer día. Gracias por haber confiado en mí.

A cada uno de los compañeros y compañeras que conocí durante esta etapa y con quienes compartí de forma ideológica. Gracias por cada conversación, por cada debate, por cada aprendizaje y, sobre todo, por materializar verdaderamente lo que significa ser estudiante de una universidad pública. Por ustedes, conservo mi esperanza de que Colombia algún día tenga un nuevo amanecer.

Kevin Santiago Gutierrez Mendez

Dedicado a

A mis padres, tanto mi madre como mi padre, siempre fueron un apoyo constante en esta etapa universitaria, siempre confiaron y creyeron desde el inicio y nunca desistieron.

A mis amigos que siempre estuvieron para alegrarme y escucharme en los momentos en que lo requería.

A mi madrina y mi abuela, quienes nunca dudaron que se conseguiría este logro.

Y a T, quien llegó en el momento menos esperado, pero se convirtió en un gran apoyo en la etapa más dura.

Javier Andrés Carrillo Infante

Agradecimientos

Al profesor Jose Geralbert Rubiano por su dirección y guía constante en el desarrollo del en este proyecto.

Al Dr. Ivan Pena por su capacitación, apoyo, paciencia y disposición a la hora de instruirnos.

A nuestros compañeros y profesores que conocimos e interactuamos a lo largo de la carrera, ya que cada uno de ellos nos enseñó y brindó un poco para lograr el objetivo.

A nuestras familias por nunca desistir.

¡Muchas gracias!

Tabla de Contenido

Pág.

Lista de Tablas

Pág.

Lista de Figuras

Pág.

Glosario

Alerta: Notificación que indica una condición inusual o evento importante.

API Rest: Interfaz de programación de aplicación que sigue la lógica RESTful.

Async Storage: Almacenamiento asincrónico.

Back end: Parte de la aplicación que no es visible, pero se encarga de la lógica.

Controladores: Componentes que gestionan las solicitudes y generan las respuestas.

Emergencia: Situación que requiere una respuesta inmediata.

Endpoint: Punto final de una conexión de red, donde se envía la solicitud.

Front end: Parte de una aplicación que es visible para el usuario, la interfaz gráfica.

I2C: Inter-Integrated Circuit.

IoT: Internet de las Cosas

JSON: JavaScript Object Notation.

JWT: JSON Web token.

Local Storage: Almacenamiento local.

Microcontroladores: Pequeños ordenadores que controlan dispositivos electrónicos.

Monitoreo: Proceso de observar y registrar el comportamiento de un sistema para detectar anomalías.

Notificaciones: Mensajes enviados a un usuario para informarle de un evento o actualización.

Roles: Permisos y responsabilidades asignados a un usuario dentro de un sistema.

Sensores: Dispositivos que detectan cambios en el entorno.

Websocket: Protocolo de comunicación que permite una conexión entre un cliente y un servidor.

Resumen

Título: Diseño de un prototipo de plataforma basada en IoT para la detección, registro y alerta de crisis convulsivas en pacientes epilépticos[1*]

Autor: Javier Andrés Carrillo Infante y Kevin Santiago Gutiérrez Méndez[2*3*]

Palabras Clave: Plataforma IoT, epilepsia, ritmo cardiaco, sensores, salud, detección

Descripción: En el ámbito de la salud y la tecnología, surge la necesidad de mejorar el tratamiento de pacientes con epilepsia, una enfermedad neurológica común que requiere atención continua y precisa (Patel et al., 2016). La epilepsia, diagnosticada frecuentemente a temprana edad, impacta no solo a los pacientes, sino también en las familias que están al cuidado del paciente. En países de bajos recursos, 3/4 partes de la población tienen dificultades para acceder a los tratamientos necesarios debido a su costo y disponibilidad, lo que se conoce como la "brecha de tratamiento" (OMS, 2023). Frente a esta situación se hace clara la necesidad de una herramienta que acompañe a esta población.

Las alternativas de monitoreo y detección actuales incluyen dispositivos basados en deep learning y redes neuronales que analizan señales de EEG para generar alertas de crisis. Sin embargo, estos dispositivos presentan desafíos como complejidad técnica, altos costos y la necesidad de hardware especializado (Daoud et al., 2020). Ante estas limitaciones, el Internet de las Cosas (IoT) y los dispositivos basados en sensores ofrecen una solución más accesible y adaptable, permitiendo el monitoreo continuo de los pacientes en cualquier entorno (Mchale & Pereira, 2021).

Este trabajo se centra en desarrollar un prototipo de plataforma IoT que permita la detección, registro y alerta de crisis convulsivas a través de un dispositivo portátil con una aplicación móvil y un dashboard web. Esta solución busca ofrecer una alternativa de monitoreo, mejorar la calidad de vida de los pacientes y proporcionarles mayor autonomía y seguridad. El prototipo permitirá emitir alertas de emergencia a centros de atención médica y contactos telefónicos del paciente cada que se detecte una convulsión asociada a una crisis epiléptica, buscando garantizar una atención rápida y oportuna, reduciendo así los riesgos asociados a las crisis epilépticas.

[1*] Trabajo de Grado

[2**] Facultad de Ingenierías Físico-mecánicas. Escuela de Ingeniería de Sistemas e Informática. Director: Jose Geralbert Rubiano. MSc En TIC para la educación. Codirector: Iván Mauricio Peña Castellanos. Médico Especialista en Neurología

[3]

Abstract

Title: Design of a prototype IoT-based platform for the detection, recording and alerting of seizures in epileptic patients.[4*]

Author(s): Javier Andres Carrillo Infante and Kevin santiago Gutierrez Mendez[5]

Keywords: IoT platform, epilepsy, heart rate, sensors, health, sensing

Description: In the field of health and technology, there is a need to improve the treatment of patients with epilepsy, a common neurological condition that requires continuous and accurate care (Patel et al., 2016). Epilepsy, often diagnosed at a young age, impacts not only patients, but also the families caring for the patient. In low-resource countries, 3/4 of the population has difficulty accessing necessary treatments due to cost and availability, known as the "treatment gap" (WHO, 2023). Faced with this situation, the need for a tool to accompany this population is clear.

Current monitoring and detection alternatives include devices based on deep learning and neural networks that analyze EEG signals to generate crisis alerts. However, these devices present challenges such as technical complexity, high costs and the need for specialized hardware (Daoud et al., 2020). In the face of these limitations, the Internet of Things (IoT) and sensor-based devices offer a more accessible and adaptable solution, enabling continuous monitoring of patients in any environment (Mchale & Pereira, 2021).

This work focuses on developing a prototype IoT platform that allows the detection, recording and alerting of seizures through a portable device with a mobile application and a web dashboard. This solution aims to offer a monitoring alternative, improve the quality of life of patients and provide them with greater autonomy and safety. The prototype will enable emergency alerts to be issued to healthcare centers and the patient's telephone contacts whenever a seizure associated with an epileptic seizure is detected, seeking to ensure rapid and timely care, thus reducing the risks associated with epileptic seizures.

[4*] Degree Work

[5]Faculty of Physical-mechanical Engineering. School of Systems Engineering and Informatics. Director: Jose Geralbert Rubiano. MSc In ICT for Education. Co-director: Iván Mauricio Peña Castellanos. Medical Specialist in Neurology

Introducción

En el mundo actual, más específicamente en el ámbito de la salud y la tecnología, surge una pregunta: ¿Es posible mejorar el tratamiento de los pacientes que sufren de epilepsia? Esta es una enfermedad neurológica bastante común en el mundo, la cual demanda una atención continua y precisa (Patel et al., 2016). Este trabajo surge como una solución a esto. El propósito primordial de este trabajo es desarrollar una plataforma basada en tecnologías IoT (Internet de las cosas) que apoye al registro, detección y alerta de crisis convulsivas en pacientes epilépticos. Esto no solo busca una modernización frente al tratamiento que se tiene hoy en día por parte de los centros de salud para esta enfermedad, sino se busca que esta solución abra las puertas para la innovación en el monitoreo de pacientes, de una forma práctica y eficaz, que lleve a la mejora en la calidad de vida de esta población.

La epilepsia es diagnosticada muchas veces a temprana edad, afectando desde niños la vida diaria de las personas, no solo directamente al paciente que la padece, sino también a los familiares y cuidadores del mismo. Una problemática latente de esta enfermedad es que en los países considerados de bajos recursos es muy complicado para 3 ⁄ 4 partes de la población acceder a los tratamientos necesarios, esto se conoce como una "brecha de tratamiento". (World Health Organization: WHO, 2023). Estos tratamientos van desde los más comunes, como los del tipo farmacológico, que debido a su costo o disponibilidad se hacen de difícil acceso en ciertas regiones del mundo o no se hacen efectivos, contando con cifras de cerca del 40% de pacientes que generan resistencia a los mismos (Patel et al., 2016), hasta tratamientos que incluyen dispositivos clínicos o cirugías que por sus altos costos y complejidad no representan una verdadera solución al alcance de esta población. Sumado a esto, algunos pacientes presentan problemas de índole psiquiátrica, generalmente ansiedad o depresión, estos dificultan aún más el tratamiento de las crisis y sobre todo en la calidad de vida de los pacientes. (Ali, 2018).

A medida que avanzan los estudios en torno a esta enfermedad, surgen nuevas alternativas para su tratamiento. Una de las áreas que toma gran relevancia, son los dispositivos que tengan la capacidad de detectar y en un caso más optimista predecir las convulsiones antes

que se produzcan. (Fisher et al., 2008). Algunos estudios y desarrollos han propuesto soluciones que van desde dispositivos basados en deep learning y redes neuronales que extraen las señales de electroencefalogramas (EEG) y a través de su procesamiento buscan generar alertas y reportes a un especialista encargado ante crisis epilépticas que se puedan presentar (Daoud et al., 2020). Pese a que estos dispositivos están logrando unos avances prometedores, plantean ciertos desafíos y limitaciones. Los cuales parten desde presentar una complejidad respectos a las características técnicas, pues para pacientes y profesionales de la salud sin experiencia en manejo de estas plataformas genera una gran brecha en la accesibilidad. Adicionalmente, por las tecnologías empleadas, materiales de fabricación y costos de distribución, en términos económicos, genera una brecha grande para su adquisición, esto golpea fuertemente a ciertos sectores de pacientes que sufren de esta enfermedad. Por último, existen dispositivos basados en monitorización de EEG, estos requieren hardware especializado para obtener resultados óptimos, con esto se crea una barrera para los centros de salud que no cuenten con el hardware pertinente y tampoco tengan al funcionamiento los especialistas indicados que interpreten y gestionen estos resultados en los dispositivos.

Ante esta demanda y necesidad de monitorear los pacientes, los servicios de atención médica ven una facilidad de lograr esto con ayuda del Internet de las cosas (IoT) y dispositivos basados en sensores. (Mchale & Pereira, 2021) con un enfoque inclusivo y que tenga la menor cantidad de restricciones para los pacientes eliminando las barreras de acceso y permitiendo que cualquier usuario, desde el paciente hasta el profesional de la salud, vea en esto una herramienta que se adapte y abra nuevos caminos para la alerta, registro y detección de crisis convulsivas, pero que no solo se centre específicamente en los centros de atención hospitalaria, sino que busque facilitar la calidad de vida de las personas, esto llevándolo a un entorno cotidiano independientemente de la ubicación o actividades del paciente.

En este trabajo se plantea desarrollar un prototipo de plataforma IoT que permita la detección, registro y alertas de crisis convulsivas en pacientes epilépticos mediante el uso de un dispositivo electrónico portátil con una aplicación móvil y dashboard web, orientados a

garantizar el acceso a tratamientos de menor costo para la población en general y mejorar el bienestar de los pacientes, brindándoles mayor autonomía, independencia y seguridad en el desarrollo de sus actividades cotidianas. Este prototipo busca identificar crisis convulsivas asociadas a ataques epilépticos, permitiendo así emitir una alerta de emergencia a centros de atención médica que garanticen la atención oportuna del paciente, minimizando así factores de riesgo producto de la crisis presentadas.

1. Planteamiento y justificación del problema

El avance tecnológico que se presenta en la actualidad ha supuesto un aporte importante en diversos aspectos para la sociedad. Pese a esto, el aporte de desarrollo tecnológico en el campo de la salud aún sigue siendo poco visible y más en aplicaciones que se empleen y repercutan en la calidad de vida y cotidianidad de pacientes con diversas enfermedades. Este retraso es todavía más evidente en ciertas regiones donde la adopción de nuevas técnicas empleando tecnologías al servicio de la población en general no se hace notorio.

Esta problemática es fácil de evidenciar en el tratamiento de pacientes con enfermedades crónicas como la epilepsia, siendo personas que por las características de su enfermedad requieren un monitoreo y vigilancia constante, con un alto grado de rigurosidad, según la gravedad de su caso (*Understanding seizures*, s. f.). Este seguimiento normalmente está ligado a centros de atención especializada de la mano de profesionales.

Se estima que esta enfermedad afecta a cerca de 50 millones de personas en el mundo y según estimaciones se diagnostican alrededor de cinco millones de casos nuevos de epilepsia cada año, convirtiéndola así en uno de los trastornos neurológicos más comunes en el mundo (*Epilepsia*, s. f.). Esta enfermedad se caracteriza por desencadenar episodios convulsivos de forma recurrente, los cuales se materializan de diferentes formas en los pacientes, entre ellas, los

movimientos de forma involuntaria e incontrolables en diferentes partes del cuerpo, grupos musculares o en general de todo el cuerpo (World Health Organization: WHO, 2023) Estos episodios convulsivos representan un riesgo latente entre las personas que padecen esta enfermedad al ser afecciones que pueden ir desde una breve ausencia de control hasta ser episodios más graves y con gran frecuencia que se prolongan por una cantidad de tiempo considerable.

Estos episodios convulsivos o crisis convulsivas cobran relevancia al ser causantes de lesiones accidentales que repercuten en la salud y bienestar de las personas que padecen esta enfermedad. Entre esas afecciones se encuentran caídas, golpes, ahogamientos, quemaduras y fracturas. Aunque esto representa un desafío significativo para este grupo de personas, el gran índice de mortalidad prematura que se presenta en los pacientes es preocupante, con cifras de hasta tres veces más que las de la población en general (World Health Organization: WHO, 2023). Se calcula que alrededor del mundo mueren 125,000 personas al año por epilepsia (Beghi et al., 2019), siendo las principales causas de mortalidad la muerte súbita inesperada por epilepsia (SUDEP), el estado epiléptico, las lesiones producidas por movimientos involuntarios y el suicidio (Trinka et al., 2023).

En concordancia con lo anterior, es evidente que la población que sufre de esta enfermedad tiene carencias en el desarrollo de su vida con normalidad. Su emancipación como persona y como individuos se ve limitada por esta enfermedad producto del constante miedo y sensación de estar en peligro al sufrir una afección física grave o mortal producto de una crisis convulsiva que no sea atendida oportunamente. Sumado a esto, el estigma que se tiene hacia las personas que la padecen, ha generado en esta población una autopercepción negativa, lo cual lleva a inminentes crisis mentales que desencadenan finalmente en el suicidio (O. Y. Kwon & Park, 2014).

Por último, es importante señalar que la cifras de pacientes epilépticos son de mayor cantidad en países de ingresos medios y bajos en comparación con las cifras de los países con ingresos altos, eso sumado a que en dichos territorios(ingresos medios y bajos) cerca del 75% de

las personas afectadas por epilepsia podrían no estar recibiendo el tratamiento necesario (World Health Organization: WHO, 2023). En lo que se refiere a Colombia, país el cual se encuentra focalizado en estas regiones de ingresos medios y bajos, se afirma que el 1,3% de su población padece de epilepsia y cerca del 0,8% de las causas de mortalidad en el país está asociado a esta enfermedad (*Epilepsia: Mucho más que convulsiones*, 2017).

Todo lo mencionado con anterioridad lleva a plantear la idea que se hace necesario y prioritario implementar otras medidas que ayuden con el registro, alertas, monitoreo o detección de este tipo de crisis asociadas a la epilepsia en estas poblaciones, con el fin de garantizar su bienestar. De la idea expuesta surgen los siguientes interrogantes: ¿Es posible desarrollar e implementar tecnologías IoT al alcance de pacientes epilépticos para ayudar con la detección, alertas y registro de crisis convulsivas? ¿Es posible gracias a una plataforma IoT que identifique y alerte a pacientes epilépticos garantizar una atención rápida y oportuna en casos de emergencia? ¿Puede una plataforma IoT orientada al monitoreo de pacientes epilépticos ayudar a que desarrollen con mayor normalidad sus actividades sin una mayor preocupación? A partir de estos interrogantes se identifica el objetivo principal de este trabajo el cual buscará diseñar un prototipo de plataforma IoT con el fin de detectar, registrar y alertar sobre crisis convulsivas en pacientes que padezcan de epilepsia mediante un dispositivo electrónico portátil con una aplicación móvil y un dashboard web, que identifique patrones de movimientos corporales y mediciones de ritmo cardiaco anormales asociados a episodios convulsivos permitiendo llevar un registro de los episodios y brindar una alerta mediante la notificación a centros de emergencia garantizando así una atención rápida y oportuna frente a estos casos, ayudando a mitigar factores de riesgo que desencadenan en complicaciones de mayor gravedad.

2. Objetivos

2.1 Objetivo General

Diseñar un prototipo de plataforma IoT que combine un dispositivo electrónico portátil con una aplicación móvil y un dashboard web, que permita la detección, registro y alerta de crisis convulsivas en pacientes con epilepsia identificando patrones de movimientos corporales y mediciones de ritmo cardiaco anormales.

2.2 Objetivos Específicos

Identificar y documentar variables claves, como movimientos corporales y ritmo cardiaco, necesarias para la detección , registro y alerta de crisis convulsivas en pacientes con epilepsia.

Diseñar una arquitectura de software que permita la integración de un dispositivo de medición y detección, asegurando la compatibilidad e interacción efectiva entre ellos.

Desarrollar un prototipo funcional de la plataforma IoT que combine un dispositivo electrónico portátil, una aplicación móvil y un dashboard web.

3. Marco Referencial

3.1 Marco Teórico

3.1.1 Trastornos Neurológicos

Los trastornos neurológicos son aquellas enfermedades ocasionadas por una disfunción en el cerebro o en el sistema nervioso central y periférico (médula espinal, cerebro, nervios craneales, nervios periféricos, sistema nervioso autónomo y músculos). Estos trastornos neurológicos comprenden un amplio espectro, sus causas son diversas y puede llegar a tener implicaciones en la salud que requieran tratamiento de por vida. Estos trastornos comprenden enfermedades como la epilepsia, Alzheimer, enfermedades cardiovasculares, entre otros (*Mental health: Neurological disorders*, 2016).

3.1.2 Epilepsia

La epilepsia es un trastorno neurológico no transmisible cuya causa está asociada a diversos factores. Esta enfermedad se caracteriza por una cantidad considerable de crisis recurrentes producto de descargas eléctricas en exceso y desordenadas del tejido nervioso. Estas pueden provocar episodios de convulsiones focales, generalizadas o desconocidas (*Epilepsia: Mucho más que convulsiones*, 2017).

3.1.3 Factores de mortalidad en epilepsia

Las principales causas de muerte o factores de mortalidad que presentan las personas que sufren de epilepsia están asociados al estado epiléptico, los traumatismos producto de golpes generados por crisis convulsivas severas, quemaduras, ahogamientos, aspiración grave, bolo de comida, enfermedades cardio respiratorias, la muerte súbita e inesperada en epilepsia, la cual supone la mayor tasa de mortalidad en esta población y el suicidio (Campos, 2000).

3.1.4 Convulsiones

Las convulsiones son episodios repentinos y temporales de actividad cerebral anormal que pueden manifestarse como contracciones musculares involuntarias, pérdida del conocimiento y cambios en la conciencia. Cuando están asociadas con la epilepsia,

estas convulsiones representan el síntoma principal, caracterizándose por la repetición de episodios convulsivos debido a la actividad eléctrica alterada en el cerebro (Fisher et al., 2014).

3.1.5 Internet de las cosas (IoT)

Hace referencia a una arquitectura basada en internet que facilita que varios dispositivos estén conectados y en constante intercambio de datos entre sí, generando un impacto importante en la percepción de la información, servicios inteligentes más completos e interconectividad extensa (Salazar & Silvestre, 2016). Estos dispositivos van desde objetos cotidianos hasta herramientas de índole industrial más sofisticadas. El IoT se ha convertido en una de las tecnologías más importantes de este siglo, ya que con toda la conexión entre los diferentes objetos electrónicos es posible lograr una comunicación fluida entre estos y las personas. Con ayuda de informática de bajo costo, la nube, big data y tecnologías móviles es posible compartir y recopilar datos con la mínima intervención humana (¿Qué es el Internet de las cosas (IoT)?, s. f.)

3.1.6 Microcontrolador

Son consideradas “computadoras” de bajo costo, las cuales pueden ser usadas para tener un control en tiempo real de sistemas. Estos también hacen referencia a sistemas embebidos en tiempo real. Los microcontroladores son de bajo costo, cuentan con un único chip y presentan una facilidad al programar. Tradicionalmente, su programación se hace por medio de lenguaje ensamblador, pero en la actualidad es muy común programar estos dispositivos con lenguajes de alto nivel como Basic, Pascal o C. Con esto se abre la posibilidad de desarrollar e implementar algoritmos complejos en estos dispositivos (İbrahim, 2006).

3.1.7 Placa de desarrollo

En el desarrollo de un proyecto basado en microcontroladores se hacen necesarios distintos tipos de hardware y software, pero existen algunos productos que son usados en la mayoría de estos proyectos, como es el caso de las placas de desarrollo. Estas son placas que tienen integrados desde leds, switches y alarmas para que los usuarios puedan probar programas simples hasta algunos con complejidad moderada. Algunas incorporan

un chip el cual permite programar el microcontrolador elegido desde la misma placa. Los proyectos complejos pueden iniciar su desarrollo en estas placas y si estos llegan a ser usados en entornos más comerciales o industriales, es necesario la adquisición de una placa de circuitos diseñada en específico para dicho proyecto (İbrahim, 2006).

3.1.8 Sensores

Son dispositivos los cuales tienen la capacidad de detectar magnitudes físicas o químicas, que son llamadas variables y transformarlas en señales. Son considerados unos de los principales pilares a la hora del desarrollo en el Internet de las Cosas. Pueden ser implementados en todas partes, independientemente de su tamaño o distancia, siguen recogiendo y enviando información. Poseen una ventaja clave y es poderse anticipar a una necesidad con base en la información recopilada en su entorno (Salazar & Silvestre, 2016).

3.1.9 Sensores biométricos

Los sensores biométricos son los dispositivos que se encargan de recopilar, medir e identificar las características deseadas para el estudio. Una vez obtenidas estas mediciones se realizan las tareas de acondicionamiento necesarias. Estos generalmente hacen parte de sistemas biométricos, los cuales son necesarios para el proceso de reconocer características propias de una persona, control de acceso y sistemas de seguridad (Borja & Bueno, 2006).

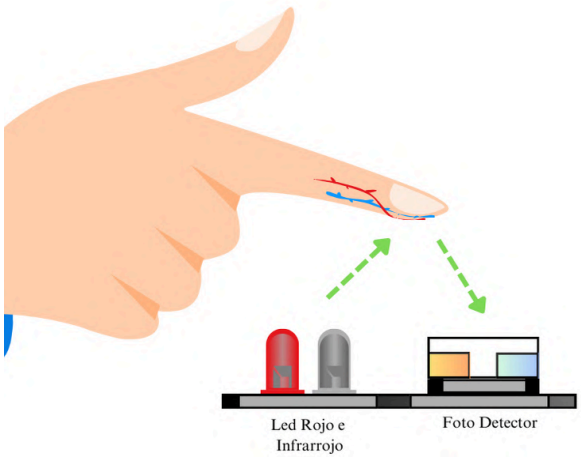
3.1.10 Sensor ritmo cardiaco

“La frecuencia cardiaca es un indicador útil de la adaptación fisiológica y la intensidad del esfuerzo”. La monitorización de esta toma importancia, ya que permite evaluar la aptitud cardiovascular. Esta se ayuda de sensores de ritmo cardiaco, los cuales son dispositivos que registran la actividad eléctrica del corazón para medir la frecuencia cardiaca. (Laukkanen & Virtanen, 1998). Estos son sensores ópticos que se apoyan con el uso de dos leds, uno de espectro infrarrojo y otro de espectro rojo que iluminan a través de la piel, esto busca medir los cambios en la luz en el flujo de sangre a través de sus vasos con la ayuda de un fotodetector (Ver Figura 1), esta frecuencia en la medición

aumenta a medida que se detecta un movimiento mayor. El método de detección de pulso a través de la luz se llama Fotopletismografía. (LME Editorial Staff, 2022).

Figura 1

Representación del proceso de Fotopletismografía.



Nota. Se presenta el método de detección de pulso mediante el uso de leds y fotodetector, este método conocido como Fotopletismografía. Creación propia.

3.1.11 Acelerómetro

La acelerometría es una de las técnicas más confiables que busca registrar y almacenar la cantidad y el nivel de actividad física realizada por cada persona en un tiempo determinado. Esta se apoya en los acelerómetros, dispositivos sensores que miden la aceleración, puede ser en una o tres dimensiones. Mayormente usados para detallar cambios de velocidad y posiciones de un objeto (Aguilar Cordero et al., 2014).

3.1.12 Giroscopio

El giroscopio es un dispositivo sensor que permite conocer la variación de un ángulo en el tiempo o también conocido como la tasa de rotación angular. Con esto se es posible determinar la orientación y movimiento del objeto en el cual se encuentra dicho sensor (*Pozo Espín, 2010).*

3.1.13 Arquitectura IoT

Los sistemas basados en ingeniería de las cosas presentan una arquitectura la cual puede ser dividida en cuatro capas o fases, las cuales son: Fase de detección de objetos, fase de intercambio de datos, fase de integración de la información y fase de servicios de aplicaciones (Salazar & Silvestre, 2016*)*

Tabla 1

Capas de arquitectura IoT.

Arquitectura IoT de cuatro capas.	
Capa de detección	Sensores, los objetos físicos y la obtención de datos.
Capa de Intercambio de Datos	Transmisión transparente de datos a través de redes de comunicación.
Capa de integtración de la información	El procesamiento de la información incierta adquirida de las redes, filtrado de datos no deseados e integración de información principal en conocimiento útil para los servicios y los usuarios finales.
Capa de servicio de aplicación	Da servicios de contenido a los usuarios.

Nota. De Internet de las cosas. (p. 16), por Salazar, J., & Silvestre, S., 2016, Techpedia. České vysoké učení technické v Praze Fakulta elektrotechnická.

3.1.14 Arquitectura de Software

La arquitectura de software se refiere a la estructura fundamental de un sistema de software, incluyendo los componentes del sistema, sus relaciones, y los principios y pautas que guían su organización y evolución a lo largo del tiempo (Bass et al., 2012).

3.1.15 Cloud computing

"Cloud computing es un modelo que permite el acceso bajo demanda y a través de internet a un conjunto compartido de recursos informáticos configurables (por ejemplo, redes, servidores, almacenamiento, aplicaciones y servicios) que pueden ser rápidamente

aprovisionados y liberados con un esfuerzo mínimo de gestión o interacción con el proveedor de servicios" (Mell & Grance, 2011).

3.1.16 Modelo y diseño 3D

El modelado 3D implica el uso de software para generar una representación matemática de objetos o formas en tres dimensiones, creando lo que se conoce como un modelo 3D. Este tipo de modelos son utilizados en una variedad de industrias, incluyendo el cine, la televisión, los videojuegos, la arquitectura, la construcción, el diseño de productos, la ciencia y la medicina, para propósitos como visualización, simulación y renderización de diseños gráficos (*Software de modelado 3D | versiones de prueba y tutoriales gratuitos | Autodesk*, s. f.)

La impresión 3D, conocida también como fabricación por adición, consiste en una serie de procedimientos que crean objetos mediante la superposición de material en capas que corresponden a secciones transversales sucesivas de un modelo 3D. Aunque los plásticos y las aleaciones metálicas son los materiales más comunes en la impresión 3D, prácticamente cualquier sustancia, desde hormigón hasta tejido biológico, puede ser utilizada en este proceso (*¿Qué es la impresión 3D? | Programa para impresora 3D | Autodesk*, s.f).

3.2 Estado del Arte

3.2.1 Embrace2

Dispositivo inalámbrico con la apariencia de reloj inteligente diseñado para identificar ataques convulsivos y la notificación de estos ataques a los respectivos contactos proporcionados. El dispositivo tiene la capacidad de detectar ataques convulsivos que presentan una duración mayor a 20 segundos, los datos reportados son enviados a un teléfono emparejado por medio de Bluetooth y este envía los datos a los servidores de Empatica. Cuando el dispositivo detecta una posible convulsión, enviará llamadas y SMS a los teléfonos del cuidador o familiares (Empatica, s. f.).

3.2.2 NightWatch

Dispositivo inalámbrico utilizado para detectar ataques epilépticos en pacientes mientras duermen. Consta de un brazalete para el brazo que registra la medición de frecuencia cardiaca y movimientos mientras se encuentra descansando el usuario, detecta el ataque y alerta a cuidadores, los cuales mediante un módulo recibe una alerta sonora y visual. Adicionalmente, cuenta con la opción de oprimir un botón de emergencia que envíe una alerta a un centro de atención médica si el usuario lo desea (NightWatch Epilepsy Seizure Detection, 2024).

3.2.3 LifeMinder

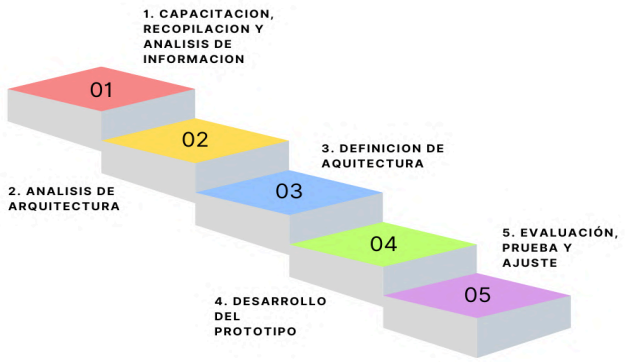
Dispositivo de uso portátil que detecta caídas y permite enviar un SMS alertando sobre la caída a números de emergencia definidos con ubicación en donde ocurrió; permite tener una llamada bidireccional con la incorporación de manos libres y la opción de tener un botón de emergencia en caso de ser necesario. Es resistente al agua, funciona con cobertura 4G y necesita una SIM Card para la comunicación y alertas (Life Minder, 2023).

4. Marco Metodológico

Para la elección de la metodología, se analizaron tanto las metodologías de desarrollo tradicionales como las metodologías ágiles. Se decidió optar por un enfoque de metodología tradicional como la metodología en cascada, ya que es fuertemente usada para el desarrollo de prototipos. La opción de una metodología de desarrollo ágil fue descartada, ya que estas son usadas en contextos con alta participación del cliente y para equipos de desarrollo de gran tamaño.

Figura 2

Esquema de la metodología de trabajo.



Nota. Esquema de la metodología a emplearse en el presente trabajo. Creación propia.

4.1. Capacitación, recopilación y análisis de información

En esta etapa se abarcan tanto la capacitación de conceptos técnicos, tecnológicos y médicos. En la capacitación de conceptos médicos se busca de la mano del médico especialista en neurología Iván Peña (Codirector del trabajo de investigación) Instruirse acerca de diferentes temas concernientes a la enfermedad, conceptos que no queden claros en la etapa de investigación o conceptos cuyo alcance para estudiarlos sean más complejos. En la capacitación de conceptos técnicos y tecnológicos se busca analizar cada una de las herramientas y tecnologías empleadas para cumplir con el desarrollo de los objetivos, cubriendo la totalidad de

conceptos necesarios en los integrantes del equipo de trabajo. Finalmente, se analizará la información obtenida con el fin de determinar su impacto.

4.2. Análisis de la arquitectura

En esta etapa se busca identificar cada uno de los elementos necesarios en el desarrollo del trabajo; desde las herramientas empleadas para la construcción del prototipo del dispositivo, hasta cada uno de los requerimientos y componentes que conforman la plataforma. Se estudiará sensores, proveedores de software, tecnologías, servicios en la nube, entre otros, analizando la viabilidad que presenten a lo largo del desarrollo.

4.3. Definición de la arquitectura

En esta etapa, una vez identificados los requerimientos, se plantea diseñar una arquitectura de sistema, que cumpla con ciertos criterios, como portabilidad, conectividad de dispositivos, capacidad de procesamiento oportuno, y eficiencia en términos de costo, para el desarrollo del prototipo de la plataforma IoT. Se tendrá en cuenta el desarrollo de la estructura del software, se establecerá una línea gráfica y guía de estilo con el desarrollo de mockups y se planteará un modelo 3D que albergará todo el hardware.

4.4. Desarrollo del prototipo

En esta etapa se pondrá en marcha el desarrollo del prototipo de la plataforma IoT (software y hardware) que dé cumplimiento con los objetivos definidos. Se buscará dar ejecución al desarrollo del elemento software de la plataforma (app móvil y dashboard web) siguiendo los lineamientos de diseño establecidos en la etapa anterior; de igual forma, se desarrollará el elemento hardware (dispositivo electrónico) el cual será el encargado de la toma de datos. Finalmente, la integración de estos dos elementos, dará como resultado el prototipo de plataforma IoT.

4.5. Evaluación, pruebas y ajustes del prototipo

Con la etapa de desarrollo culminada, se establecerá una etapa de pruebas del prototipo; donde teniendo en cuenta el plan diseñado, se validará la efectividad, inconvenientes, riesgos y posibles mejoras que presente el prototipo en un entorno controlado. La finalidad será realizar los ajustes necesarios con el fin de dar cumplimiento a la mayoría de requerimientos descritos.

5. Desarrollo del Proyecto

5.1 Capacitación, recopilación y análisis de información

5.1.1 Recopilación y análisis de información sobre la enfermedad

Para la realización de este proyecto fue importante recopilar y estudiar información relevante sobre la enfermedad, para ello se partió con una investigación y estudio exploratorio sobre la enfermedad con información proveniente de diversas fuentes tales como: libros, revistas académicas y artículos científicos, gracias a esto nos pudimos dar una primera idea de la representación de la enfermedad y cada una de sus implicaciones en los pacientes que la sufre.

A su vez, como primer encuentro con la enfermedad, se realizaron reuniones con nuestro codirector y médico especialista en neurología, Iván Mauricio Peña, con el fin de obtener un primer vistazo general sobre la enfermedad proveniente de un especialista. Estas reuniones fueron de vital importancia, ya que nos permitió conocer de primera mano las vivencias de pacientes que poseen esta enfermedad y a los cuales el médico Iván Peña ha tenido la oportunidad de acompañar y tratar. De estas reuniones pudimos obtener consejos sobre las necesidades de los pacientes, sobre los retos y adversidades que enfrentan estos, las limitaciones que contraen sistemas de monitoreo y por sobre todo, aclaraciones sobre conceptos y generalidades de la enfermedad.

5.1.2 Delimitación de la enfermedad

Para llevar a cabo este proyecto fue necesario definir una ruta clara, limitar el alcance del proyecto. Debido a la naturaleza y complejidad de la enfermedad, fue necesario hacer un análisis de los tipos de convulsiones y representaciones más comunes de la enfermedad en pacientes epilépticos con el fin de concluir hacia donde enfocar él todo el esfuerzo del proyecto.

En primer lugar, examinamos los diversos tipos de convulsiones que se presentan en la epilepsia. Los episodios de convulsión tónico-clónica, ausencias y convulsiones mioclónicas fueron los principales tipos que consideramos teniendo en cuenta las recomendaciones de nuestro codirector. Las convulsiones tónico-clónicas son conocidas por su presentación dramática, que comprende una fase tónica de rigidez muscular, seguida por una fase clónica de movimientos rítmicos y sacudidas. De acuerdo con la Fundación Americana de Epilepsia, estas convulsiones se caracterizan por "una pérdida de conciencia precedida por una rigidez muscular y movimientos rítmicos y sacudidas" Por el contrario, las crisis de ausencias se caracterizan por episodios de pérdida de conciencia sin movimientos corporales significativos, y las mioclónicas se presentan como sacudidas breves e involuntarias de un grupo de músculos.

Durante el análisis de las características de estos tipos de convulsiones en los pacientes, observamos que las convulsiones tónico-clónicas son más habituales en adultos, mientras que las ausencias y las convulsiones mioclónicas son más habituales en niños. De acuerdo con un estudio realizado por Sillanpää y Schmidt (2010), "las convulsiones tónico-clónicas son más comunes en adultos, mientras que las ausencias y las convulsiones mioclónicas son más frecuentes en la población pediátrica" Dado que nuestro objetivo contempla una amplia gama de edades, las convulsiones tónico-clónicas brindarán una base sólida para un análisis más inclusivo y representativo.

Asimismo, consideramos las consecuencias de cada tipo de convulsión en la calidad de vida y el bienestar de los pacientes, que es uno de los focos de nuestro proyecto. Las convulsiones tónico-clónicas suelen tener consecuencias más graves, tales como lesiones físicas y mayor morbilidad, en comparación con otros tipos de convulsiones. Por todo lo anterior, enfocarnos en estas convulsiones nos brindaría la oportunidad de diseñar una estrategia más eficaz para reducir los riesgos a los que están sometidos quienes padecen de este tipo de crisis y mejorar la calidad de vida de los pacientes.

5.1.3 Análisis de variables comunes

Una vez con la información necesaria sobre la enfermedad y seleccionado el foco de población hacia el cual se encuentra destinado el desarrollo del proyecto, se procedió a identificar variables medibles con el fin estudiar cuáles podrían ser de ayuda para caracterizar una crisis convulsiva tónico-clónica asociada a un ataque epiléptico. De lo anterior pudimos identificar las siguientes variables:

- Patrones de ondas cerebrales anormales
- Movimiento del cuerpo, como movimientos bruscos o sacudidas espontáneas en extremidades
- Rigidez o contracciones en músculos de extremidades
- Aceleración o desaceleración súbita del ritmo cardiaco
- Sudoración y cambio de temperatura corporal

Después de una revisión cuidadosa y discusiones con el médico especialista Iván Peña, decidimos centrarnos en dos variables clave: el movimiento corporal y la frecuencia cardíaca.

El movimiento corporal fue la principal variable que elegimos porque proporciona una señal clara de la actividad motora durante las crisis epilépticas. Las convulsiones a menudo sé

asocian con movimientos repentinos o sacudidas espontáneas de las extremidades, lo que puede indicar claramente que ha ocurrido un evento epiléptico. Al monitorear y analizar estas actividades, podemos identificar patrones comunes durante las crisis. Además, la aceleración o desaceleración repentina de la frecuencia cardíaca fue otra variable que seleccionamos porque estaba asociada con ataques epilépticos. Durante una crisis, pueden producirse cambios significativos en la actividad autónoma del cuerpo, que se manifiesta por cambios significativos en la frecuencia cardíaca. Estos cambios brindan información adicional sobre el estado del paciente y la gravedad de la convulsión.

Las otras variables consideradas, como los patrones anormales de ondas cerebrales, la rigidez muscular y los cambios en la temperatura corporal, aunque son importantes para el diagnóstico y el tratamiento clínico de las crisis epilépticas, plantean desafíos adicionales para el seguimiento y el análisis continuo. La detección precisa y consistente de patrones anormales de ondas cerebrales requiere equipo especializado y tecnología sofisticada, mientras que la rigidez muscular y los cambios de temperatura corporal pueden ser menos específicos y difíciles de medir de forma no invasiva

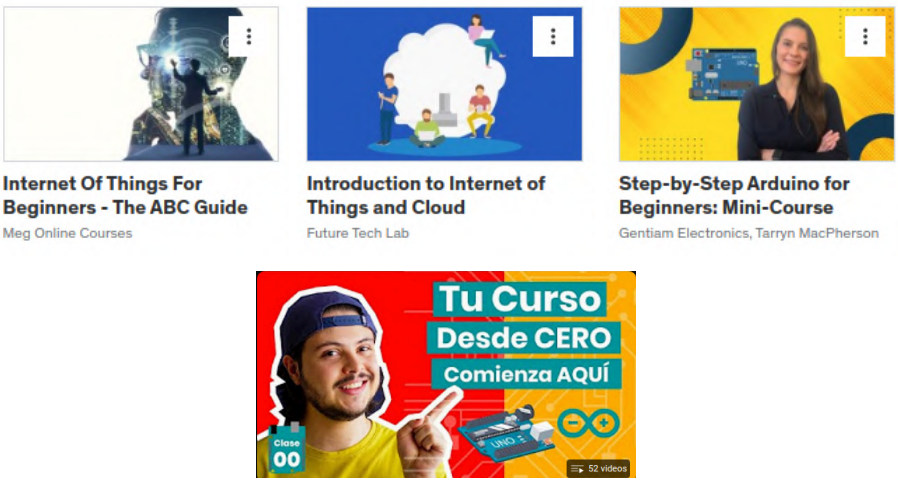
5.1.4 Capacitación de aspectos tecnológicos

Partiendo del hecho que nuestro proyecto busca emplear tecnologías IoT, fue necesario para la realización del mismo, consultar diversas fuentes de información para entender más en profundidad cómo emplearemos estas tecnologías para dar solución a nuestro problema. Para ello se hizo uso de cursos virtuales, los cuales seguimos en su totalidad en plataformas como Udemy y plataformas de video como YouTube. Los cursos seguidos fueron: Internet Of Things For Beginners - The ABC Guide de Meg Online Courses, Learn the basics of the IoT and Cloud Systems de Future Tech Lab. Con estos cursos fue evidente que la principal plataforma de

desarrollo y programación de placas y sensores empleado en proyectos generales de IoT es Arduino, por ello, también tomamos cursos sobre esta plataforma de desarrollo y poder darnos una idea general de cómo se estructura y desarrolla un proyecto empleando sensores, entre otros elementos. Los cursos seguidos fueron: Step-by-Step Arduino for Beginners: Mini-Course de Gentiam Electronics y Tarryn MacPherson y Curso de ARDUINO desde CERO en Español de Johann Pérez E.

Figura 3.

Cursos para capacitación de aspectos tecnológicos.



Nota. Cursos que fueron realizados para capacitarse en las respectivas tecnologías necesarias para el proyecto.

5.2 Análisis de arquitectura

5.2.1 Análisis de alcance

Para el desarrollo del proyecto fue necesario partir de la definición del alcance del proyecto. En el contexto de nuestro proyecto se requiere una plataforma que permita monitorear y procesar datos de sensores para identificar patrones de movimientos corporales para alertar sobre una crisis. Esta toma de datos se hará a través de un dispositivo electrónico portátil que le

permita al usuario desarrollar su vida cotidiana sin estar necesariamente en centro de atención médica, que no le interrumpa la mayoría de sus actividades y que le brinde un grado alto de autonomía; es por eso que se hace necesario que el dispositivo sea lo menos invasivo posible, que tenga autonomía de funcionalidad y además que permita cierto grado de conectividad y transmisión de datos por parte del dispositivo mientras que el usuario se encuentre desarrollando sus actividades. Adicionalmente, la plataforma debe permitir integrar una aplicación móvil, la cual le permitirá al usuario realizar diversas funciones y una app web para gestionar emergencias.

Dejando claro que la finalidad del proyecto es realizar un prototipo funcional, acotamos el desarrollo del proyecto teniendo en cuenta consideraciones de limitaciones y mínimos que nos garantizaran cumplir todos los objetivos planteados, es así como buscamos una arquitectura que nos permitiera cumplir la definición inicial del proyecto.

5.2.2 Requerimientos funcionales, no funcionales y definición de roles

Inicialmente, un proyecto de software se constituye a partir que el cliente hace la solicitud de desarrollo de un producto software, indicando por completo cada una de las necesidades, funcionalidades y usos con los que tiene que cumplir el sistema, a partir de esto, se define y clarifican cada uno de los requerimientos funcionales o no funcionales y en adición se establecen los roles que sean necesarios. En este caso, los requerimientos obedecen y fueron seleccionados con base en el propósito del proyecto y sus objetivos establecidos limitados por el alcance definido anteriormente.

5.2.2.1 Definición de roles. Dentro del contexto del proyecto se identificaron los siguientes roles:

Tabla 2

Definición de roles del proyecto

Rol	Descripción
Administrador	Tiene acceso al dashboard de gestión de emergencias
Usuario	Tiene acceso a la app móvil en donde podrá consultar y gestionar información y hacer uso de funcionalidades

5.2.2.2 Requerimientos funcionales. Para el desarrollo del proyecto, se definieron unos requerimientos funcionales mínimos, los cuales nos ayudaran a cumplir los objetivos definidos

Tabla 3

Definición de requerimientos funcionales.

No	Descripción	Roles
RF-01	La aplicación móvil permitirá iniciar y cerrar sesión con correo y contraseña	Paciente
RF-02	La aplicación móvil permitirá conectar y desconectar el dispositivo electrónico con el fin de comenzar a recibir datos	Paciente
RF-03	La aplicación móvil permitirá registrar, editar y eliminar contactos de emergencia con	Paciente

	nombre y número de teléfono, con el fin de alertar cuando se detecte una emergencia	
RF-04	La aplicación móvil permitirá que se envíen notificaciones de emergencia si el usuario lo desea	Paciente
RF-05	La aplicación móvil debe permitir visualizar información al usuario sobre la última crisis registrada	Paciente
RF-06	La aplicación web permitirá iniciar y cerrar sesión con correo y contraseña	Administrador
RF-07	La aplicación web permitirá al usuario crear credenciales de pacientes para que hagan uso de la aplicación móvil	Administrador
RF-08	La aplicación web permitirá al usuario visualizar el histórico de los registros de las crisis de atendidas	Administrador
RF-09	La aplicación web permitirá al usuario consultar, por	Administrador

	nombre de paciente, los registros de crisis atendidas	
RF-10	La aplicación web permitirá guardar un registro de crisis atendidas una vez se confirme la recepción de la emergencia	Administrador
RF-11	La aplicación móvil deberá notificar a los contactos de emergencia del paciente por SMS y enviar la alerta a la aplicación web del administrador una vez se detecte una crisis con información de la ubicación, valor de ritmo cardiaco y saturación de oxígeno en sangre del momento de la emergencia	Ningún usuario
RF-12	La aplicación web mostrará las notificaciones de emergencia con una alerta visual cuando recién se reciban, priorizando que las más recientes sean las más visibles	Ningún usuario

5.2.2.3 Requerimientos no funcionales. Con el fin de garantizar un prototipo que funcione de forma óptima, se definieron los siguientes requerimientos no funcionales

Tabla 4

Definición de requerimientos no funcionales

No	Nombre	Descripción
RNF-01	Escalabilidad	Diseñar la arquitectura de forma que los servicios y vistas sean modulares y tengan su lógica separada
RNF-02	Usabilidad	Generar interfaces amigables e intuitivas para el usuario
RNF-03	Seguridad	● Los servicios tendrán asignadas autorizaciones por medio de roles ● Los servicios contarán con autenticación por medio de tokens ● Las contraseñas solo serán almacenadas de forma encriptada
RNF-04	Conectividad	● La aplicación web y aplicación móvil requieren conectividad a internet ● La aplicación móvil debe permitir conexión de dispositivos por bluetooth

5.2.3 Análisis de lenguajes de programación y frameworks

Una vez definido por completo el alcance y requerimientos que cubre nuestro proyecto, con una idea clara de las necesidades que se tienen que cubrir, se realizó un análisis minucioso de diversos lenguajes y frameworks, tanto para el back-end, front-end y para la aplicación móvil. Este análisis dio la oportunidad de detectar las tecnologías que mejor se ajustan a nuestras

necesidades y objetivos, teniendo en cuenta factores como la eficacia, la escalabilidad, la comunidad que lo soporta y la capacidad de integración. A continuación, se presenta el análisis en el que se fundamentó la decisión.

5.2.3.1 Tecnologías Front End. Para el análisis de las tecnologías front end, optamos por frameworks o bibliotecas basadas en JavaScript por su fuerte protagonismo y consolidación a nivel de desarrollo web. Exploramos varios frameworks populares para el desarrollo del front end, tales como Angular, Vue.js y React.

- **Angular:** brinda una arquitectura completa y soluciones integradas, lo cual resulta beneficioso para proyectos grandes. Sin embargo, su curva de aprendizaje es más pronunciada y puede ser excesiva para proyectos de tamaño medio a pequeño.
- **Vue.js:** Es un framework progresivo que es fácil de aprender y usar. Aunque es muy flexible, su ecosistema no es tan robusto como el de otros frameworks.
- **React:** React es una biblioteca de JavaScript mantenida por Facebook que se centra en la construcción de interfaces de usuario. Su enfoque en los componentes permite una gran reutilización de código y facilita la gestión del estado de la aplicación. Además, su gran comunidad y la amplia disponibilidad de recursos y bibliotecas de terceros la convierten en una opción muy atractiva.

En este caso tomamos la decisión de optar por React para nuestro desarrollo front end debido a su flexibilidad, la fuerte comunidad de soporte, y la facilidad para integrarse con otras herramientas y librerías. React tiene gran capacidad para manejar interfaces de usuario dinámicas y tiene una buena compatibilidad con una amplia gama de

complementos que nos permite construir aplicaciones escalables y mantenibles de manera eficiente. A su vez, también tuvo un peso importante en la decisión el hecho del conocimiento previo que poseíamos sobre la biblioteca, lo cual facilita en gran medida el tiempo de familiarización con las tecnologías.

5.2.3.2 Tecnologías Back End. Para el back-end, consideramos varias opciones como Express.js, Django, Spring Boot, y Nest JS que abarcan gran variedad de lenguajes como JavaScript, Python y Java, pero con el fin de mantener una línea de desarrollo con el mismo lenguaje, en este caso JavaScript que fue el que seleccionamos por el front-end, optamos por analizar dos de las opciones más fuertes que tiene este para el desarrollo back-end.

- **Express.js:** Es uno de los frameworks más antiguos y populares para Node.js, tiene una gran comunidad. Express.js es conocido por su naturaleza minimalista, lo que permite a los desarrolladores construir aplicaciones web y API de manera muy flexible y personalizada. Además, la curva de aprendizaje es relativamente baja, lo que facilita la incorporación de nuevos desarrolladores al proyecto.
- **NestJS:** Está construido con una arquitectura modular que permite organizar el código en módulos claramente definidos, facilitando el mantenimiento y la escalabilidad de la aplicación. Al estar escrito en TypeScript, NestJS proporciona un tipado fuerte y detección de errores en tiempo de desarrollo, lo que mejora la calidad del código y la productividad del desarrollo. Además, NestJS incorpora conceptos modernos de desarrollo como inyección de dependencias, lo que permite construir aplicaciones robustas y modernas

Elegimos NestJS con TypeScript para nuestro back-end debido a su fuerte tipado, modularidad y facilidad para construir aplicaciones escalables y mantenibles. NestJS aprovecha el ecosistema de Node.js y ofrece una estructura clara y robusta que mejora la productividad del desarrollo.

5.2.3.3 Tecnologías aplicación móvil. Para la elección de las tecnologías para el desarrollo de la aplicación móvil, nos decantamos fuertemente por el desarrollo de aplicación usando React Native, debido a que podíamos aprovechar el conocimiento en React obtenido para el desarrollo del front-end y mantener cierta consistencia al trabajar con un stack tecnológico completo alrededor de JavaScript. Además, el uso de React Native con Expo facilita el desarrollo y despliegue de aplicaciones, reduciendo el tiempo de configuración y sin la necesidad de trabajar en gran cantidad a bajo nivel.

5.2.4 Análisis de base de datos

Otra parte importante para el desarrollo del proyecto es, seleccionar la base de datos más adecuada para nuestro proyecto IoT, se analizó varias opciones populares en el mercado, tanto relacionales como no relacionales.

MySQL ofrece una amplia adopción, soporte robusto y consistencia ACID, pero la rigidez del esquema y las operaciones de escritura intensivas pueden limitar la velocidad, especialmente en un entorno IoT. PostgreSQL proporciona soporte para operaciones complejas, integridad de datos y extensibilidad, aunque su rendimiento puede verse afectado cuando se enfrenta a un alto volumen de escrituras rápidas típicas en aplicaciones IoT. Entre las opciones no relacionales, Cassandra destaca por su escalabilidad horizontal, alta disponibilidad y soporte para grandes volúmenes de datos, pero su configuración y mantenimiento son complejos y presenta una curva de aprendizaje pronunciada. Redis es extremadamente rápido debido a su

naturaleza en memoria, siendo adecuado para almacenamiento en caché y operaciones de datos en tiempo real, pero no es ideal para el almacenamiento persistente de grandes volúmenes de datos debido a la dependencia de la memoria RAM. Sin embargo, MongoDB, con su almacenamiento de documentos flexible, escalabilidad horizontal, y rendimiento rápido en operaciones de lectura y escritura, se adapta perfectamente a la variabilidad de los datos proveniente de sensores. Su naturaleza sin esquema permite almacenar datos heterogéneos y ajustar fácilmente la estructura según sea necesario. Las características de MongoDB están acorde con la necesidad de manejar grandes volúmenes de datos de manera eficiente y adaptable. Por estas razones, hemos decidido optar por MongoDB para nuestro proyecto.

5.2.5 Análisis de proveedores de software y servicios

Algunas de las necesidades que se necesitaban cubrir dentro del desarrollo del proyecto era el envío de mensajes SMS a los contactos del usuario, la plataforma en donde se realizaría el despliegue del software y de la base de datos.

5.2.5.1 Servicios de mensajería. El análisis comienza definiendo qué servicio de mensajería empleamos para poder enviar la información de la emergencia a los contactos registrados por el usuario. A continuación se enuncian cada una de las opciones que se tuvieron en cuenta y el análisis que se hizo para escoger entre alguna de ellas.

- **Twilo.** Twilio es conocido por su alta fiabilidad y capacidad para manejar grandes volúmenes de mensajes sin comprometer la calidad de entrega. Su infraestructura global está diseñada para garantizar la entrega rápida y consistente de SMS. Twilio ofrece una API robusta y flexible que permite una integración fácil y rápida con una amplia gama de aplicaciones y servicios. Twilio ofrece

características avanzadas de seguridad, incluyendo cifrado de datos en tránsito y en reposo, lo que es crucial para proteger la información sensible.

- ***Vonage.*** Nexmo ofrece una excelente cobertura global, permitiendo enviar mensajes SMS a casi cualquier país. Proporciona una API fácil de usar para la integración de SMS en aplicaciones y servicios. Además de SMS, Nexmo ofrece servicios de voz, verificación de números y mensajes de texto para aplicaciones de dos factores.
- **Plivo.** Plivo proporciona una API de SMS robusta que facilita la integración de servicios de mensajería en aplicaciones. Su API es conocida por su facilidad de uso y capacidad para manejar grandes volúmenes de mensajes. Ofrece servicios adicionales como voz, verificación de números y autenticación de dos factores.
- **Amazon SnS.** Amazon SNS es un servicio de notificación que ofrece soporte para SMS, entre otros tipos de mensajes. Es parte del ecosistema de AWS y se integra bien con otros servicios de Amazon.

Todas las opciones analizadas, cuentan con un plan gratuito de prueba, el cual ofrece la ventaja y facilidad de integrar el servicio para ver si se adapta a las características que se esperan de los mismos. La mayoría de servicios, funcionan bajo un costo por mensaje básico. Dentro del plan de prueba se brindan una serie de créditos o dinero con el cual podremos realizar cierta cantidad de mensajes hasta finalizar los créditos. A continuación se presenta una tabla de costo por mensaje y los créditos brindados dentro del plan de prueba.

Tabla 5

Costos por mensaje y créditos en el plan de prueba

	Twilo	Vonage	Plivo	Amazon sns
Precio sms USD	$0.0075	$0.0068	$0.0065	$0.00157
Créditos de prueba	$15 USD	$5 USD	$10 USD	100,000 sms x mes

De la anterior tabla se pudo calcular la cantidad total de mensajes con los cuales podremos hacer uso dependiendo de los créditos que nos brinda el servicio, en este sentido con el plan gratuito de Twilo tenemos la oportunidad de enviar 2000 mensajes en total hasta finalizar los créditos de prueba, con el plan gratuito de Vonage tendremos la posibilidad de enviar 736 mensajes dentro del plan gratuito de prueba, con Plivo podremos enviar 1539 y Amazon ya tienen una cantidad fija de mensajes por mes durante el periodo que dura el periodo de prueba que es alrededor de un año.

De lo anterior es evidente que Twilo y Amazon sns son las dos mejores opciones, considerando a Amazon sns como una opción fuerte por la gran cantidad de tiempo de la prueba gratuita y la gran cantidad de mensajes que podemos enviar mensajes. A pesar de esto, vemos un servicio de Amazon, un reto en la implementación, ya que esto requiere un cierto conocimiento previo en el servicio cloud de Amazon, complicando en cierto nivel la implementación del servicio. Por su parte, Twilo tiene una API robusta y fácilmente implementable sumando a que es una herramienta de la que se tiene previo conocimiento. Es por eso que se decide tomar la opción de Twilo como la opción adecuada para nuestro proyecto.

5.2.5.2 Plataforma de despliegue software. El análisis comienza definiendo qué servicio de mensajería empleamos para poder enviar la información de la emergencia a los contactos registrados por el usuario. A continuación se enuncian cada unas de las opciones que se tuvieron en cuenta y el análisis que se hizo para escoger entre alguna de ellas.

- ***AWS.*** AWS ofrece una amplia gama de servicios que permiten desplegar desde simples aplicaciones web hasta arquitecturas complejas y escalables. AWS ofrece un nivel gratuito durante 12 meses con acceso limitado a servicios como EC2, S3, y RDS. Los costos después del nivel gratuito dependen del uso y pueden ser elevados.
- ***DigitalOcean.*** Es conocido por su simplicidad y precios competitivos, DigitalOcean ofrece una plataforma amigable para desarrolladores. A pesar de esto, no ofrece un plan gratuito, pero tiene precios iniciales bajos que son atractivos para pequeñas y medianas aplicaciones. Digital Ocean cuenta con un panel de control intuitivo y fácil de usar.
- ***Railway.*** Railway es conocido por su enfoque en la simplicidad y la automatización del despliegue y la gestión de aplicaciones. Ofrece un plan gratuito con $5 en créditos, lo que permite probar y desplegar aplicaciones pequeñas sin costo inicial. Los planes de pago son flexibles y escalan según las necesidades del proyecto. El proceso de despliegue es extremadamente sencillo y rápido, ideal para desarrolladores que buscan una experiencia sin complicaciones.

Después de evaluar cuidadosamente todas las opciones, decidimos que Railway es la mejor plataforma para nuestras necesidades de despliegue. Aunque plataformas como AWS ofrecen características avanzadas y robustez, Railway destaca por su combinación de simplicidad, precios competitivos y un plan gratuito generoso. Además, Railway proporciona una experiencia de usuario sencilla, facilitando el proceso de despliegue y la gestión de aplicaciones.

5.2.5.3 Plataforma de despliegue base de datos. Para la elección de la opción para el despliegue de MongoDB, se tomó como opción el servicio en la nube que brinda Mongo conocido como Mongo Atlas. En el plan gratuito, se obtiene una instancia de base de datos compartida de bajo rendimiento. La instancia es adecuada para proyectos pequeños y de desarrollo, pero no para aplicaciones de producción de gran escala. En este plan gratuito se brindan 512 MB de uso de disco, lo cual es suficiente para el caso de uso de nuestro proyecto.

5.2.6 Análisis componentes hardware

5.2.6.1 Placa de desarrollo. Para el análisis de las placas de desarrollo se partió de las necesidades que tenía que cumplir este, con el fin de darle cumplimiento a los objetivos planteados. En este caso se necesita una placa de desarrollo que tenga un tamaño considerablemente pequeño para garantizar portabilidad, debe tener bluetooth para garantizar una conectividad por esta vía, debe tener la capacidad de gestionar una fuente de alimentación portátil(batería) para garantizar la autonomía del dispositivo. Teniendo

en cuenta esto, se tuvieron en cuenta las siguientes placas con el fin de seleccionar el adecuado para nuestro proyecto.

- **Esp32 38 pines Wroom 32**

  Procesador: Dual-core Tensilica LX6 microprocessor, con una frecuencia de reloj de hasta 240 MHz.

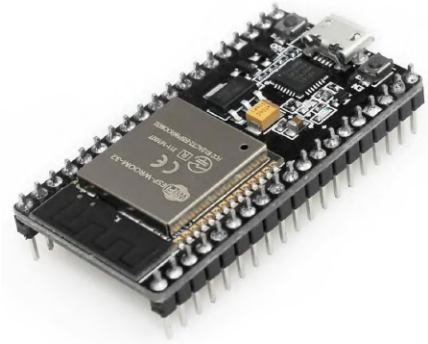
  Memoria: 520 KB de SRAM y 4MB de flash.

  Conectividad: Wi-Fi 802.11 b/g/n y Bluetooth 4.2 (clásico y BLE).

  GPIO: 38 pines de entrada/salida, ofreciendo múltiples opciones de conexión para sensores y actuadores.

Figura 4.

ESP 32 38 pines Wroom 32



- **ESP32 D1 Mini**

  Procesador: Similar al ESP32 estándar, con un dual-core Tensilica LX6 microprocessor.

  Memoria: 520 KB de SRAM y 4MB de flash.

Conectividad: Wi-Fi 802.11 b/g/n y Bluetooth 4.2 (clásico y BLE).

GPIO: Menos pines comparado con la versión de 38 pines, pero aún suficiente para muchos proyectos.

Figura 5.

ESP 32 D1 Mini

- **Raspberry pi pico**

Procesador: ARM Cortex-M0+ dual-core, con una frecuencia de reloj de hasta 133 MHz.

Memoria: 264 KB de SRAM y 2 MB de flash.

Conectividad: No incluye conectividad Wi-Fi o Bluetooth integrada, requiere módulos adicionales.

GPIO: 26 pines de entrada/salida, ofreciendo buenas opciones de conexión para sensores y actuadores.

Figura 6.

Raspberry pi pico

- **Esp32 TTGO T Display**

Procesador: Dual-core Tensilica LX6 microprocessor, con una frecuencia de reloj de hasta 240 MHz.

Memoria: 520 KB de SRAM y 4MB de flash.

Conectividad: Wi-Fi 802.11 b/g/n y Bluetooth 4.2 (clásico y BLE).

Pantalla: Pantalla LCD integrada de 1.14 pulgadas con resolución de 135x240 píxeles.

GPIO: Suficientes pines de entrada/salida para la mayoría de proyectos.

Batería: Módulo integrado para conectar y cargar baterías LiPo.

Figura 7.

Esp32 TTGO T Display

Después de considerar todas las opciones, decidimos que el ESP32 TTGO T Display es la mejor opción para nuestro proyecto. Su pantalla LCD integrada proporciona una solución conveniente para la visualización de datos e información relevante, sin la necesidad de componentes adicionales. Además, su capacidad de conectar y cargar baterías LiPo facilita la creación de proyectos con autonomía en su uso. Finalmente, cuenta con la conectividad bluetooth necesaria.

5.2.6.2 Sensores. Para el caso del sensor encargado de tomar las mediciones de ritmo cardiaco y el sensor que permite identificar patrones de movimientos corporales, acatamos las sugerencias de nuestro director el cual con base en su experiencia ya tenía conocimientos previos de cuáles podrían ser adecuados teniendo el alcance y contexto del proyecto. De esta forma, los sensores elegidos fueron los siguientes.

- **Max30102**

  Tecnología: Sensor óptico de oxímetro de pulso y ritmo cardíaco.

Mediciones: Proporciona datos de ritmo cardíaco y saturación de oxígeno en sangre (SpO2).

Consumo de Energía: Bajo consumo de energía, ideal para dispositivos portátiles.

Interfaz: Comunicación I2C, fácil integración con microcontroladores.

Tamaño: Compacto, adecuado para aplicaciones de monitoreo de salud portátiles.

Figura 8.

Max30102

- **Acelerómetro y giroscopio MPU6050**

Ejes: Acelerómetro y giroscopio de 3 ejes.

Consumo de Energía: Bajo consumo de energía, adecuado para dispositivos portátiles.

Interfaz: Comunicación I2C, fácil integración con microcontroladores.

Integración: Sensor de movimiento integrado con procesamiento digital de movimiento (DMP).

Figura 9.

Acelerómetro y giroscopio MPU6050

5.3 Definición de la arquitectura

La definición de la arquitectura es una fase crucial en la que establecimos el diseño detallado del sistema, cubriendo tanto los aspectos de software como de hardware. Acá buscamos asegurar que todos los componentes estuvieran claramente definidos y que su integración fuera eficiente y coherente. Se dividió en varias sub fases para abordar diferentes aspectos del diseño.

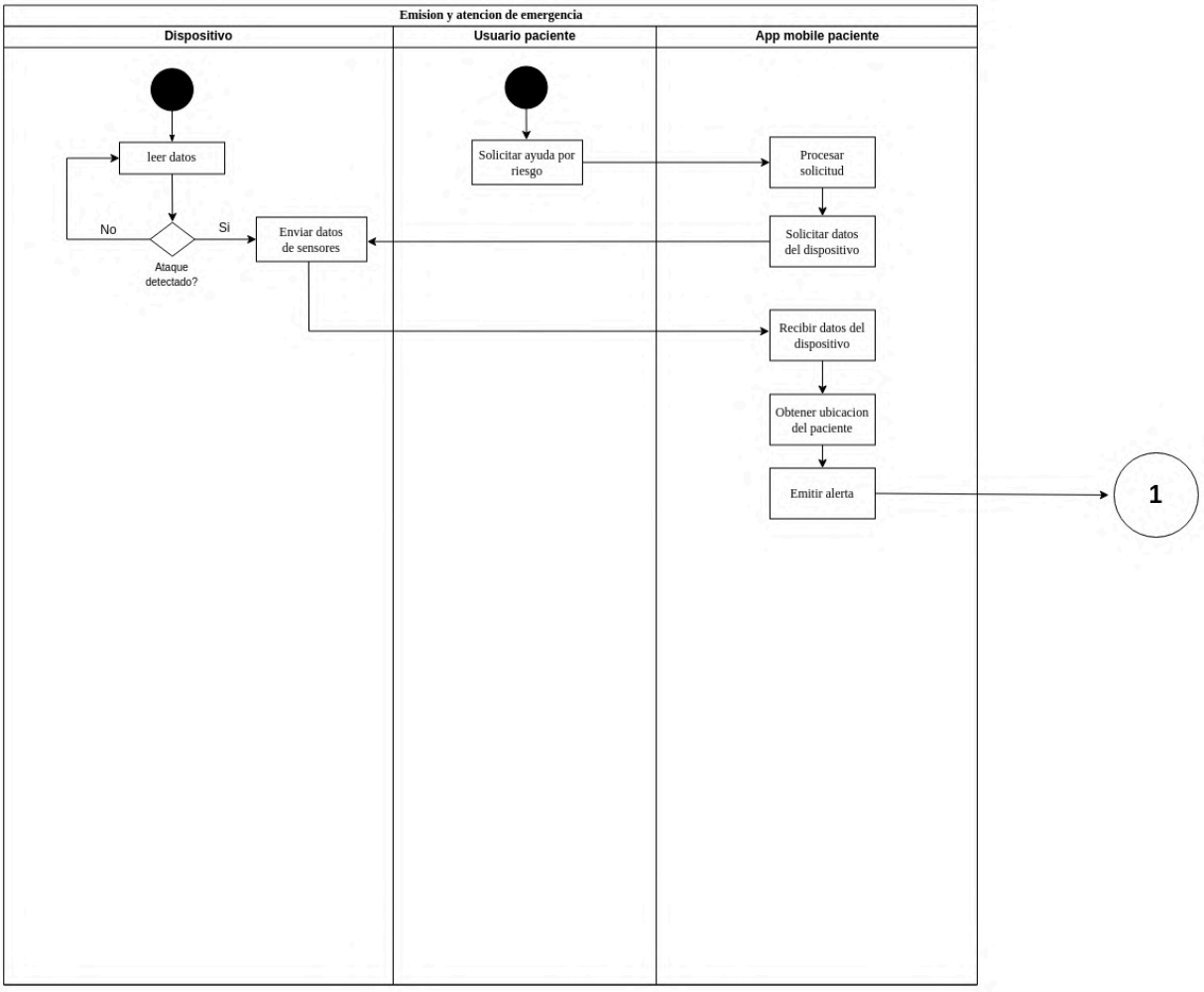
5.3.1 Diseño software

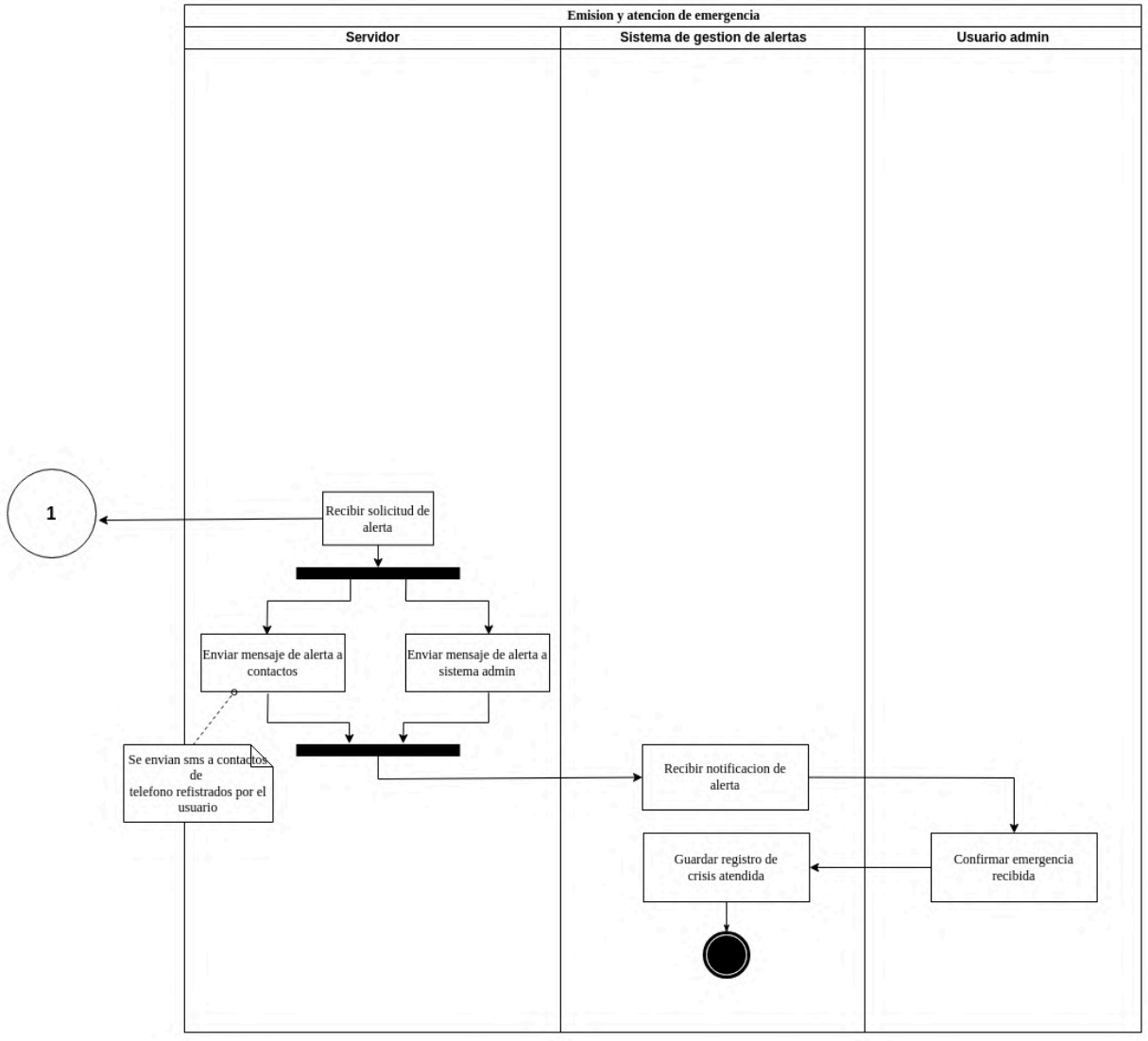
5.3.1.1 Diagrama de procesos. Habiendo planteado nuestros requerimientos, realizamos una visualización de los flujos de trabajo y procesos clave dentro del sistema, esto con el fin de facilitar la comprensión clara de cómo interactúan los diferentes componentes del

sistema y asegurando una implementación coherente. Este diagrama evidencia todo el proceso que se realiza al momento de emitir y atender una emergencia.

Figura 10.

Diagrama de procesos.



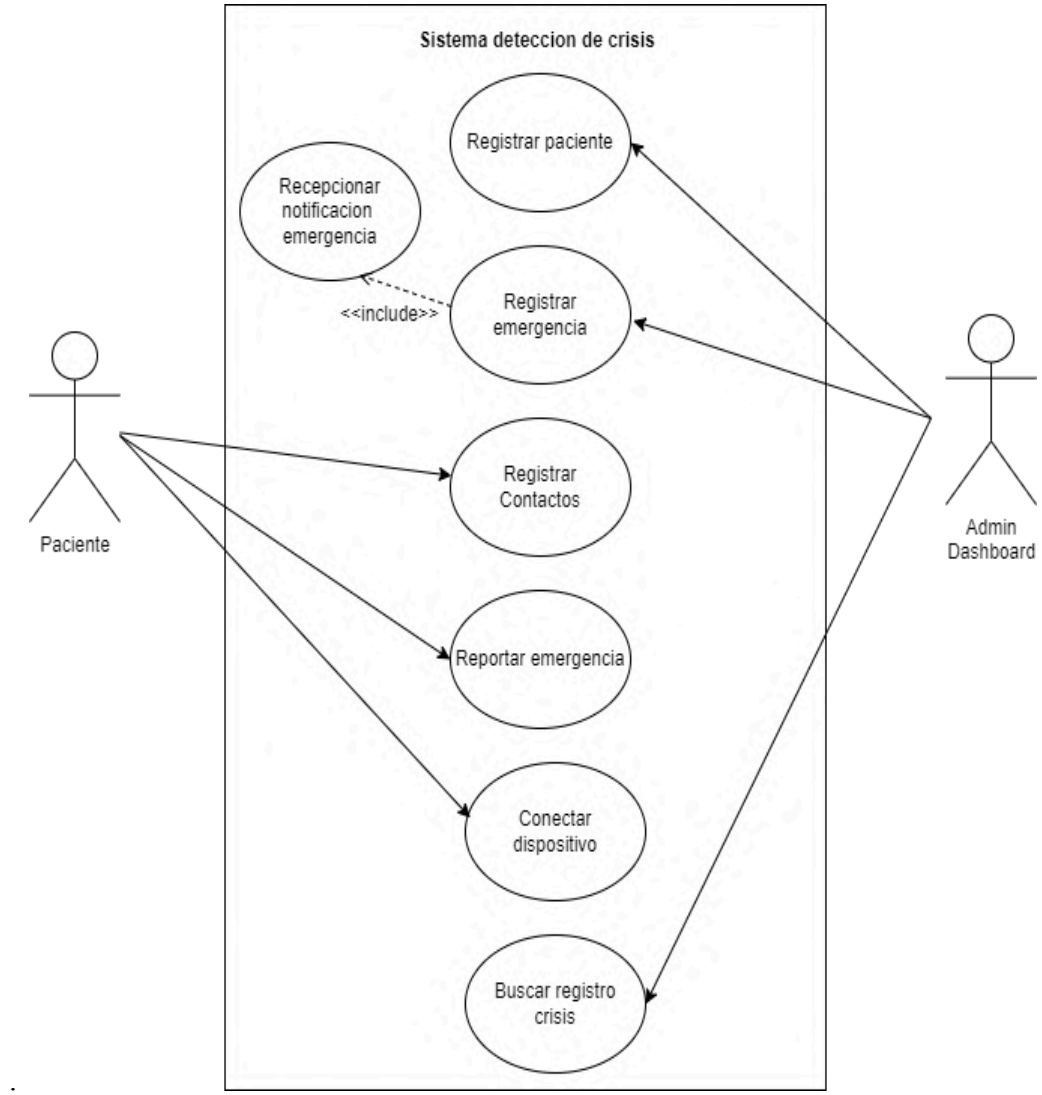


Nota. El diagrama de procesos fue dividido en dos partes para facilitar su visualización

5.3.1.2 Diagrama de casos de uso. En este plantea el desarrollo del escenario principal que describe cómo los roles asignados interactúan con el sistema para cumplir con sus respectivos objetivos.

Figura 11.

Diagrama de casos de uso.

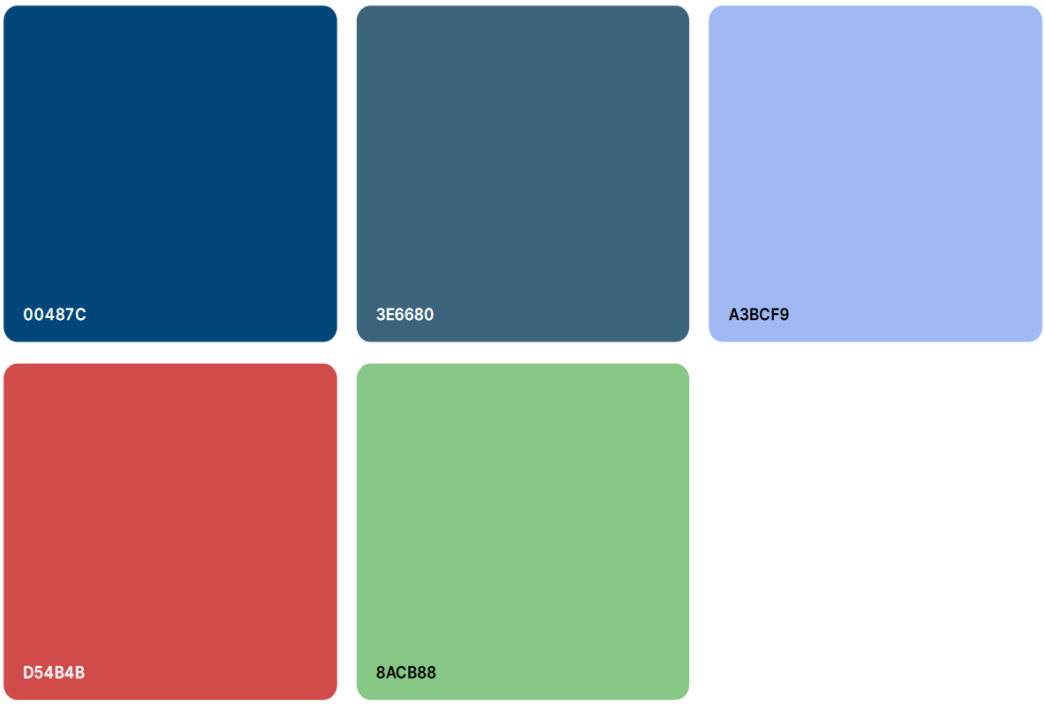


5.3.1.3 Diseño gráfico de la aplicación móvil y dashboard web.

5.3.1.3.1 Definición de una línea gráfica y guía de estilo. Planteamos una serie de directrices visuales con las cuales se pudiera mantener una apariencia coherente y armoniosa en las respectivas interfaces de usuario. Esto incluye la elección de una paleta de colores para la aplicación móvil como para el dashboard web, la cual fue elegida con base en colores pasteles con tonalidades azules, que tienen relación con el contexto de la salud. La elección de colores suaves y calmantes busca transmitir tranquilidad y confianza, elementos de suma importancia en el contexto médico. Además, la elección de la tipografía e iconos fueron seleccionados con el objetivo de ser lo más sencillo e intuitivos posibles.

Figura 12.

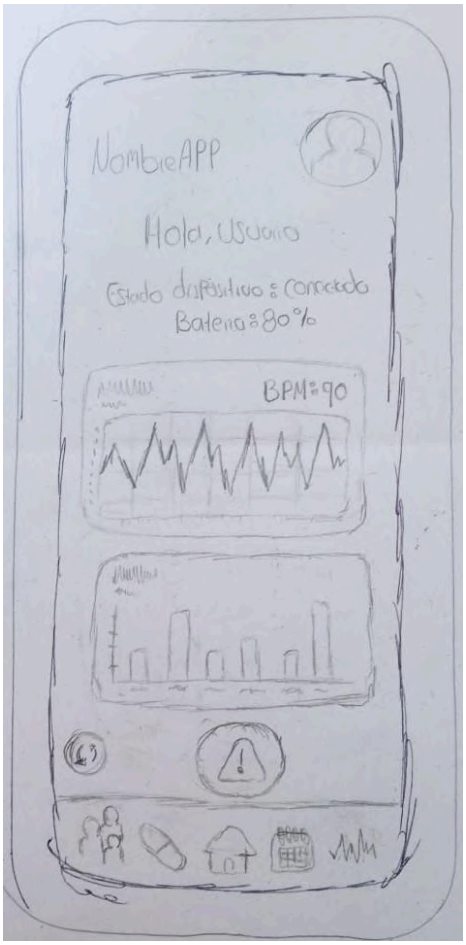
Paleta de colores usada en la línea gráfica del proyecto.



Antes de pasar a desarrollar los prototipos visuales, se diseñó como un boceto a mano de cómo podría llegar a verse o que debería tener la pantalla de inicio en la aplicación móvil, esto con el fin de tener claro cuáles son los datos y funcionalidades importantes para el usuario.

Figura 13.

Primer boceto a mano de la idea gráfica de la pantalla principal

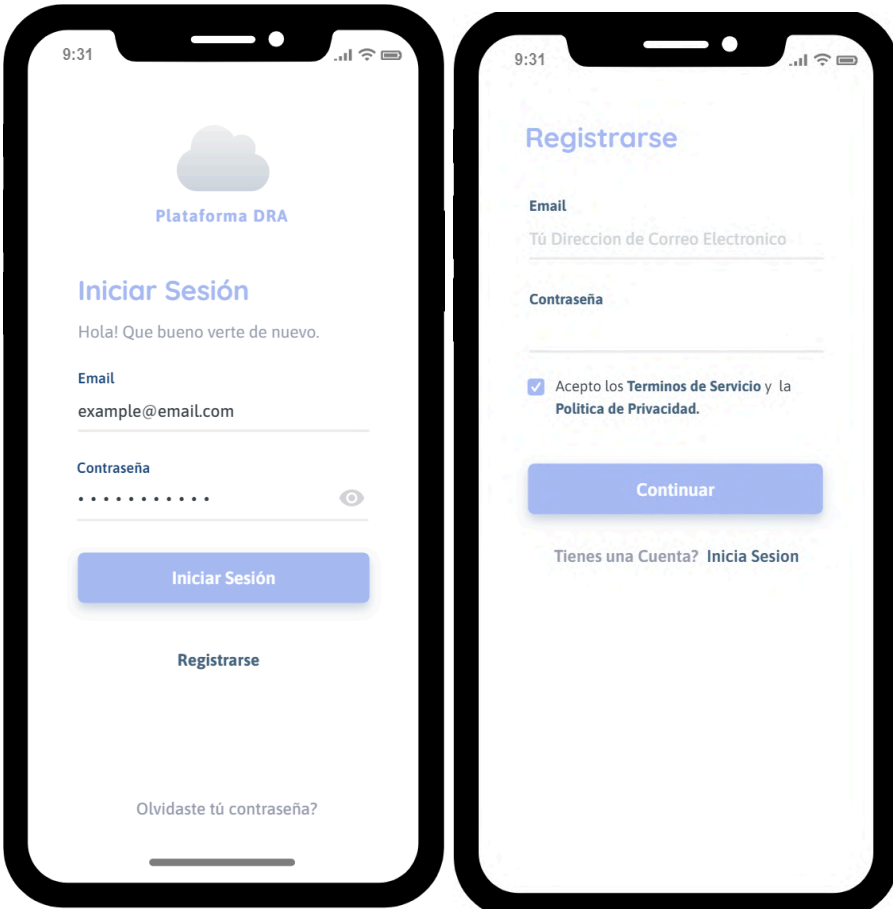


5.3.1.3.2 Desarrollo de mockups. En esta fase se crearon prototipos visuales (mockups) en la aplicación Moqups, la cual permitió realizar los prototipos con una gran cantidad de detalles, tanto de la aplicación móvil como del dashboard web. Para la aplicación móvil,

los mockups incluyen la pantalla de inicio de sesión, la pantalla de registro, la pantalla que muestra principal, la cual busca mostrar la información necesaria del usuario y todos sus detalles importantes, así como la pantalla de contactos. Esto nos sirvió para tener una evaluación temprana del diseño y la funcionalidad.

Figura 14.

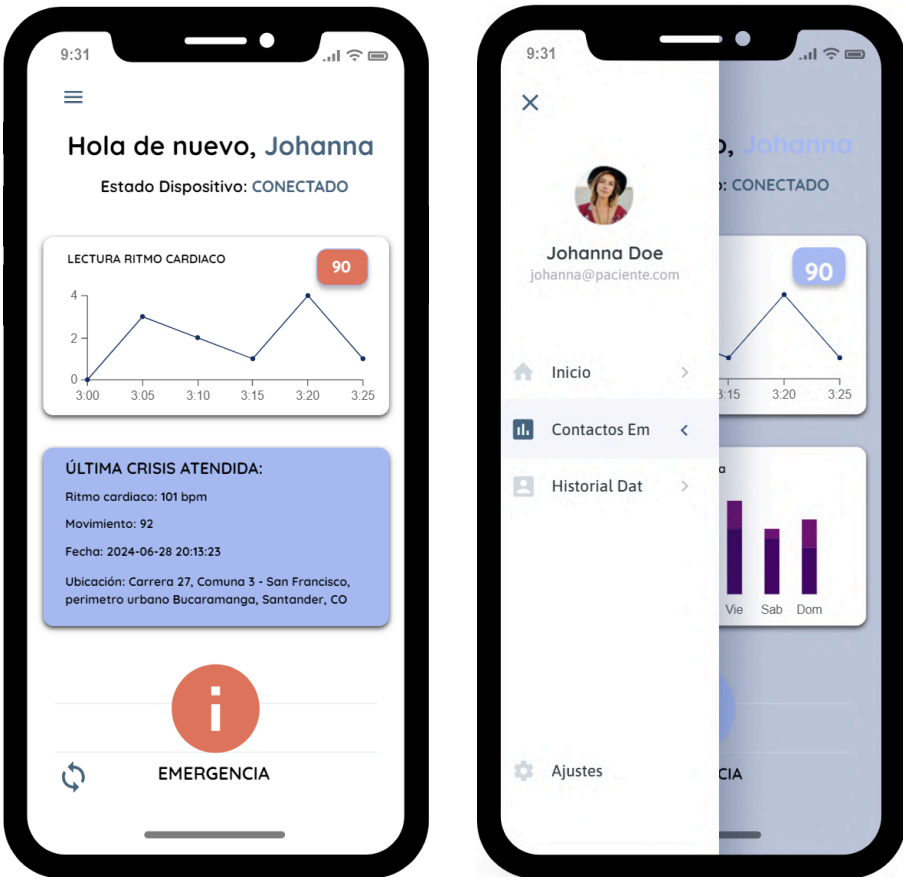
Mockups de las pantallas de inicio en la aplicación



Nota. Mockups pantalla de inicio de sesión y pantalla de registro respectivamente.

Figura 15.

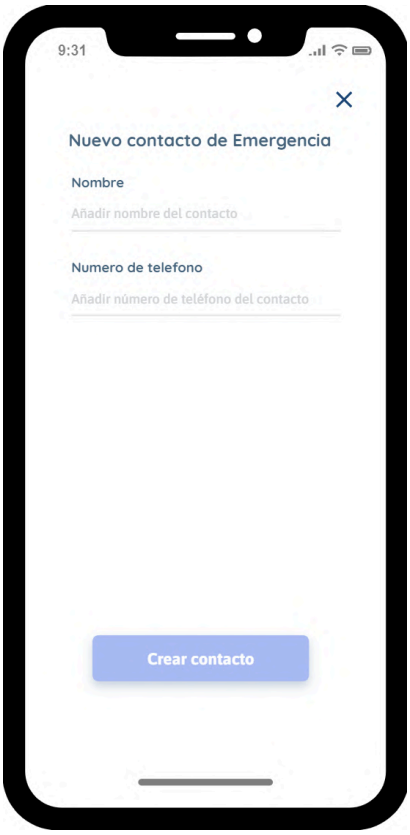
Mockups de pantallas luego de realizar un inicio de sesión



Nota. Mockups de la pantalla principal o "home" de la aplicación y el menú de navegación correspondiente

Figura 16.

Mockup de pantalla de contactos y funcionalidades



Nota. Mockup correspondiente a la pantalla de contactos y las respectivas funcionalidades que esta presenta, desde editar o eliminar, hasta crear un nuevo contacto.

Para el dashboard web, los mockups incluyen la pantalla de inicio de sesión y la pantalla principal que muestra toda la información y funciones relevantes para el uso médico.

Figura 17.

Mockup inicio de sesión en el panel del administrador del servicio médico

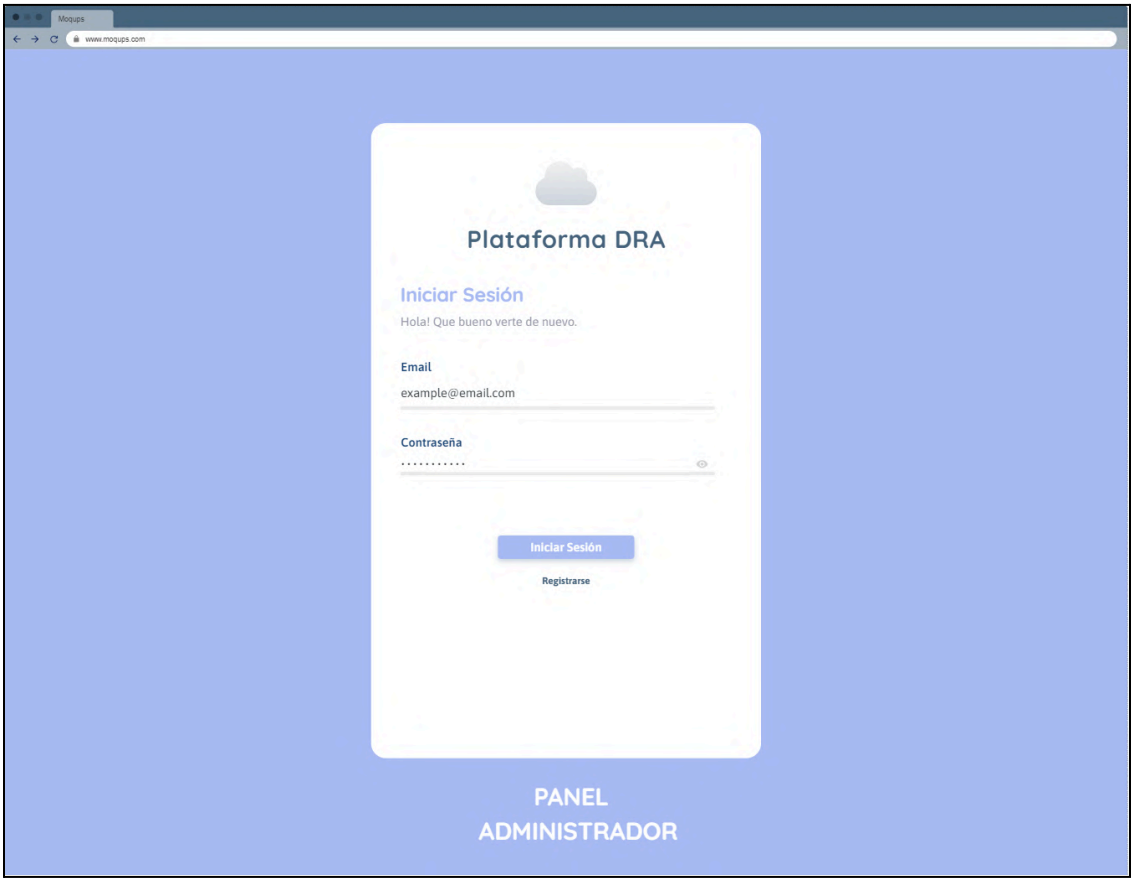
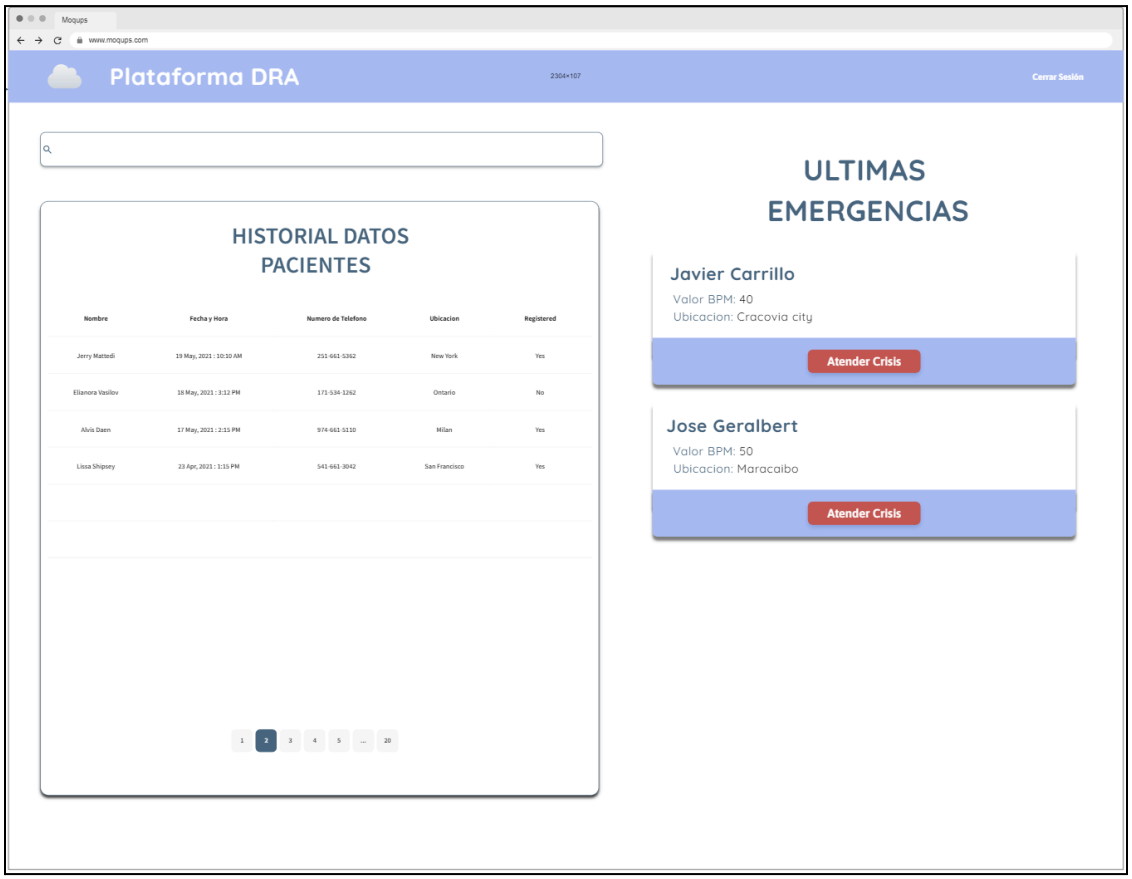


Figura 18.

Mockup de la pantalla de inicio con las respectivas funcionalidades en el panel de administrador.

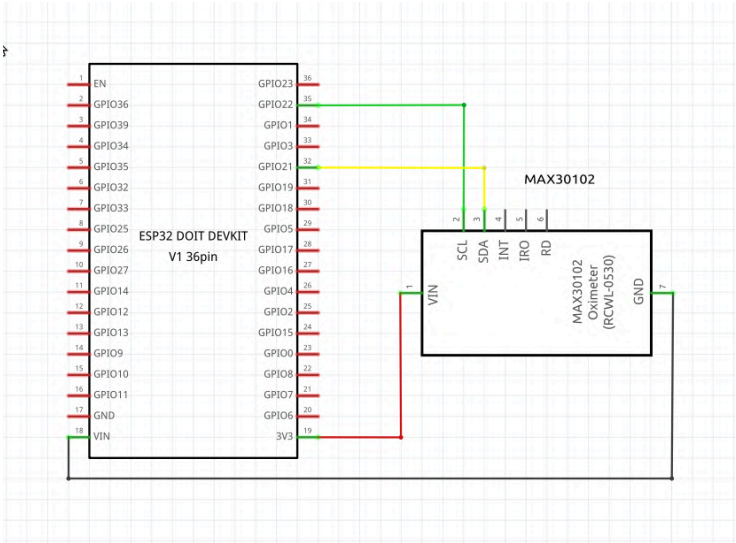


5.3.2 Diseño del hardware

5.3.2.1 Diseño conectividad componentes electrónicos. Los dos sensores, tanto el max 30102 y mpu6050 cuentan con protocolo de comunicación I2C, por lo tanto, la conexión con el microcontrolador se hizo por medio de este protocolo. A continuación se evidencia el esquema de conexiones de cada sensor con la placa de desarrollo.

Figura 19.

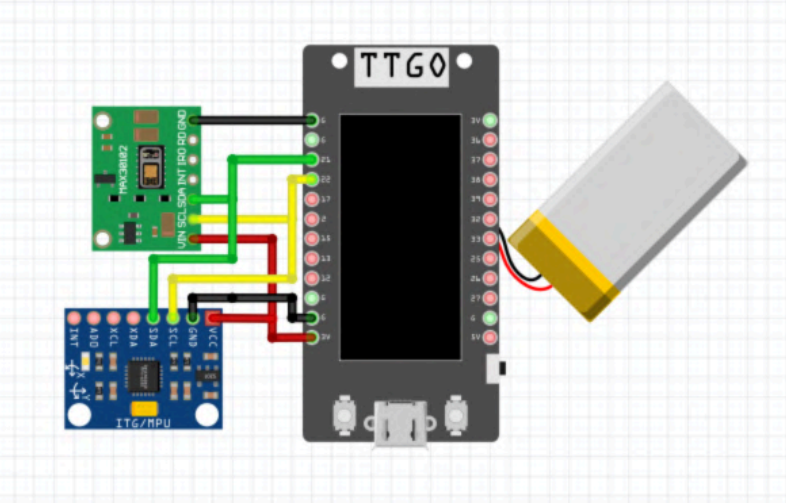
Esquema de conexiones de los sensores con la placa de desarrollo



Finalmente, ya planteado el esquema de conexiones individuales, fue posible realizar el esquema de conexión general, el cual alberga todos los componentes del dispositivo, incluido la batería.

Figura 20.

Esquema de conexión general



5.3.2.2 Diseño del modelo 3D para el dispositivo electrónico. Se diseñaron modelos 3D del dispositivo portátil, esto con el fin de asegurar su ergonomía, su funcionalidad y que sea estéticamente agradable y cómodo para el usuario final. Este diseño abarcó tanto el exterior del dispositivo como la disposición interna de los componentes. Este diseño del dispositivo se ha conceptualizado en dos partes: una parte similar a un reloj que es la encargada de contener el acelerómetro, giroscopio y microcontrolador, y la otra parte es una especie de pinza, la cual va sujeta al dedo índice del usuario, esta contiene el sensor de ritmo cardiaco. Todo esto conectado por un cable que toma los datos del sensor de ritmo cardíaco y los transmite al microcontrolador.

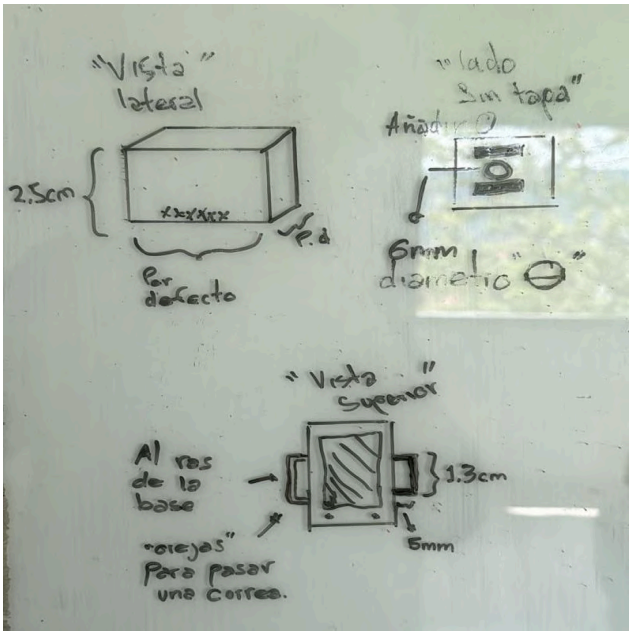
Se tomó esta decisión para optimizar la precisión de las mediciones del sensor de ritmo cardíaco. El sensor, que utiliza un LED infrarrojo para identificar la circulación de sangre, necesita estar en constante contacto con un medio que permita identificar fácilmente el paso de sangre. El uso del sensor en la muñeca puede generar medidas

imprecisas debido a los movimientos variados que pueden hacer que el sensor pierda contacto. Además, la ubicación en la muñeca puede dificultar la obtención de mediciones precisas. Por lo tanto, se decidió ubicar el sensor en el dedo índice del paciente para asegurar un contacto directo y obtener datos de forma precisa y estable.

Primero, se optó por realizar los bocetos a mano de la parte “reloj” del dispositivo en el que se permitiera visualizar o plasmar las ideas de cómo esperamos que fuera estéticamente el dispositivo. Esto con el fin que fuera como una especie de reloj inteligente y a su vez fácil de transportar en la cotidianidad del usuario, al realizar el respectivo boceto se tienen en cuenta las medidas de los sensores, microcontroladores y demás dispositivos que deben ir dentro del diseño, para así tener claras las medidas y en su caso tener un margen de error si llega a ser necesario.

Figura 21.

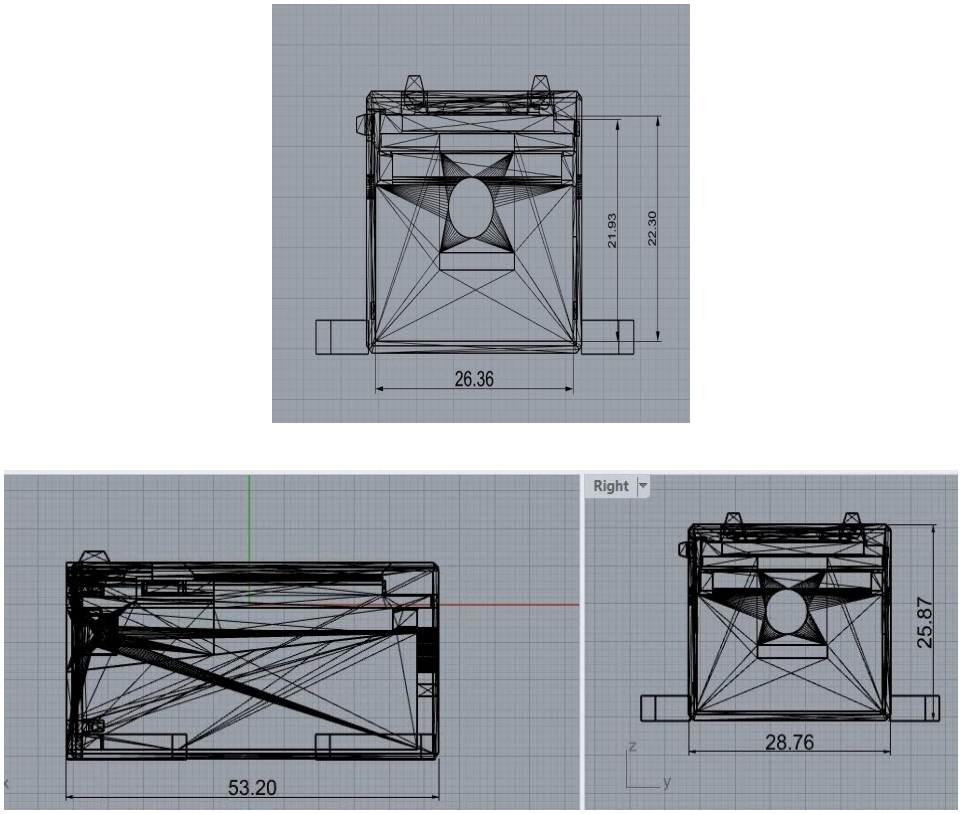
Boceto a mano de la idea de diseño del prototipo



Tras realizar estos bocetos y con las medidas presentes, se opta por llevar dicha idea a un modelado en 3D, partiendo de la generación de las respectivas medidas y demás ajustes que vayan apareciendo durante el proceso, hasta la respectiva culminación e impresión de los modelos.

Figura 22.

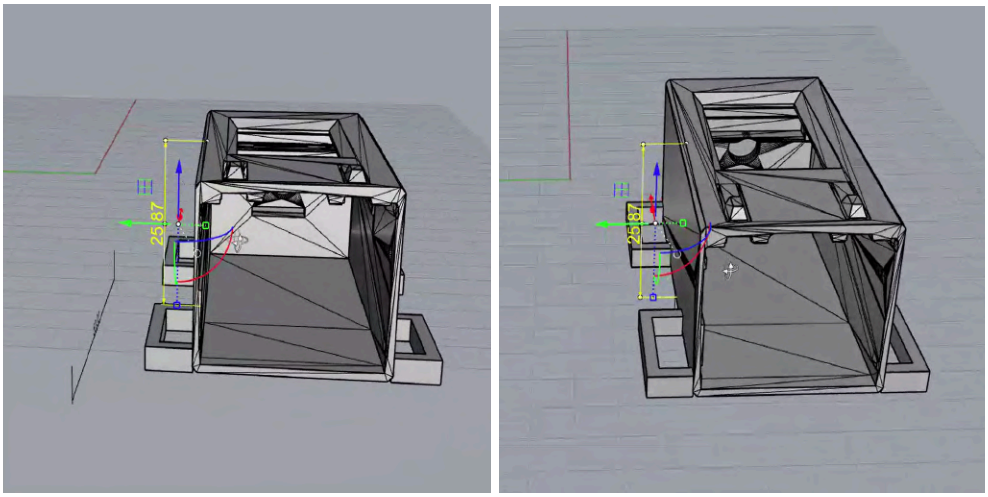
Medidas internas y externas del modelo



Nota. Medidas internas del modelo, medidas externas del modelo y caras laterales y frontales.

Figura 23.

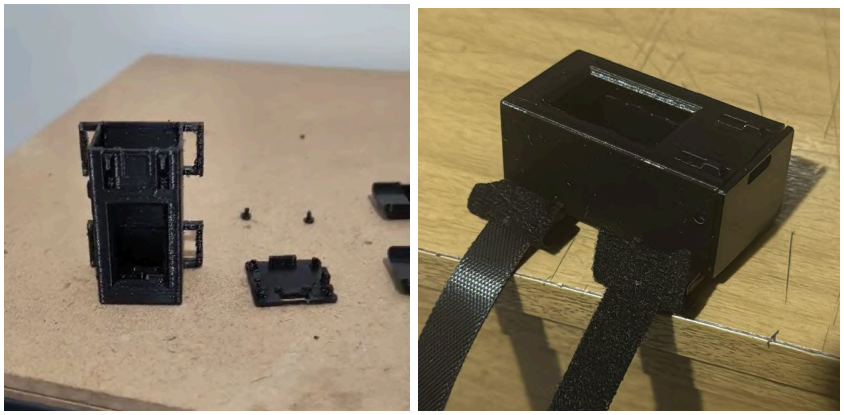
Modelo 3D



Ya con las medidas y ajustes realizados, como el de añadir dos “orejas” en cada cara lateral para que le dé mayor rigidez y estabilidad a la hora de colocar el dispositivo en la respectiva zona correspondiente y modificar el tamaño externo para dar un poco de margen en dado caso de ser necesario. Se procede a realizar la impresión del modelo, esta impresión se hizo en un color negro, esto con el fin de que sea lo menos llamativo posible, porque al buscar la comodidad del usuario tampoco queremos que sea notorio que tiene un dispositivo médico en su muñeca.

Figura 24.

Impresión y disposición final del modelo 3D



Nota. Impresión y disposición final del modelado 3D, sujeto a unas cintas de velcro para añadir estabilidad.

Ahora, para la segunda parte que complementa el dispositivo, tomamos la idea de los oxímetros portátiles. Estos se basan en una “pinza” que tiene el sensor y toma los datos en un determinado tiempo, con esto en mente optamos por seguir esta idea. Un diseño el cual consta de dos partes que estarán unidas por un resorte y un par de tornillos , esto con el fin de que se genere una presión agradable y manejable a la hora de usar el dispositivo. También este diseño tiene su respectivo campo para soldar y asegurar el sensor de ritmo cardiaco, garantizando un contacto adecuado con el dedo del usuario, esto conectado por un cable dirigido a la parte del dispositivo en la muñeca.

Figura 25.

Modelo 3D "pinza" del dedo

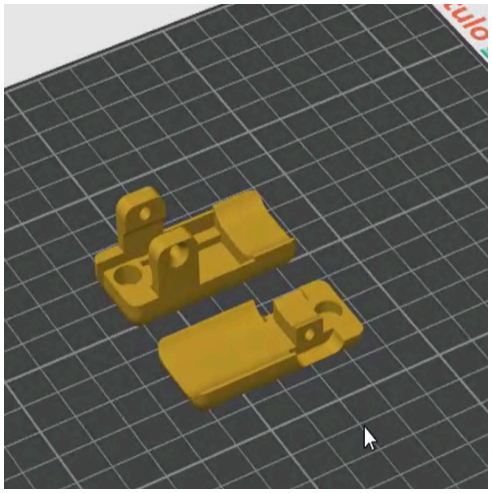


Figura 26.

Impresión y disposición final del modelo 3D "Pinza"

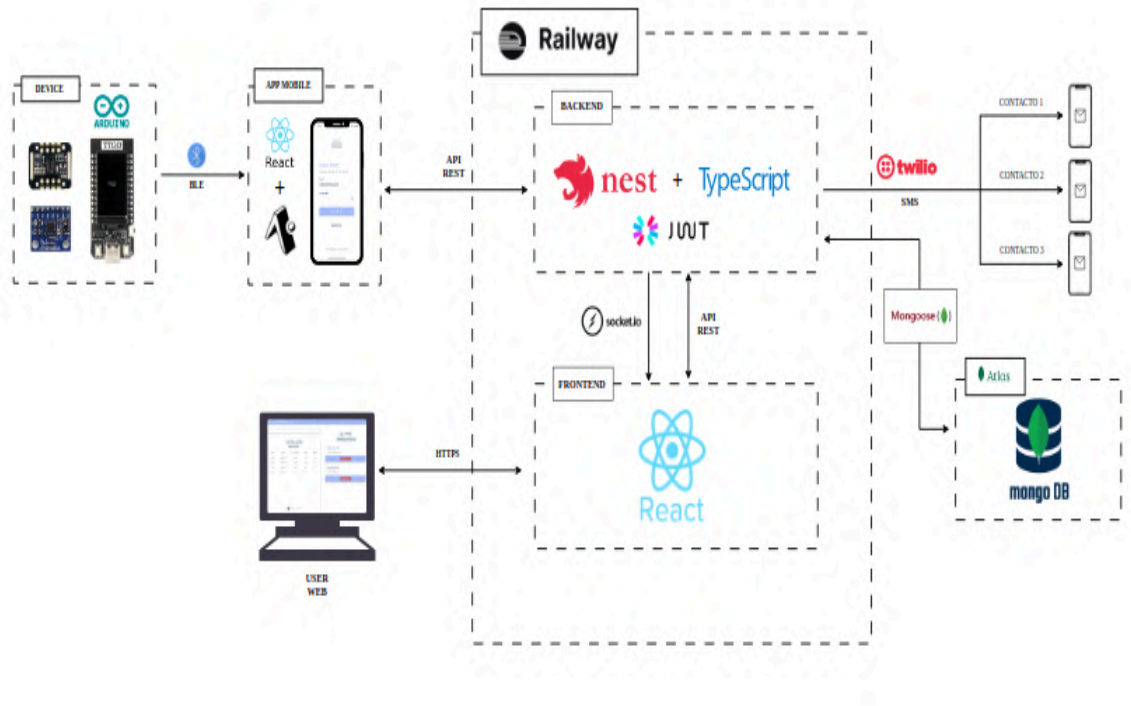
Nota. Modelo “pinza” luego de ser impreso y ubicar el respectivo resorte y tornillo.

5.3.3 Diseño de la arquitectura

A continuación se muestra como fue el planteamiento de la arquitectura final, teniendo en cuenta todo lo descrito hasta el momento.

Figura 27.

Diagrama de arquitectura.



5.4 Desarrollo del prototipo

En esta fase del desarrollo del prototipo es fundamental convertir el diseño teórico de la arquitectura en una implementación práctica y funcional. En esta fase buscamos incluir tanto el desarrollo del software como del hardware, asegurando que todos los componentes se integraran adecuadamente.

5.4.1 Desarrollo del software

Nos centramos en la correcta implementación de los componentes software necesarios para el funcionamiento de la plataforma IoT.

5.4.1.1 Desarrollo Back-end.

5.4.1.1.1 Desarrollo de API. Como se mencionó anteriormente en la sección del análisis, el framework de desarrollo para el back-end escogido fue NestJS. El inicio del proyecto fue realizado con ayuda del CLI que tiene este framework el cual ya brinda comandos para crear nuevos proyectos.

Figura 28.

Instalación Nest.Js

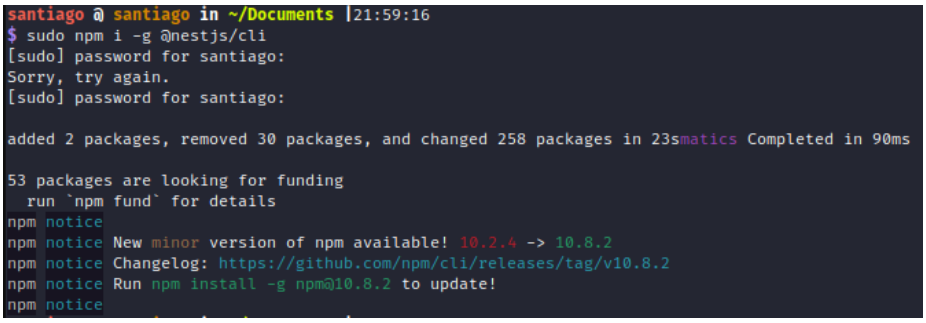


Figura 29.

Creación de un nuevo proyecto con Nest.js

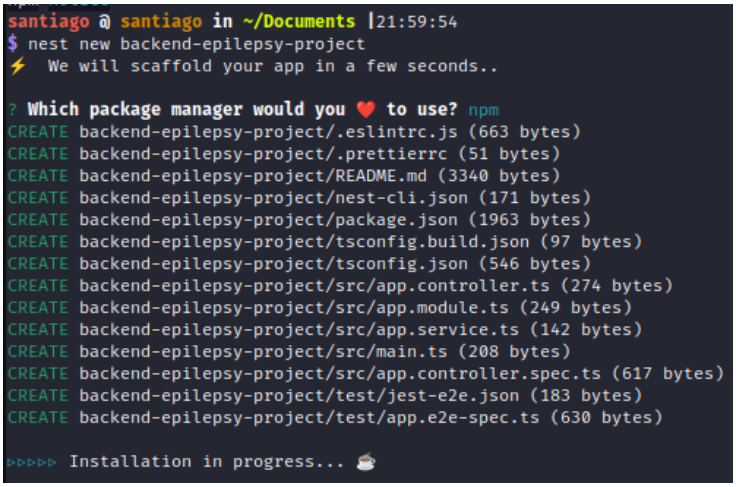
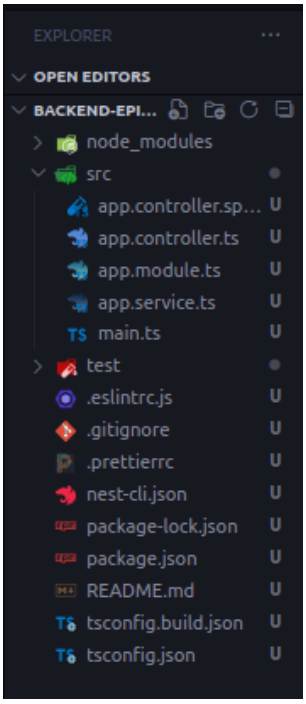


Figura 30.

Carpeta raíz del proyecto recién creado.



En este caso, los archivos más importantes creados son con extensión ts, haciendo referencia al uso de Typescript que hace NestJS para el tipado.

El archivo main.ts es el archivo de entrada o principal que crean la instancia de una aplicación Nest.

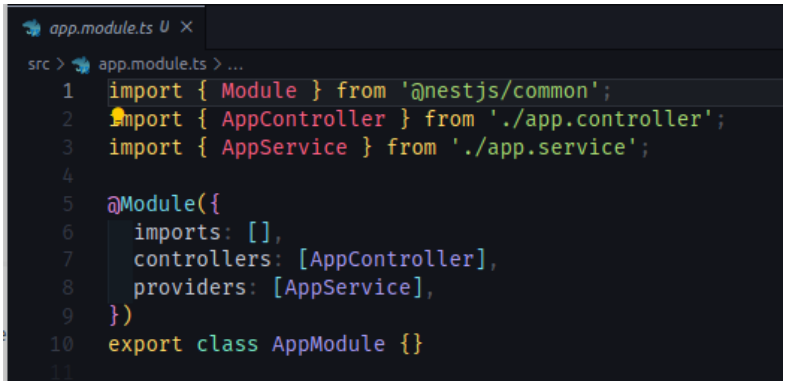
Figura 31

Creación instancia de una aplicación Nest

El archivo app.module.ts es el módulo raíz de la aplicación. Nest hace uso del principio de desarrollo modular. Estos módulos representan una parte o sección del proyecto en específico.

Figura 32.

Representación de módulos en el proyecto



Cada módulo tiene asociados controladores, los cuales servirán como punto de entrada o rutas, las cuales serán consultadas por el cliente para acceder a alguna funcionalidad en específico. En este caso, el archivo app.controller.ts hace referencia a estos.

Figura 33.

Controladores asociados a los módulos

Cada controlador tiene asociado un servicio, el cual es ejecutado cuando el cliente accede a la ruta definida del controlador. En este caso la ruta raíz por el método Get ejecuta el servicio app service que contiene la función getHello(), esta función se encuentra dentro del archivo app.service.ts.

Figura 34.

Ejecución del servicio que contiene la función getHello()

A partir de esta configuración inicial, comienza el desarrollo del proyecto. Para este caso, se trató de implementar principios de desarrollo de software con la intención de

mantener una estructura que se pudiera considerar escalable y mantenible. Se trató de seguir el principio de única responsabilidad en el cual se desarrollan los módulos, garantizando que se encarguen únicamente del contexto propio para el cual fue construido y no se extralimite implementando funcionalidades que no le corresponden.

Figura 35.

Modulos back-end

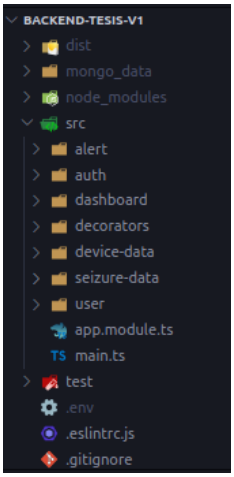
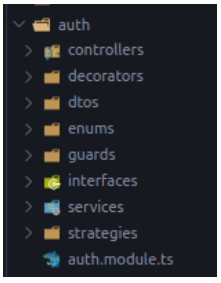


Figura 36.

Distribución de archivos para el módulo de autenticación



Para el desarrollo, como se comentó con anterioridad, la base de datos seleccionada fue MongoDB. Para la interacción entre la base de datos y el entorno de ejecución de desarrollo del back-end se hizo uso de la librería Mongoose.

Figura 37.

Fragmento de conexión de NestJS con MongoDB

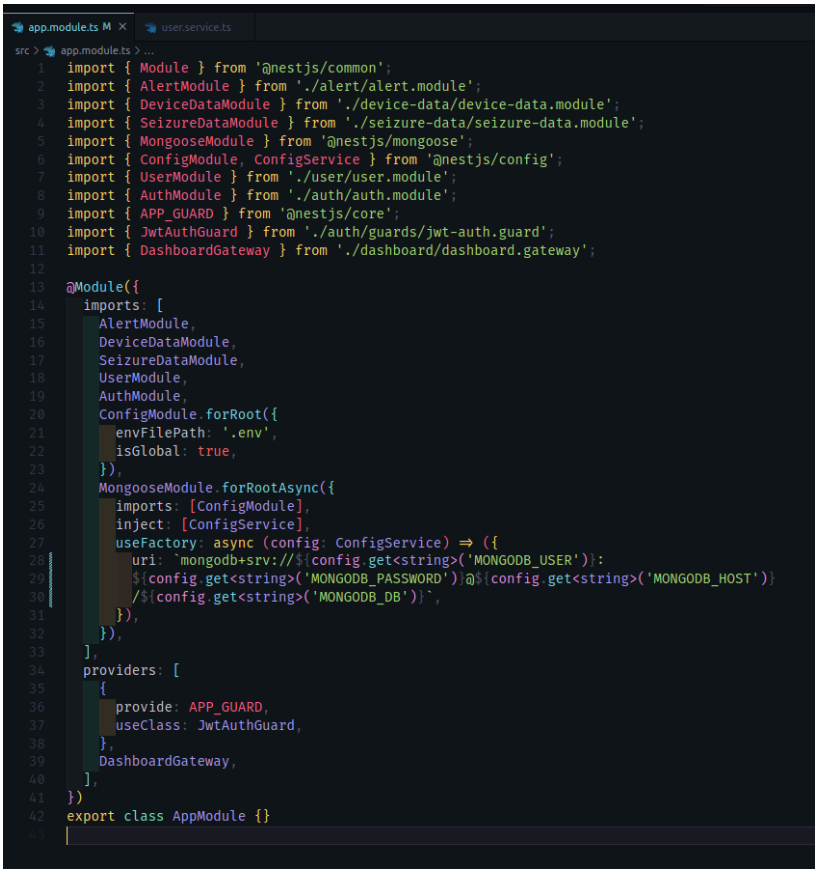
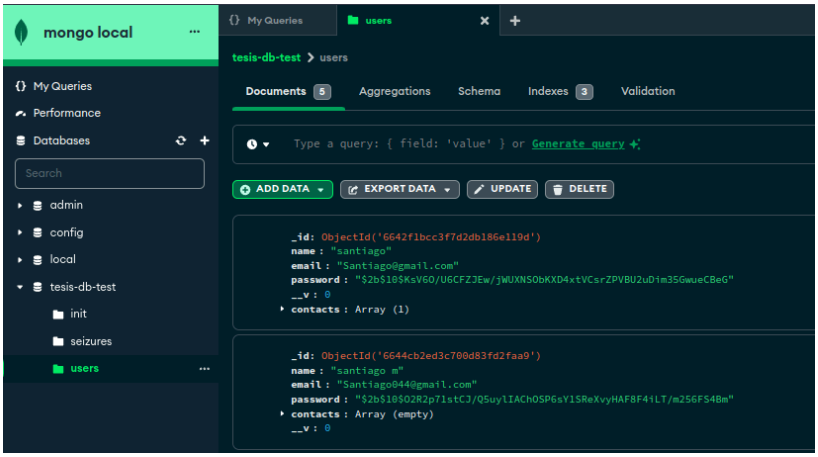


Figura 38.

Captura base de datos MongoDB usando gestor mongo compass



Otra parte importante del desarrollo del back-end, es la autenticación y autorización por medio de estrategias con tokens y roles, en este caso, se hizo uso de JWT y protección de rutas con roles.

Figura 39.

Fragmento de código estrategia JWT

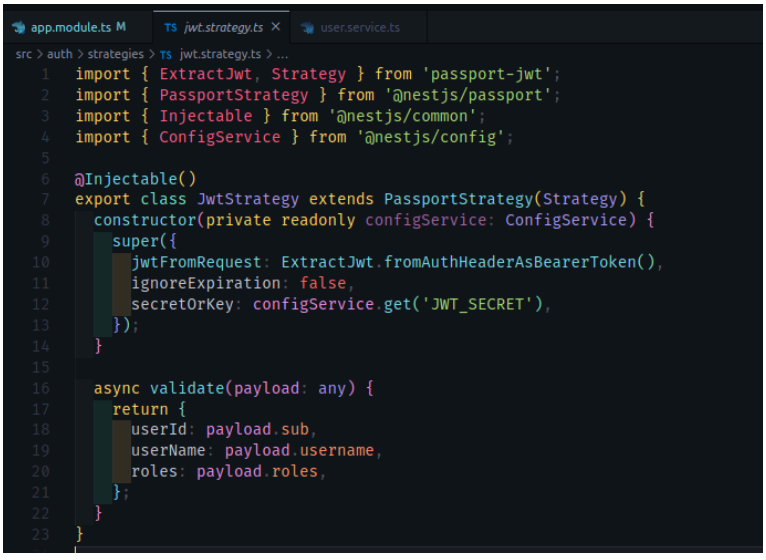
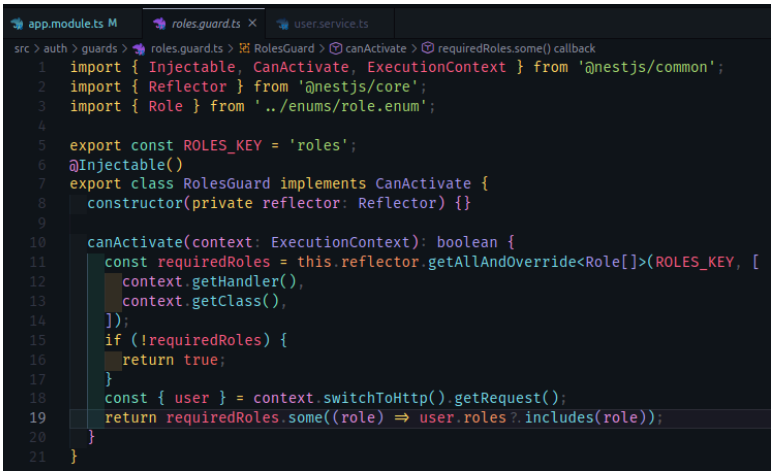


Figura 40.

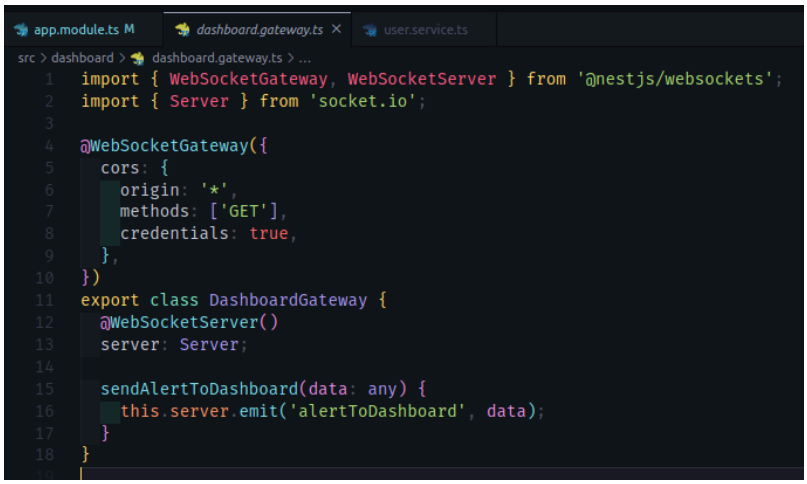
Fragmento de código de roles guard.



Finalmente, otra de las funciones claves del back-end fue el uso de websockets para manejar el envío de notificaciones en tiempo real al cliente (dashboard admin). Para ello se realizó la implementación de un websocket con Socket io.

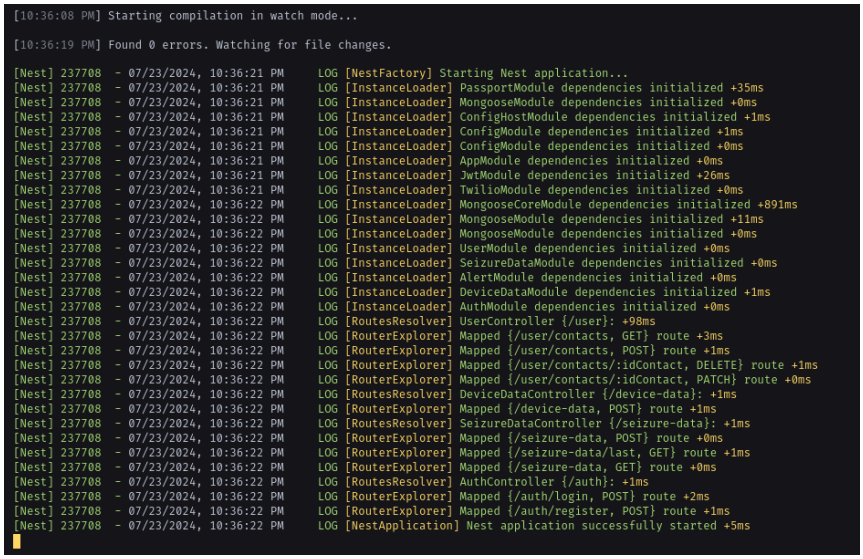
Figuro 41.

Fragmentos de código, implementación de Websockets



Figuro 42.

Servicio back-end en ejecución



5.4.1.1.2 Implementación de servicios. Como se había mencionado, uno de los objetivos principales del proyecto era enviar notificaciones a través de SMS a los contactos registrados por el usuario. Dentro del análisis seleccionamos la plataforma de servicio de mensajería Twilo. Dentro de twilo, con los créditos brindados dentro del plan gratuito podemos adquirir un número de teléfono, el cual se encargará de enviar mensajes, a su vez, tenemos un ID de cuenta y un token con el cual podremos hacer uso del API de mensajería de twilo en nuestro back-end.

Figura 43.

Fragmento panel de administración Twilo

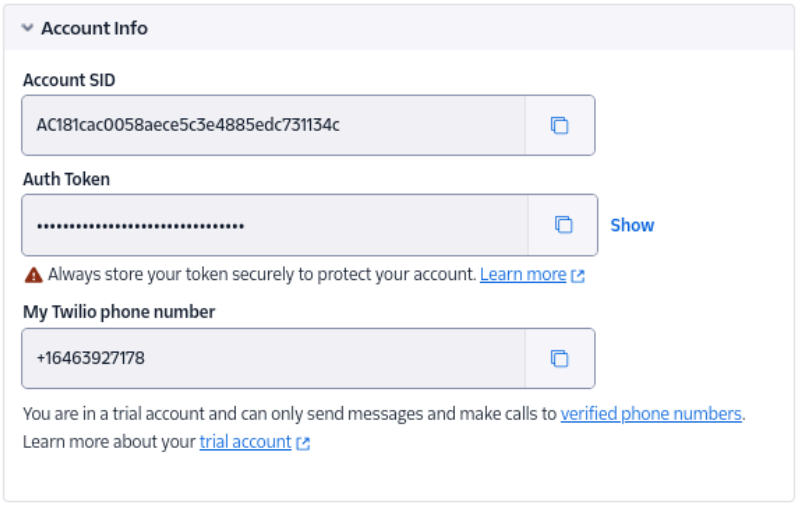
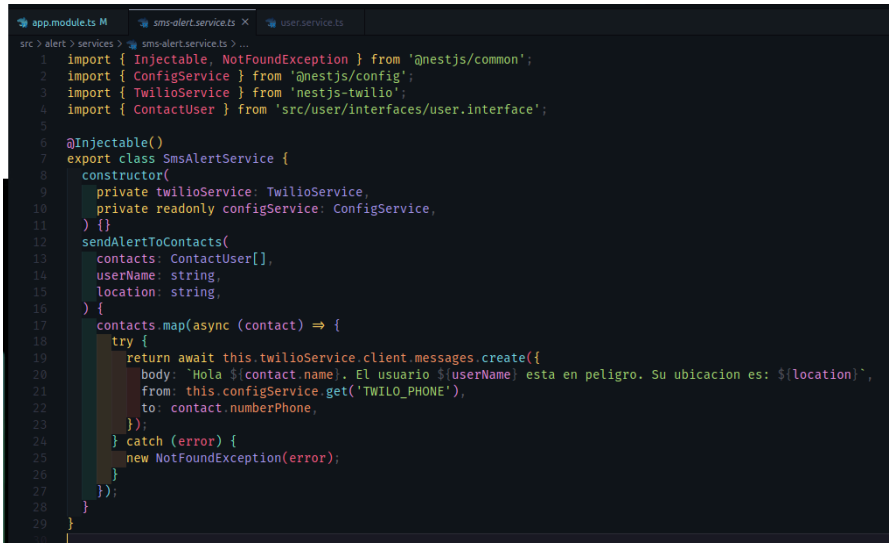


Figura 44.

Implementación de servicio de mensajería de Twilo en back-end



```
import { Injectable, NotFoundException } from '@nestjs/common';
import { ConfigService } from '@nestjs/config';
import { TwilioService } from 'nestjs-twilio';
import { ContactUser } from 'src/user/interfaces/user.interface';

@Injectable()
export class SmsAlertService {
  constructor(
    private twilioService: TwilioService,
    private readonly configService: ConfigService,
  ) {}
  sendAlertToContacts(
    contacts: ContactUser[],
    userName: string,
    location: string,
  ) {
    contacts.map(async (contact) => {
      try {
        return await this.twilioService.client.messages.create({
          body: `Hola ${contact.name}. El usuario ${userName} esta en peligro. Su ubicacion es: ${location}`,
          from: this.configService.get('TWILO_PHONE'),
          to: contact.numberPhone,
        });
      } catch (error) {
        new NotFoundException(error);
      }
    });
  }
}
```

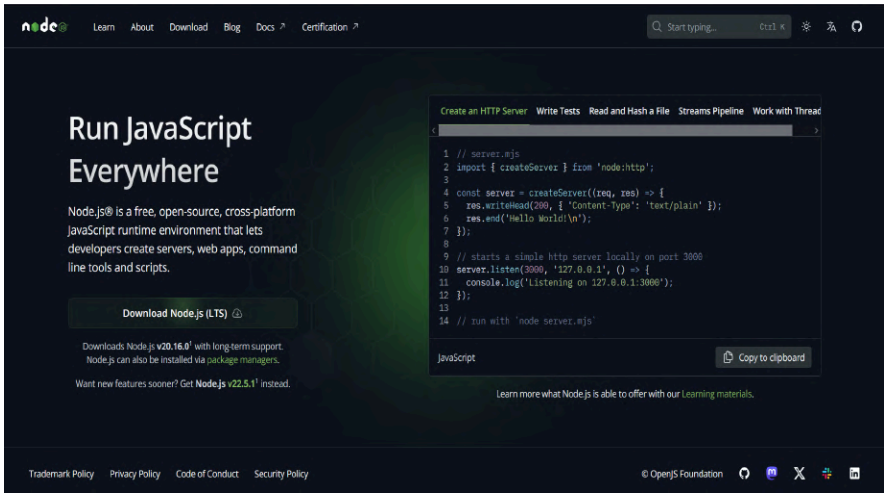
5.4.1.2 Desarrollo Front End.

5.4.1.2.1 Desarrollo de la aplicación móvil. A continuación detallamos el proceso desde la configuración inicial.

Para iniciar el proceso de desarrollo debemos configurar nuestro entorno de desarrollo, para poder ejecutar tanto React Native cómo Expo debemos instalar Node.JS este se puede descargar e instalar desde su web oficial nodejs.org

Figura 45.

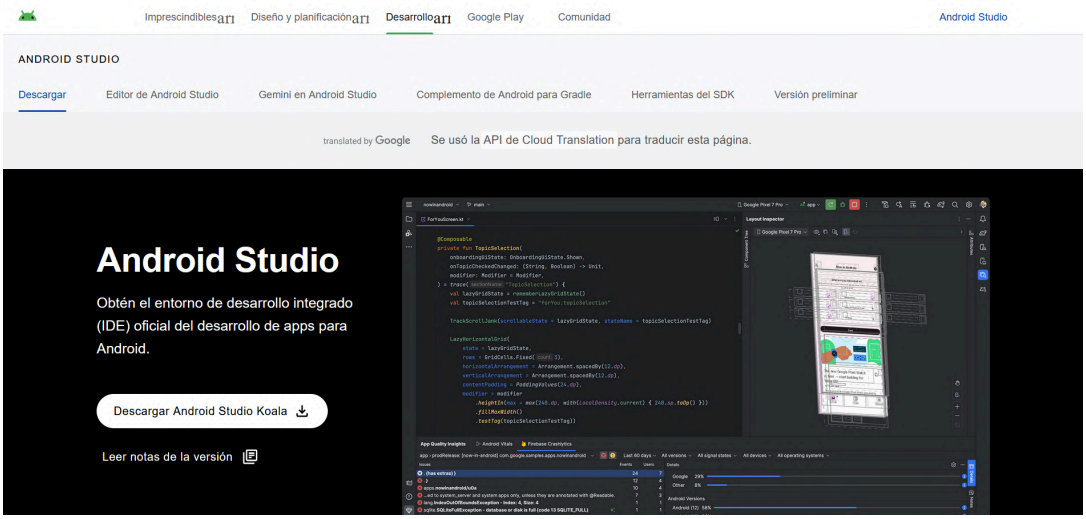
Página oficial para descargar Node.js



Y una herramienta que nos será de ayuda para la creación y gestión de proyectos Expo será Expo CLI, esta herramienta la podemos instalar desde la terminal con “npm”. Luego de haber instalado estas herramientas continuamos con la configuración del entorno, esta vez instalamos Android Studio, pues enfocamos la aplicación móvil a los sistemas operativos Android. La respectiva descarga e instalación se realizaron desde su página web oficial developer.android.com/studio.

Figura 46.

Página oficial para descargar Android Studio

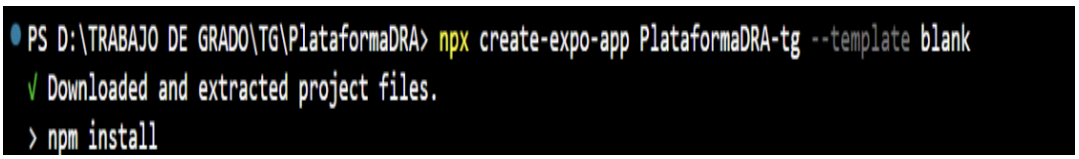


Luego de haber instalado Android Studio procedimos a configurar nuestro Virtual Device Manager (VDM) que es el emulador en el que trabajaremos para ejecutar las funcionalidades de la aplicación que no requieren tanta demanda de recursos.

Con nuestro entorno ya configurado y todas nuestras herramientas descargadas e instaladas, se realizó el procedimiento de iniciar y crear el respectivo proyecto.

Figura 47.

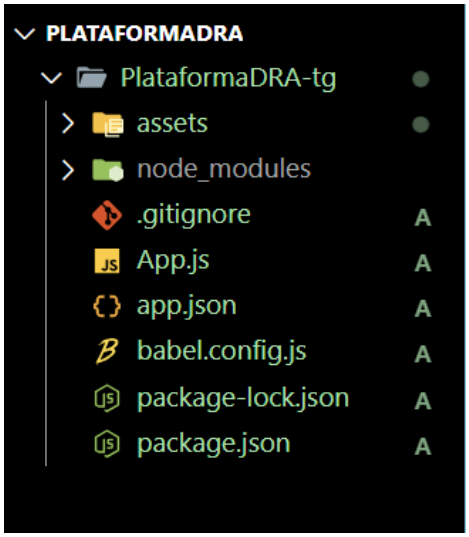
Creación de un proyecto en React Native con Expo.



El comando ejecutado permite seleccionar la plantilla para el proyecto, se seleccionó la plantilla “blank” para generar un proyecto limpio.

Figura 48.

Carpeta raíz del proyecto



En esta carpeta contamos con los archivos iniciales del proyecto, el proyecto inicial se compone del archivo App.js que es el archivo principal donde se ubican los componentes desarrollados.

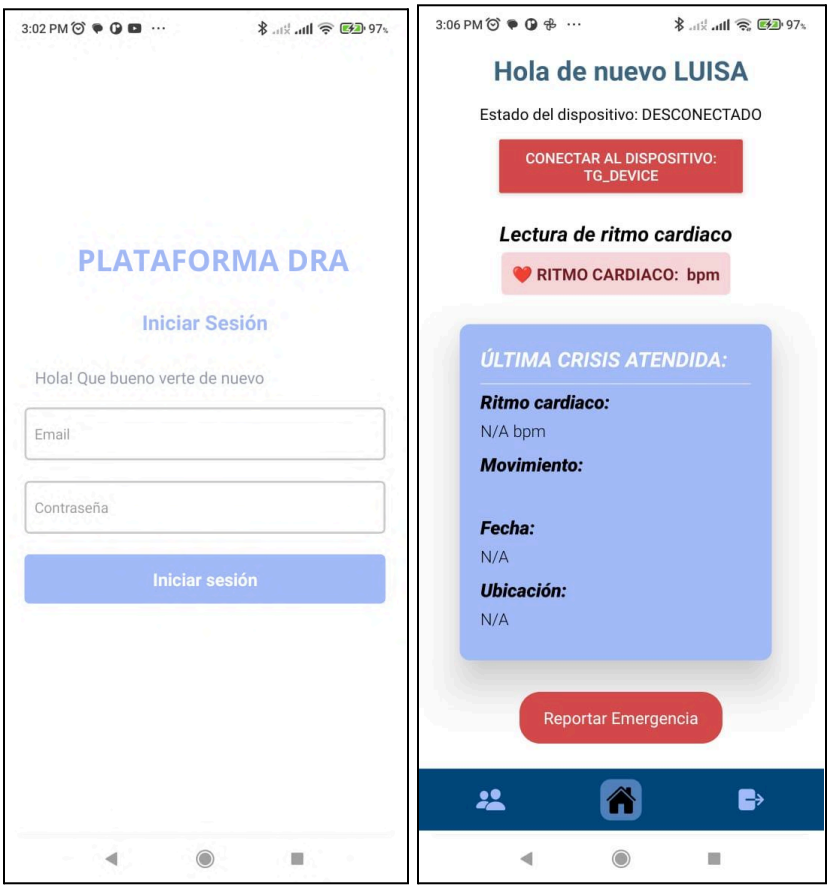
Figura 49.

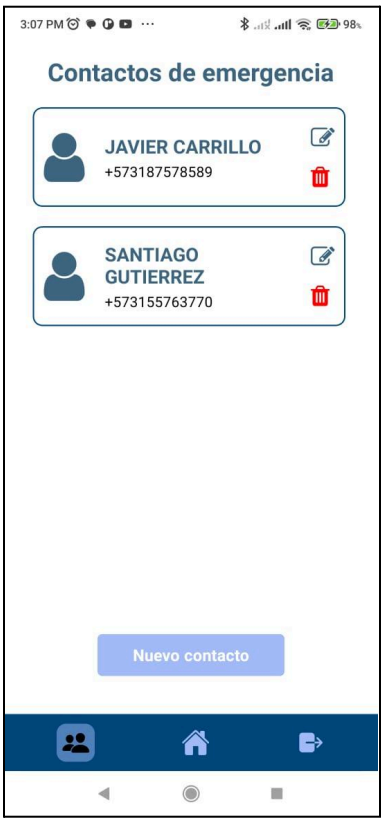
Pantalla de ejecución del proyecto.

Para proceder con la emulación del ambiente virtual elegimos la opción de abrir con Android “a” y esto ejecutará nuestro proyecto en el simulador y podemos empezar a desarrollar nuestras respectivas pantallas con la ayuda de los mockups que se diseñaron previamente como guías.

Figura 50.

Resultados finales de las pantallas de la aplicación móvil





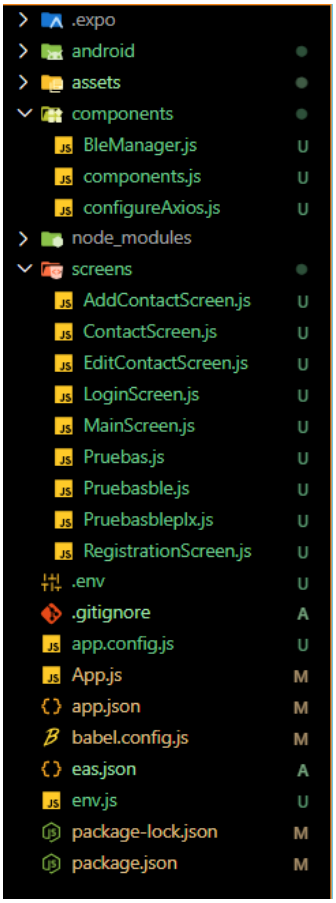
Nota. Resultado final de las pantallas de inicio de sesión, pantalla principal y pantalla de contactos.

Estas fueron algunas de las pantallas que se desarrollaron para el funcionamiento de la aplicación móvil, las cuales cumplieron con el diseño previamente establecido y con las funcionalidades requeridas para cada una.

Como resultado final, la distribución de archivos quedo de la siguiente forma en el desarrollo móvil.

Figura 51.

Carpeta raíz del proyecto una vez finalizado



Acá podemos evidenciar la cantidad de pantallas y componentes que fueron el resultado del desarrollo.

5.4.1.2.2 Desarrollo del dashboard web. El desarrollo del dashboard web, como ya se había previsto, fue realizado en el framework React. Al haber configurado nuestro entorno de desarrollo previamente, no debimos realizar nuevas instalaciones para el desarrollo web.

Por lo que se procedió directamente con la creación del proyecto.

Figura 52.

Comando para crear un nuevo proyecto en React Web

```
npx create-react-app my-app
```

Al ejecutar este comando nuestra carpeta raíz quedo conformada por los siguientes archivos:

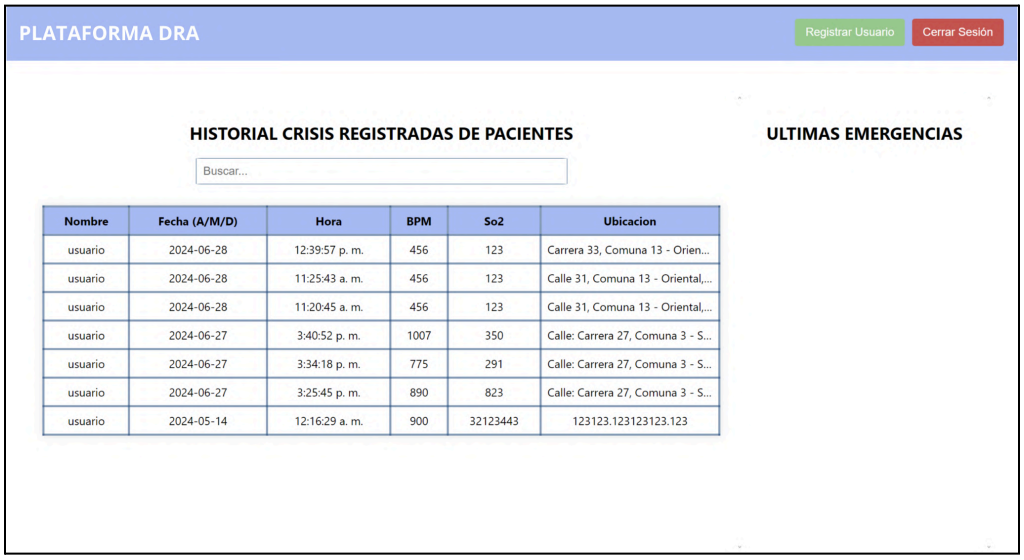
Figura 53.

Ejemplo de la carpeta raíz al crear un nuevo proyecto de React Web

Una vez se creó el proyecto, se procedió a seguir la guía de diseño y desarrollar las respectivas pantallas y funcionalidades del dashboard web.

Figura 54.

Resultados finales de pantalla del dashboard web.



Nota. Se evidencia los resultados tanto de la pantalla de inicio de del administrador como su respectiva pantalla principal del dashboard web.

Como se evidenció, las pantallas cumplieron con los diseños propuestos y funcionalidades requeridas. Una de las funcionalidades importantes a la hora del desarrollo del dashboard fue la del momento de implementar la escucha de un websocket, esto se realizó con el fin de poder obtener las alertas en tiempo real al momento de ser generadas por el dispositivo portátil o la aplicación móvil.

Figura 55.

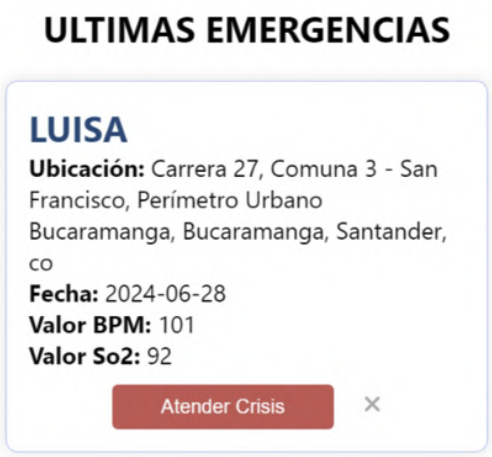
Desarrollo del websocket para las alertas en tiempo real

```
useEffect(() => {
  if (!socket) return;

  // Manejar la recepción de datos de emergencia del servidor
  const handleEmergencyData = (data) => {
    console.log('Nueva emergencia recibida:', data);
    const emergencyId = new Date().toISOString();
    const newEmergency = { ...data, id: emergencyId };

    // Guardar la nueva emergencia en localStorage
    const storedEmergencies = JSON.parse(localStorage.getItem('emergencies')) || [];
    localStorage.setItem('emergencies', JSON.stringify([newEmergency, ...storedEmergencies]));

    // Actualizar el estado local
    setEmergencies(prevEmergencies => [newEmergency, ...prevEmergencies]);
    setShowNewEmergency(true);
    setHighlightedEmergency(newEmergency);
    setTimeout(() => {
      setShowNewEmergency(false);
      setHighlightedEmergency(null);
    }, 2000);
  };

  // Escuchar eventos de datos de emergencia del servidor
  socket.on('alertToDashboard', handleEmergencyData);

  return () => {
    socket.off('alertToDashboard', handleEmergencyData);
    console.log('Conexión WebSocket cerrada');
  };
}, [socket]);
```

Como resultado de la escucha de estas alertas, el dashboard generaba una notificación para que esta sea atendida por los administradores del servicio médico.

Figura 56.

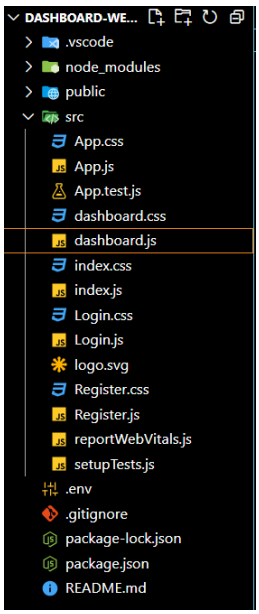
Emergencia generada al escuchar el websocket



En el momento en que se finalizó el desarrollo del dashboard web la carpeta del proyecto quedo conformada de la siguiente manera.

Figura 57.

Carpeta raíz del proyecto una vez finalizado

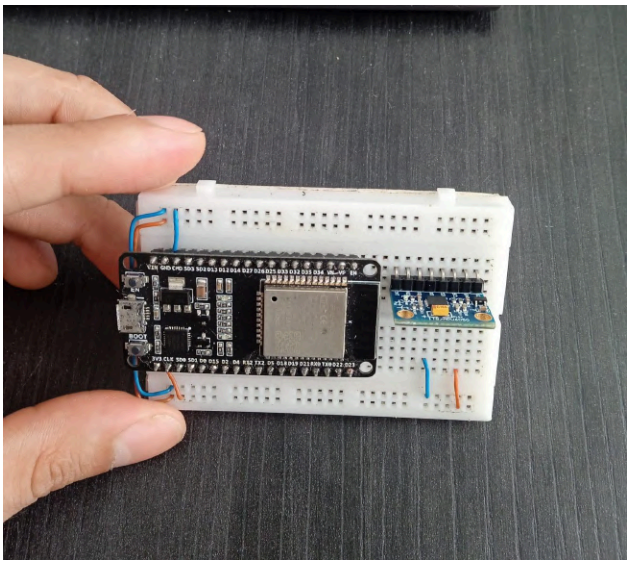


5.4.2 Desarrollo del Hardware

5.4.2.1 Montaje inicial de componentes electrónicos. Inicialmente, se realizó un montaje básico y sencillo haciendo uso de una protoboard para facilitar las conexiones. En relación con la placa de desarrollo se hizo uso de una placa de desarrollo esp 32 de 38 pines que nos brinda las funcionalidades necesarias para hacer pruebas iniciales del funcionamiento de los sensores.

Figura 58.

Montaje inicial sensor mpu6050 con esp32



5.4.2.2 Toma y análisis de datos con los componentes electrónicos. Con el fin de saber cómo abordaremos el problema de la detección de patrones de convulsiones tónico clónicas en pacientes epilépticos, el enfoque fue observar el comportamiento del problema que queremos solucionar y darnos una idea de la variabilidad de los datos que

se presenta en esta situación. Con ayuda y asistencia de nuestro codirector, el doctor Iván Peña, realizamos una visita a los simuladores de grado clínico con los que cuenta la facultad de salud de la universidad; simuladores con los cuales los estudiantes del área de salud practican y refuerzan sus conocimientos en un ambiente simulado pero lo más cercano a la realidad posible. En nuestro caso en específico, dentro del área de simulación se encuentra un simulador de convulsiones tónico-clónicas y mioclónicas con el cual pudimos hacer una toma de la variabilidad de los datos de aceleración presentes durante una crisis convulsiva.

Estos datos fueron tomados ubicando el sensor en la muñeca, uno de los lugares en donde más se presentan representaciones de movimientos durante una crisis convulsiva tónico-clónica.

Para poder tomar mediciones y facilitar el uso con el sensor MPU6050 es necesario instalar una librería de la compañía Adafruit, la cual es una librería que nos brinda una serie de funciones para la toma y el procesamiento de los datos. El sensor MPU6050 es un sensor triaxial, esto quiere decir que podemos tomar datos de aceleración tanto en el eje X, Y y Z. Teniendo en cuenta la variabilidad de movimiento de una convulsión y de la posición del usuario, con el fin de analizar la variación de la aceleración de forma general, se hizo uso del vector de magnitud de la señal, el cual representa la magnitud combinada de las tres componentes de la aceleración. El cálculo de este vector se realizó con la siguiente fórmula.

$$VMS = \sqrt{a_x^2 + a_y^2 + a_z^2}$$

Donde:

a_x^2 es la medida de la aceleración en el eje x

a_y^2 es la medida de la aceleración en el eje y

a_z^2 es la medida de la aceleración en el eje z

Estos datos quedan almacenados en un archivo csv con el tiempo de la toma en milisegundos y el valor del vector de magnitud de la señal.

Figura 59.

Fotografías de la visita para toma de datos.

Nota. Toma de datos, comportamiento de aceleración en simulador de convulsiones tónico clónicas y mioclónicas.

Figura 60.

Fragmento de código de toma de datos de aceleración con sensor

```
void setup() {
  Serial.begin(115200);
  Wire.begin();
  mpu.initialize();

  if (!mpu.testConnection()) {
    Serial.println("MPU6050 connection failed");
    while (1);
  }

  // Configurar el rango del acelerómetro a ±8g
  mpu.setFullScaleAccelRange(MPU6050_ACCEL_FS_8);
  // Etiquetas para el Serial Plotter
  Serial.println("a_x\ta_y\ta_z\tSMV");
}

void loop() {
  int16_t ax, ay, az;
  mpu.getAcceleration(&ax, &ay, &az);

  // Convertir a g basado en el rango configurado a ±8g
  float a_x = ax / 4096.0;
  float a_y = ay / 4096.0;
  float a_z = az / 4096.0;

  // Calcular la Magnitud del Vector de Señal (SMV)
  float smv = sqrt(a_x * a_x + a_y * a_y + a_z * a_z);

  // Imprimir los resultados en una línea separada por tabuladores para el Serial Plotter
  Serial.print(a_x); Serial.print("\t");
  Serial.print(a_y); Serial.print("\t");
  Serial.print(a_z); Serial.print("\t");
  Serial.println(smv);
```

```
 1 Tiempo,smv
 2 379777,1.46
 3 379828,0.54
 4 379879,0.86
 5 379930,1.53
 6 379981,0.64
 7 380032,1.04
 8 380083,1.02
 9 380134,0.56
10 380185,0.80
11 380236,1.06
12 380287,0.67
13 380338,1.13
14 380389,1.22
15 380440,0.54
16 380491,1.23
17 380542,0.75
```

Una vez generado el csv, con los datos de mediciones, es posible realizar un análisis exploratorio del comportamiento registrado. Este análisis se hizo haciendo uso del lenguaje Python el cual nos ofrece diferentes librerías para explorar los datos. Haciendo uso de pandas, se pudo almacenar esta información en un dataframe, haciendo más sencillo el manejo de los datos. Adicionalmente, haciendo uso de la librería Matplotlib fue posible graficar los datos.

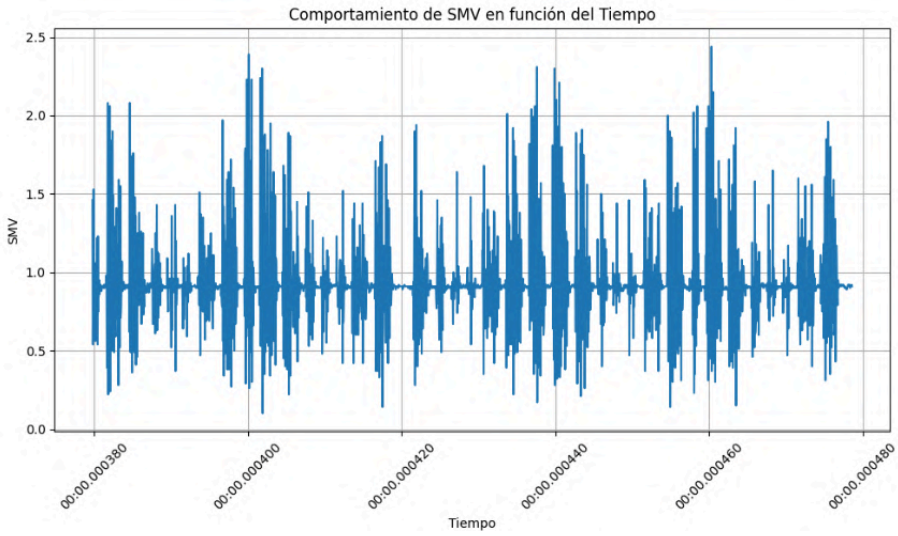
Figura 61.

Dataframe datos de aceleración

```
[ ]  1 import pandas as pd
     2 import matplotlib.pyplot as plt
     3 import numpy as np

[ ]  1 df = pd.read_csv('aceleracion_datos_tonico_clonico_8g.csv')

     1 df_copy = df.copy()
     2 df_copy['Tiempo'] = pd.to_datetime(df_copy['Tiempo'])
     3 plt.figure(figsize=(10, 6))
     4 plt.plot(df_copy['Tiempo'], df_copy['smv'])
     5 plt.xlabel('Tiempo')
     6 plt.ylabel('SMV')
     7 plt.title('Comportamiento de SMV en función del Tiempo')
     8 plt.grid(True)
     9 plt.xticks(rotation=45)
    10 plt.tight_layout()
    11
    12 plt.show()
```

Figura 62.

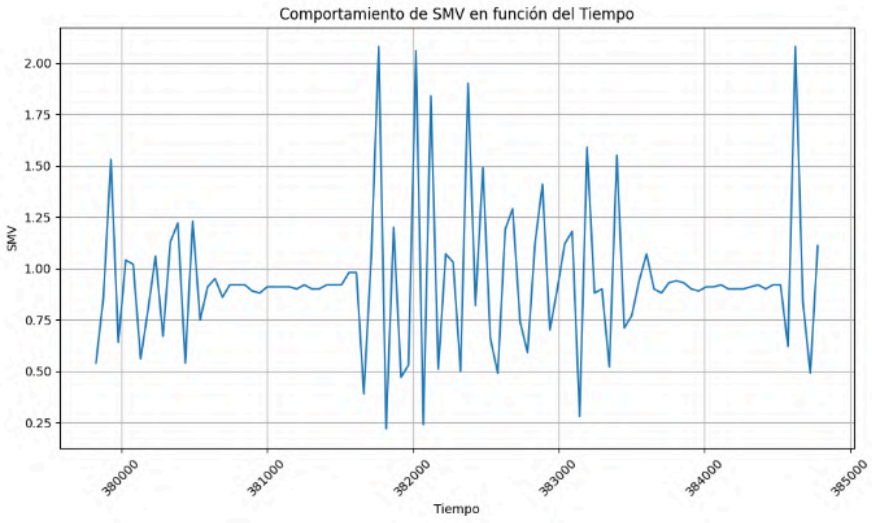
Gráfica de vector vms vs Tiempo



Con el fin de observar de forma más detallada el comportamiento de los datos en un fragmento de tiempo más corto, se tomaron intervalos de 5 segundos de la toma total.

Figura 63.

Fragmento 5 segundos de datos



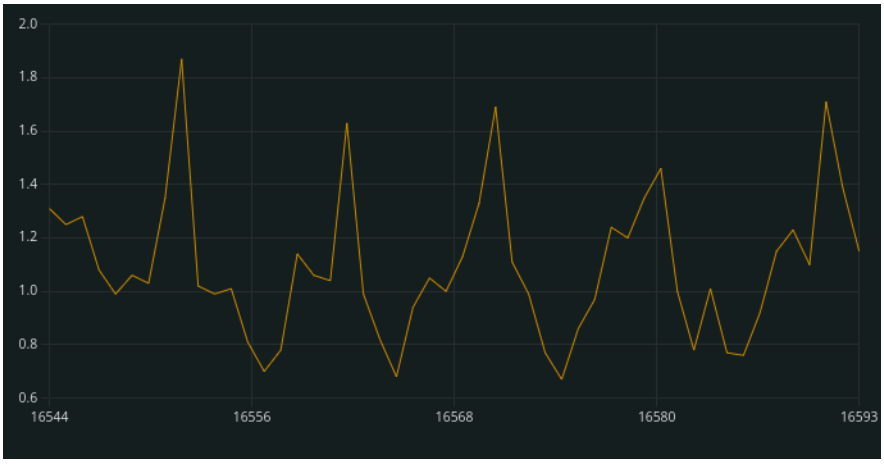
Adicionalmente, también se tomaron datos exploratorios de diferentes actividades cotidianas, la cuales también incluyen movimientos considerables en las extremidades superiores. Esto se hizo con el fin de estudiar si el comportamiento es similar al generado por un ataque convulsivo de carácter tónico clónico, esto nos dio una perspectiva de diferentes actividades que podrían confundirse durante la detección con una crisis y generar falsos positivos. Para esto tuvimos ayuda de un voluntario que realizó actividades como caminar, correr y bajar escaleras.

Figura 64.

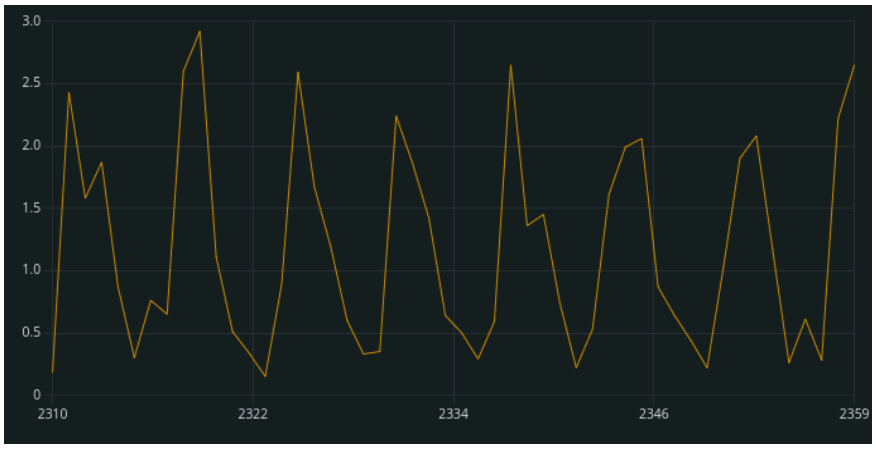
Evidencia fotográfica de toma de datos de actividades cotidianas.

Figura 65.

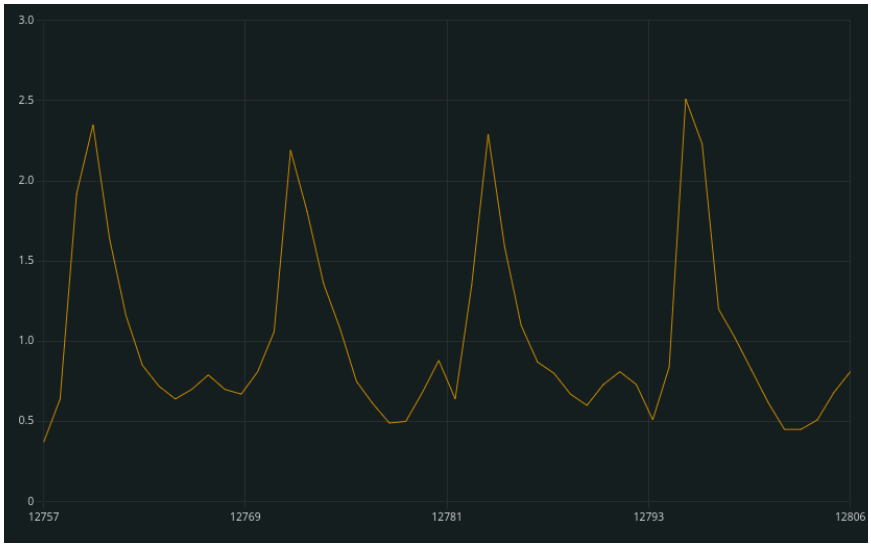
Gráficas de los datos de las actividades cotidianas.



Nota. Actividad: caminando.



Nota. Actividad: corriendo



Nota. Actividad: bajando escaleras.

Con base en la exploración de los datos tomados con el simulador de las convulsiones y de las actividades cotidianas, fue posible identificar ciertos comportamientos.

El comportamiento de la aceleración en las crisis convulsivas tónico-clónicas albergan valores de aceleración con picos de amplitud máxima de un poco más 2g y valores mínimos de amplitud de alrededor de 1,2g. Además, en comparación con las actividades cotidianas, es evidente que la distancia entre los picos de aceleración de las crisis es mucho menor, esto quiere decir que son más frecuentes en un lapso corto de tiempo, esto logra diferenciar el comportamiento de las crisis en comparación a las medidas de aceleración de actividades como correr cuya amplitud de aceleración es similar, pero que son más espaciadas en función del tiempo.

Con esta información, se procedió a analizar la cantidad de picos producidos durante ventanas de tiempo de 5 segundos para la totalidad de muestras en 100 segundos, de este análisis se concluyó que existen alrededor de 9,5 picos en promedio por ventana, que están dentro del umbral definido anteriormente.

5.4.2.3 Implementación de algoritmos de detección. Para la implementación del algoritmo de detección se siguieron los siguientes aspectos claves:

- Leer valores de aceleración y calcular el VMS
- Si el VMS está dentro del rango definido dentro del umbral de 1.2g y 2.0g, se considera un posible pico.
    - Si no se ha detectado una convulsión previamente, se inicia la detección, se registra el tiempo de inicio y se cuenta el primer pico.

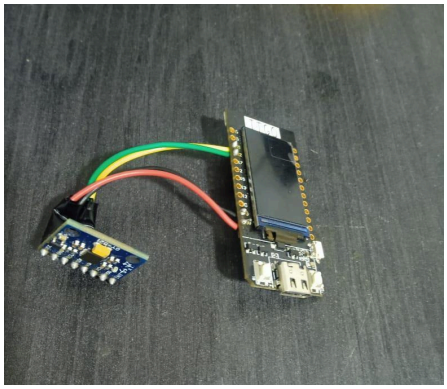
- Si ya se está detectando una convulsión, se calcula la pendiente de cambio del VMS y se verifica si supera un umbral de pendiente para confirmar que es un pico pronunciado y aumentar el conteo de picos

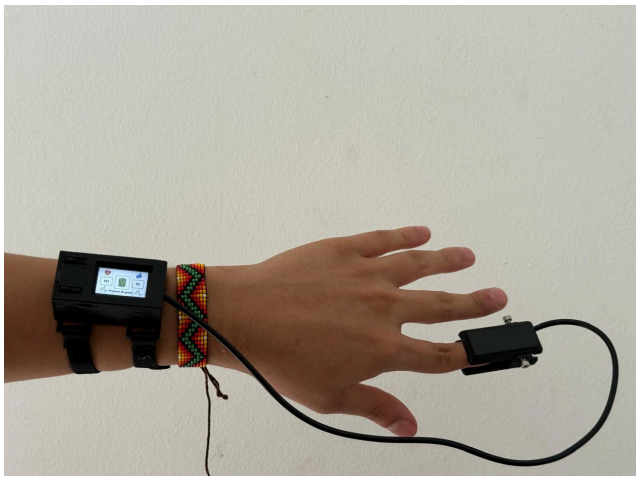
- Si el VMS sale del rango umbral y el pico es demasiado largo o el intervalo entre picos es demasiado grande, se descarta la detección y se reinicia.
- Durante la detección, se monitorea el tiempo transcurrido. Si se supera el período de detección y se han registrado suficientes picos (al menos 9), se considera que se ha detectado una convulsión

5.4.2.4 Montaje final del dispositivo. Finalmente, cada componente fue soldado a la placa de desarrollo y ensamblado dentro del diseño impreso en 3D, dando así como resultado el prototipo final del dispositivo.

Figura 66.

Montaje final del dispositivo.





Nota. Imágenes correspondientes al momento de soldar los componentes y como se ve ensamblado el prototipo en su totalidad

5.4.3 Integración del hardware y software

Para la integración del hardware y software se realizó la configuración del servidor bluetooth en la placa de desarrollo. En esta se definió un identificador, en este caso UUID tanto para el servicio como para cada una de las características(booleano para identificar convulsión, ritmo cardiaco y saturación de oxígeno en sangre). La conexión entre el hardware y software se hará a través de comunicación por vía Bluetooth, más específicamente haciendo uso de BLE. El uso de BLE supone ciertos beneficios sobre el uso del bluetooth convencional, entre ellos y a la razón que debe su nombre es menor coste en términos de uso de energía, teniendo en cuenta que el factor energético es importante en el dispositivo, ya que hará uso de una batería para su autonomía.

Figura 67.

Definición de UUID para el servicio bluetooth y características del servicio

```
// UUIDs para el servicio y las características
#define SERVICE_UUID "91bad492-b950-4226-aa2b-4ede9fa42f59"
#define CHARACTERISTIC_UUID_1 "7e5e9973-50f1-4a4a-a5e5-1e0e4e90f16e"
#define CHARACTERISTIC_UUID_2 "7e5e9973-50f1-4a4a-a5e5-1e0e4e90f16f"
#define CHARACTERISTIC_UUID_3 "7e5e9973-50f1-4a4a-a5e5-1e0e4e90f170"

BLECharacteristic *pCharacteristic1;
BLECharacteristic *pCharacteristic2;
BLECharacteristic *pCharacteristic3;
bool deviceConnected = false;
```

Esta conexión BLE se llevó a cabo gracias a una librería de React Native llamada "react-native-ble-plx" la cual ya tiene ciertas funciones, las cuales permitían establecer la conexión y solicitar permisos necesarios para el correcto funcionamiento de esta.

Para esta implementación se creó un componente que albergó dichas funciones, desde iniciar el escaneo para encontrar el dispositivo hasta obtener los datos y monitorear las características previamente definidas.

Figura 68.

Funciones de solicitud de permisos y escaneo

```
class BleService {
    constructor() {
        this.manager = new BleManager();
    }

    async requestPermissions() {
        if (Platform.OS === 'android') {
            await PermissionsAndroid.requestMultiple([···
            ]);
        } else {
            await request(PERMISSIONS.IOS.BLUETOOTH_PERIPHERAL);
        }
    }

    startScan(setDevices) {
        this.manager.startDeviceScan(null, null, (error, device) => {
            if (error) {
                Alert.alert('Error', error.message);
                return;
            }
            setDevices(prevDevices => {
                const newDevices = new Map(prevDevices);
                newDevices.set(device.id, device);
                return newDevices;
            });
        });
    }
```

Figura 69.

Definición de los UUID del servicio y sus caracteristicas en el front-end

```
import BleService from '../components/BleManager';
const SERVICE_UUID = "91bad492-b950-4226-aa2b-4ede9fa42f59";
//?True
const CHARACTERISTIC_UUID_1 = "7e5e9973-50f1-4a4a-a5e5-1e0e4e90f16e";
//?Oxigeno
const CHARACTERISTIC_UUID_2 = "7e5e9973-50f1-4a4a-a5e5-1e0e4e90f16f";
//?Ritmo cardiaco
const CHARACTERISTIC_UUID_3 = "7e5e9973-50f1-4a4a-a5e5-1e0e4e90f170";
```

Figura 70.

Funcionalidades BLE

```
const connectToDevice = async () => {
    try {
        await BleService.requestPermissions();
        setDeviceStatus('CONECTANDO...');
        const device = await BleService.connectToTGDevice(setConnectedDevice);
        if (device) {
            setDeviceStatus('CONECTADO');
            startMonitoring(device);
        } else {
            setDeviceStatus('DESCONECTADO');
            Alert.alert('Error', 'No se pudo conectar al dispositivo TG_DEVICE');
        }
    } catch (error) {
        console.error('Error en la conexión:', error);
        setDeviceStatus('DESCONECTADO');
        Alert.alert('Error', 'No se pudo conectar al dispositivo TG_DEVICE');
    }
};
```

```
const startMonitoring = async (device) => {
  await BleService.monitorConvulsion(
    device,
    SERVICE_UUID,
    CHARACTERISTIC_UUID_1,
    async () => {
      try {
        const oxigeno = await BleService.readData(device, SERVICE_UUID, CHARACTERISTIC_UUID_2);
        const ritmoCardiaco = await BleService.readData(device, SERVICE_UUID, CHARACTERISTIC_UUID_3);
        const location = await handleGetLocation();

        const data = { oxigeno, ritmoCardiaco, location };
        console.log(data);
        await sendDataToBackend(data);
        Alert.alert('Convulsión detectada', 'Datos enviados al backend');
      } catch (error) {
        console.error('Error al obtener datos adicionales:', error);
      }
    }
  );
```

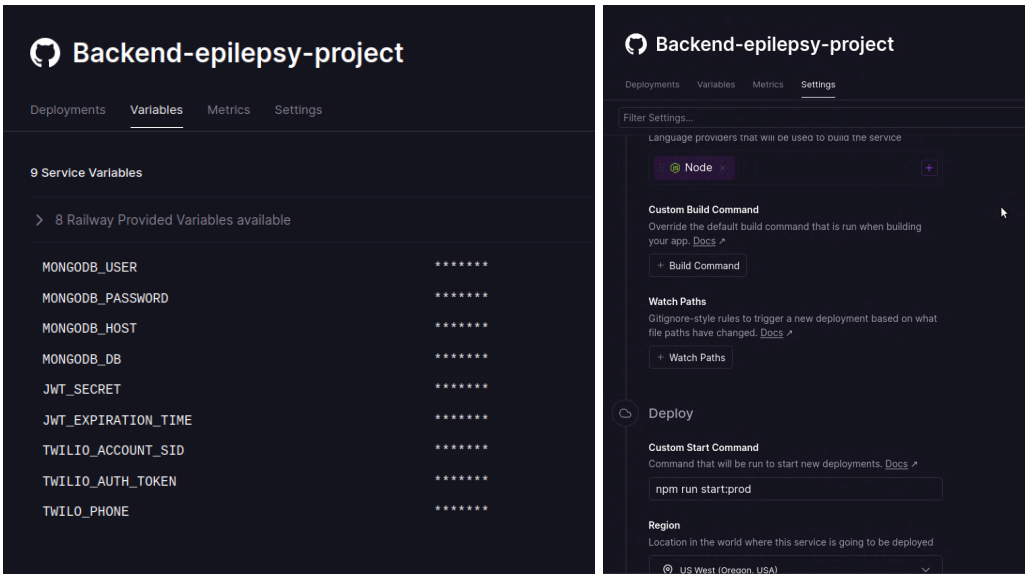
Nota. Funcionalidad que permite conectarse a un dispositivo específico y funcionalidad para dar inicio al monitoreo una vez se tiene establecida una conexión.

5.4.4 Despliegue del proyecto

Como se había mencionado con anterioridad, la plataforma escogida para el despliegue del proyecto fue Railway. Esta nos da la posibilidad de conectar nuestra cuenta de GitHub y seleccionar el repositorio donde está albergado el código del proyecto. Railway toma ese proyecto y procede a iniciar el proceso de despliegue en donde se tiene que indicar las variables de entorno necesarias para el proyecto y comandos de inicio y compilación del proyecto dependiendo del entorno de desarrollo.

Figura 71.

Variables de entorno y comandos de deploy para proyecto en railway



Una vez configurado y en ejecución el proyecto, ya es posible acceder al proyecto mediante un dominio público que nos brinda railway.

Figura 72.

Documentación del api del proyecto a través del dominio de railway

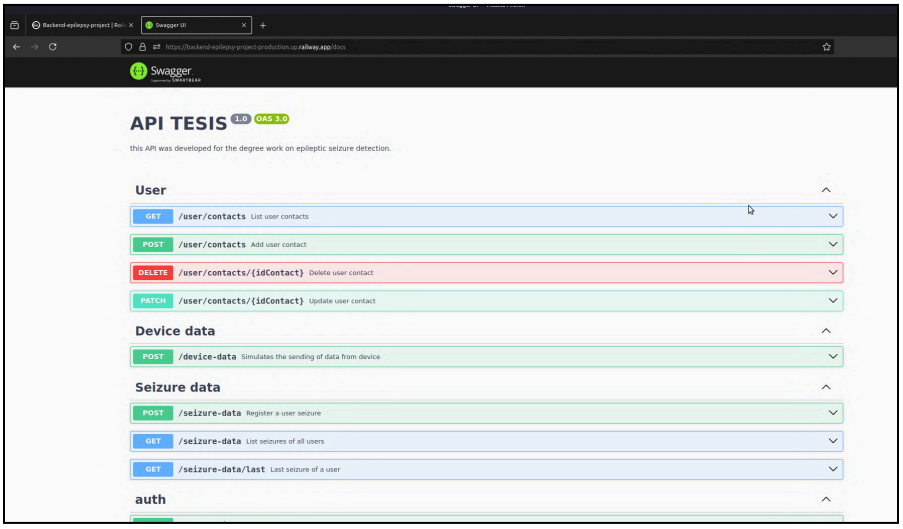
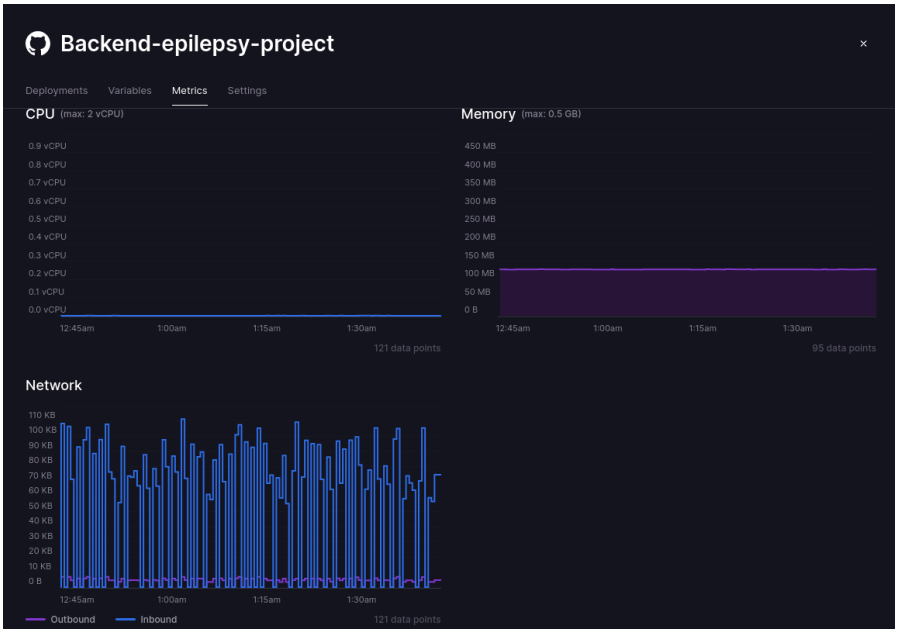


Figura 73.

Panel de monitoreo del proyecto en railway



5.5 Evaluación y pruebas del prototipo

Una vez integrados los componentes que conforman la plataforma, se planteó un plan de pruebas con la finalidad de analizar y estudiar el comportamiento en conjunto de la plataforma.

5.5.1 Realización de Pruebas

5.5.1.1 Validación de efectividad y precisión del prototipo. En esta fase se realizó una segunda visita a los simuladores de la facultad de salud de la universidad. En esta segunda visita se tuvo como objetivo realizar pruebas de efectividad de los algoritmos de detección construidos con anterioridad. El plan de pruebas consistió en realizar 15 simulaciones de crisis convulsivas del tipo tónico-clónico y 15 simulaciones de crisis convulsivas del tipo mioclónico con el fin de observar la totalidad y el tiempo promedio de detección.

Figura 74.

Evidencia fotográfica de las pruebas de simulación de crisis convulsivas

De las pruebas realizadas se obtuvieron los siguientes resultados:

Tabla 6:

Pruebas realizadas en una simulación de crisis mioclónicas

Crisis Mioclónicas		
TOMA	***TIEMPO***	***ESTADO***
1	00:06,38	DETECTADA
2	00:06,35	DETECTADA
3	00:06,81	DETECTADA
4	00:06,45	DETECTADA
5	00:06,21	DETECTADA
6	00:06,80	NO DETECTADA
7	00:05,76	NO DETECTADA
8	00:07,06	DETECTADA
9	00:06,05	DETECTADA
10	00:07,25	DETECTADA
11	00:06,28	DETECTADA
12	00:06,08	DETECTADA
13	00:06,65	DETECTADA
14	00:06,55	DETECTADA
15	00:06,26	DETECTADA

(Toma de 15 datos de crisis Mioclónicas)

Tabla 7:

Pruebas realizadas en una simulación de crisis Tónico-clónicas.

Crisis Tónico-clónicas		
TOMA	***TIEMPO***	***ESTADO***
1	00:06,48	DETECTADA
2	00:07,03	DETECTADA
3	00:12,85	NO DETECTADA
4	00:06,63	DETECTADA
5	00:06,90	DETECTADA

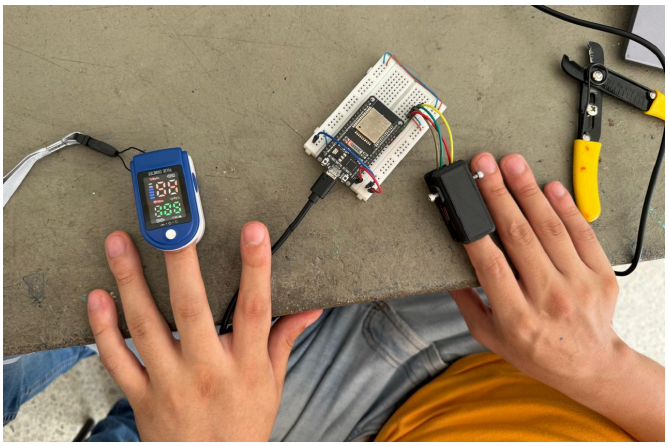
6	00:05,65	DETECTADA
7	00:06,46	DETECTADA
8	00:06,40	DETECTADA
9	00:05,68	DETECTADA
10	00:09,27	DETECTADA
11	00:05,48	DETECTADA
12	00:12,78	NO DETECTADA
13	00:08,96	DETECTADA
14	00:06,08	DETECTADA
15	00:06,08	DETECTADA

Como se evidencia del conjunto de 30 pruebas realizadas para la detección de crisis convulsivas (Tonico-clónicas y Mioclónicas) nuestro dispositivo fue capaz de detectar 26 crisis convulsivas con un promedio de 00:07,51 s para las crisis tónico-clónicas y de 00:06,46 s para las mioclónicas. Estos resultados indican que el dispositivo fue capaz de detectar el 86,7% de las crisis simuladas.

Otro punto a analizar fue la precisión en la toma del ritmo cardíaco con el sensor. Para esta situación se tomaron una serie de mediciones con el sensor empleado para el proyecto y se compararon con las mediciones obtenidas por un oxímetro de uso comercial empleado para este mismo fin.

Figura 75.

Comparación entre Oximetro y sensor de ritmo cardiaco



Para esta prueba, se tomaron alrededor de 30 muestras.

Tabla 8:

Comparativa toma de datos sensor ritmo cardiaco vs. oxímetro

Toma de datos BPM Sensor vs. Oxímetro		
TOMA	**SENSOR**	**OXÍMETRO**
1	101	100
2	96	94
3	99	99
4	97	99
5	100	100
6	97	99
7	95	92
8	98	99
9	94	96
10	96	92
11	91	93
12	98	98

13	97	95
14	100	103
15	96	95
16	98	99
17	96	99
18	99	100
19	97	95
20	97	98
21	95	96
22	99	101
23	99	100
24	98	98
25	100	98
26	98	95
27	92	93
28	99	97
29	98	100
30	101	99

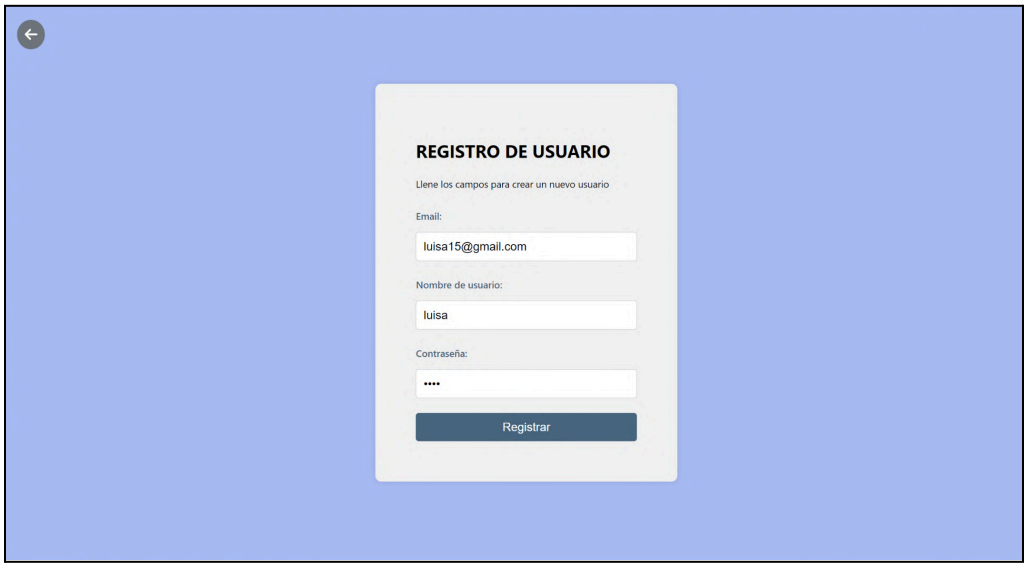
de estas 30 muestras se obtuvo un porcentaje de error de aproximadamente 1,68%

5.5.1.2 Pruebas de Integración en ambiente simulado. Para este escenario lo que se buscó fue simular el flujo completo que tendría la plataforma en el escenario en el cual se detecta una emergencia. Antes de comenzar la prueba, el usuario tiene que haber sido

creado por el administrador. Para este escenario las pruebas se realizaron con un usuario de nombre Luisa.

Figura 76.

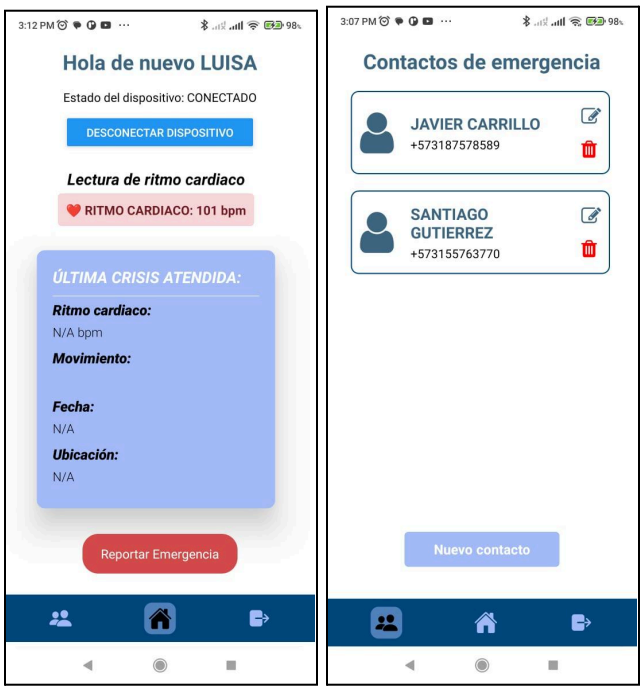
Formulario de registro de usuario en el panel del administrador



Una vez creado el usuario, este tiene que haber iniciado sesión en la aplicación, tener conectado el dispositivo de detección y tener registrado por lo menos mínimo un contacto de emergencia.

Figura 77.

Inicio de sesión y pantalla de contactos para el flujo



La prueba comienza en el instante en el cual el usuario está teniendo una crisis convulsiva asociada a un ataque epiléptico.

Figura 78.

Simulación de un ataque epiléptico en los simuladores

Una vez detectada la crisis, el dispositivo envía la alerta a la app móvil y esta procede a hacer la petición al back-end del envío de las alertas tanto a los contactos registrados como al dashboard web del médico con el nombre del usuario y la ubicación del lugar en donde tuvo la emergencia.

Figura 79.

Recepción de las alertas de parte de los contactos.

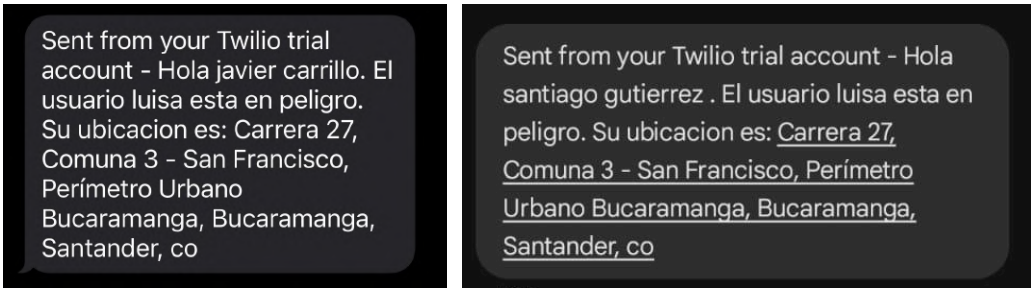
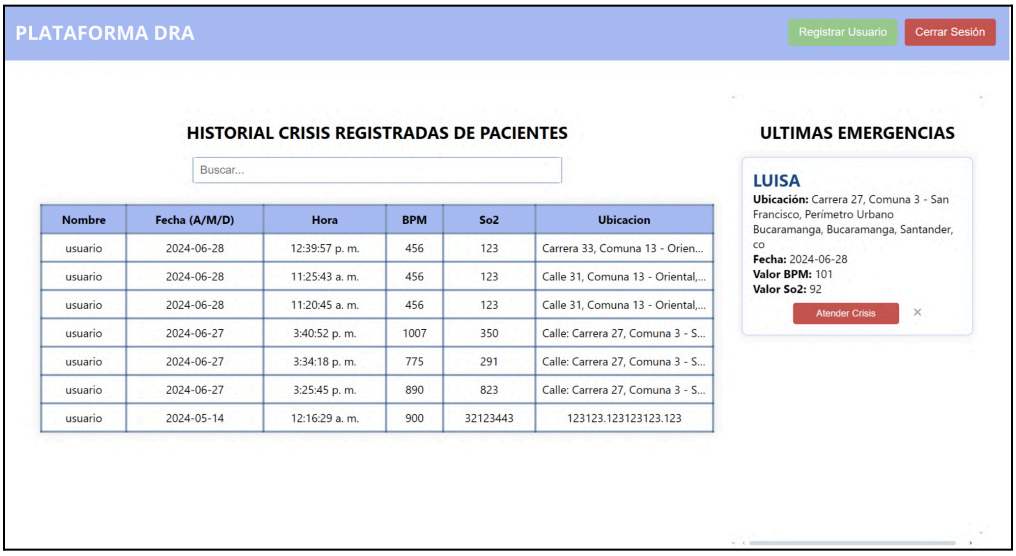


Figura 80.

Recepción de alerta en el dashboard



Una vez recepcionada la emergencia en el dashboard web, el administrador, haciendo clic en el botón “atender crisis”, generará un registro nuevo en la tabla de

histórico de crisis atendidas. A su vez, este registro ya puede ser visible en la app del usuario.

Figura 81.

Alerta atendida y guardada en el historial de crisis registradas

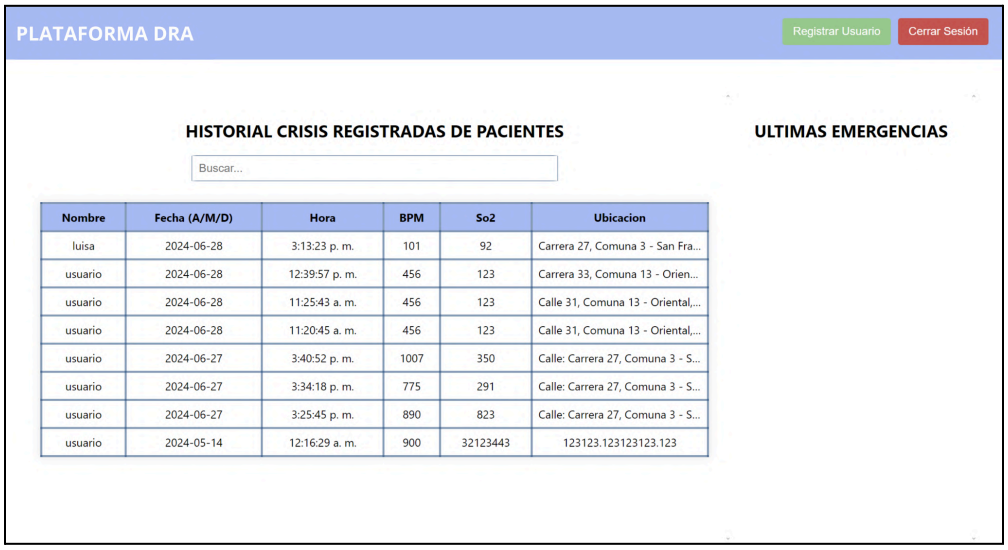


Figura 82.

Alerta recibida y guardada en el historial del paciente

5.5.2 Retroalimentación y sugerencias de expertos especialistas

El Dr. Iván Peña, luego de haber observado las pruebas realizadas y todo el funcionamiento y flujo de la plataforma, concluyó que se logró con éxito el objetivo planteado inicialmente. A su vez, brindó ciertas sugerencias para tener en consideración al momento de realizar una mejora a la plataforma, las cuales fueron:

- Al momento de generar las alertas para el centro de salud, estas deberían tener un campo de información en el que se haga mención del medicamento que toma él

paciente para tratar sus respectivas crisis, esto con el fin de que el servicio médico esté al tanto del tratamiento e historial del paciente.

- Sugiere que en el panel de administrador, se tenga la posibilidad que al momento de filtrar por el nombre del paciente para saber en específico las crisis presentadas y atendidas por este, se pueda observar la cantidad y un promedio de las crisis en cierto periodo de tiempo.
- Sugiere que el material en el que está albergado el dispositivo debería ser más resistente, esto con el fin de evitar que se estropee o dañen los componentes internos, ya que algunos pacientes al momento de las crisis presentan caídas y golpes.

6. Conclusiones

El proyecto nació con la finalidad de desarrollar una plataforma que permitiera detectar, realizar un registro y alertar las crisis convulsivas en pacientes que sufren de epilepsia mediante un dispositivo portátil que pudiera ser portado por el usuario brindándole cierto grado de autonomía.

El proyecto cumplió con todos los objetivos planteados, ya que se logró diseñar y desarrollar un sistema que haciendo uso de un dispositivo portátil permite detectar y alertar sobre emergencias generadas por crisis tonico clonicas en pacientes epileptico identificando patrones de movimientos corporales. Con esta solución no solo se facilitó la detección temprana de las crisis, si no que se brindaron soluciones de software que permitieron al usuario hacer gestión de

cierta información relacionada con su enfermedad, notificar emergencias, y una panel de administrador que centralizaba la recepción e información de emergencias.

En relación con el cumplimiento de los objetivos, es importante recalcar que: la precisión que se obtuvo en la detección de las crisis convulsivas asociadas a ataques epilépticos fue considerablemente buena pudiendo detectar alrededor del 80% de las crisis evaluadas. La arquitectura logró cumplir a cabalidad los requerimientos descritos desde el comienzo del proyecto, cumpliendo satisfactoriamente cada una de las pruebas planteadas. La aplicación móvil permite y dashboard web de igual forma, respondieron de forma satisfactoria dando asi cumplimiento a los requerimientos descritos al inicio del proyecto

La elección de la metodología en cascada en lugar de una de las famosas metodologías ágiles resultó ser una elección fundamental para el desarrollo del proyecto. Pues su enfoque secuencial y estructurado, permitió que cada fase se completará antes de pasar a la siguiente. Al tener esta secuencia bien definida, cada etapa fue planificada con detalle, asignando recursos y tiempo de manera efectiva. Desde el inicio del proyecto se tuvo una estructura clara y definida, desde la recopilación y análisis de la información hasta la respectiva evaluación y pruebas del prototipo, facilitando la medición del progreso y el cumplimiento de los objetivos.

El uso de las tecnologías IoT también fue un factor clave en la obtención de los objetivos, pues con ayuda de estas fue que se permitió un monitoreo continuo y en tiempo real, proporcionando datos de carácter primordial para el tratamiento de la condición médica de los pacientes. Tanto los sensores, los cuales, fueron capaces de recolectar datos sobre el ritmo

cardíaco y los movimientos corporales y la capacidad de generar alertas cuando se detectaban patrones anómalos por medio del dispositivo y su conexión con la aplicación móvil.

El prototipo desarrollado fue validado satisfactoriamente utilizando ambientes de simulación de grado clínico, lo que nos permite afirmar que este prototipo representa un excelente punto de partida para el desarrollo de herramientas que permitan a los pacientes con epilepsia llevar una vida cotidiana sin miedo ni incertidumbre. Gracias a este dispositivo, los pacientes podrán contar con una herramienta con cierto grado de confiabilidad para detectar y alertar sobre emergencias, mejorando significativamente su calidad de vida y proporcionando una mayor tranquilidad tanto a ellos como a sus seres queridos.

7. Recomendaciones

Una alternativa al uso de algoritmos tradicionales y con el auge que presenta hoy en día, es el uso de la inteligencia artificial. Aprovechando la toma de datos de los comportamientos y particularidades de las crisis convulsivas, sería posible desarrollar un modelo predictivo haciendo uso de técnicas como machine learning para identificar patrones en el comportamiento de la aceleración durante una crisis convulsiva, siendo capaz de generar una detección más precisa y con mayor confiabilidad, evitando escenarios de posibles falsos positivos en donde se confunda una actividad que realice el usuario con una crisis convulsivas

Sería recomendable analizar y estudiar qué beneficios traería en la precisión de la detección la implementación de más sensores que se encarguen de monitorear algunos de los otros aspectos o variables asociados a la enfermedad que se mencionaron. El uso de sensores que capturan variabilidad en la temperatura corporal, o sensores electromiográficos que permitan saber cuando se presentan contracciones musculares en conjunto con lo ya desarrollado, podría crear un método más robusto para la detección de crisis asociadas a la enfermedad.

Finalmente por parte del director y codirector se recomienda analizar la posibilidad de continuar con el estudio que se planteó este trabajo, llevando el análisis y pruebas a un entorno con pacientes reales, los cuales puedan ser monitoreados durante un cierto periodo de tiempo y concluir alrededor de estas pruebas.

Referencias Bibliográficas

Patel, A. D., Moss, R., Rust, S. W., Patterson, J., Strouse, R., Gedela, S., Haines, J., & Lin, S. (2016). Patient-centered design criteria for wearable seizure detection devices. Epilepsy & Behavior, 64, 116-121. https://doi.org/10.1016/j.yebeh.2016.09.012

Ali, A. (2018). Global Health: Epilepsy. Seminars in Neurology, 38(02), 191-199. https://doi.org/10.1055/s-0038-1646947

Fisher, N., Talathi, S. S., Cadotte, A. J., & Carney, P. R. (2008). Epilepsy Detection and Monitoring. Quantitative EEG Analysis Methods and the Application. http://www.sachintalathi.com/wp-content/uploads/2013/04/BookChapter_Detection1.pdf

Daoud, H., Williams, P., & Bayoumi, M. (2020). IoT based Efficient Epileptic Seizure Prediction System Using Deep Learning. IEEE 6th World Forum on Internet of Things (WF-IoT). https://doi.org/10.1109/wf-iot48130.2020.9221169

Mchale, S., & Pereira, E. (2021). An IoT based epilepsy monitoring model. En Lecture notes in networks and systems (pp. 192-207). https://doi.org/10.1007/978-3-030-80129-8_15

Understanding seizures. (s. f.-b). Epilepsy Foundation. https://www.epilepsy.com/what-is-epilepsy/understanding-seizures

Epilepsia. (s. f.). OPS/OMS | Organización Panamericana de la Salud. https://www.paho.org/es/temas/epilepsia

World Health Organization: WHO. (2023, 9 febrero). Epilepsia. https://www.who.int/es/news-room/fact-sheets/detail/epilepsy

Beghi, E., Giussani, G., Nichols, E., Abd‐Allah, F., Abdela, J., Abdelalim, A., Abraha, H. N., Adib, M. G., Agrawal, S., Alahdab, F., Awasthi, A., Ayele, Y., Barboza, M. A., Belachew, A. B., Biadgo, B., Bijani, A., Bitew, H., Carvalho, F., Chaiah, Y., . . . Tsegay, A. (2019).

Global, Regional, and National Burden of Epilepsy, 1990–2016: A Systematic Analysis for the Global Burden of Disease Study 2016. The Lancet Neurology, 18(4), 357-375. https://doi.org/10.1016/s1474-4422(18)30454-x

Trinka, E., Rainer, L., Granbichler, C. A., Zimmermann, G., & Leitinger, M. (2023). Mortality, and life expectancy in Epilepsy and Status epilepticus—current trends and future aspects. Frontiers, 3. https://doi.org/10.3389/fepid.2023.1081757

Kwon, O. Y., & Park, S. P. (2014). Depression and anxiety in people with epilepsy. Journal of Clinical Neurology, 10(3), 175. https://doi.org/10.3988/jcn.2014.10.3.175

Epilepsia: Mucho más que convulsiones. (2017, 13 febrero). Ministerio de Salud y Protección Social. Recuperado 31 de enero de 2024, de https://www.minsalud.gov.co/Paginas/Epilepsia-mucho-mas-que-convulsiones.aspx

Mental health: neurological disorders. (2016). https://www.who.int/news-room/questions-and-answers/item/mental-health-neurological-disorders

Campos, M. R. (2000). Mortalidad en las epilepsias. Revista De Neurología, 30(S1), 110. https://doi.org/10.33588/rn.30s1.2000138

Fisher, R. S., Acevedo, C. A., Arzimanoglou, A., Bogacz, A., Cross, J. H., Elger, C. E., Engel, J., Forsgren, L., French, J. A., Glynn, M., Hesdorffer, D. C., Lee, B. I., Mathern, G. W., Moshé, S. L., Perucca, E., Scheffer, I. E., Tomson, T., Watanabe, M., & Wiebe, S. (2014). ILAE Official Report: A Practical Clinical Definition of Epilepsy. Epilepsia, 55(4), 475-482. https://doi.org/10.1111/epi.12550

Salazar, J., & Silvestre, S. (2016). Internet de las cosas. Techpedia. České vysoké učení technické v Praze Fakulta elektrotechnická. https://psm.fei.stuba.sk/pages/95/LM08_F_ES.pdf

¿Qué es el Internet de las cosas (IoT)? (s. f.). Oracle Colombia. https://www.oracle.com/co/internet-of-things/what-is-iot/

İbrahim, D. (2006). Microcontroller based applied digital Control. https://www.amazon.com/Microcontroller-Based-Applied-Digital-Control/dp/0470863358

Borja, C. T., & Bueno, Á. G. (2006). Sistemas biométricos. Disponible en internet: http://www.dsi.uclm..es/asignaturas/42635/web_BIO/Documentacion/Trabajos/Biometria/Trabajo% 20Biometria. pdf.

Laukkanen, R., & Virtanen, P. (1998). Heart rate monitors: state of the art. Journal of Sports Sciences, 16(sup1), 3-7. https://doi.org/10.1080/026404198366920

LME Editorial Staff. (2022, 6 febrero). Interfacing MAX30102 pulse oximeter and heart rate sensor with arduino. Last Minute Engineers. https://lastminuteengineers.com/max30102-pulse-oximeter-heart-rate-sensor-arduino-tutorial/

Aguilar Cordero, M. J., Sánchez López, A. M., Barrilao, G., Rodríguez Blanque, R., Noack Segovia, J., & Cano, P. (2014). Descripción del acelerómetro como método para valorar la actividad física en los diferentes periodos de la vida: revisión sistemática. Nutrición hospitalaria, 29(6), 1250-1261.

Pozo Espín, D. F. (2010). Diseño y construcción de una plataforma didáctica para medir ángulos de inclinación usando sensores inerciales como acelerómetro y giroscopio (Bachelor's thesis, QUITO/EPN/2010).

Bass, L., Clements, P., & Kazman, R. (2012). Software Architecture in practice. Addison-Wesley.

Mell, P., & Grance, T. (2011). The NIST definition of cloud Computing. https://doi.org/10.6028/nist.sp.800-145

Software de modelado 3D | versiones de prueba y tutoriales gratuitos | Autodesk. (s. f.). https://latinoamerica.autodesk.com/solutions/3d-modeling-software

¿Qué es la impresión 3D? | Programa para impresora 3D | Autodesk. (s. f.). https://latinoamerica.autodesk.com/solutions/3d-printing

Empatica. (s. f.). Embrace2 Seizure Monitoring | Smarter Epilepsy Management | Embrace Watch | Empatica. https://www.empatica.com/en-eu/embrace2/

NightWatch Epilepsy Seizure Detection. (2024, 24 enero). NightWatch | Epilepsy Seizure Detection During Sleep. NightWatch. https://nightwatchepilepsy.com/

Life Minder. (2023, 18 enero). Emergency Pendants Alarms for Elderly Australia | Fall Detector Pendant for Seniors. https://lifeminder.com.au/shop-emergency-pendants-alarm-for-seniors/