



UNIVERSIDAD POLITÉCNICA SALESIANA

SEDE: GUAYAQUIL

CARRERA DE INGENIERÍA EN BIOMEDICINA

**“DISEÑO Y PROTOTIPADO DE UN SISTEMA
AUTOMÁTICO PARA LA TOMA DE SIGNOS VITALES EN EL
PROCESO DE TRIAJE EN CENTROS DE SALUD DE PRIMER
NIVEL DEL GUASMO CENTRAL”**

Trabajo de titulación previo a la obtención del

Título de Ingeniero Biomédico

AUTORES: Jabes Lucas Ordóñez Núñez & Miguel Ángel Yagual Lima.

Tutor: Ing. Flavio Vicente Moreno Villamarin, Mgtr.

Guayaquil-Ecuador

2024

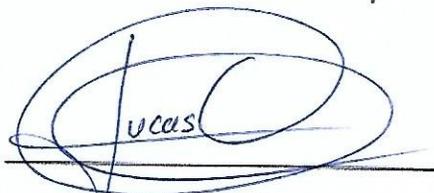
**CERTIFICADO DE RESPONSABILIDAD Y AUTORÍA DEL TRABAJO
DE TITULACIÓN.**

Nosotros, **Jabes Lucas Ordóñez Núñez** con documento de identificación N°0955597919 y **Miguel Ángel Yagual Lima** con documento de identificación N°0915991244; manifestamos que:

Somos los autores y responsables del presente trabajo; y, autorizamos a que sin fines de lucro la Universidad Politécnica Salesiana pueda usar, difundir, reproducir o publicar de manera total o parcial el presente trabajo de titulación.

Guayaquil, 23 de agosto del año 2024

Atentamente,

A handwritten signature in blue ink, appearing to read 'Jabes Lucas', enclosed within a large, loopy scribble.

Jabes Lucas Ordóñez Núñez

C.I. 0955597919

A handwritten signature in blue ink, appearing to read 'Miguel Ángel Yagual Lima', written in a cursive style.

Miguel Ángel Yagual Lima

C.I. 0955597919

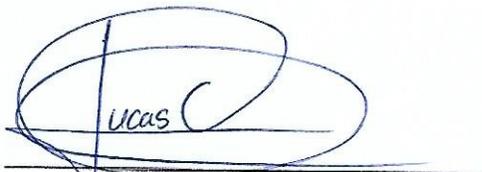
**CERTIFICADO DE CESIÓN DE DERECHOS DE AUTOR DEL TRABAJO DE
TITULACIÓN A LA UNIVERSIDAD POLITÉCNICA SALESIANA**

Nosotros, **Jabes Lucas Ordóñez Núñez** con documento de identificación No. **0955597919** **Miguel Ángel Yagual Lima** con documento de identificación No. **0915991244**, expresamos nuestra voluntad y por medio del presente documento cedemos a la Universidad Politécnica Salesiana la titularidad sobre los derechos patrimoniales en virtud de que somos autores del **Proyecto técnico: DISEÑO Y PROTOTIPADO DE UN SISTEMA AUTOMÁTICO PARA LA TOMA DE SIGNOS VITALES EN EL PROCESO DE TRIAJE EN CENTROS DE SALUD DE PRIMER NIVEL DEL GUASMO CENTRAL**, el cual ha sido desarrollado para optar por el título de: **Ingeniero en Biomedicina**, en la Universidad Politécnica Salesiana, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente.

En concordancia con lo manifestado, suscribimos este documento en el momento que hacemos la entrega del trabajo final en formato digital a la Biblioteca de la Universidad Politécnica Salesiana.

Guayaquil, 23 de agosto del año 2024

Atentamente,

Handwritten signature of Jabes Lucas Ordóñez Núñez in blue ink, featuring a stylized 'J' and 'L' with the name 'ucas' written in the middle.

Jabes Lucas Ordóñez Núñez

C.I. 0955597919

Handwritten signature of Miguel Ángel Yagual Lima in blue ink, consisting of a series of loops and strokes.

Miguel Ángel Yagual Lima

C.I. 0955597919

CERTIFICADO DE DIRECCIÓN DEL TRABAJO DE TITULACIÓN
SUSCRITO POR EL TUTOR.

Yo, **Flavio Vicente Moreno Villamarin** con documento de identificación N° **1205480542**, docente de la Universidad Politécnica Salesiana , declaro que bajo mi tutoría fue desarrollado el trabajo de titulación: **DISEÑO Y PROTOTIPADO DE UN SISTEMA AUTOMÁTICO PARA LA TOMA DE SIGNOS VITALES EN EL PROCESO DE TRIAJE EN CENTROS DE SALUD DE PRIMER NIVEL DEL GUASMO CENTRAL**, realizado por **Jabes Lucas Ordóñez Núñez** con documento de identificación N°**0955597919** y por **Miguel Ángel Yagual Lima** con documento de identificación N°**0915991244**, obteniendo como resultado final el trabajo de titulación bajo la opción **Proyecto Técnico** que cumple con todos los requisitos determinados por la Universidad Politécnica Salesiana.

Guayaquil, 23 de agosto del año 2024

Atentamente,



Ing. Flavio Vicente Moreno Villamarin

C.I. 1205480542

DEDICATORIA Y AGRADECIMIENTO.

Lucas Ordóñez.

Deseo, en primer lugar, agradecerle a Dios por cada día de vida que me ha permitido, así como permitirme gozar de esta experiencia llamada Universidad. De la misma forma, le agradezco por haberme permitido encontrarme rodeado de una familia que me ha otorgado su constante apoyo y amor de forma incondicional. Le agradezco a mi padre, Sr. Luis Ordóñez, por abrirme las puertas al mundo del conocimiento. Le agradezco a mi madre, Sra. Sonia Núñez, por enseñarme de su resiliencia y sabiduría. Le agradezco a mi tía y segunda madre, Sra. Maritza Ordóñez, por ser un modelo para seguir por el resto de mi vida. Por último, le agradezco mi novia Melany Sánchez puesto que sin ella este proyecto no hubiera sido posible.

Miguel Yagual.

Agradezco en primer lugar a Dios por permitirme culminar una etapa que tenía inconclusa, a mi Madre Sra. Guillermina Lima que fue uno de los motores principales que me impulsó a seguir estudiando, a mi familia por darme su apoyo incondicional.

RESUMEN

El proyecto técnico presentado en este documento tiene como enfoque el diseño y prototipado de un equipo con capacidad de toma de signos vitales y ponderación de estos para la realización de una clasificación de triaje. Siendo su propósito principal ser soporte en centros de salud de primer nivel del Guasmo Central durante la atención de urgencias, evitando de esta forma que las urgencias se vuelvan emergencias.

El prototipo funcional se encuentra implementado por el microcontrolador Arduino Mega, pantalla HMI Nextion y sensores para la toma de los signos vitales: temperatura, presión arterial, oximetría, frecuencia cardíaca y frecuencia respiratoria. Además, cuenta con una valoración de la escala de Glasgow con el fin de obtener información sobre el estado de conciencia de los pacientes que se harán atender.

Palabras clave: triaje, signos vitales, microcontroladores, Glasgow, Nextion, Arduino.

ABSTRACT.

The technical project presented in this document focuses on the design and prototyping of a device capable of measuring vital signs and evaluating them to perform triage classification. Its main purpose is to support first-level health centers in Guasmo Central during emergency care, thereby preventing emergencies from escalating further.

The functional prototype is implemented with an Arduino Mega microcontroller, a Nextion HMI display, and sensors for measuring vital signs: temperature, blood pressure, oximetry, heart rate, and respiratory rate. Additionally, it includes a Glasgow Coma Scale assessment to gather information about the consciousness level of patients to be attended.

Key words: triage, vital signs, microcontroller, Glasgow, Nextion, Arduino.

ÍNDICE GENERAL

CERTIFICADO DE RESPONSABILIDAD Y AUTORÍA DEL TRABAJO DE TITULACIÓN.....	ii
CERTIFICADO DE CESIÓN DE DERECHOS DE AUTOR DEL TRABAJO DE TITULACIÓN A LA UNIVERSIDAD POLITÉCNICA SALEASIANA	iii
CERTIFICADO DE DIRECCIÓN DEL TRABAJO DE TITULACIÓN SUSCRITO POR EL TUTOR.....	iv
DEDICATORIA Y AGRADECIMIENTO.....	v
RESUMEN	vi
ABSTRACT.....	vii
ÍNDICE DE TABLAS.....	xiii
ÍNDICE DE FIGURAS	xiv
INTRODUCCIÓN.....	1
PLANTEAMIENTO DEL PROBLEMA.....	3
ANTECEDENTES.....	3
IMPORTANCIA Y ALCANCES	5
Importancia.....	5
Alcances.....	6
OBJETIVOS.....	7

OBJETIVO GENERAL	7
OBJETIVOS ESPECÍFICOS.....	7
FUNDAMENTOS TEÓRICOS.	8
1. Sistema de Triage.....	8
1.1 Evolución histórica del Triage.....	8
1.2 Importancia del Triage en la modernidad.....	10
1.2.1 Definiciones del Triage en la modernidad.	10
1.2.2 Triage en el manejo de Urgencias y Emergencias.....	10
1.2.3 Sistema de Triage durante situaciones de pandemia: COVID-19 /	
Gestión clínica en situaciones de alta demanda.....	11
1.3 Tipos de Triage.	13
1.3.1 Manchester Triage System (MTS)	14
1.3.2 Emergency Severity Index (ESI).....	16
1.3.3 Sistema Español de Triage (SET).....	18
1.3.4 Australasian Triage Scale (ATS).....	20
2. Signos vitales.....	22
2.1 Definición de signos vitales.....	22
2.2 Tipos de signos vitales.....	22
2.2.1 Temperatura Corporal.....	22
2.2.2 Frecuencia Respiratoria	23
2.2.3 Presión arterial.....	24
2.2.4 Oximetría.....	24

2.2.5 Pulso arterial.....	24
2.3. Relación entre los signos vitales y el nivel de urgencia.	25
3. Escala de Glasgow	25
4. Procesos actuales en la adquisición de signos vitales.	27
4.1 Métodos Tradicionales de Toma de Signos Vitales.....	27
4.2 Avances Tecnológicos en la adquisición de Signos Vitales	28
4.3 Impacto de la Tecnología en la Atención Médica	29
5. Descripción del Proceso de Triage en Centros de Salud de Primer Nivel en Ecuador.....	30
5.1 Proceso de Triage	30
5.2 Desafíos del Triage en Ecuador	31
5.3 Mejoras Prácticas	32
6. Sistemas Automáticos y Dispositivos Médicos para la Adquisición de Signos Vitales.....	34
6.1 Principales Sistemas Automáticos y Dispositivos Médicos	34
6.2 Impacto General en la Atención Sanitaria	36
7. Microcontroladores Arduino.	38
7.1 Origen	38
7.2 Tipos de Arduino.	38
7.2.1 Arduino UNO	38
7.2.2 Arduino Mega.....	39
MARCO METODOLÓGICO	40

1. Microcontrolador y sensores.	40
1.1 Arduino Mega	40
1.2 Sensores para la adquisición de signos vitales.....	41
1.2.1 Sensor KY-037.	41
1.2.2 Sensor digital de temperatura DS18B20A10.....	42
1.2.3 Sensor MAX-30102.....	43
1.2.4 Monitor automático de presión arterial de muñeca.	44
2. Pantalla Nextion.	45
3. Selección de la escala de gravedad basada en signos vitales.	46
4. Diseño del Sistema.	48
4.1 Adaptación física de los sensores.	48
4.2 Conexión de sensores.....	50
4.3 Diseño de la interfaz de máquina-usuario en Nextion Editor.	54
4.3.1 Pantalla Inicio.	54
4.3.2 Pantalla de Información.....	55
4.3.3 Pantalla de Escala de Glasgow.	56
4.3.4 Pantalla de Toma de signos vitales.....	58
4.3.5 Pantalla de Resultados del Triage.	59
4.4 Implementación física del sistema Nextion-Arduino.	60
4.5 Conexión Nextion-Arduino.	63
5. Programación del Sistema.	63
5.1 Programación del Arduino Mega secundario.	63

5.2 Programación del Arduino Mega Principal.	67
5.2.1 Librerías.....	67
5.2.2 Definición de pines a utilizar.....	67
5.2.3 Variables Globales.....	68
5.2.4 Matrices de triaje	69
5.2.5 Void Setup ()	70
5.2.6 Void Loop ()	71
5.2.7 Funciones para procesamiento de datos.	79
5.2.8 Código completo	81
Resultados.....	95
1. Puesta a prueba del equipo	95
1.1 Ingreso de nombre y edad.....	95
1.2 Realización de la escala de Glasgow.....	97
1.3 Toma de signos vitales.....	98
1.4 Realización del Triage	100
2. Pruebas de verificación.	100
CONCLUSIONES.....	105
RECOMENDACIONES	107
BIBLIOGRAFÍA.....	108
ANEXOS.....	113

ÍNDICE DE TABLAS

Tabla 1. Niveles de clasificación del Emergency Severity Index	17
Tabla 2. Niveles de clasificación Sistema Español de Triage	19
Tabla 3. Clasificación de triaje basado en signos vitales y escala de Glasgow..	47
Tabla 4. Signos vitales del prototipo	101
Tabla 5. Signos vitales del monitor multiparámetros	101

ÍNDICE DE FIGURAS

Figura 1. Patrullas ambulatorias de Dominique-Jean Larrey, 1809	8
Figura 2. Niveles de la escala de triaje de Manchester, códigos de colores y tiempos de evaluación	15
Figura 3. Diagrama de flujo ESI. Fuente. ESI Handbook 5th Edition.	17
Figura 4. Categorías de la Escala de Triaje de Australasia	21
Figura 5. Escala de Glasgow. Fuente: Elaboración propia.....	26
Figura 6. Arduino UNO. Fuente: https://store.arduino.cc/products/arduino-uno- rev3	39
Figura 7. Arduino Nano. Fuente: https://store.arduino.cc/products/arduino-nano	¡Error! Marcador no definido.
Figura 8. Uso de 2 Arduino Mega	41
Figura 9. Sensor de sonido (KY-037 Datasheet.)	42
Figura 10. Sensor DS18B20 (DS18B20 Datasheet)	43
Figura 11. Sensor MAX30102 (DS18B20 Datasheet)	44
Figura 12. Tensiómetro digital de muñeca	44
Figura 13. PANTALLA NEXTION 7" NX8048T070 HMI SERIAL	45
Figura 14. Adaptación sensor KY-037	48
Figura 15. Adaptación física de sensor MAX-30102	49
Figura 16. Conexión de sensores en Proteus 8 Professional.	50
Figura 17. Conexión física interior de los sensores de signos vitales	51
Figura 18. Conexión física exterior de los sensores de signos vitales.....	51

Figura 19. Conexión de sensor MAX-30102 en Proteus 8 Professional.....	52
Figura 20. Conexión física interior del sensor MAX-30102.....	52
Figura 21. Conexión física exterior del sensor MAX-30102	53
Figura 22. Pantalla Inicio del sistema QuickTriage	54
Figura 23. Pantalla de Información del paciente	55
Figura 24. Teclado qwerty desplegado del cuadro de texto “Nombre”.....	56
Figura 25. Pantalla de Escala de Glasgow.....	56
Figura 26. Checkboxes de cada valoración de la escala de Glasgow.....	57
Figura 27. Touch Release Event y programación de Checkbox.....	58
Figura 28. Pantalla de Toma de signos vitales	58
Figura 29. Pantalla de Resultados del Triage	59
Figura 30. Diseño de la caja del sistema QuickTriage (a).....	60
Figura 31. Diseño de la caja del sistema QuickTriage (b).....	61
Figura 32. Caja del sistema QuickTriage (a).....	61
Figura 33. Caja del sistema QuickTriage (b).....	62
Figura 34. Caja del sistema QuickTriage (c).....	62
Figura 35. Ingreso de nombre y edad	96
Figura 36. Pantalla de Información del Paciente.....	96
Figura 37. Realización de la escala de Glasgow	97
Figura 38. Toma de signos vitales (a).....	98
Figura 39. Toma de signos vitales (b)	98
Figura 40. Toma de signos vitales (c).....	99
Figura 41. Realización del Triage	100
Figura 42. Paciente 1 signos vitales – Prototipo.....	102
Figura 43. Paciente 1 signos vitales - Monitor	102

Figura 44. Paciente 3 signos vitales - Prototipo y Monitor Multiparámetros ..	103
Figura 45. Paciente 3 Temperatura – Prototipo	103
Figura 46. Paciente 3 Temperatura - Termómetro infrarrojo	104

INTRODUCCIÓN

Los centros de salud de primer nivel en Guayaquil proveen servicios médicos ambulatorios y gestionan la atención de urgencias. Uno de sus principales roles es la determinación de la gravedad/urgencia del estado de salud de los pacientes, de forma que, puedan realizar una derivación, diagnóstico y atención terapéutica eficiente y prioritaria.

Para poder determinar la urgencia y el orden de atención, es óptimo registrarse por un sistema objetivo y parametrizado. Dicho sistema es conocido en el ámbito médico como “Proceso de Triage de Emergencia.” Un triaje podría definirse como un procedimiento permite la realización de una gestión del riesgo clínico con el fin de manejar adecuadamente los flujos de pacientes (Soler et al., 2010, p. 56.).

Alrededor del mundo, se han establecido una gran variedad de sistemas de triaje contextualizados a cada país. Aun así, es posible utilizar estos sistemas de triaje y adecuarlos en un contexto ecuatoriano. Entre los sistemas de triaje con mayor reconocimiento y mayor porcentaje de aplicación: se encuentran el Manchester Triage System (MTS), Escala de triaje australiana (ATS), la Escala de agudeza y triaje del Servicio de Urgencias de Canadá, el Sistema Español de Triage (SET) y el Índice de gravedad de emergencia (ESI) (Ávila y De la Rosa, 2022).

Actualmente en Ecuador, específicamente en la ciudad de Guayaquil, generalmente los centros de salud manejan una atención al paciente en dependencia del orden de llegada, a la vez, no se cuenta con un proceso ágil de aviso al personal médico sobre la situación del paciente. Finalmente, cuando se habla de triaje en el contexto ecuatoriano, se relaciona con una toma de signos vitales y mediciones antropométricas

que no conllevan a la correspondiente clasificación de la urgencia del estado de salud del paciente y, por ende, la priorización de su atención médica y terapéutica.

El proceso de triaje podría ser perfectamente implementado en los diversos centros de salud de primer nivel especializados en la atención de urgencias, evitando que las urgencias se vuelvan emergencias médicas poniendo la vida de los pacientes en un riesgo mortal. Dicho proceso de triaje puede recibir soporte de un prototipo de sistema inteligente y automatizado, de forma que la atención médica sea rápida y eficiente.

PLANTEAMIENTO DEL PROBLEMA

Como mencionan Chérrez-Anguizaca y León-Micheli (2021) actualmente las casas de salud públicas en el Ecuador poseen fallas en la atención inmediata, siendo así necesario la implementación de estrategias que representen una mejora en el proceso de triaje. Partiendo de esta premisa, el proceso de triaje puede ser optimizado con la implementación de la ingeniería biomédica.

El proyecto descrito en este documento propone el desarrollo de un prototipo de sistema inteligente de toma de información y signos vitales correspondientes al triaje, que además tenga la capacidad de catalogar automáticamente la urgencia en la que se encuentra el paciente y por ende establecer un orden de atención, así mismo, realizar un aviso al personal médico en caso de urgencias catalogadas como graves. Se pretende que este sistema inteligente represente un soporte para el proceso de triaje, obteniendo de esta forma una gestión rápida del flujo de pacientes. Es por eso, que este proyecto busca desde el ámbito tecnológico y biomédico dar soporte a los centros de salud de primer nivel en el proceso de triaje, mejorando de esta forma la atención inmediata al paciente.

ANTECEDENTES

Previo al desarrollo de un sistema inteligente y automatizado que asista en el proceso de Triaje, fue requerido la indagación de información sobre el estado actual del proceso de Triaje en el Ecuador. A continuación, se presentan estudios realizados en Ecuador sobre la situación e importancia del proceso de Triaje:

- Chérrez-Anguizaca y León-Micheli (2021) realizaron un estudio sobre la aplicación del Triaje en el Ecuador, partiendo su idea del ineficiente proceso de atención llevado a cabo durante la pandemia del COVID-19, todo esto con el fin de mejorar la atención en las casas de salud. Una vez realizada su investigación,

podieron concluir que el sistema de Triage en Ecuador es ineficiente y presenta varias falencias, incluso previo a la pandemia del COVID-19. Uno de los puntos claves mencionados, fue la importancia del tiempo de atención, donde mencionan “el tiempo de atención en el área de Triage de emergencias debe de ser el mínimo posible...” (Chérrez-Anguizaca y León-Micheli, 2021).

- Ávila-Cárdenas & Rosa-Ferrera (2022) mencionan que a pesar de la presencia de sistemas de triaje en el Ecuador, todavía es posible encontrar pacientes que presentan quejas sobre el tiempo de atención. Así mismo, determinan que el Triage puede ser considerado “clave de la eficiencia y efectividad clínica del servicio de emergencias.” (Ávila-Cárdenas & Rosa-Ferrera, 2022)

No cabe duda de que el proceso de Triage es pilar fundamental en la atención de urgencias y emergencias. En Ecuador, es posible encontrar una deficiencia en la atención en los distintos lugares de salud, siendo uno de los problemas más frecuentes el tiempo de atención y el reconocimiento rápido de complicaciones médicas. Un sistema que automatice el proceso de signos vitales durante el proceso de triaje, a la vez que automatiza la clasificación de prioridad de atención, puede significar una mejora sustancial en la calidad y eficacia de la atención médica.

IMPORTANCIA Y ALCANCES

Importancia.

El sistema automático de toma de signos vitales para el triaje en centros de salud de primer nivel en Ecuador presenta una importancia crucial para mejorar la calidad de la atención médica y la eficiencia del sistema sanitario en general.

En primer lugar, una característica principal del prototipo de sistema automático de toma de signos vitales para el triaje radica en su capacidad para agilizar el proceso de triaje y, en consecuencia, reducir significativamente los tiempos de espera de los pacientes. Esto se logra mediante la automatización de la toma de signos vitales y la clasificación de pacientes según su nivel de urgencia. Esta automatización no solo libera al personal de enfermería para que se concentre en tareas más complejas que requieren mayor interacción con los pacientes, sino que también permite identificar de manera rápida y precisa a aquellos pacientes que requieren atención médica urgente. Esta identificación temprana de casos graves es crucial para garantizar que reciban el tratamiento adecuado de manera oportuna, lo que a su vez reduce el riesgo de complicaciones y mejora los resultados en salud.

Por otro lado, el uso de un sistema de automatización aumenta la precisión y confiabilidad de los datos obtenidos. La toma automatizada de signos vitales minimiza la posibilidad de errores humanos en la medición y registro de datos, lo que a su vez contribuye a un diagnóstico y tratamiento más precisos. A su vez, el sistema garantiza que el proceso de triaje se realice de manera uniforme y estandarizada, independientemente del profesional que lo lleve a cabo, lo que mejora la calidad y confiabilidad del proceso.

Finalmente, una vez agilizado el proceso, es posible que los pacientes reciban una atención inmediata y oportuna de forma que el riesgo de mortalidad en pacientes críticos disminuya.

Alcances.

El sistema inteligente de triaje propuesto se ha diseñado para ser implementado en centros de salud de primer nivel en Ecuador. Su escalabilidad permitirá adaptarlo a diferentes realidades y necesidades, desde pequeñas unidades de salud hasta hospitales más grandes.

La implementación de un sistema inteligente de toma de signos vitales para el triaje en centros de salud de primer nivel en Ecuador representa una oportunidad significativa para mejorar la atención inmediata de los pacientes, optimizar el uso de recursos y contribuir a una mejor experiencia para los usuarios del sistema de salud pública. La ingeniería biomédica, en conjunto con el conocimiento médico y la experiencia en gestión de servicios de salud, puede ser un factor clave para lograr este objetivo.

OBJETIVOS

OBJETIVO GENERAL

- Diseñar y prototipar un sistema automático para la toma de información en el proceso de triaje en centros de salud de primer nivel, mediante un sistema de microcontroladores.

OBJETIVOS ESPECÍFICOS

- Identificar y recopilar los datos necesarios para el registro de pacientes en el sistema automático de triaje.
- Desarrollar un sistema de clasificación de gravedad de los pacientes, basados en los síntomas.
- Elaborar un sistema automático intuitivo y fácil de usar para que los pacientes puedan registrar su información de manera rápida y precisa.
- Realizar las pruebas de uso para verificar que sus lecturas y resultados estén de acuerdo con el proceso de triaje que se usa actualmente en Ecuador
- Diseñar un prototipo funcional de un sistema automático para la toma de signos vitales.

FUNDAMENTOS TEÓRICOS.

1. Sistema de Triage.

1.1 Evolución histórica del Triage.

Etimológicamente, la palabra “traje” proviene del francés “trier” que se puede definir como categorizar, separar o escoger. Su primer uso en el ámbito de la salud fue durante las Guerras Napoleónicas a manos del doctor Dominique Jean Larrey (Mitchell, 2008). Jean Larrey notó una deficiencia en la atención médica recibida por parte de los soldados heridos, a su vez dio cuenta de la falta de un sistema estructurado de atención (Mitchell, 2008). Ambos factores serían los principales culpables de una alta tasa de mortalidad durante la guerra. Siendo consciente de esto y en busca de una solución, decidió establecer patrullas ambulantes de médicos (Figura 1) cerca del campo de batalla y un sistema donde los soldados más graves serían los primeros en ser atendidos sin tomar en consideración el rango militar (previamente a mayor rango mayor prioridad).



Figura 1. Patrullas ambulatorias de Dominique-Jean Larrey, 1809 (Mitchell, 2008)

El objetivo principal sería atender a los gravemente heridos de forma que puedan regresar a sus posiciones de guerra rápidamente. Una vez realizada la valoración y en caso de que el soldado herido no pueda regresar prontamente al campo de guerra, sería enviado a la zona hospitalaria más cercana. Este sistema fue rápidamente implementado en otros países como Inglaterra y Turquía, aunque aún no bajo el nombre de triaje. Históricamente, este suceso sería reconocido como la primera forma de implementación del Sistema de Triaje.

La presencia continua de guerras a lo largo de los años, tales como la Guerra Civil de Estados Unidos, la Primera Guerra Mundial, la Segunda Guerra Mundial y la Guerra de Vietnam, además del involucramiento creciente de la población civil en las mismas, permitió extender este sistema de triaje a lo largo de toda la comunidad médica. Siendo la idea principal priorizar el tratamiento de los civiles gravemente heridos. A su vez, la evolución y efectividad de los primeros auxilios médicos impulsó el establecimiento de un triaje que provea de una atención rápida para la incorporación inmediata de los soldados gravemente heridos y con esperanza de vida. Otro factor importante por tomar en consideración es la evolución de los métodos de transporte que permitieron desplazar los hospitales más cercanos a aquellos soldados con bajas esperanzas de vida, priorizando la atención de estos una vez llegada a la casa de salud.

En la modernidad y basado en este sistema de atención, se empezaron a utilizar servicios ambulatorios para la atención del público civil, utilizando ambulancias y equipo médico estandarizado para la atención médica de cada país. La atención salió de las puertas de los hospitales para dirigirse a locaciones donde los siniestros habían ocurrido. Un sistema de Triaje empezó a establecerse con mayor fuerza. Finalmente, es posible observar diversos sistemas de Triaje llevados a cabo en la sala de emergencia de un hospital, en centros de salud enfocados en el manejo de urgencias, entre otros.

1.2 Importancia del Triage en la modernidad.

1.2.1 Definiciones del Triage en la modernidad.

Según Ganley & Closter (2011) el triaje puede ser definido como un proceso dinámico de clasificación médica que habilida la ubicación de los pacientes en el servicio de atención adecuado, con el fin de realizar un tratamiento rápido. A la vez, uno de los objetivos que el triaje debe tener es el de ser de fácil entendimiento y aplicación.

Por su parte, Ajani (2011) considera al Triage como el proceso en el que un paciente es evaluado, luego de su llegada, para determinar su urgencia y tipo de problema, con la finalidad de designar recursos de atención médica adecuados para atender el problema identificado. Posee como uno de sus objetivos ubicar al paciente en el área adecuada para que reciba el tratamiento adecuado en el momento adecuado.

Finalmente, Christ et. al. (2010) define al Triage como los métodos utilizados para evaluar la gravedad de la lesión o enfermedad de los pacientes en un corto período de tiempo después de su llegada, asignar prioridades y transferir a cada paciente al lugar apropiado para su tratamiento.

De esta forma y en base a una revisión bibliográfica, definimos el Triage como un proceso de selección y priorización de pacientes en situaciones de emergencia o con recursos limitados. Su objetivo principal es brindar la atención médica más adecuada y oportuna a los pacientes que la necesitan con mayor urgencia.

1.2.2 Triage en el manejo de Urgencias y Emergencias.

Uno de los conceptos más básicos manejados en el Triage es “lo urgente no siempre es grave y lo grave no siempre es urgente” (Soler et. al., 2010). De manera general, lo urgente implica una atención inmediata de forma que puedan evitarse consecuencias negativas, mientras que, lo grave refiere a un impacto significativo a largo plazo y que puede ser atendido de forma progresiva y no necesariamente inmediata. Para

el triaje, esto significa clasificar a aquellos pacientes que presenten situaciones urgentes de forma que los más urgentes puedan ser atendidos prioritariamente, mientras que los menos urgentes o graves puedan esperar hasta ser nuevamente evaluados por el médico.

Los departamentos de emergencia en los hospitales no tienen forma de predecir la cantidad de pacientes que tendrán que admitir en determinado momento. De esta forma, existirán situaciones donde los departamentos de emergencias y urgencias deben ser capaces de evaluar con rapidez y eficiencia la severidad clínica de los pacientes, asignando prioridades y permitiendo que cada uno sea atendido oportunamente. Para esto, es importante tomar en consideración las correspondientes funciones que un Triage debe cumplir (Soler et. al., 2010):

- Identificar pacientes que presenten una urgencia médica.
- Clasificar a aquellos pacientes con el fin de priorizar su atención.
- Considerar a los pacientes menos urgentes en espera de una reevaluación.
- Mejorar el flujo de los pacientes y la congestión clínica en el departamento.
- Determinar áreas apropiadas para la atención de los pacientes.
- Proporcionar información precisa sobre el paciente, tanto para el mismo como para los familiares.
- Gestión de los tiempos de espera.

1.2.3 Sistema de Triage durante situaciones de pandemia: COVID-19 / Gestión clínica en situaciones de alta demanda.

A lo largo del tiempo, la forma de implementación del Triage ha ido cambiando y evolucionando, sin embargo, su esencia se ha mantenido: clasificar pacientes tomando en consideración el grado de urgencia teniendo como fin que los más graves sean atendidos con prioridad. Además de este objetivo, el Triage posee un importante rol en la gestión

clínica y el flujo de pacientes, sobre todo en casos donde la demanda y necesidad clínica es superior a los recursos (Soler et. al., 2010). Dicha situación donde la demanda supera a los recursos puede ejemplificarse por medio de uno de los sucesos médicos más peligrosos en los últimos años: Pandemia del COVID-19.

Durante la pandemia del COVID-19 el flujo de pacientes en hospitales y centros de salud superó a la capacidad de atención de estos. Como mencionan Ávila et. al (2018) alrededor del mundo los sistemas sanitarios se encontraron en situaciones de escasez de insumos, falta de equipos y personal, volviéndose incapaces de una atención eficiente frente a tantos pacientes. Esto provocó la necesidad de tomar una decisión fundamental: reconocer qué pacientes requerían de hospitalización y atención inmediata y qué pacientes podrían simplemente realizar un aislamiento temporal en su hogar.

Al igual que el resto del mundo, Ecuador fue sorprendido ante la inesperada y crítica situación médica, donde en un inicio lamentablemente sus casas de salud no estaban preparadas para el gran flujo de paciente, ni contaban con un sistema óptimo de clasificación de paciente en dependencia de su situación médica. Prontamente, el Ministerio de Salud Pública estableció protocolos de atención inicial con la ayuda de un personal de Triage.

Ávila et. al (2018) en su investigación sobre el desafío sanitario que tuvo un hospital en Ecuador durante la pandemia, menciona como uno de los primeros protocolos implementados fue el de un personal de Triage. Dicho personal analizaba la sintomatología de cada paciente por medio de diversas evaluaciones. La evaluación CURV 65 determina si el paciente necesita ser ingresado al área de infectología clínica en dependencia de los siguientes signos y síntomas: estado de conciencia, niveles de urea, frecuencia respiratoria, presión arterial sistólica por debajo de 90mm Hg y poseer una edad de 65 años. Además de esta evaluación de triaje, se utilizaba la valoración quick

SOFA que tomaba en consideración: estado de conciencia, frecuencia respiratoria y presión arterial, si la calificación de esta valoración era mayor a 2 significaba la necesidad de ingresar al paciente a terapia intensiva. Caso contrario el paciente se realizaría un hisopado y sería enviado a casa.

El establecimiento de un sistema de Triage permitió a este hospital centinela identificar rápidamente a pacientes que tenían mayor riesgo de presentar futuras complicaciones, de forma que reciban atención médica de urgencia. A su vez, facilitó la asignación eficiente de recursos limitados y reducir la congestión en servicios de emergencia y UCI. Por otro lado, permitió la redistribución del personal sanitario hacia las áreas de mayor necesidad optimizando su uso.

1.3 Tipos de Triage.

Los sistemas de Triage deben poseer una característica principal: adaptabilidad a las circunstancias médicas del entorno. Esto deriva en la capacidad del sistema de triaje de cambiar en base a las necesidades médicas de donde se lo esté aplicando, capacidad de cambio en base a las especificidades de una enfermedad, etc. Es por esto que diversos países han optado por desarrollar sistemas de Triage que cumplan sus condiciones médicas particulares.

Por otro lado, existen diferentes sistemas porque las necesidades varían según la situación. Los servicios de urgencias priorizan la evaluación rápida de una amplia gama de afecciones, mientras que las zonas de desastre necesitan gestionar una afluencia masiva de pacientes. El triaje pediátrico se adapta a los niños que podrían no comunicarse de manera eficaz, y las limitaciones de recursos en algunas zonas pueden influir en qué pacientes reciben atención primero. Además, algunos sistemas pueden centrarse más en cuestiones específicas, como la salud mental. Los factores locales, como la prevalencia de enfermedades y las consideraciones culturales, también pueden influir en la forma en

que se prioriza a los pacientes. Sin embargo, a pesar de estas variaciones, todos los sistemas de triaje comparten el mismo objetivo principal: asegurar que cada uno de los pacientes sean atendidos de la formas más adecuada y oportuna posible.

De esta manera es posible encontrar diversos modelos de triaje, siendo los más reconocidos los siguientes (Ávila y De la Rosa, 2022):

- Sistema de Triaje Manchester (MTS)
- Índice de gravedad de emergencia (ESI)
- Sistema Español de Triaje (SET)
- Escala de triaje de Australasia (ATS)

1.3.1 Manchester Triage System (MTS)

El Sistema de Triaje Manchester (STM) es un modelo de clasificación para la priorización y predicción de riesgos en pacientes que busquen atención médica de emergencia. Es utilizado ampliamente en la Unión Europea, a la vez que puede llegar a ser aplicado en países latinoamericanos, como Ecuador, Brasil, Chile o México (Picallo, 2019).

El STM tiene como objetivo principal servir como soporte en momentos donde un paciente busca de un servicio de salud de emergencia, siendo atendidos en dependencia de la severidad clínica en vez de ser atendidos en dependencia del orden de llegada. El sistema de triaje de Manchester (STM) optimiza la evaluación de los pacientes en el departamento de urgencias mediante el uso de una biblioteca de 52 condiciones preprogramadas. Estos árboles de decisión, o diagramas de flujo, actúan como pautas para el departamento de enfermería. Después de que un paciente informa su queja principal, el enfermero selecciona el diagrama de flujo más relevante de la biblioteca del MTS. Al seguir las indicaciones del diagrama de flujo y considerar los signos y síntomas

específicos del paciente, el enfermero responsable puede categorizar de manera eficiente el nivel de urgencia del paciente.

Su clasificación se divide en 5 niveles de prioridad, cada uno identificado con un color, a su vez cada color determina el tiempo posible de espera hasta una atención: rojo (atención inmediata), naranja (muy urgente), amarillo (urgente), verde (menos urgente) y azul (no urgente) (Picallo, 2019).

Level	Status	Colour	Time to assessment
1	Immediate	Red	0 minutes
2	Very urgent	Orange	10 minutes
3	Urgent	Yellow	60 minutes
4	Standard	Green	120 minutes
5	Non-urgent	Blue	240 minutes

Mackway-Jones (2006)

Figura 2. Niveles de la escala de triaje de Manchester, códigos de colores y tiempos de evaluación (Lähdet, 2009)

Los signos vitales que pueden ser tomados en consideración durante el sistema de triaje Manchester son:

- Temperatura.
- Frecuencia cardiaca.
- Saturación de oxígeno.
- Frecuencia respiratoria.
- Presión arterial.

El Sistema de Triaje de Manchester (STM) se ha convertido en una herramienta invaluable para los departamentos de emergencia, desempeñando un papel fundamental en la priorización de la atención médica, la optimización de recursos y la mejora de los

resultados de salud para los pacientes. Su enfoque sistemático y flexible lo convierte en un modelo adaptable a diversos entornos de atención médica, contribuyendo a la eficiencia y calidad de la atención de emergencia en todo el mundo.

1.3.2 Emergency Severity Index (ESI)

El índice de gravedad de emergencia (ESI) es un sistema de clasificación de cinco niveles que se utiliza en los departamentos de emergencias (ED) para clasificar a los pacientes en función de la urgencia de su afección médica y sus necesidades de recursos previstos, fue principalmente desarrollado para mejorar tanto la confiabilidad como la validez en la toma de decisiones durante un triaje (Tanabe et. al., 2008).

A diferencia del SMT, que utiliza afecciones predefinidas, el ESI se centra en dos aspectos clave:

- **Agudeza:** se refiere a la gravedad de la afección del paciente y al potencial de deterioro. Los pacientes con afecciones potencialmente mortales o con alto riesgo de deterioro se consideran más agudos.
- **Necesidades de recursos:** considera la cantidad de personal médico, equipo y diagnósticos que es probable que requiera un paciente durante su tratamiento. Esto implica si el paciente requerirá recursos que van más allá de una examinación física.

En base a estos dos principales aspectos, el departamento de enfermería es capaz de categorizar al paciente en uno de los 5 niveles que maneja el ESI (Tabla 1).

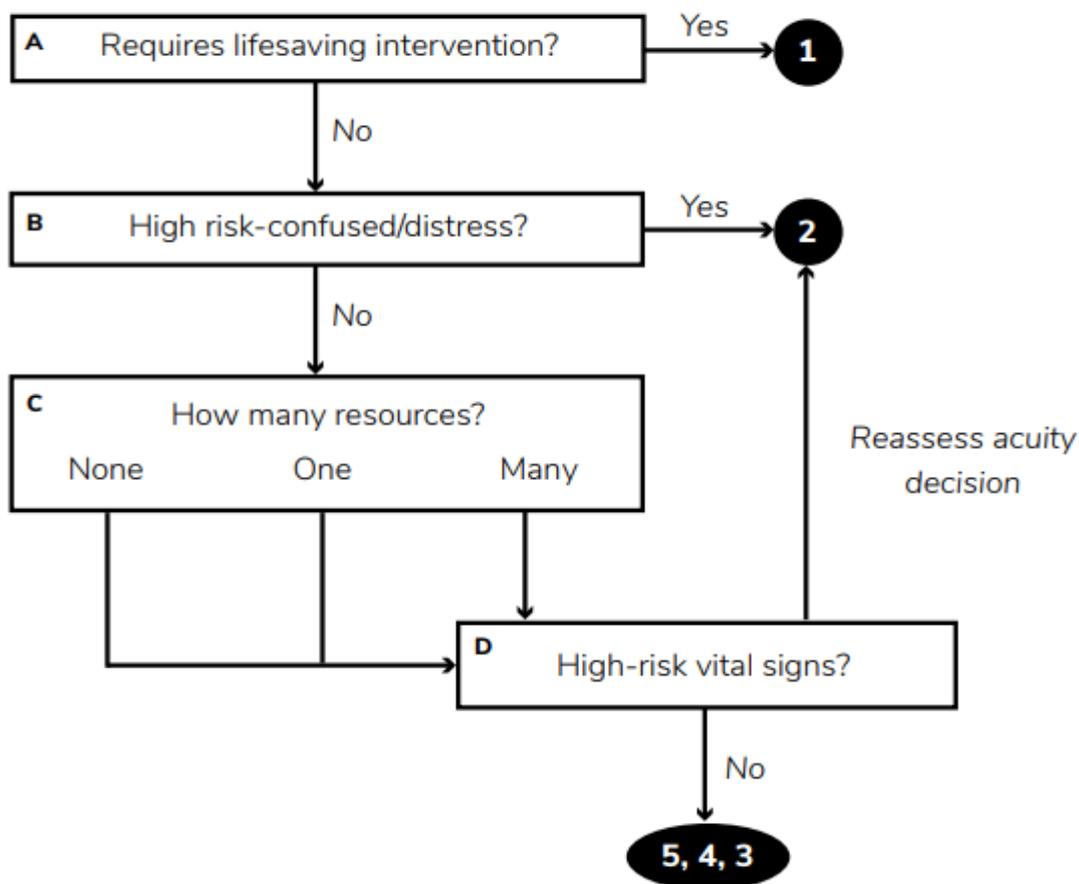


Figura 3. Diagrama de flujo ESI. Fuente: ESI Handbook 5th Edition.

Niveles	Descripción
ESI 1	Condiciones críticas que requieren reanimación inmediata o intervenciones para salvar vidas.
ESI 2	Condiciones deteriorativas de alto riesgo que requieren una evaluación rápida y posible intervención vital.
ESI 3	Condición estable, requiere el uso de múltiples recursos para una evaluación y tratamiento adecuado.
ESI 4	Condición estable, requiere un solo recurso para evaluación y tratamiento.
ESI 5	Condición estable, no requiere uso de ningún recurso para tratamiento, excepto medicaciones o prescripciones.

Tabla 1. Niveles de clasificación del Emergency Severity Index

La Figura 3 representa el algoritmo en el que se basa el procedimiento de asignación de cada nivel del ESI. El personal médico toma una serie de decisiones y consideraciones al evaluar al paciente y ubicarlo en un nivel específico. El nivel ESI 1, responde afirmativamente a la decisión A de si el paciente requiere una intervención de vida o muerte. De forma contrario, se procede con la decisión B de si la condición del paciente tiene riesgo de deterioro mortal y por tanto requiera una atención más inmediata, siendo esta decisión correspondiente al nivel ESI 2. Los dos primeros niveles ESI se basan principalmente en la agudeza de la condición del paciente.

La decisión C y los 3 siguientes niveles (3,4 y 5) se toman en consideración únicamente cuando se determina que el paciente se encuentra fisiológicamente estable con bajo riesgo de deterioro. Una vez fuera de los dos primeros niveles, el personal médico empieza a tomar en consideración los recursos necesarios para evaluar la condición del paciente y en dependencia de eso asignar un nivel ESI. De forma complementaria, se pueden tomar en consideración los signos vitales de cada paciente (decisión D) en buscas de anomalías con el fin de realizar una reconsideración de la agudeza de la condición del paciente.

El ESI es un sistema de triaje efectivo que proporciona un método uniforme para priorizar a los pacientes en los distintos departamentos de urgencias. A su vez, garantiza que los pacientes más críticos reciban atención inmediata y, al mismo tiempo, se asignan los recursos de manera eficiente. Por último, al ser un sistema de triaje, reduce los tiempos de espera de los pacientes no críticos al dirigirlos a los entornos de atención adecuados.

1.3.3 Sistema Español de Triage (SET)

El sistema español de Triage desarrollado por la Sociedad Española de Medicina de Urgencias y Emergencias (SEMES) fue en el 2003 denominado el modelo estándar de triaje en castellano para todo el territorio español (Soler et. al., 2010). El SET es un

modelo estructurado de triaje que consta de 5 niveles de clasificación con un color de identificación, categoría y tiempo de atención respectivo.

Tabla 2. Adaptado de Niveles de clasificación Sistema Español de Triage (Soler et. al., 2010).

Nivel	Color	Categoría	Tiempo de atención
1	Azul	Reanimación	Inmediato
2	Rojo	Emergencia	7 minutos
3	Naranja	Urgente	30 minutos
4	Verde	Menos urgente	45 minutos
5	Negro	No urgente	60 minutos

El SET reconoce una serie de categorías sintomáticas y subcategorías que agrupan una amplia variedad de motivos clínicos de consulta. Estas categorías permiten una evaluación más detallada y precisa de la condición del paciente. El Sistema Español de Triage (SET) clasifica 578 motivos de consulta médica en 32 categorías principales y 14 subcategorías, estableciendo así una estructura detallada para evaluar la urgencia de los pacientes (Soler et. al., 2010). Estas categorías sintomáticas son tomadas en consideración por el profesional de la salud partiendo del motivo de consulta del paciente, permitiendo ubicarlo en una clasificación dentro del SET.

Por otro lado, el SET posee 3 discriminantes principales que permiten realizar una diferenciación entre los grados de urgencia del triaje realizado. Siendo los discriminantes los siguientes:

- Signos vitales: se consideran los signos vitales y la posible anormalidad presente. Entre los signos vitales más comunes se encuentran la frecuencia cardíaca, presión arterial sistólica, temperatura, saturación de oxígeno (SpO2), glucemia capilar, frecuencia y profundidad respiratoria, estado de confusión.

- Dolor: Se evalúan los niveles de dolor manifestados por el paciente por medio de una guía clínica objetiva.

Los discriminantes, las categorías sintomáticas y las escalas de clasificación permiten una aplicación completa y eficiente del triaje (Soler et al., 2010).

1.3.4 Australasian Triage Scale (ATS)

La Escala de triaje de Australasia (ATS) es una herramienta clínica empleada en los departamentos de emergencia australianos para priorizar la atención al paciente en función de la urgencia. Diseñado para garantizar que los pacientes sean tratados en orden de gravedad clínica, el ATS clasifica a los pacientes en cinco niveles. Este sistema ayuda a asignar a los pacientes al área de evaluación y tratamiento más adecuada dentro del departamento. Considera factores como el problema actual del paciente, la apariencia general y los signos potencialmente vitales para determinar el nivel de urgencia. El ATS es fundamental para gestionar el flujo de pacientes y optimizar la asignación de recursos en entornos de emergencia concurridos.

La ATS ubica y clasifica a los pacientes en cinco categorías según su urgencia médica, que van desde afecciones que tienen la capacidad de poner en peligro la vida del paciente inmediatamente y que requieren intervención inmediata hasta problemas menores que pueden abordarse en dos horas. El proceso de clasificación, que normalmente lo completa personal médico capacitado en unos pocos minutos, se centra únicamente en determinar la urgencia de la atención. Es importante señalar que la ATS no mide la gravedad de la enfermedad, la complejidad del tratamiento ni la carga de trabajo del departamento; es exclusivamente para priorizar el tratamiento del paciente en función de la urgencia.

Australasian Triage Scale (ATS)

Triage Category	Description	Maximum Clinically Appropriate Triage Time	Performance Benchmark
1	Immediately life-threatening,	Immediate simultaneous triage and treatment	100%
2	Imminently life-threatening, or important time-critical	10 minutes	80%
3	Potentially life-threatening, potential adverse outcomes from delay > 30 min, or severe discomfort or distress	30 minutes	75%
4	Potentially serious, or potential adverse outcomes from delay > 60 min, or significant complexity or severity, or discomfort or distress	60 minutes	70%
5	Less urgent, or dealing with administrative issues only	120 minutes	70%

Figura 4. Categorías de la Escala de Triage de Australasia (Miller, 2019)

2. Signos vitales

2.1 Definición de signos vitales.

Los signos vitales consisten en mediciones esenciales que le permiten al profesional de la salud realizar una evaluación de cómo se encuentran funcionando la respiración, la circulación y las funciones neurológicas básicas, así como la respuesta del individuo a diversos estímulos catalogados como normales o patológicos. Es fundamental que los médicos supervisen estos indicadores con precisión. Gracias a los avances tecnológicos, es posible detectar fácilmente cualquier alteración en los SV, lo que exige una intervención rápida y adecuada por parte del médico, (Villegas González et al., 2012)

Los SV representan la medición de funciones fisiológicas como la frecuencia y el ritmo cardíaco (FC), la frecuencia respiratoria o cantidad de respiraciones por minuto, la temperatura del cuerpo, la tensión arterial y la oximetría. Estos indicadores muestran que una persona se encuentra con vitalidad y brindan información sobre la calidad del funcionamiento de sus órganos. Los valores normales de los SV pueden variar entre diferentes personas y en la misma persona a lo largo del día, además de ser influenciados por diversos factores. Cualquier desviación de estos valores normales podría señalar un problema en el funcionamiento de los órganos y sugiere la posibilidad de una enfermedad. (Villegas González et al., 2012).

2.2 Tipos de signos vitales.

2.2.1 Temperatura Corporal.

El proceso a través del cual un organismo tiene la capacidad de mantener una temperatura corporal es denominado termorregulación. El hipotálamo actúa como centro de control, activando mecanismos como la vasodilatación, hiperventilación y sudoración para disipar el calor cuando la temperatura del organismo aumenta. En la situación donde

se presentan temperaturas bajas o también llamado hipotermia, se desencadenan respuestas como el aumento del metabolismo y los escalofríos para generar calor, (Penagos & De Urgencias, 2005)

La Asociación Médica Americana establece que la temperatura normal del organismo puede variarte entre los 36,5°C y 37,2°C. Para medir la TC se utilizan termómetros clínicos, que han evolucionado con la introducción de termómetros digitales, reduciendo los riesgos asociados al mercurio. Existen distintos tipos de termómetros, como el axilar, el rectal, los digitales de oído, y los de contacto con la piel. Cada uno tiene sus características específicas y tiempos de lectura que varían desde 60 segundos en termómetros digitales hasta 5 segundos en los de contacto con la piel.

2.2.2 Frecuencia Respiratoria

La frecuencia respiratoria se refiere a la cantidad de respiraciones que una persona hace por minuto. Esta medición generalmente se lleva a cabo cuando la persona está en reposo y no es consciente de que se está midiendo, observando cuántas veces se eleva el tórax en el transcurso de un minuto. (González et al., 2012)

La frecuencia respiratoria puede tender a aumentar en presencia de altas temperaturas corporales o también en presencia de otras condiciones médicas. Además de contar las respiraciones, es importante observar si la persona presenta problemas al realizar cada respiración. En un adulto en reposo, la FR normal tiene a variar entre las 15 y 20 respiraciones por minuto. Una frecuencia respiratoria en reposo que presente valores superiores a 25 las respiraciones por ciclo o que sea inferior a 12 puede contemplarse como anormal (González et al., 2012).

2.2.3 Presión arterial

La presión arterial es la fuerza que ejerce la sangre contra las paredes de las arterias mientras el corazón bombea. Se mide en dos valores: la presión sistólica (cuando el corazón late) y la presión diastólica (cuando el corazón está en reposo entre latidos).

2.2.4 Oximetría

La saturación de oxígeno (SpO₂) se mide a través de un fotodetector que capta la luz transmitida por la piel, correlacionando el radio de absorbanza de la luz con la proporción de hemoglobina saturada y desaturada en el tejido. Una SpO₂ de aproximadamente 85% suele corresponder a una presión arterial de oxígeno (PaO₂) superior a 50 mmHg. Los valores normales de saturación de oxígeno, medidos por oximetría de pulso durante la respiración regular, son entre 97% y 100% en recién nacidos a término al nivel del mar, y entre 95% y 100% en recién nacidos pretérmino, (González et al., 2012).

2.2.5 Pulso arterial.

El pulso es la onda de sangre causada por la contracción del ventrículo izquierdo del corazón, que provoca la expansión de las arterias. Refleja el latido cardíaco, la capacidad arterial y el funcionamiento de la válvula aórtica. Se palpa en áreas donde las arterias están cerca de huesos, como muñecas y cuello. Su velocidad, en latidos por minuto, suele coincidir con la frecuencia cardíaca (González et al., 2012).

En un adulto sano en reposo, la frecuencia cardíaca suele variar entre sesenta y cien latidos por minuto, ya que en este estado el corazón requiere bombear menos sangre. Sin embargo, este rango cambia con la edad: al nacer, la frecuencia cardíaca es alta debido a la intensa actividad del organismo, pero disminuye a partir del primer mes de vida y se estabiliza después de la infancia.

2.3. Relación entre los signos vitales y el nivel de urgencia.

Los signos vitales junto a la sintomatología constituyen el primer paso para un diagnóstico preventivo frente a una emergencia o urgencia, esta información permite al profesional de la salud realizar un diagnóstico rápido con el fin de conocer la situación del paciente. Una vez valorada la situación clínica del paciente, se establecerá un tiempo estimado de espera para su atención, teniendo en cuenta la necesidad de priorizar a los pacientes con mayor gravedad,

3. Escala de Glasgow

La Escala de Coma de Glasgow (GCS), fue desarrollada por Graham Teasdale y Bryan Jennett en 1974, es un instrumento clave en la valoración del nivel de conciencia en pacientes que presenten daño cerebral, como los traumatismos craneoencefálicos. Compuesta por tres subescalas que miden la respuesta verbal, la apertura de los ojos y la motora, la GCS asigna un puntaje que refleja la mejor respuesta en cada área. Esta escala es valorada por su facilidad de uso y capacidad para estandarizar la evaluación neurológica, lo que facilita la comunicación entre el personal médico. No obstante, su uso puede presentar inconsistencias, lo que resalta la necesidad de una formación continua para asegurar evaluaciones precisas.

La GCS es una herramienta de amplio reconocimiento a nivel internacional, teniendo como objetivo evaluar y medir el nivel de conciencia de un individuo. Evalúa dos aspectos clave: el estado de alerta, que implica la conciencia del entorno, y el estado cognitivo, que refleja la capacidad del paciente para obedecer órdenes. Clínicamente, la escala tiene tres objetivos principales: discriminación (evaluar la profundidad de la alteración de la conciencia y la gravedad del daño cerebral), evaluación,

y predicción (pronosticar la evolución del paciente según su nivel de conciencia en el momento de la evaluación).

El nivel de conciencia no es observable directamente, por lo que la evaluación clínica depende de la observación y las inferencias sobre el estado subyacente del paciente. La GCS es valiosa porque proporciona un marco común que reduce la variabilidad entre evaluadores. Diversos estudios han respaldado su uso, demostrando su validez y confiabilidad, incluso entre enfermeras con diferentes niveles de experiencia. Además, se ha validado como un índice de severidad para lesiones cerebrales en numerosos estudios, mostrando una alta correlación entre la puntuación de la GCS y la mortalidad.

ABERTURA OCULAR	Espontanea (4)	<input type="checkbox"/>	<input type="checkbox"/>
	Al estímulo verbal (3)	<input type="checkbox"/>	<input type="checkbox"/>
	Al estímulo doloroso (2)	<input type="checkbox"/>	<input type="checkbox"/>
	Ninguna (1)	<input type="checkbox"/>	<input type="checkbox"/>
RESPUESTA VERBAL	Orientada (5)	<input type="checkbox"/>	<input type="checkbox"/>
	Confusa (4)	<input type="checkbox"/>	<input type="checkbox"/>
	Palabras inadecuadas (3)	<input type="checkbox"/>	<input type="checkbox"/>
	Sonidos Incomprensibles (2)	<input type="checkbox"/>	<input type="checkbox"/>
	Ninguna (1)	<input type="checkbox"/>	<input type="checkbox"/>
RESPUESTA MOTORA	Obedece órdenes (6)	<input type="checkbox"/>	<input type="checkbox"/>
	Localiza el dolor (5)	<input type="checkbox"/>	<input type="checkbox"/>
	Movimiento de retirada (4)	<input type="checkbox"/>	<input type="checkbox"/>
	Flexión hipertónica (3)	<input type="checkbox"/>	<input type="checkbox"/>
	Extensión hipertónica (2)	<input type="checkbox"/>	<input type="checkbox"/>
	Ninguna (1)	<input type="checkbox"/>	<input type="checkbox"/>

Figura 5. Escala de Glasgow. Fuente: Elaboración propia

4. Procesos actuales en la adquisición de signos vitales.

La medición de signos vitales es fundamental en la atención médica, proporcionando información esencial sobre la salud de los pacientes. Los signos vitales más comunes son la frecuencia respiratoria, la frecuencia cardíaca, la presión arterial y la temperatura corporal. Con los avances tecnológicos, los métodos y dispositivos para medir estos signos han evolucionado, mejorando la precisión, eficiencia y accesibilidad de los datos obtenidos.

4.1 Métodos Tradicionales de Toma de Signos Vitales

1. Temperatura Corporal

- Termómetros de Mercurio: Fueron utilizados en grandes cantidades, pero han caído en desuso por preocupaciones ambientales y de seguridad.
- Termómetros Digitales: Utilizan sensores electrónicos para medir la temperatura, proporcionando resultados rápidos y precisos.

2. Presión Arterial

- Esfigmomanómetros de Mercurio: Considerados el estándar de oro durante muchos años, aunque su uso ha disminuido por razones ambientales.

- Esfigmomanómetros Aneroides: Utilizan un dial y una aguja para indicar la presión, y son portátiles y relativamente económicos.

3. Frecuencia Cardíaca

- Palpación Manual: Contar la cantidad de palpaciones cardiacas mediante la palpación de arterias, como la radial o la carótida, durante un tiempo específico.
- Estetoscopio: Utilizado en combinación con la palpación para una evaluación más detallada.

4. Frecuencia Respiratoria

- Observación Visual: Contar la cantidad respiraciones por minuto, tomando en consideración el movimiento del pecho.

4.2 Avances Tecnológicos en la adquisición de Signos Vitales

1. Dispositivos Electrónicos y Digitales:

- Termómetros Infrarrojos: Permiten la medición de la temperatura sin contacto, ideales para situaciones donde la higiene es una preocupación.
- Monitores Electrónicos de Presión Arterial: Automáticos y fáciles de usar, ofrecen lecturas precisas y pueden almacenar múltiples lecturas para seguimiento.

2. Wearables (Usables):

- Relojes Inteligentes y Monitores de Fitness: Capaces de medir la frecuencia cardíaca, el ritmo cardíaco, la variabilidad de la frecuencia cardíaca y, en algunos casos, la oxigenación de la sangre.
- Parche de Frecuencia Respiratoria: Dispositivos adhesivos que monitorean continuamente la frecuencia respiratoria y otros signos vitales.

3. Telemedicina y Dispositivos Conectados:

- Monitores Remotos: Dispositivos que transmiten datos de signos vitales a profesionales de la salud en tiempo real, facilitando el monitoreo a distancia y la gestión de pacientes con enfermedades crónicas.

4.3 Impacto de la Tecnología en la Atención Médica

- Precisión y Confiabilidad: Los dispositivos electrónicos y digitales han mejorado la precisión y la confiabilidad de la adquisición de signos vitales.
- Accesibilidad y Conveniencia: Los wearables (usables) y aplicaciones móviles permiten a los pacientes monitorear sus signos vitales de manera continua y en cualquier lugar.
- Telemedicina: La capacidad de realizar una monitorización de signos vitales a distancia ha mejorado la gestión de enfermedades crónicas y ha permitido una respuesta más rápida a las emergencias médicas.

Los avances tecnológicos han transformado significativamente los procesos de toma de signos vitales, haciéndolos más precisos, accesibles y eficientes. La adopción de dispositivos digitales y la integración de la telemedicina han facilitado una mejor atención de los pacientes, mejorando los resultados de salud.

5. Descripción del Proceso de Triage en Centros de Salud de Primer Nivel en Ecuador

El triaje es un proceso fundamental en la atención médica que permite priorizar a los pacientes en función de la urgencia de sus condiciones. En los centros de salud de primer nivel en Ecuador, el triaje es crucial para gestionar de manera eficiente los recursos limitados y asegurar que aquellos con condiciones más críticas reciban atención inmediata. Este análisis examina cómo se lleva a cabo el triaje en estos centros, incluyendo sus protocolos, desafíos y las mejores prácticas.

5.1 Proceso de Triage

1. Recepción del Paciente:

- **Descripción:** Al llegar al centro de salud, los pacientes son recibidos por el personal de recepción, quien registra sus datos y el motivo de su visita.
- **Importancia:** Este primer paso es esencial para identificar rápidamente a aquellos pacientes que pueden necesitar atención urgente.

2. Evaluación Inicial:

- **Descripción:** Un profesional de salud, como una enfermera o un médico, realiza una evaluación rápida para determinar la gravedad del estado del paciente.
- **Métodos Utilizados:** La evaluación incluye la medición de signos vitales y la observación de síntomas visibles.

- **Clasificación:** Los pacientes son clasificados en categorías de urgencia (urgente, semiurgente, no urgente) según la evaluación inicial.

3. Clasificación de Prioridad:

- **Categorías de Triage:**
 - **Urgente:** Pacientes con condiciones que ponen en riesgo inmediato su vida y requieren atención de inmediato.
 - **Semiurgente:** Pacientes cuya condición podría deteriorarse si no se atienden pronto, pero no es una amenaza inmediata.
 - **No Urgente:** Pacientes con condiciones estables que pueden esperar sin riesgo significativo.
- **Protocolos de Triage:** Se emplean sistemas estandarizados, como el Sistema de Triage de Manchester, adaptados a las necesidades locales.

4. Atención Según Prioridad:

- **Descripción:** Los pacientes son atendidos de acuerdo con su clasificación de urgencia. Los casos urgentes reciben atención inmediata, mientras que los semiurgentes y no urgentes esperan su turno.
- **Recursos:** Los centros asignan recursos y personal según la prioridad de los pacientes para maximizar la eficiencia.

5.2 Desafíos del Triage en Ecuador

1. Recursos Limitados:

- **Problema:** La falta de personal capacitado y equipos médicos puede dificultar la implementación efectiva del triaje.
- **Soluciones:** Mejorar la formación del personal y asegurar un suministro adecuado de equipos básicos.

2. **Capacitación del Personal:**

- **Problema:** Las diferencias en la capacitación y experiencia del personal pueden afectar la consistencia y precisión del triaje.
- **Soluciones:** Establecer programas de capacitación continua y estandarizar los protocolos de triaje.

3. **Infraestructura:**

- **Problema:** Algunas instalaciones pueden no tener la infraestructura adecuada para realizar el triaje eficazmente.
- **Soluciones:** Invertir en mejorar la infraestructura y asegurar que los centros cuenten con áreas designadas para el triaje.

5.3 Mejoras Prácticas

● **Capacitación Continua:**

- **Descripción:** Proporcionar formación regular y actualizada al personal en técnicas de triaje y manejo de emergencias.
- **Beneficios:** Mejora la precisión del triaje y asegura una respuesta rápida y adecuada a emergencias.

● **Uso de Protocolos Estandarizados:**

- **Descripción:** Implementar y seguir protocolos estandarizados de triaje adaptados a las necesidades locales.
- **Beneficios:** Garantiza un enfoque consistente y basado en evidencia para la clasificación de pacientes.

● **Mejora de Infraestructura:**

- **Descripción:** Invertir en la mejora de las instalaciones y el equipamiento necesario para realizar el triaje de manera eficaz.

- **Beneficios:** Facilita el proceso de triaje y mejora la experiencia del paciente.

El triaje en los centros de salud de primer nivel en Ecuador es esencial para gestionar el flujo de pacientes y priorizar la atención según la gravedad de sus condiciones. A pesar de los desafíos, la implementación de mejores prácticas y la inversión en recursos y capacitación pueden mejorar significativamente la eficiencia y efectividad del proceso de triaje.

6. Sistemas Automáticos y Dispositivos Médicos para la Adquisición de Signos Vitales

La integración de tecnología avanzada en la atención médica ha llevado a un cambio en la forma en que los signos vitales son adquiridos. En Ecuador, el uso de sistemas automáticos y dispositivos médicos ha transformado significativamente el monitoreo de la salud, permitiendo mediciones más precisas y accesibles. Este estudio examina los dispositivos más destacados y cómo han impactado la atención sanitaria en el país.

6.1 Principales Sistemas Automáticos y Dispositivos Médicos

- **Monitores Multifuncionales de Signos Vitales**
 - **Descripción:** Estos equipos combinan múltiples funciones en un solo dispositivo, como medir la saturación de oxígeno, la presión arterial, la frecuencia cardíaca y la temperatura. Son ampliamente usados en clínicas y hospitales para un monitoreo constante y detallado.
 - **Impacto:** Ofrecen una recolección de datos más eficiente, eliminando la necesidad de múltiples dispositivos y proporcionando una visión completa del estado de salud del paciente en tiempo real.
- **Sensores de Frecuencia Cardíaca y Oxímetros de Pulso**
 - **Descripción:** Estos dispositivos miden la frecuencia cardíaca y la saturación de oxígeno en sangre. Los sensores pueden ser portátiles y se usan en diversos contextos, desde hospitales hasta el hogar.

- **Impacto:** Proporcionan mediciones continuas y precisas de la frecuencia cardíaca y el nivel de oxígeno, facilitando el seguimiento de pacientes con problemas cardiovasculares y respiratorios.
- **Equipos Automatizados para la toma de Presión Arterial**
 - **Descripción:** Estos aparatos permiten medir la presión arterial automáticamente mediante un manguito y un sensor, proporcionando lecturas rápidas y precisas. A menudo, incluyen funciones para almacenar y analizar datos a lo largo del tiempo.
 - **Impacto:** Facilitan el control regular de la presión arterial, lo cual es crucial para manejar la hipertensión y otras condiciones relacionadas, ayudando a prevenir complicaciones y mantener un control riguroso.
- **Termómetros Digitales e Infrarrojos**
 - **Descripción:** Los termómetros digitales e infrarrojos miden la temperatura corporal con alta precisión. Mientras que los digitales requieren contacto con la piel, los infrarrojos permiten la medición sin contacto.
 - **Impacto:** Permiten una medición rápida y precisa de la temperatura, esencial para identificar fiebre y otras condiciones febriles. Los termómetros infrarrojos son especialmente útiles en entornos con alto volumen de pacientes, como hospitales.
- **Dispositivos de Monitoreo Remoto**
 - **Descripción:** Estos dispositivos permiten la monitorización continua de los signos vitales desde una distancia, utilizando tecnologías como el Internet de las cosas (IoT) para transmitir datos en tiempo real a los profesionales de salud.

- **Impacto:** Facilitan el seguimiento de pacientes desde sus hogares, especialmente aquellos con enfermedades crónicas, lo que permite una intervención temprana y reduce la necesidad de visitas frecuentes a centros médicos.
- **Monitores de Actividad Física**
 - **Descripción:** Los monitores de actividad física, también conocidos como rastreadores de fitness, miden la actividad diaria, el sueño y la frecuencia cardíaca. Son útiles para promover la salud y monitorear el bienestar general.
 - **Impacto:** Proporcionan datos valiosos sobre la actividad física y el estado general de salud, ayudando a los usuarios a adoptar comportamientos más saludables y a manejar condiciones como la obesidad y la diabetes.

6.2 Impacto General en la Atención Sanitaria

- **Mejora en Precisión y Eficiencia**
 - **Descripción:** Los sistemas automáticos y los dispositivos modernos permiten una medición más exacta y eficiente de los signos vitales, reduciendo errores humanos y mejorando la calidad de los datos obtenidos.
 - **Beneficios:** Facilitan una evaluación más precisa del estado de salud del paciente, optimizan la toma de decisiones clínicas y mejoran el flujo de trabajo en los centros de salud.
- **Accesibilidad y Monitoreo Continuo**
 - **Descripción:** Los dispositivos portátiles y de monitoreo remoto han incrementado el acceso a la atención médica, permitiendo a los pacientes

realizar un seguimiento de su salud desde el hogar y reduciendo la necesidad de visitas físicas frecuentes.

- **Beneficios:** Aumentan la accesibilidad a la atención médica, particularmente para aquellos en zonas remotas o con movilidad reducida, y permiten un monitoreo continuo que puede llevar a intervenciones más oportunas.

- **Eficiencia en la Gestión de Pacientes**

- **Descripción:** Los dispositivos automáticos y de monitoreo remoto han mejorado la gestión de pacientes al proporcionar datos en tiempo real y permitir respuestas más rápidas a cambios en los signos vitales.
- **Beneficios:** Incrementan la eficiencia en la atención médica, permiten una mejor gestión de enfermedades crónicas y reducen el riesgo de complicaciones mediante una monitorización proactiva.

Los avances en sistemas automáticos y dispositivos médicos han revolucionado la manera en que se toman los signos vitales, mejorando la precisión, la eficiencia y la accesibilidad en la atención médica. Estas tecnologías están transformando el cuidado de los pacientes y prometen continuar influyendo positivamente en la práctica médica en Ecuador y más allá.

7. Microcontroladores Arduino.

7.1 Origen

Arduino surgió en 2005 en Ivrea, Italia, como respuesta a la urgencia de contar con un dispositivo asequible para su implementación en las aulas. En un inicio, el objetivo era desarrollar una placa que pueda ser utilizada en el instituto educativo. Sin embargo, el instituto tuvo que cerrar, lo que puso en riesgo el proyecto Arduino. Para evitar su desaparición, se decidió hacer el proyecto accesible al público, permitiendo que cualquiera pudiera ayudar en la evolución del proyecto.

En la actualidad, Arduino es una plataforma de desarrollo de hardware libre con una placa de circuito impreso (PCB) que integra un microcontrolador reprogramable y pines para conectar sensores y actuadores fácilmente. Existen varios modelos de Arduino, todos compatibles en software, que usan un entorno de programación basado en C++. Es versátil y reutilizable, permitiendo desmontar componentes y empezar nuevos proyectos con facilidad.

7.2 Tipos de Arduino.

7.2.1 Arduino UNO

El Arduino Uno Rev3 es la placa más robusta y cuenta con la mayor cantidad de documentación disponible, lo que la convierte en la tarjeta más utilizada y accesible para principiantes (Figura 5). El microcontrolador ATmega328P funciona a un voltaje de 5V y tiene un rango de voltaje de entrada recomendado de 7-12V, con un límite máximo de

20V (Arduino, n.d.). Ofrece 14 pines de entrada/salidas digitales, de los cuales 6 están habilitados para salida PWM, y 6 pines de entrada analógica. En términos de memoria, dispone de 32 KB de memoria Flash, de los cuales 0.5 KB están reservados para el bootloader, 2 KB de SRAM y 1 KB de EEPROM. Su velocidad de reloj es de 16 MHz (Arduino, n.d.). El microcontrolador mide 68.6 mm de longitud y 53.4 mm de anchura, con un peso de 25 g, (Arduino Uno Rev3 — Arduino Official Store, 2024)



Figura 6. Arduino UNO. Fuente: <https://store.arduino.cc/products/arduino-uno-rev3>

7.2.2 Arduino Mega.

El Arduino Mega 2560 es una placa de desarrollo basada en el microcontrolador ATmega2560, diseñada para proyectos que requieren una mayor capacidad de procesamiento y una mayor cantidad de pines de entrada/salida en comparación con otros modelos de Arduino, como el Uno. Opera a un voltaje de 5V y acepta un voltaje de entrada recomendado de 7-12V (Arduino Mega 2560 Rev 3, Arduino Official Store, 2024). La placa cuenta con 54 pines de entrada/salidas digitales, de los cuales 15 pueden ser utilizados como salidas PWM, y 16 pines de entrada analógica (Arduino Mega 2560 Rev 3, Arduino Official Store, 2024). Su memoria incluye 256 KB de memoria Flash, de

los cuales 8 KB están destinados al bootloader, 8 KB de SRAM y 4 KB de EEPROM. Con una velocidad de reloj de 16 MHz, el Arduino Mega tiene un consumo típico de alrededor de 70-100 mA, aunque esto puede variar según el uso (Arduino Mega 2560 Rev 3, Arduino Official Store, 2024).

MARCO METODOLÓGICO

1. Microcontrolador y sensores.

1.1 Arduino Mega

El primer paso del proyecto fue la elección del microcontrolador encargado de realizar el procesamiento de la información enviada por los sensores de signos vitales, así mismo, el microcontrolador sería el encargado de poseer el conjunto de datos adecuado para el procesamiento de los signos vitales y la realización del triaje.

El microcontrolador escogido fue el conjunto de microcontroladores Arduino, esto debido a su lenguaje de programación sencillo y amigable con el usuario, además de poseer en su mayoría sensores de signos vitales junto a librerías fáciles de implementar en su interfaz de programación Arduino IDE. En específico se hizo uso del Arduino Mega debido a su gran capacidad de procesamiento (256 kb de Flash Memory), elevada cantidad de pines digitales y analógicos y su capacidad de comunicación serial con hasta 3 dispositivos. Se hizo uso de 2 Arduino Mega para evitar la sobrecarga del equipo y un mejor flujo de envío y recepción de información.

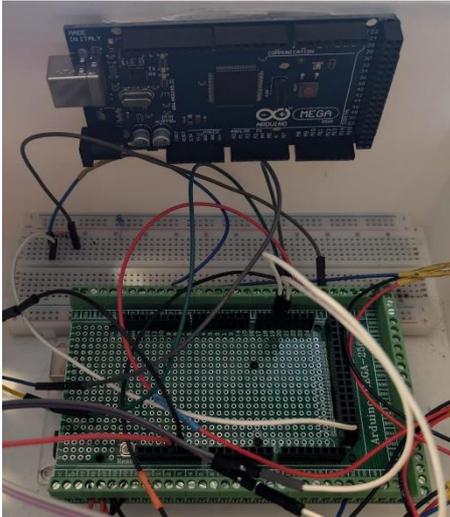


Figura 7. Uso de 2 Arduino Mega

1.2 Sensores para la adquisición de signos vitales.

Para los signos vitales se utilizaron sensores ya establecidos en el mercado y con librerías elaboradas, facilitando el proceso de programación del equipo. Los sensores utilizados fueron: KY-037, MAX-30102, DS18B20 y monitor automático de presión arterial de muñeca ZUEN.

1.2.1 Sensor KY-037.

El sensor KY-037 es un sensor de sonido digital/analógico que permite la recepción de señales sónicas. El voltaje manejado por el sensor es de 5V, posee dos pines de alimentación VCC y GND y un pin de salida OUT por el cuál la señal detectada será enviada al Arduino. La forma utilizada será la digital, de forma que el micrófono realice lecturas durante las respiraciones de cada persona.

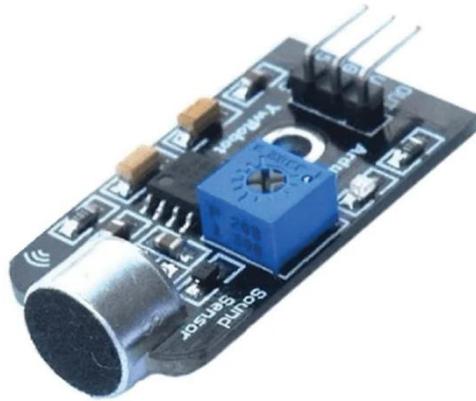


Figura 8. Sensor de sonido. Fuente: <https://roboticsec.com/producto/sensor-de-sonido-microfono-grande-version-mejorada-alta-sensibilidad/>

1.2.2 Sensor digital de temperatura DS18B20A10.

El sensor digital DS18B20 permite la obtención de valores de temperatura, utiliza el protocolo de comunicación 1-Wire que permite la conexión y transmisión de datos por medio de una sola línea. Maneja un voltaje de trabajo entre 2.5 y 5 Voltios, sus rangos de medición abarcan entre los -55 y 125 grados Celsius, utiliza 3 terminales de conexión: VDD, GND y la salida digital de información. Así mismo, maneja una precisión de ± 0.4 °C entre los rangos -10°C a $+70^{\circ}\text{C}$.



Figura 9. Sensor DS18B20. Fuente: <https://roboticsec.com/producto/sensor-de-temperatura-sumergible-ds18b20/>

1.2.3 Sensor MAX-30102.

El sensor MAX-30102 es un sensor de alta sensibilidad para la obtención de valores de oximetría y frecuencia cardíaca. Cuenta con 4 pines de conexión, 2 de alimentación VIN y GND y 2 de transmisión de información SDA y SCL. Utiliza un LED rojo, un LED infrarrojo y un fotodiodo para captar las señales de reflectancia que permiten estimar la saturación de oxígeno, además del ritmo cardíaco.

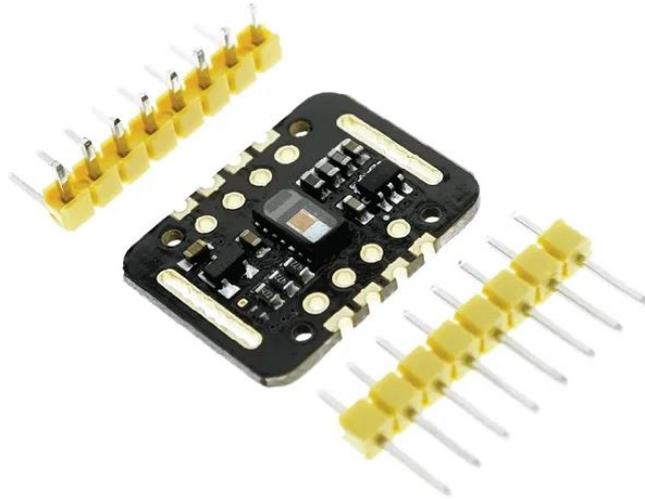


Figura 10. Sensor MAX30102. Fuente:
<https://roboticsec.com/producto/sensor-de-frecuencia-cardiaca-oximetro-max30102-negro/>

1.2.4 Monitor automático de presión arterial de muñeca.

Este equipo permite la medición de presión arterial en la muñeca, maneja un voltaje de entrada de 3.3V y posee dos pines de comunicación SDA y SCL.



Figura 11. Tensiómetro digital de muñeca

2. Pantalla Nextion.

Una vez escogidos el microcontrolador y los sensores por utilizar, se procedió a decidir de qué forma se desarrollaría una interfaz intuitiva para nuestro sistema y se decidió utilizar una pantalla Nextion de 7" por su tamaño y su manejo de interfaz táctil.



Figura 12. PANTALLA NEXTION 7" NX8048T070 HMI SERIAL. Fuente:

<https://grupoelectrostorec.com/shop/displays-y-pantallas/pantalla-nextion-7-nx8048t070-serial/>

Esta pantalla cuenta con una resolución de 800x400 píxeles que permite una representación de información e imágenes más clara y detallada, así mismo, posee una memoria flash de 16MB, 3584 bytes de RAM y un procesador integrado que alivia la carga de trabajo del microcontrolador al manejar los elementos visuales y eventos táctiles. Por último, maneja un sistema de comunicación UART (Nextion, n.d.).

3. Selección de la escala de gravedad basada en signos vitales.

El método de triaje más utilizado en Ecuador es el sistema Manchester, el cual permite clasificar a cada paciente en distintos niveles de prioridad, utilizando como base la gravedad de sus síntomas y signos vitales. Sin embargo, este enfoque pone un fuerte énfasis en la sintomatología del paciente, lo que lo aleja del objetivo de la presente tesis. En el proyecto propuesto, se busca que el triaje pueda ser determinado exclusivamente mediante signos vitales, sin depender de la interpretación subjetiva de los síntomas.

Con el fin de lograr este objetivo, se realizó una investigación exhaustiva en busca de un enfoque alternativo que permita una clasificación precisa basada únicamente en parámetros medibles como la temperatura, presión arterial, frecuencia cardíaca, saturación de oxígeno y frecuencia respiratoria. Durante este proceso, se identificó una tabla de triaje que se ajusta perfectamente a esta necesidad, permitiendo una clasificación eficaz del paciente basada únicamente en los valores de sus signos vitales, sin necesidad de considerar la sintomatología. Así mismo, esta tabla de triaje muestra cierto parecido a los parámetros evaluados en el triaje manchester.

Además, la tabla incorpora la aplicación de la escala de Glasgow, lo cual consideramos de gran relevancia, ya que permite medir de manera rápida y efectiva el estado de conciencia del paciente. Esta inclusión es fundamental para complementar el análisis de los signos vitales, ofreciendo una visión más completa de la condición del paciente sin desviarse del objetivo de basar el triaje en parámetros objetivos y medibles.

La tabla encontrada proviene de un proyecto de tesis enfocado en el triaje y ha sido valorada por personal de salud, lo que garantiza su validez y aplicabilidad en contextos clínicos (Cervantes, Reyes y Brack, 2017). Esta validación le otorga un

respaldo adicional y refuerza la utilidad de utilizarla como base en la presente tesis, donde se busca una clasificación precisa de pacientes mediante signos vitales.

Tabla 3. Clasificación de triaje basado en signos vitales y escala de Glasgow

Clasificación de Rangos

<i>Signo Vital</i>	Nivel I	Nivel II	Nivel III	Nivel IV	Nivel V
<i>Presión arterial sistólica</i>	0-49 mmHg	50-69 mmHg >170 mmHg	70-89 mmHg 150-169 mmHg	130-149 mmHg	90-129 mmHg
<i>Presión arterial diastólica</i>	0-40 mmHg	41-49 mmHg >120 mmHg	50-59 mmHg 110-120 mmHg	90-109 mmHg	60-89 mmHg
<i>Frecuencia Respiratoria</i>	0-6/min	7-10 /min	11-13/min	21-24/min	14-20/min
<i>Frecuencia cardiaca</i>	>41/min	31-40/min	25-30/min		
<i>Temperatura</i>	0-40/min	41-54/min	55-59/min	90-99/min	60-89/min
<i>Oximetría</i>	>180/min	140-179/min	100-139/min		
<i>Escala de Glasgow</i>	0-29°C	30-33.9°C >41°C	34-35.9°C 39-40.9°C	37.6- 38.9°C	36-37.5°C
		<90%	90%-94%	94%-96%	>96%
	3-8	9-12	13-14	15	15

Fuente: Adaptado de Cervantes, Reyes y Brack, 2017. Propuesta de tabla de triaje basada en signos vitales.

Finalmente, se han agregado valores de oximetría tentativos que han sido conversados con un personal médico.

4. Diseño del Sistema.

4.1 Adaptación física de los sensores.

Para un mejor manejo de los sensores y una mejor respuesta a las señales medidas, se consideró necesario realizar adaptaciones físicas en ciertos sensores: MAX-30102 y KY-037. Además, se soldaron conectores metálicos a cada uno de los sensores a utilizar.

El sensor KY-037 fue ubicado en un micrófono de diadema para que al momento de tomar la frecuencia respiratoria, pueda ubicarse fácilmente por debajo de la nariz del paciente.

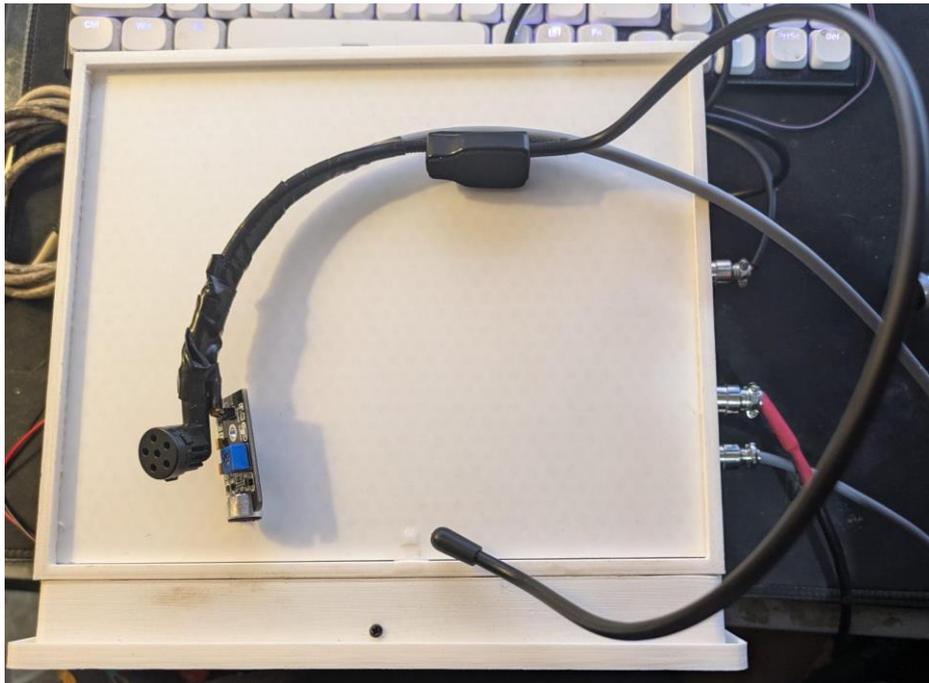


Figura 13. Adaptación sensor KY-037

Por otro lado, el sensor de oximetría MAX-30102 fue ubicado en un armazón impreso en material PLA para que al ser ubicado en el dedo del paciente pueda mantenerse firme y seguro.

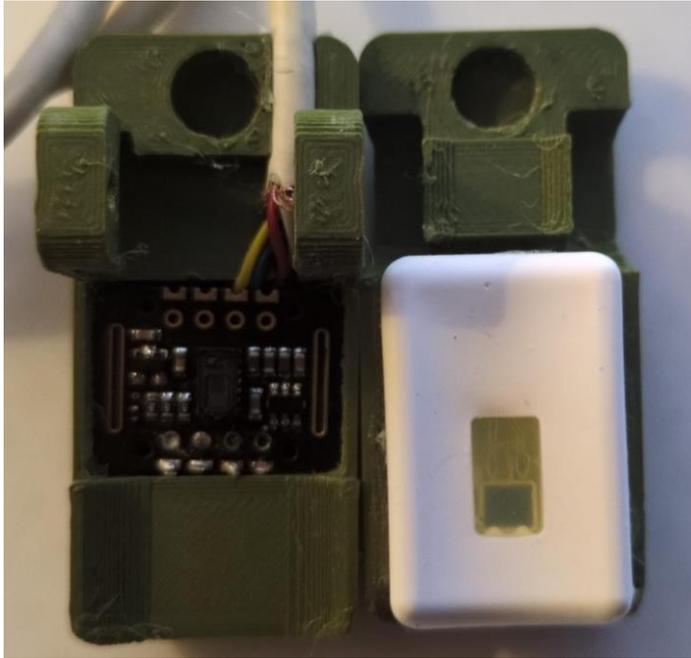


Figura 14. Adaptación física de sensor MAX-30102

4.2 Conexión de sensores.

Una vez establecido el microcontrolador y los sensores por utilizar, se procedió a realizar la conexión de los sensores al microcontrolador.

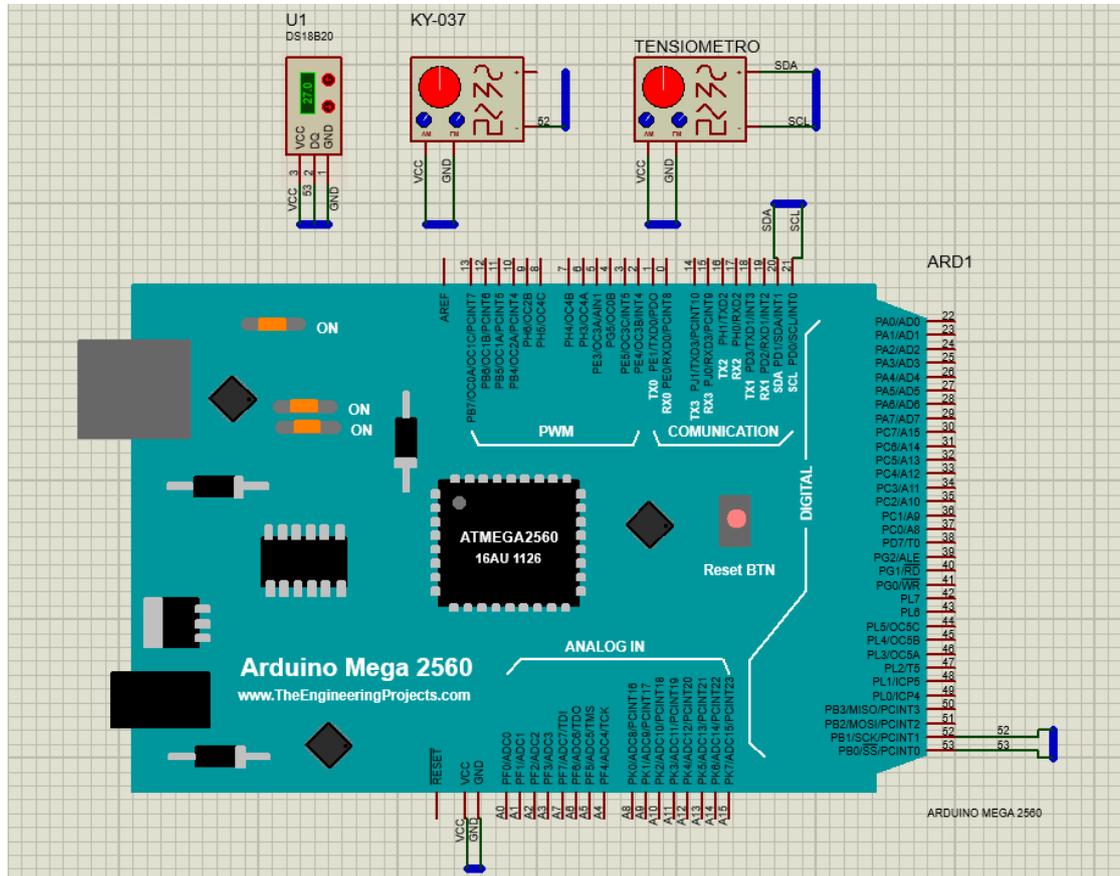


Figura 15. Conexión de sensores en Proteus 8 Professional.

Como se puede observar en la Figura 15(Ordóñez & Yagual, 2024), los sensores KY-037, DS18B20 y Tensiómetro fueron conectados al Arduino mega principal. El sensor KY-037 correspondiente a la toma de frecuencia respiratoria fue conectado al pin digital 52 y alimentado a 5V por medio de los pines VCC y GND. El sensor de temperatura DS18B20 fue conectado al pin digital 53 y alimentado a 5V por medio de los pines VCC y GND. Finalmente, el tensiómetro fue conectado a los puertos SDA y SCL encargados de la comunicación i2c. Ya en la parte física, fue requerido ubicar una resistencia pull-up en la conexión de datos del sensor de temperatura DS1820, la resistencia utilizada fue de 4.7kΩ.

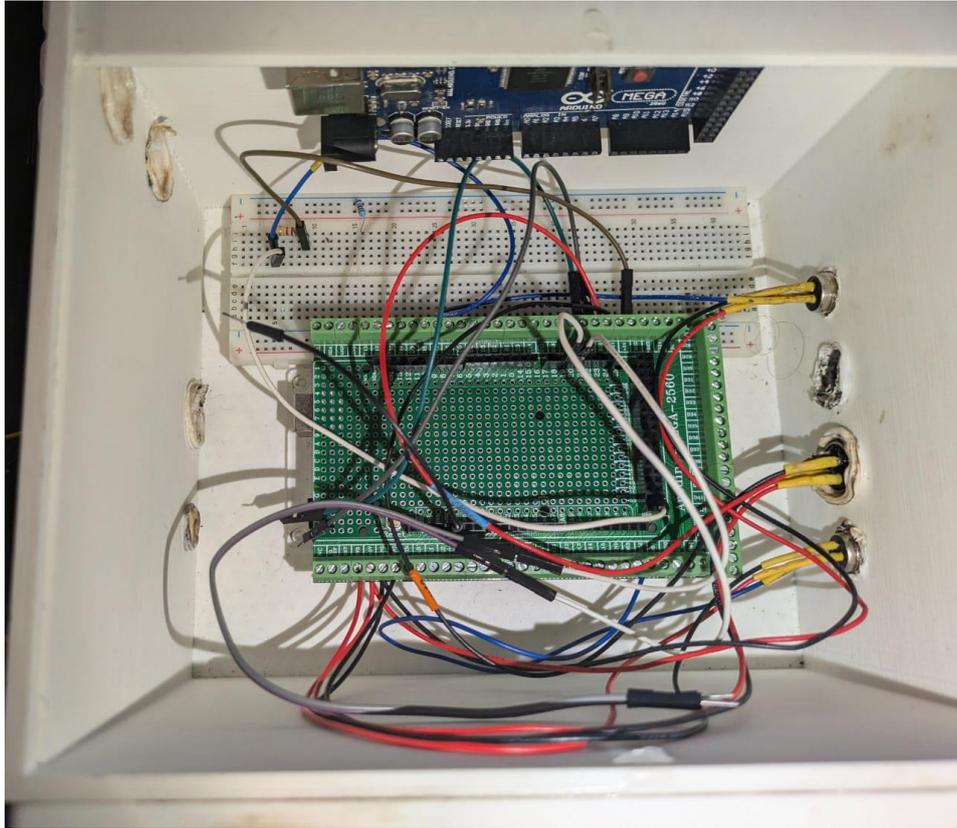


Figura 16. Conexión física interior de los sensores de signos vitales

Así mismo, los cables de cada sensor fueron soldados a conectores metálicos de pines como se puede observar en la Figura 16 (Ordóñez & Yagual, 2024).



Figura 17. Conexión física exterior de los sensores de signos vitales

El segundo Arduino Mega se encargará de procesar el sensor MAX-30102 y enviar la información al Arduino Mega principal. Debido a esto, el segundo Arduino Mega se conectó a través de sus puertos RX1 y TX1 a los puertos RX2 y TX2 del otro Arduino.

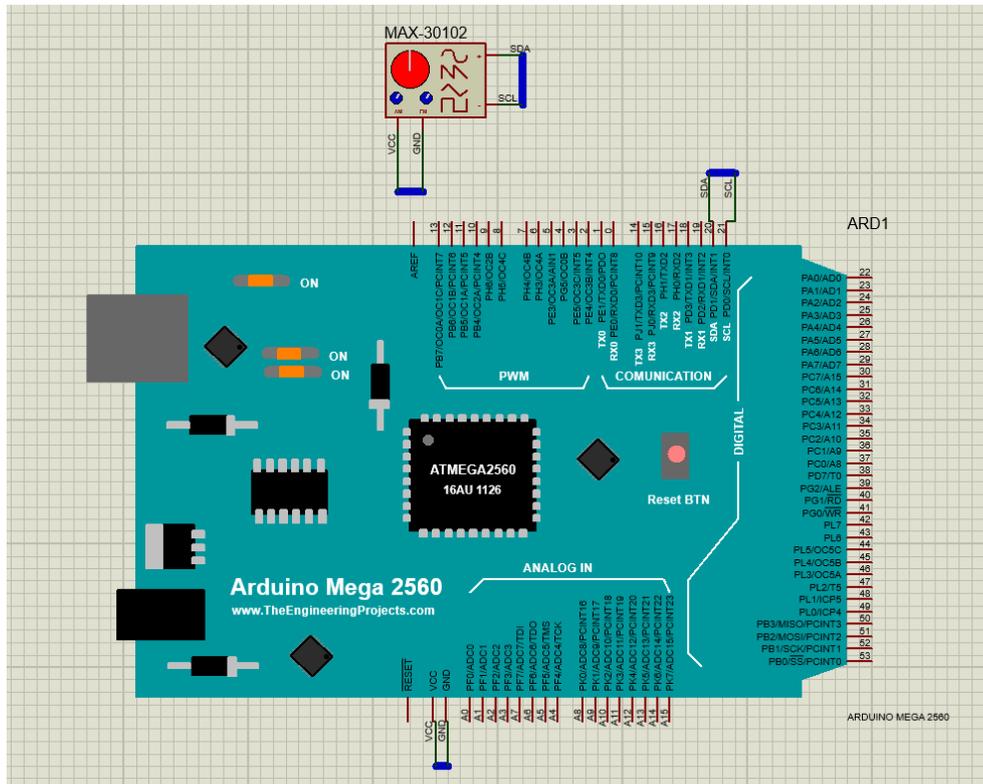


Figura 18. Conexión de sensor MAX-30102 en Proteus 8 Professional

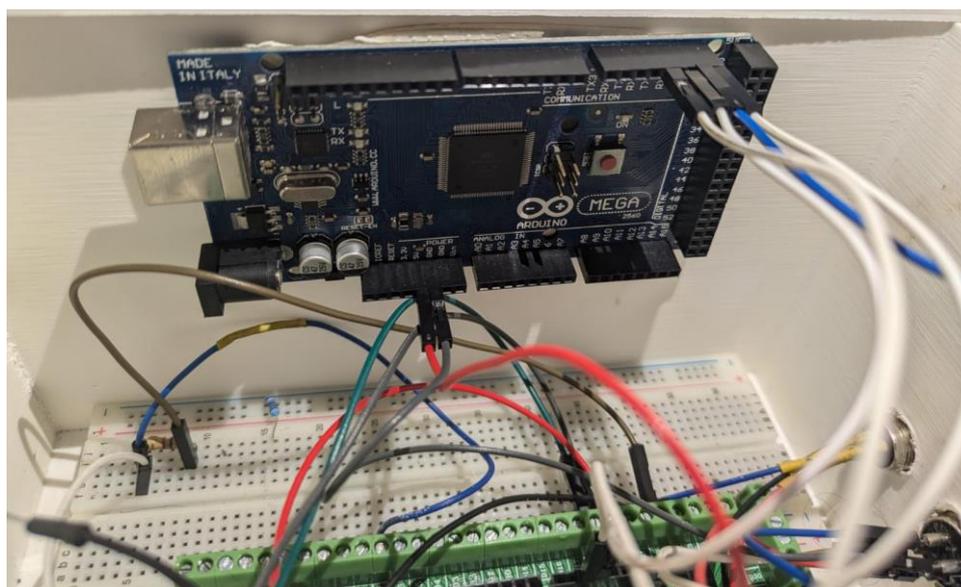


Figura 19. Conexión física interior del sensor MAX-30102

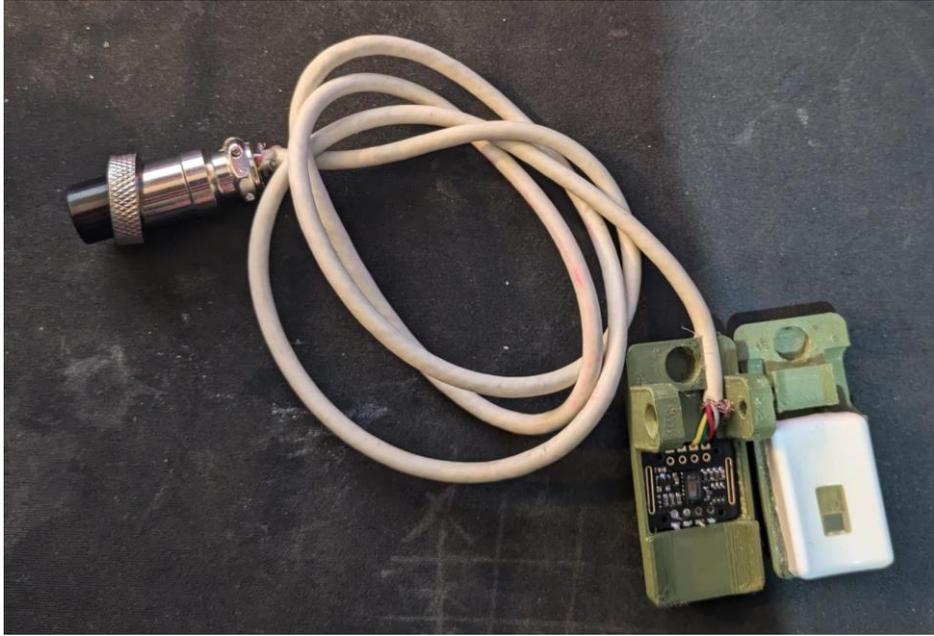


Figura 20. Conexión física exterior del sensor MAX-30102

4.3 Diseño de la interfaz de máquina-usuario en Nextion Editor.

La interfaz para que el usuario sea capaz de tomar signos vitales y se realice el triaje fue desarrollada en Nextion Editor.

4.3.1 Pantalla Inicio.



Figura 21. Pantalla Inicio del sistema QuickTriage

La pantalla de inicio será la primera en aparecer al encender el sistema. Posee el nombre del sistema y el nombre de los autores del sistema, finalmente, cuenta con un botón de Inicio que redirigirá al usuario a la siguiente página, la de información del paciente. La programación desarrollada dentro del botón b0, en el Touch Release Event, redirigirá al usuario a la siguiente página *'page page1'*.

4.3.2 Pantalla de Información.

Información del paciente

Nombre: nombre _____

Edad: edad _____

Sexo: masculino M femenino F

b1 BACK b0 NEXT

Figura 22. Pantalla de Información del paciente

En la pantalla de información del paciente será posible ingresar el nombre y edad del paciente por medio de un teclado qwerty que se desplegará al presionar sobre el cuadro de texto correspondiente, como se muestra en la Figura 23 (Ordóñez & Yagual, 2024). Además, se podrá encontrar en la pantalla dos botones que permitirán seleccionar el sexo del paciente. Por último, se ubicaron 2 botones en las esquinas inferiores, el botón b1 “Back” permitirá regresar a la pantalla de Inicio, mientras que el botón b0 “Next” permitirá avanzar a la siguiente página del sistema, correspondiente a la escala de Glasgow.

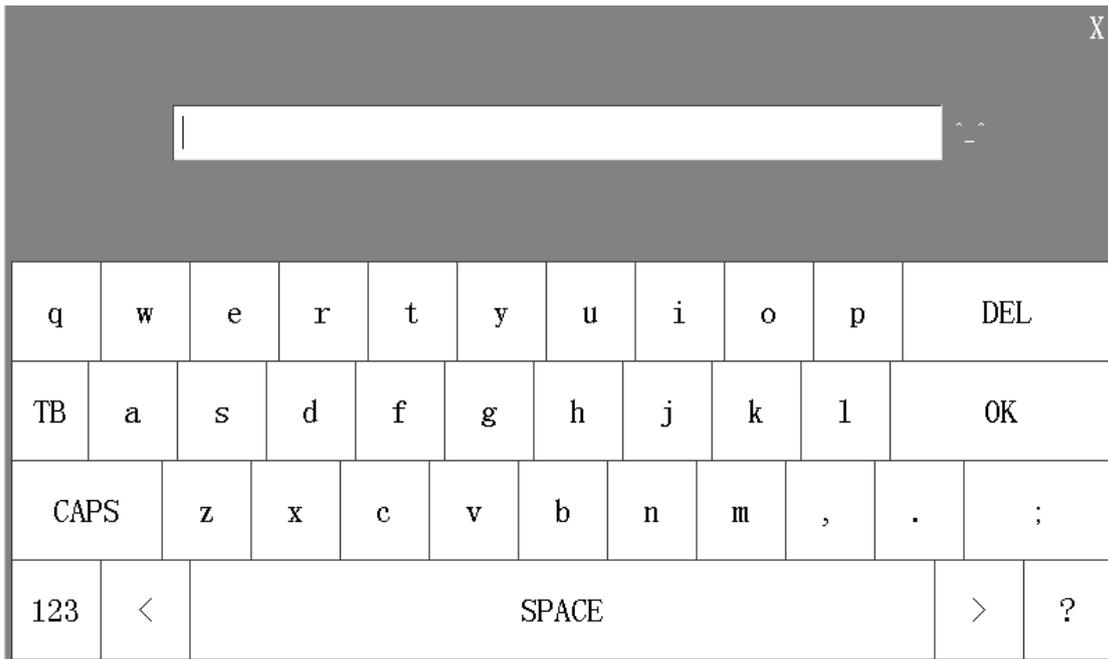


Figura 23. Teclado qwerty desplegado del cuadro de texto “Nombre”

4.3.3 Pantalla de Escala de Glasgow.



Figura 24. Pantalla de Escala de Glasgow

Al avanzar a la siguiente página, encontraremos la página de la escala de Glasgow. En esta pantalla encontraremos la escala de Glasgow diseccionada entre sus 3 valoraciones: abertura ocular, respuesta verbal y respuesta motora. Cada una de estas valoraciones va a contar con la herramienta CheckBoxes que podrán seleccionarse en dependencia de la situación del paciente, tal y como se muestra en la Figura 24 (Ordóñez & Yagual, 2024).

Categoría	Valoración	Estado
ABERTURA OCULAR	Espontánea (4)	<input checked="" type="checkbox"/>
	Al estímulo verbal (3)	<input type="checkbox"/>
	Al estímulo doloroso (2)	<input type="checkbox"/>
	Ninguna (1)	<input type="checkbox"/>
RESPUESTA VERBAL	Orientada (5)	<input type="checkbox"/>
	Confusa (4)	<input type="checkbox"/>
	Palabras inadecuadas (3)	<input checked="" type="checkbox"/>
	Sonidos incomprensibles (2)	<input type="checkbox"/>
RESPUESTA MOTORA	Ninguna (1)	<input type="checkbox"/>
	Obedece órdenes (6)	<input type="checkbox"/>
	Localiza el dolor (5)	<input type="checkbox"/>
	Movimiento de retirada (4)	<input checked="" type="checkbox"/>
	Flexión hipertónica (3)	<input type="checkbox"/>
	Extensión hipertónica (2)	<input type="checkbox"/>
	Ninguna (1)	<input type="checkbox"/>

Resultado: 0

CALCULAR

BACK NEXT

Figura 25. Checkboxes de cada valoración de la escala de Glasgow

Posterior a que se realice la valoración se presiona el botón glasgow1 “CALCULAR” para que se realice una sumatoria de los valores de cada valoración, dicha suma aparecerá en el cuadro de texto Glasgow. Esta sumatoria servirá para realizar el triaje posteriormente.

Respecto a la programación de cada Checkbox, se programó cada Checkbox para que solo pueda estar únicamente 1 Checkbox activado dentro de cada valoración, como se muestra en la Figura 25 (Ordóñez & Yagual, 2024). Cada Checkbox posee un bloque de programación en su Touch Release Event. De esta forma es posible evitar que aparezcan valores equivocados en el resultado de la sumatoria.

Finalmente, al presionar el botón b2 “Next” se avanzará a la siguiente pantalla de toma de signos vitales.



Figura 26. Touch Release Event y programación de Checkbox

4.3.4 Pantalla de Toma de signos vitales.



Figura 27. Pantalla de Toma de signos vitales

En la pantalla de Toma de signos vitales se encontrará cada signo vital importante para la realización de un triaje: Frecuencia respiratoria, temperatura, saturación de oxígeno, frecuencia cardíaca y presión arterial. En cada cuadro de texto de cada signo vital aparecerá el valor del signo vital correspondiente. Esta pantalla consta de 4 botones: INICIAR, CONGELAR, REINICIAR y TRIAJE.

4. INICIAR: El botón INICIAR permite empezar la toma de signos vitales y habilitar los sensores para obtener información.
5. CONGELAR: Una vez obtenidos los valores de los signos vitales se congelará la obtención de información de los sensores, lo que permitirá facilitar el procesamiento de los datos para el triaje. Esto será posible al presionar el botón CONGELAR.
6. REINICIAR: En caso de errores en la obtención de signos vitales o el deseo de volver a realizar una toma de información es posible hacerlo al presionar el botón REINICIAR.
7. TRIAJE: Finalmente, una vez obtenidos y congelados los datos de signos vitales se presiona el botón de TRIAJE que nos redirigirá a la siguiente página y se hará la ponderación de los valores obtenidos para realizar una clasificación de gravedad del paciente.

4.3.5 Pantalla de Resultados del Triage.

nombre

edad

NIVEL: 0

Nivel	Color	Tiempo de espera
1	ROJO	Atención de forma inmediata
2	NARANJA	10 - 15 MINUTOS
3	AMARILLO	60 MINUTOS
4	VERDE	2 HORAS
5	AZUL	4 HORAS

BACK

INICIO

Figura 28. Pantalla de Resultados del Triage

La última pantalla de la interfaz del sistema corresponde a la pantalla de Resultados del Triage. En esta pantalla se imprimirá el resultado de la clasificación de gravedad en base a los signos vitales tomados, así mismo se volverán a imprimir el nombre y edad del paciente. Además, será posible encontrar una tabla que mostrará cada código de color y tiempo de espera correspondiente a cada nivel de triaje.

Por último, encontramos dos botones: BACK e INICIO. El botón BACK nos permitirá regresar a la pantalla anterior y el botón INICIO reiniciará el sistema para una nueva toma de signos vitales y triaje.

4.4 Implementación física del sistema Nextion-Arduino.

Para integrar el sistema de triaje en un solo equipo se decidió diseñar una caja e imprimirla con material PLA para una correcta integración del sistema.

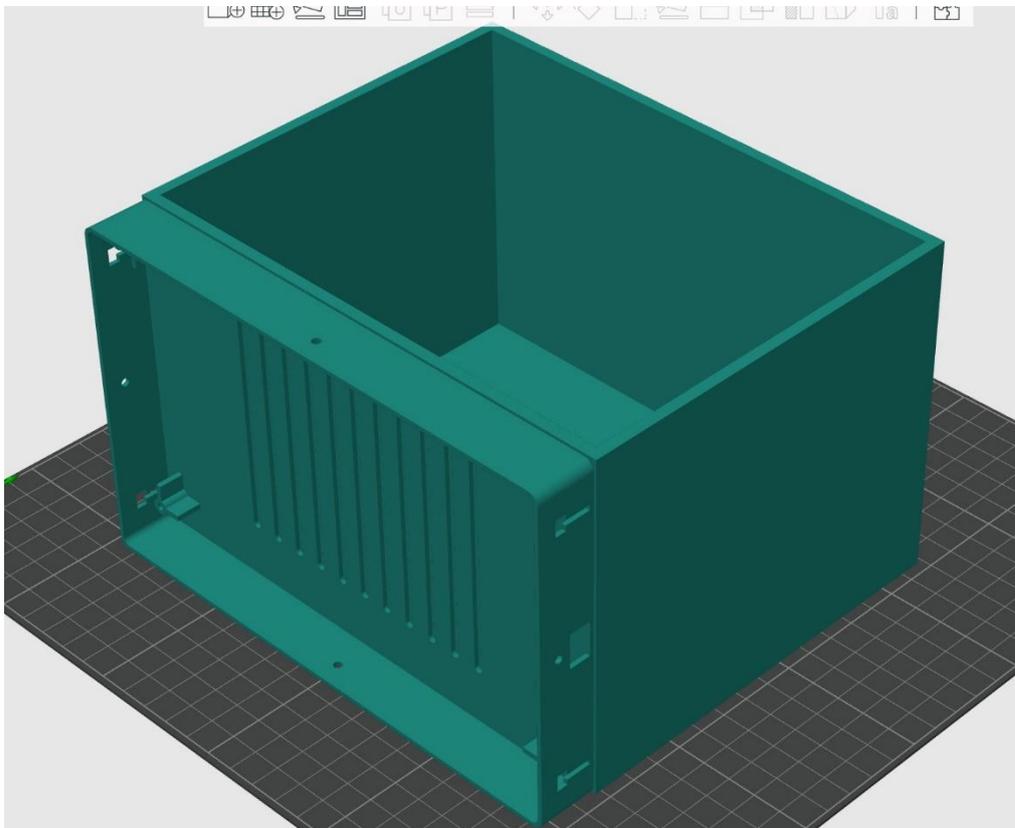


Figura 29. Diseño de la caja del sistema QuickTriage (a)

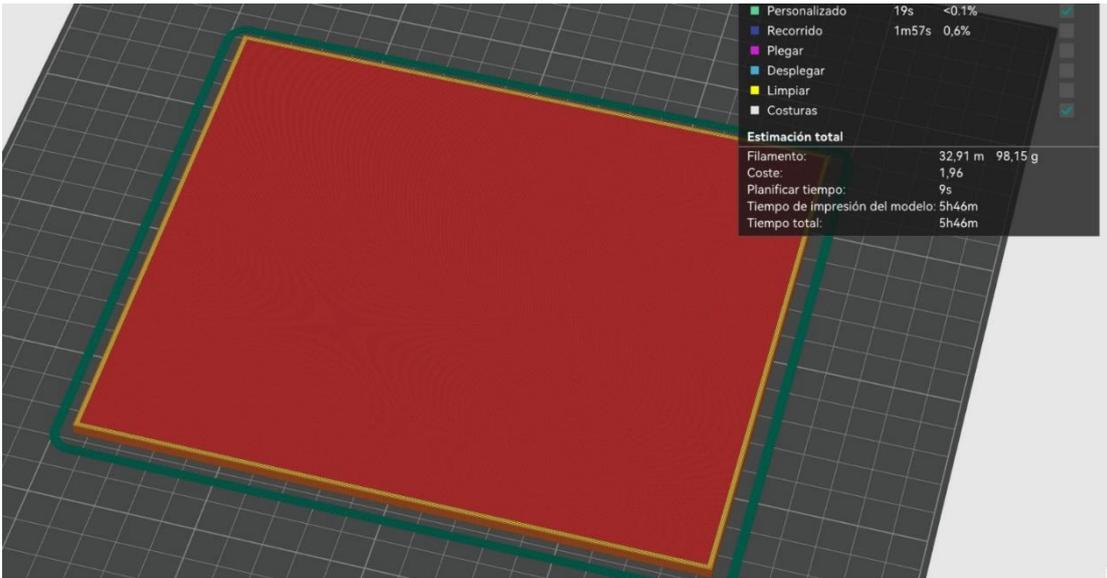


Figura 30. Diseño de la caja del sistema QuickTriage (b)



Figura 31. Caja del sistema QuickTriage (a)



Figura 32. Caja del sistema QuickTriage (b)

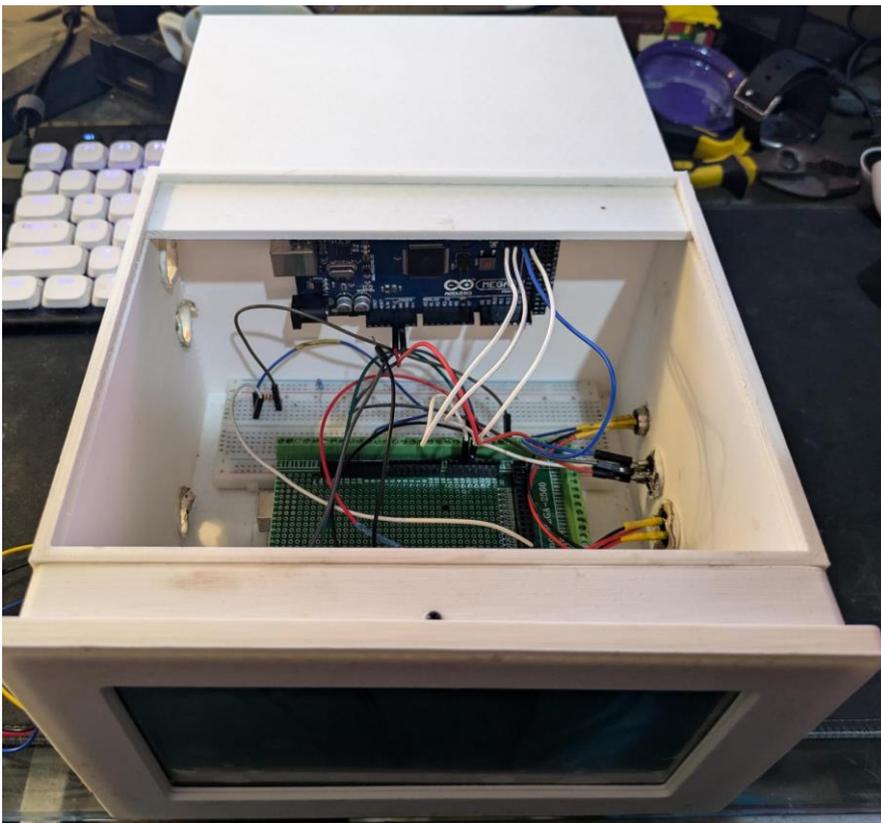


Figura 33. Caja del sistema QuickTriage (c)

4.5 Conexión Nextion-Arduino.

La conexión entre el Arduino y la Nextion es sencilla debido a su sistema de comunicación UART que utiliza los puertos Rx para recepción y Tx para la transmisión de datos. El Arduino mega principal utilizará los puertos Rx1 y Tx1 para comunicarse (recepción y envío de datos) con la pantalla Nextion.

5. Programación del Sistema.

La programación del sistema fue desarrollada en Arduino IDE, a continuación, se expondrá bloque por bloque el funcionamiento del código desarrollado específicamente para el sistema, más no se explicará el uso de las librerías ya creadas de los sensores.

5.1 Programación del Arduino Mega secundario.

En el segundo Arduino se encuentra conectado el sensor MAX-30102 que será el encargado de recibir el signo vital de oximetría y enviarlo al Arduino Mega principal, esto debido a que el tensiómetro y el sensor MAX-30102 pueden entrar en conflicto durante el envío de información. Para desarrollar el código de oximetría se utilizó en primera instancia la librería del sensor MAX-3012 desarrollada por SparkFun (SparkFun, 2020) junto a los ejemplos ofrecidos, para obtener valores de oximetría, siendo una de las funciones principales en esta librería la siguiente:

```
maxim_heart_rate_and_oxygen_saturation(irBuffer, bufferLength,  
redBuffer, &spo2, &validSPO2, &heartRate, &validHeartRate);
```

Este bloque de programación permite obtener los valores sensados por la luz roja e infrarroja y mide la saturación de oxígeno en base a cálculos. Posteriormente, se procedió a programar la comunicación serial entre en el Arduino Mega principal y secundario por medio del siguiente bloque de programación:

```

if (validSPO2) {
    Serial.print("SPO2=");
    Serial.println(spo2);
    Serial1.write((byte*)&spo2, sizeof(spo2));
}

```

En caso de que el valor obtenido de oximetría sea válido se imprimirá en el monitor Serial y a su vez el valor será enviado al Arduino Mega Principal por medio del código Serial1.write.

A continuación, se adjunta la programación en su totalidad:

```

#include <Wire.h>
#include "MAX30105.h"
#include "spo2_algorithm.h"

MAX30105 particleSensor;

#define MAX_BRIGHTNESS 255

#if defined(__AVR_ATmega328P__) || defined(__AVR_ATmega168__)
uint16_t irBuffer[100]; //infrared LED sensor data
uint16_t redBuffer[100]; //red LED sensor data
#else
uint32_t irBuffer[100]; //infrared LED sensor data
uint32_t redBuffer[100]; //red LED sensor data
#endif

int32_t bufferLength; //data length
int32_t spo2; //SPO2 value
int8_t validSPO2; //indicator to show if the SPO2 calculation
is valid
int32_t heartRate; //heart rate value
int8_t validHeartRate; //indicator to show if the heart rate
calculation is valid

byte pulseLED = 11; //Must be on PWM pin
byte readLED = 13; //Blinks with each data read

void setup()
{
    Serial.begin(115200); // initialize serial communication at
115200 bits per second:

```

```

//Serial1.begin(115200);
pinMode(pulseLED, OUTPUT);
pinMode(readLED, OUTPUT);

// Initialize sensor
if (!particleSensor.begin(Wire, I2C_SPEED_FAST)) //Use
default I2C port, 400kHz speed
{
    Serial.println(F("MAX30105 was not found. Please check
wiring/power."));
    while (1);
}

byte ledBrightness = 60; //Options: 0=Off to 255=50mA
byte sampleAverage = 4; //Options: 1, 2, 4, 8, 16, 32
byte ledMode = 2; //Options: 1 = Red only, 2 = Red + IR, 3 =
Red + IR + Green
byte sampleRate = 100; //Options: 50, 100, 200, 400, 800,
1000, 1600, 3200
int pulseWidth = 411; //Options: 69, 118, 215, 411
int adcRange = 4096; //Options: 2048, 4096, 8192, 16384

particleSensor.setup(ledBrightness, sampleAverage, ledMode,
sampleRate, pulseWidth, adcRange); //Configure sensor with these
settings
}

void loop()
{
    bufferLength = 100; //buffer length of 100 stores 4 seconds
of samples running at 25sps

    //read the first 100 samples, and determine the signal range
for (byte i = 0 ; i < bufferLength ; i++)
{
    while (particleSensor.available() == false) //do we have
new data?
        particleSensor.check(); //Check the sensor for new data

    redBuffer[i] = particleSensor.getRed();
    irBuffer[i] = particleSensor.getIR();
    particleSensor.nextSample(); //We're finished with this
sample so move to next sample
}
}

```

```

    }

    //calculate heart rate and SpO2 after first 100 samples (first
4 seconds of samples)
    maxim_heart_rate_and_oxygen_saturation(irBuffer,
bufferLength,    redBuffer,    &spo2,    &validSPO2,    &heartRate,
&validHeartRate);

    //Continuously taking samples from MAX30102. Heart rate and
SpO2 are calculated every 1 second
    while (1)
    {
        //dumping the first 25 sets of samples in the memory and
shift the last 75 sets of samples to the top
        for (byte i = 25; i < 100; i++)
        {
            redBuffer[i - 25] = redBuffer[i];
            irBuffer[i - 25] = irBuffer[i];
        }

        //take 25 sets of samples before calculating the heart
rate.
        for (byte i = 75; i < 100; i++)
        {
            while (particleSensor.available() == false) //do we have
new data?

                particleSensor.check(); //Check the sensor for new data

                digitalWrite(readLED, !digitalRead(readLED)); //Blink
onboard LED with every data read

                redBuffer[i] = particleSensor.getRed();
                irBuffer[i] = particleSensor.getIR();
                particleSensor.nextSample(); //We're finished with this
sample so move to next sample
        }

        //After gathering 25 new samples recalculate HR and SP02
        maxim_heart_rate_and_oxygen_saturation(irBuffer,
bufferLength,    redBuffer,    &spo2,    &validSPO2,    &heartRate,
&validHeartRate);

```

```

        // Enviar la información de oximetría por comunicación
        serial al otro Arduino Mega.
        if (validSPO2) {
            Serial.print("SPO2=");
            Serial.println(spo2);
            Serial1.write((byte*)&spo2, sizeof(spo2));
        }

        delay(500); // Delay to send data every second
    }
}

```

5.2 Programación del Arduino Mega Principal.

5.2.1 Librerías

En el Arduino Mega principal se llevará a cabo la recepción de señales de los siguientes sensores: DS18B20, KY-037 y TENSÍOMETRO DIGITAL. Además, este Arduino tendrá la tarea de comunicarse con la pantalla Nextion para la impresión de datos. El primer paso de la programación es la definición de las librerías a utilizar:

```

// Definición de librerías a utilizar
#include <Wire.h>
#include <OneWire.h>
#include <DallasTemperature.h>

```

Las librerías utilizadas son Wire.h (Arduino Core Team, 2024) para la comunicación I2C, OneWire.h (Jim Studt, 2007) y DallasTemperature.h (Miles Burton, 2015) para el sensor de temperatura DS18B20.

5.2.2 Definición de pines a utilizar

Posteriormente se definen los pines que utilizarán los sensores, en este caso el sensor DS18B20 el pin 53 y el sensor KY-037 el pin 52:

```

#define TEMP_PIN (53) // Pin del sensor de temperatura
#define FR_PIN (52) // Pin del sensor de temperatura

```

5.2.3 Variables Globales

A continuación, también se definen las variables globales que se utilizarán a lo largo de la programación:

```
//-----  
VariablesGlobales      -----  
-----//  
  
    // Variables globales para la comunicación I2C del tensiómetro  
    volatile byte i2c_data_rx;  
    volatile uint16_t count;  
    volatile uint8_t sys, dia, hr;  
  
    //Variables globales para guardar valores de signos vitales.  
    int ultimaTemperatura = 0;  
    int ultimaFrecuenciaCardiaca = 0;  
    int ultimaPresionSistolica = 0;  
    int ultimaPresionDiastolica = 0;  
    int ultimaFrecuenciaRespiratoria = 0;  
    int32_t ultimaSaturacion = 0;  
  
    int SENSOR_KY037 = FR_PIN;          // Pin al que está conectado  
    el sensor de sonido digital  
    int RESPIRACIONES = 0;              // Variable para contar el número  
    de inhalaciones  
    int Frecuencia_Respiratoria = 0;    // Variable para guardar la  
    frecuencia respiratoria  
    unsigned long startTime = 0;       // Tiempo de inicio del conteo  
  
    //Banderas de control.  
    bool medicionCompletada = false;    // Bandera para saber si se  
    ha medido la Frecuencia Respiratoria y Presión Arterial.  
    bool congelarDatos = false;        // Bandera para congelar valores  
    de signos vitales  
    bool medirRespiraciones = false;  
    bool leerTemperatura = false;  
    bool glasgowCompleto = false;      // Bandera para saber si la escala  
    de Glasgow fue realizada  
    bool in_measurement = false;       // Bandera para saber si el Arduino  
    está en modo esclavo
```

```

//Variables para escala de Glasgow
int abertura_ocular = 0;
int respuesta_verbal = 0;
int respuesta_motora = 0;
int sumaGlasgow = 0;
int nivelGlasgow = 0;

```

El apartado de Banderas de control nos va a permitir controlar en qué instancias se ejecutan ciertos bloques de código y no otros. Las variables de los sensores guardarán los valores enviados por los sensores, y las variables de la escala de Glasgow guardarán el valor que impriman los checkboxes en la pantalla de Escala de Glasgow.

5.2.4 Matrices de triaje

Por último, se van a establecer las matrices de datos con los cuales los valores de signos vitales se ponderarán y asignarán un nivel de triaje:

```

//-----
-----Matrices para la ponderación de valores de signos vitales
y clasificación-----
-----//
//-----
-----Cada Signo vital con sus rangos secundarios de
clasificación.-----
-----//

const int rows = 5; // 5 Niveles
const int cols = 2; // Rango inferior y superior

int matrizTemperatura[rows][cols] = {
    {0, 29}, {30, 33.9}, {34, 35.9}, {37.6, 38.9}, {36, 37.5}
};

int RangosSecundariosTemperatura[2][2] = {
    {41, 100}, // Nivel 2 secundario
    {39, 40.9} // Nivel 3 secundario
};

int matrizFrecuenciaCardiaca[rows][cols] = {
    {0, 40}, {41, 54}, {55, 59}, {90, 99}, {60, 89}
};

```

```

int RangosSecundariosFrecuenciaCardiaca[3][2] = {
    {180, 300}, // Nivel 1 secundario
    {140, 179}, // Nivel 2 secundario
    {100, 139} // Nivel 3 secundario
};

int matrizFrecuenciaRespiratoria[rows][cols] = {
    {0, 6}, {7, 10}, {11, 13}, {14, 20}, {21, 24}
};

int RangosSecundariosFrecuenciaRespiratoria[3][2] = {
    {41, 100}, // Nivel 1 secundario
    {31, 40}, // Nivel 2 secundario
    {25, 30} // Nivel 3 secundario
};

int matrizPresionSistolica[rows][cols] = {
    {0, 49}, {50, 69}, {70, 89}, {130, 149}, {90, 129}
};

int RangosSecundariosPresionSistolica[2][2] = {
    {170, 300}, // Nivel 2 secundario
    {150, 169} // Nivel 3 secundario
};

int matrizPresionDiastolica[rows][cols] = {
    {0, 40}, {41, 49}, {50, 59}, {90, 109}, {60, 89}
};

int RangosSecundariosPresionDiastolica[2][2] = {
    {121, 300}, // Nivel 2 secundario
    {110, 120} // Nivel 3 secundario
};

int matrizGlasgow[4][2] = {
    {3, 8}, {9, 12}, {13, 14}, {15, 15}
};

```

5.2.5 Void Setup ()

```

void setup() {
    Serial.begin(9600); //Comunicación serial encargada de
comunicarse con el Arduino IDE
    Serial1.begin(115200); //Comunicación serial encargada de
comunicarse con la pantalla Nextion
    Serial2.begin(115200); //Comunicación serial encargada de
comunicarse con el Arduino Mega Secundario
    pinMode(BP_START_PIN, OUTPUT);
}

```

```

pinMode (VALVE_PIN, INPUT);

pinMode (SENSOR_KY037, INPUT); // Inicializa el sensor de
frecuencia respiratoria estableciendo su estado.
sensorDS18B20.begin(); // Inicializa el sensor de temperatura

Serial.println("Esperando comando para iniciar
medición..."); // Aviso de que el Arduino se encuentra a la espera
de información de la pantalla Nextion
}

```

En el setup del programa se activarán los monitores seriales necesarios para la recepción y envío de información, así mismo, se establecerá en el estado de los pines a utilizar y se dará inicio al sensor de temperatura DS18B20.

5.2.6 Void Loop ()

En el loop del programa se encuentra el código que se repetirá una y otra vez a lo largo de la toma de signos vitales. La principal característica encontrada aquí es la recepción de información que proviene de la Nextion y la acción inmediata del Arduino en base a la información recibida. Lo primero en activarse será la recepción de información del Arduino y almacenar esa información en una variable:

```

if (Serial1.available()) {
    command = Serial1.read(); // Guardar los datos recibidos
en una variable
    Serial.println(command); // Imprimir la variable en el
monitor serial del Arduino IDE
}

```

Lo siguiente en el programa y en base al orden de las pantallas, será realizar la escala de Glasgow y obtener la sumatoria:

```

//-----
-----Escala de Glasgow-----
-----//
if (!glasgowCompleto) {
    switch (command) {

```

```

        case '1': abertura_ocular = 1; break;
        case '2': abertura_ocular = 2; break;
        case '3': abertura_ocular = 3; break;
        case '4': abertura_ocular = 4; break;
        case '5': respuesta_verbal = 1; break;
        case '6': respuesta_verbal = 2; break;
        case '7': respuesta_verbal = 3; break;
        case '8': respuesta_verbal = 4; break;
        case '9': respuesta_verbal = 5; break;
        case 'A': respuesta_motora = 1; break;
        case 'B': respuesta_motora = 2; break;
        case 'C': respuesta_motora = 3; break;
        case 'D': respuesta_motora = 4; break;
        case 'E': respuesta_motora = 5; break;
        case 'F': respuesta_motora = 6; break;
        case 'g':
            sumaGlasgow = abertura_ocular + respuesta_motora +
respuesta_verbal;
            Serial1.print("glasgow.val=");
            Serial1.print(sumaGlasgow);
            Serial1.write(0xff);
            Serial1.write(0xff);
            Serial1.write(0xff);

            nivelGlasgow = verificarNivelGlasgow(matrizGlasgow,
sumaGlasgow);
            Serial.print("Nivel de Glasgow");
            Serial.println(nivelGlasgow);

            Serial.println(sumaGlasgow);
            glasgowCompleto = true; // Marcar como completo
            break;
        }
    }
}

```

La variable command guardará el valor recibido al presionar los checkboxes observados en la pantalla Escala de Glasgow, al presionar el botón Calcular se realizará una sumatoria de los valores de cada checkbox y se imprimirá esa sumatoria en el cuadro de texto sumaGlasgow. Una vez finalizada la escala de Glasgow, el programa se verá

habilitado por medio de la bandera `glasgowCompleto` para trabajar con los sensores de signos vitales:

```
//-----  
-----Comandos encargados de activar los sensores-----  
-----//  
    if (glasgowCompleto) {  
        // Solo procesar el resto de los comandos después de  
completar los valores de Glasgow  
        switch (command) {  
            case 'i': // Inicia la medición y procesamiento de  
valores del sensor de frecuencia respiratoria y posteriormente de  
los demás sensores  
                medicionCompletada = true;  
                congelarDatos = false;  
                medirRespiraciones = true;  
                RESPIRACIONES = 0;  
                startTime = millis();  
                break;  
            case 'c': // Congela los datos  
                congelarDatos = true;  
                medicionCompletada = false;  
                medirRespiraciones = false;  
                break;  
            case 't': // Compara los valores sensados con las  
matrices y calcula el triaje  
                int nivelTemperatura =  
verificarNivel(matrizTemperatura,RangosSecundariosTemperatura, 2,  
ultimaTemperatura);  
                int nivelFrecuenciaCardiaca =  
verificarNivel(matrizFrecuenciaCardiaca,RangosSecundariosFrecuencia  
Cardiaca,3, ultimaFrecuenciaCardiaca);  
                int nivelFrecuenciaRespiratoria =  
verificarNivel(matrizFrecuenciaRespiratoria,RangosSecundariosFrecue  
nciaRespiratoria,3, ultimaFrecuenciaRespiratoria);  
                int nivelPresionDiastolica =  
verificarNivel(matrizPresionDiastolica,RangosSecundariosPresionDias  
tolica,2, ultimaPresionDiastolica);  
                int nivelPresionSistolica =  
verificarNivel(matrizPresionSistolica,RangosSecundariosPresionSisto  
lica,2, ultimaPresionSistolica);
```

```

        int sumaTotal = nivelTemperatura +
nivelFrecuenciaCardiaca + nivelFrecuenciaRespiratoria +
nivelPresionSistolica + nivelPresionDiastolica;
        int triaje = round(sumaTotal/5);
        Serial1.print("triaje.val=");
        Serial1.print(triaje);
        Serial1.write(0xff);
        Serial1.write(0xff);
        Serial1.write(0xff);

        Serial.println(nivelTemperatura);
        Serial.println(nivelTemperatura);
        Serial.println(nivelFrecuenciaCardiaca);
        Serial.println(nivelFrecuenciaRespiratoria);
        Serial.println(nivelPresionSistolica);
        Serial.println(nivelPresionDiastolica);
        Serial.println(triaje);
        delay(2000);

        break;
    case 'k': //Permite salir del bucle de
medirRespiraciones
        congelarDatos = false;
        medicionCompletada = true;
        medirRespiraciones = false;
        break;
    }
}

```

Cuando en la pantalla de Toma de signos vitales de la Nextion se presiona el botón INICIAR se enviará el caracter “i” al Arduino lo que permitirá la iniciación de la toma de frecuencia respiratoria gracias a la activación de la bandera medirRespiraciones:

```

    if (medirRespiraciones) {
        Serial.println("Se procederá a tomar la frecuencia
respiratoria");
        delay(1000);
        Serial1.print("fr.val=");
        Serial1.print(3);
        Serial1.write(0xff);
        Serial1.write(0xff);
        Serial1.write(0xff);
    }
}

```

```

    delay(1000);
    Serial1.print("fr.val=");
    Serial1.print(2);
    Serial1.write(0xff);
    Serial1.write(0xff);
    Serial1.write(0xff);
    delay(1000);
    Serial1.print("fr.val=");
    Serial1.print(1);
    Serial1.write(0xff);
    Serial1.write(0xff);
    Serial1.write(0xff);
    delay(1000);

    while (millis() - startTime < 15000) {
        int sensorValue = digitalRead(SENSOR_KY037); // Leer
el valor del sensor de sonido
        if (sensorValue == HIGH) {
            RESPIRACIONES++; // Incrementar el contador de
respiraciones
            Serial.print(RESPIRACIONES);
            Serial1.print("fr.val=");
            Serial1.print(RESPIRACIONES);
            Serial1.write(0xff);
            Serial1.write(0xff);
            Serial1.write(0xff);
            delay(1000); // Esperar para evitar conteos múltiples
por la misma inhalación
        }
    }
    // Calcular la frecuencia respiratoria
    Frecuencia_Respiratoria = RESPIRACIONES * 4; // Calcular
frecuencia respiratoria por minuto
    ultimaFrecuenciaRespiratoria = Frecuencia_Respiratoria;
    Serial1.print("fr.val=");
    Serial1.print(Frecuencia_Respiratoria);
    Serial1.write(0xff); // Se envían tres líneas después de
cada comando enviado a la pantalla Nextion.
    Serial1.write(0xff);
    Serial1.write(0xff);
    medirRespiraciones = false;
    command = 'k';

```

```
}
```

Cuando esto ocurra, el sensor KY-037 captará las respiraciones del paciente durante 15 segundos, para luego multiplicar ese valor entre 4 y obtener así la frecuencia respiratoria por minuto. Una vez obtenida la frecuencia respiratoria, se imprimirá en la pantalla Nextion y consecuentemente se habilitará el sensor de temperatura, tensiómetro y oxímetro para que empiece la impresión de valores, todo esto gracias a la activación de la bandera medicionCompletada:

```
if (mediccionCompletada) {
    iniciarMediccion();
    sensorDS18B20.requestTemperatures();
    ultimaTemperatura = sensorDS18B20.getTempCByIndex(0);

    if (Serial2.available() >= sizeof(ultimaSaturacion)) {
        Serial2.readBytes((char*)&ultimaSaturacion,
sizeof(ultimaSaturacion));
    }

    Serial1.print("saturacion.val=");
    Serial1.print(ultimaSaturacion);
    Serial1.write(0xff); // Se envían tres líneas después
de cada comando enviado a la pantalla Nextion.
    Serial1.write(0xff);
    Serial1.write(0xff);

    Serial1.print("temperatura.val=");
    Serial1.print(ultimaTemperatura);
    Serial1.write(0xff);
    Serial1.write(0xff);
    Serial1.write(0xff);

    Serial1.print("diastole.val=");
    Serial1.print(ultimaPresionDiastolica);
    Serial1.write(0xff);
    Serial1.write(0xff);
    Serial1.write(0xff);

    Serial1.print("fc.val=");
    Serial1.print(ultimaFrecuenciaCardiaca);
    Serial1.write(0xff);
}
```

```

        Serial1.write(0xff);
        Serial1.write(0xff);

        Serial1.print("sistole.val=");
        Serial1.print(ultimaPresionSistolica);
        Serial1.write(0xff);
        Serial1.write(0xff);
        Serial1.write(0xff);

        Serial1.print("fr.val=");
        Serial1.print(Frecuencia_Respiratoria);
        Serial1.write(0xff); // Se envían tres líneas después
de cada comando enviado a la pantalla Nextion.
        Serial1.write(0xff);
        Serial1.write(0xff);
    }

```

En este punto, el Arduino Mega principal se encontrará hábil para recibir la información enviada por el Arduino Mega secundario.

Cabe recalcar que para este momento el paciente debe tener puesto el tensiómetro de muñeca de la forma adecuada y se debe de presionar el botón ON, luego de la toma de datos se imprimirá la frecuencia cardíaca y la presión arterial en pantalla.

Una vez aparezcan todos los signos vitales en pantalla se presionará el botón CONGELAR para guardar los últimos valores leídos y se obtenga mayor estabilidad previo al triaje:

```

    if (congelarDatos) {
        // Imprimir los últimos valores si se ha activado la
congelación
        Serial1.print("temperatura.val=");
        Serial1.print(ultimaTemperatura);
        Serial1.write(0xff);
        Serial1.write(0xff);
        Serial1.write(0xff);

        Serial1.print("diastole.val=");
        Serial1.print(ultimaPresionDiastolica);
        Serial1.write(0xff);
        Serial1.write(0xff);
    }

```

```

        Serial1.write(0xff);

        Serial1.print("fc.val=");
        Serial1.print(ultimaFrecuenciaCardiaca);
        Serial1.write(0xff);
        Serial1.write(0xff);
        Serial1.write(0xff);

        Serial1.print("sistole.val=");
        Serial1.print(ultimaPresionSistolica);
        Serial1.write(0xff);
        Serial1.write(0xff);
        Serial1.write(0xff);

        Serial1.print("fr.val=");
        Serial1.print(Frecuencia_Respiratoria);
        Serial1.write(0xff); // Se envían tres líneas después
de cada comando enviado a la pantalla Nextion.
        Serial1.write(0xff);
        Serial1.write(0xff);

        Serial1.print("saturacion.val=");
        Serial1.print(ultimaSaturacion);
        Serial1.write(0xff); // Se envían tres líneas después
de cada comando enviado a la pantalla Nextion.
        Serial1.write(0xff);
        Serial1.write(0xff);
    }

```

Una vez congelados los datos, se presionará el botón TRIAJE, que enviará el caracter “t” donde se irá a la pantalla Resultados del Triage y se imprimirán los nombres, edad, sexo y el resultado de la clasificación del triaje:

```

    case 't': // Compara los valores sensados con las matrices y
calcula el triaje
        int nivelTemperatura =
verificarNivel(matrizTemperatura,RangosSecundariosTemperatura, 2,
ultimaTemperatura);
        int nivelFrecuenciaCardiaca =
verificarNivel(matrizFrecuenciaCardiaca,RangosSecundariosFrecuencia
Cardiaca,3, ultimaFrecuenciaCardiaca);

```

```

        int nivelFrecuenciaRespiratoria =
verificarNivel(matrizFrecuenciaRespiratoria,RangosSecundariosFrecue
nciaRespiratoria,3, ultimaFrecuenciaRespiratoria);
        int nivelPresionDiastolica =
verificarNivel(matrizPresionDiastolica,RangosSecundariosPresionDias
tolica,2, ultimaPresionDiastolica);
        int nivelPresionSistolica =
verificarNivel(matrizPresionSistolica,RangosSecundariosPresionSisto
lica,2, ultimaPresionSistolica);
        int sumaTotal = nivelTemperatura +
nivelFrecuenciaCardiaca + nivelFrecuenciaRespiratoria +
nivelPresionSistolica + nivelPresionDiastolica;
        int triaje = round(sumaTotal/5);
        Serial1.print("triaje=");
        Serial1.print(triaje);
        Serial1.write(0xff);
        Serial1.write(0xff);
        Serial1.write(0xff);

        Serial.println(nivelTemperatura);
        Serial.println(nivelTemperatura);
        Serial.println(nivelFrecuenciaCardiaca);
        Serial.println(nivelFrecuenciaRespiratoria);
        Serial.println(nivelPresionSistolica);
        Serial.println(nivelPresionDiastolica);
        Serial.println(triaje);
        delay(2000);

break;

```

5.2.7 Funciones para procesamiento de datos.

```

// Función para iniciar la medición en modo esclavo
void iniciarMedicion() {
    in_measurement = true; // Marca que estamos en modo esclavo
    count = 0; // Reinicia el contador
    Wire.begin(0x50); // Cambia el Arduino a modo esclavo
con dirección 0x50
    Wire.onReceive(receiveEvent); // Activa la recepción de
datos I2C
    Serial.println("Modo esclavo activado, esperando datos...");
}
// Función para finalizar la medición y volver a modo maestro
void finalizarMedicion() {

```

```

    in_measurement = false; // Ya no estamos en modo esclavo
    count = 0; // Reinicia el contador

    Wire.end(); // Termina la comunicación I2C en
modo esclavo

    Serial.println("Medición completada. Arduino vuelve a ser
maestro.");
}
void receiveEvent(int iData) {
    if (iData > 0 && in_measurement) { // Solo recibe datos si
está en modo esclavo
        while (iData--) {
            i2c_data_rx = Wire.read();
            count++;

            if (count == 28) {
                sys = i2c_data_rx;
                ultimaPresionSistolica = int(sys);
            }
            if (count == 29) {
                dia = i2c_data_rx;
                ultimaPresionDiastolica = int(dia);
            }
            if (count == 30) {
                hr = i2c_data_rx;
                ultimaFrecuenciaCardiaca = hr;
            }
        }
    }
}
// Función para verificar el nivel de triaje de cada signo
vital.

int verificarNivel(int matriz[rows][cols], int
secondaryRanges[][2], int numSecondaryRanges, int valor) {
    // Verifica los valores en la matriz principal
    for (int i = 0; i < rows; i++) {
        if (valor >= matriz[i][0] && valor <= matriz[i][1]) {
            return i + 1; // Regresa el nivel encontrado (1-5)
        }
    }
}

```

```

    // Verifica los rangos secundarios
    for (int i = 0; i < numSecondaryRanges; i++) {
        if (valor >= secondaryRanges[i][0] && valor <=
secondaryRanges[i][1]) {
            return i + 2; // Regresa el nivel correspondiente al
rango secundario (ajustado para que coincida con nivel 2, 3, etc.)
        }
    }

    return 0; // Si no se encuentra un nivel, regresa 0
}

// Función para verificar el nivel de Glasgow

int verificarNivelGlasgow(int matriz[4][2], int valor) {
    // Verifica los valores en la matriz de Glasgow
    for (int i = 0; i < 4; i++) {
        if (valor >= matriz[i][0] && valor <= matriz[i][1]) {
            // Si el valor es 15, asigna directamente el nivel 5
            if (valor == 15) {
                return 5;
            }
            return i + 1; // Regresa el nivel encontrado (1-5)
        }
    }

    return 0; // Si no se encuentra un nivel, regresa 0
}

```

5.2.8 Código completo

```

//-----
-----Definición de librerías a utilizar-----
//-----

#include <Wire.h>
#include <OneWire.h>
#include <DallasTemperature.h>

//-----
-----Definición de pines a utilizar-----
//-----

#define BP_START_PIN (2) // Emulación del botón de inicio del
dispositivo de medición de presión arterial
#define VALVE_PIN (3) // Verifica si la medición del
tensiometro ha terminado

```

```

#define TEMP_PIN (53) // Pin del sensor de temperatura
#define FR_PIN (52) // Pin del sensor de temperatura

//-----Variables Globales-----
//-----//

// Variables globales para la comunicación I2C del tensiómetro
volatile byte i2c_data_rx;
volatile uint16_t count;
volatile uint8_t sys, dia, hr;

//Variables globales para guardar valores de signos vitales.
int ultimaTemperatura = 0;
int ultimaFrecuenciaCardiaca = 0;
int ultimaPresionSistolica = 0;
int ultimaPresionDiastolica = 0;
int ultimaFrecuenciaRespiratoria = 0;
int32_t ultimaSaturacion = 0;

int SENSOR_KY037 = FR_PIN; // Pin al que está conectado
el sensor de sonido digital
int RESPIRACIONES = 0; // Variable para contar el número
de inhalaciones
int Frecuencia_Respiratoria = 0; // Variable para guardar la
frecuencia respiratoria
unsigned long startTime = 0; // Tiempo de inicio del conteo

//Banderas de control.
bool medicionCompletada = false; // Bandera para saber si se
ha medido la Frecuencia Respiratoria y Presión Arterial.
bool congelarDatos = false; // Bandera para congelar valores
de signos vitales
bool medirRespiraciones = false;
bool leerTemperatura = false;
bool glasgowCompleto = false; // Bandera para saber si la escala
de Glasgow fue realizada
bool in_measurement = false; // Bandera para saber si el Arduino
está en modo esclavo

//Variables para escala de Glasgow

```

```

int abertura_ocular = 0;
int respuesta_verbal = 0;
int respuesta_motora = 0;
int sumaGlasgow = 0;
int nivelGlasgow = 0;

//-----
-----Matrices para la ponderación de valores de signos vitales
y clasificación-----
-----//
//-----
-----Cada Signo vital con sus rangos secundarios de
clasificación.-----
-----//

const int rows = 5; // 5 Niveles
const int cols = 2; // Rango inferior y superior

int matrizTemperatura[rows][cols] = {
    {0, 29}, {30, 33.9}, {34, 35.9}, {37.6, 38.9}, {36, 37.5}
};
int RangosSecundariosTemperatura[2][2] = {
    {41, 100}, // Nivel 2 secundario
    {39, 40.9} // Nivel 3 secundario
};

int matrizFrecuenciaCardiaca[rows][cols] = {
    {0, 40}, {41, 54}, {55, 59}, {90, 99}, {60, 89}
};
int RangosSecundariosFrecuenciaCardiaca[3][2] = {
    {180, 300}, // Nivel 1 secundario
    {140, 179}, // Nivel 2 secundario
    {100, 139} // Nivel 3 secundario
};

int matrizFrecuenciaRespiratoria[rows][cols] = {
    {0, 6}, {7, 10}, {11, 13}, {14, 20}, {21, 24}
};
int RangosSecundariosFrecuenciaRespiratoria[3][2] = {
    {41, 100}, // Nivel 1 secundario
    {31, 40}, // Nivel 2 secundario

```

```

    {25, 30}    // Nivel 3 secundario
};

int matrizPresionSistolica[rows][cols] = {
    {0, 49}, {50, 69}, {70, 89}, {130, 149}, {90, 129}
};

int RangosSecundariosPresionSistolica[2][2] = {
    {170, 300}, // Nivel 2 secundario
    {150, 169} // Nivel 3 secundario
};

int matrizPresionDiastolica[rows][cols] = {
    {0, 40}, {41, 49}, {50, 59}, {90, 109}, {60, 89}
};

int RangosSecundariosPresionDiastolica[2][2] = {
    {121, 300}, // Nivel 2 secundario
    {110, 120} // Nivel 3 secundario
};

int matrizGlasgow[4][2] = {
    {3, 8}, // Nivel 1
    {9, 12}, // Nivel 2
    {13, 14}, // Nivel 3
    {15, 15} // Nivel 5
};

char command; //Variable de recepción de información de la
comunicación serial

// Sensor de temperatura
OneWire oneWireObjeto(TEMP_PIN);
DallasTemperature sensorDS18B20(&oneWireObjeto);

void setup() {
    Serial.begin(9600); //Comunicación serial encargada de
comunicarse con el Arduino IDE
    Serial1.begin(115200); //Comunicación serial encargada de
comunicarse con la pantalla Nextion
    Serial2.begin(115200); //Comunicación serial encargada de
comunicarse con el Arduino Mega Secundario
    pinMode(BP_START_PIN, OUTPUT);
    pinMode(VALUE_PIN, INPUT);
}

```

```

    pinMode(SENSOR_KY037, INPUT); // Inicializa el sensor de
frecuencia respiratoria estableciendo su estado.
    sensorDS18B20.begin(); // Inicializa el sensor de temperatura

    Serial.println("Esperando comando para iniciar
medición..."); // Aviso de que el Arduino se encuentra a la espera
de información de la pantalla Nextion
}

void loop() {

    // Habilitar la comunicación serial entre la Nextion y el
Arduino para la transmisión de información.
    if (Serial1.available()) {
        command = Serial1.read(); // Guardar los datos recibidos
en una variable
        Serial.println(command); // Imprimir la variable en el
monitor serial del Arduino IDE
    }

    //-----Escala de Glasgow-----//

    if (!glasgowCompleto) {
        switch (command) {
            case '1': abertura_ocular = 1; break;
            case '2': abertura_ocular = 2; break;
            case '3': abertura_ocular = 3; break;
            case '4': abertura_ocular = 4; break;
            case '5': respuesta_verbal = 1; break;
            case '6': respuesta_verbal = 2; break;
            case '7': respuesta_verbal = 3; break;
            case '8': respuesta_verbal = 4; break;
            case '9': respuesta_verbal = 5; break;
            case 'A': respuesta_motora = 1; break;
            case 'B': respuesta_motora = 2; break;
            case 'C': respuesta_motora = 3; break;
            case 'D': respuesta_motora = 4; break;
            case 'E': respuesta_motora = 5; break;
            case 'F': respuesta_motora = 6; break;
            case 'g':

```

```

        sumaGlasgow = abertura_ocular + respuesta_motora +
respuesta_verbal;
        Serial1.print("glasgow.val=");
        Serial1.print(sumaGlasgow);
        Serial1.write(0xff);
        Serial1.write(0xff);
        Serial1.write(0xff);

        nivelGlasgow = verificarNivelGlasgow(matrizGlasgow,
sumaGlasgow);
        Serial.print("Nivel de Glasgow");
        Serial.println(nivelGlasgow);

        Serial.println(sumaGlasgow);
        glasgowCompleto = true; // Marcar como completo
        break;
    }
}
//-----
-----Comandos encargados de activar los sensores-----
-----//

    if (glasgowCompleto) {
        // Solo procesar el resto de los comandos después de
completar los valores de Glasgow
        switch (command) {
            case 'i': // Inicia la medición y procesamiento de
valores del sensor de frecuencia respiratoria y posteriormente de
los demás sensores
                medicionCompletada = true;
                congelarDatos = false;
                medirRespiraciones = true;
                RESPIRACIONES = 0;
                startTime = millis();
                break;
            case 'c': // Congela los datos
                congelarDatos = true;
                medicionCompletada = false;
                medirRespiraciones = false;
                break;
            case 't': // Compara los valores sensados con las
matrices y calcula el triaje

```

```

        int nivelTemperatura =
verificarNivel (matrizTemperatura, RangosSecundariosTemperatura, 2,
ultimaTemperatura);
        int nivelFrecuenciaCardiaca =
verificarNivel (matrizFrecuenciaCardiaca, RangosSecundariosFrecuencia
Cardiaca, 3, ultimaFrecuenciaCardiaca);
        int nivelFrecuenciaRespiratoria =
verificarNivel (matrizFrecuenciaRespiratoria, RangosSecundariosFrecue
nciaRespiratoria, 3, ultimaFrecuenciaRespiratoria);
        int nivelPresionDiastolica =
verificarNivel (matrizPresionDiastolica, RangosSecundariosPresionDias
tolica, 2, ultimaPresionDiastolica);
        int nivelPresionSistolica =
verificarNivel (matrizPresionSistolica, RangosSecundariosPresionSisto
lica, 2, ultimaPresionSistolica);
        int sumaTotal = nivelTemperatura +
nivelFrecuenciaCardiaca + nivelFrecuenciaRespiratoria +
nivelPresionSistolica + nivelPresionDiastolica;
        int triaje = round((sumaTotal+nivelGlasgow)/5);
        Serial1.print("triaje.val=");
        Serial1.print(triaje);
        Serial1.write(0xff);
        Serial1.write(0xff);
        Serial1.write(0xff);

        Serial.println(nivelTemperatura);
        Serial.println(nivelTemperatura);
        Serial.println(nivelFrecuenciaCardiaca);
        Serial.println(nivelFrecuenciaRespiratoria);
        Serial.println(nivelPresionSistolica);
        Serial.println(nivelPresionDiastolica);
        Serial.println(triaje);
        delay(2000);

        break;
    case 'k': //Permite salir del bucle de
medirRespiraciones
        congelarDatos = false;
        medicionCompletada = true;
        medirRespiraciones = false;
        break;
    }
}

```

```

        if (medirRespiraciones) {
            Serial.println("Se procederá a tomar la frecuencia
respiratoria");
            delay(1000);
            Serial1.print("fr.val=");
            Serial1.print(3);
            Serial1.write(0xff);
            Serial1.write(0xff);
            Serial1.write(0xff);
            delay(1000);
            Serial1.print("fr.val=");
            Serial1.print(2);
            Serial1.write(0xff);
            Serial1.write(0xff);
            Serial1.write(0xff);
            delay(1000);
            Serial1.print("fr.val=");
            Serial1.print(1);
            Serial1.write(0xff);
            Serial1.write(0xff);
            Serial1.write(0xff);
            delay(1000);

            while (millis() - startTime < 15000) {
                int sensorValue = digitalRead(SENSOR_KY037); // Leer
el valor del sensor de sonido
                if (sensorValue == HIGH) {
                    RESPIRACIONES++; // Incrementar el contador de
respiraciones
                    Serial.print(RESPIRACIONES);
                    Serial1.print("fr.val=");
                    Serial1.print(RESPIRACIONES);
                    Serial1.write(0xff);
                    Serial1.write(0xff);
                    Serial1.write(0xff);
                    delay(1000); // Esperar para evitar conteos múltiples
por la misma inhalación
                }
            }
            // Calcular la frecuencia respiratoria

```

```

        Frecuencia_Respiratoria = RESPIRACIONES * 4; // Calcular
frecuencia respiratoria por minuto
        ultimaFrecuenciaRespiratoria = Frecuencia_Respiratoria;
        Serial1.print("fr.val=");
        Serial1.print(Frecuencia_Respiratoria);
        Serial1.write(0xff); // Se envían tres líneas después de
cada comando enviado a la pantalla Nextion.
        Serial1.write(0xff);
        Serial1.write(0xff);
        medirRespiraciones = false;
        command = 'k';

    }
// Procesa los datos del tensiómetro si han sido recibidos
if (count == 35) {

    Serial1.print("diastole.val=");
    Serial1.print(int(dia));
    Serial1.write(0xff);
    Serial1.write(0xff);
    Serial1.write(0xff);

    Serial1.print("fc.val=");
    Serial1.print(hr);
    Serial1.write(0xff);
    Serial1.write(0xff);
    Serial1.write(0xff);

    Serial1.print("sistole.val=");
    Serial1.print(int(sys));
    Serial1.write(0xff);
    Serial1.write(0xff);
    Serial1.write(0xff);

    // Después de recibir los datos, finalizar la
comunicación I2C y reiniciar los estados
    finalizarMedicion();
    medicionCompletada = true;
} else if (count > 0 && count != 35) {
    Serial.println("Error en la recepción de datos.");
    count = 0;
}

```

```

if (mediccionCompletada) {
    iniciarMediccion();
    sensorDS18B20.requestTemperatures();
    ultimaTemperatura = sensorDS18B20.getTempCByIndex(0);

    if (Serial2.available() >= sizeof(ultimaSaturacion)) {
        Serial2.readBytes((char*)&ultimaSaturacion,
sizeof(ultimaSaturacion));
    }
    Serial1.print("saturacion.val=");
    Serial1.print(ultimaSaturacion);
    Serial1.write(0xff); // Se envían tres líneas después
de cada comando enviado a la pantalla Nextion.
    Serial1.write(0xff);
    Serial1.write(0xff);

    Serial1.print("temperatura.val=");
    Serial1.print(ultimaTemperatura);
    Serial1.write(0xff);
    Serial1.write(0xff);
    Serial1.write(0xff);

    Serial1.print("diastole.val=");
    Serial1.print(ultimaPresionDiastolica);
    Serial1.write(0xff);
    Serial1.write(0xff);
    Serial1.write(0xff);

    Serial1.print("fc.val=");
    Serial1.print(ultimaFrecuenciaCardiaca);
    Serial1.write(0xff);
    Serial1.write(0xff);
    Serial1.write(0xff);

    Serial1.print("sistole.val=");
    Serial1.print(ultimaPresionSistolica);
    Serial1.write(0xff);
    Serial1.write(0xff);
    Serial1.write(0xff);

    Serial1.print("fr.val=");
    Serial1.print(Frecuencia_Respiratoria);

```

```

        Serial1.write(0xff); // Se envían tres líneas después
de cada comando enviado a la pantalla Nextion.
        Serial1.write(0xff);
        Serial1.write(0xff);
    }

    if (congelarDatos) {
        // Imprimir los últimos valores si se ha activado la
congelación

        Serial1.print("temperatura.val=");
        Serial1.print(ultimaTemperatura);
        Serial1.write(0xff);
        Serial1.write(0xff);
        Serial1.write(0xff);

        Serial1.print("diastole.val=");
        Serial1.print(ultimaPresionDiastolica);
        Serial1.write(0xff);
        Serial1.write(0xff);
        Serial1.write(0xff);

        Serial1.print("fc.val=");
        Serial1.print(ultimaFrecuenciaCardiaca);
        Serial1.write(0xff);
        Serial1.write(0xff);
        Serial1.write(0xff);

        Serial1.print("sistole.val=");
        Serial1.print(ultimaPresionSistolica);
        Serial1.write(0xff);
        Serial1.write(0xff);
        Serial1.write(0xff);

        Serial1.print("fr.val=");
        Serial1.print(Frecuencia_Respiratoria);
        Serial1.write(0xff); // Se envían tres líneas después
de cada comando enviado a la pantalla Nextion.
        Serial1.write(0xff);
        Serial1.write(0xff);

        Serial1.print("saturacion.val=");
        Serial1.print(ultimaSaturacion);

```

```

        Serial1.write(0xff); // Se envían tres líneas después
de cada comando enviado a la pantalla Nextion.
        Serial1.write(0xff);
        Serial1.write(0xff);
    }

    delay(500);
}

// Función para iniciar la medición en modo esclavo
void iniciarMedicion() {
    in_measurement = true; // Marca que estamos en modo esclavo
    count = 0;             // Reinicia el contador
    Wire.begin(0x50);      // Cambia el Arduino a modo esclavo
con dirección 0x50
    Wire.onReceive(receiveEvent); // Activa la recepción de
datos I2C
    Serial.println("Modo esclavo activado, esperando datos...");
}
// Función para finalizar la medición y volver a modo maestro
void finalizarMedicion() {
    in_measurement = false; // Ya no estamos en modo esclavo
    count = 0;             // Reinicia el contador

    Wire.end();           // Termina la comunicación I2C en
modo esclavo

    Serial.println("Medición completada. Arduino vuelve a ser
maestro.");
}
void receiveEvent(int iData) {
    if (iData > 0 && in_measurement) { // Solo recibe datos si
está en modo esclavo
        while (iData--> 0) {
            i2c_data_rx = Wire.read();
            count++;

            if (count == 28) {
                sys = i2c_data_rx;
                ultimaPresionSistolica = int(sys);
            }
            if (count == 29) {
                dia = i2c_data_rx;

```

```

        ultimaPresionDiastolica = int(dia);
    }
    if (count == 30) {
        hr = i2c_data_rx;
        ultimaFrecuenciaCardiaca = hr;
    }
}
}

/*int verificarNivel(int matriz[rows][cols], int valor) {
    for (int i = 0; i < rows; i++) {
        if (valor >= matriz[i][0] && valor <= matriz[i][1]) {
            return i + 1; // Regresa el nivel encontrado
        }
    }
    return 0; // Si no se encuentra un nivel, regresa 0
}*/

int verificarNivel(int matriz[rows][cols], int
secondaryRanges[][2], int numSecondaryRanges, int valor) {
    // Verifica los valores en la matriz principal
    for (int i = 0; i < rows; i++) {
        if (valor >= matriz[i][0] && valor <= matriz[i][1]) {
            return i + 1; // Regresa el nivel encontrado (1-5)
        }
    }

    // Verifica los rangos secundarios
    for (int i = 0; i < numSecondaryRanges; i++) {
        if (valor >= secondaryRanges[i][0] && valor <=
secondaryRanges[i][1]) {
            return i + 2; // Regresa el nivel correspondiente al
rango secundario (ajustado para que coincida con nivel 2, 3, etc.)
        }
    }

    return 0; // Si no se encuentra un nivel, regresa 0
}

int verificarNivelGlasgow(int matriz[4][2], int valor) {
    // Verifica los valores en la matriz de Glasgow
    for (int i = 0; i < 4; i++) {

```

```
if (valor >= matriz[i][0] && valor <= matriz[i][1]) {
    // Si el valor es 15, asigna directamente el nivel 5
    if (valor == 15) {
        return 5;
    }
    return i + 1; // Regresa el nivel encontrado (1-5)
}

return 0; // Si no se encuentra un nivel, regresa 0
}
```

Resultados

1. Puesta a prueba del equipo

Una vez desarrollada la programación y que el equipo haya sido ensamblado, se procedió a hacer una prueba en un Paciente A. Es posible observar a lo largo de las fotos que la interfaz presentada en la pantalla Nextion es fácil e intuitiva de manejar, a la vez que permite conocer los signos vitales del paciente y también permite la clasificación del paciente en un nivel de gravedad tomando en consideración los signos vitales y la escala de Glasgow. El prototipo se encuentra funcional.

1.1 Ingreso de nombre y edad.

El primer paso en el sistema QuickTriage es la recolección básica de información del paciente: nombre, edad y sexo.

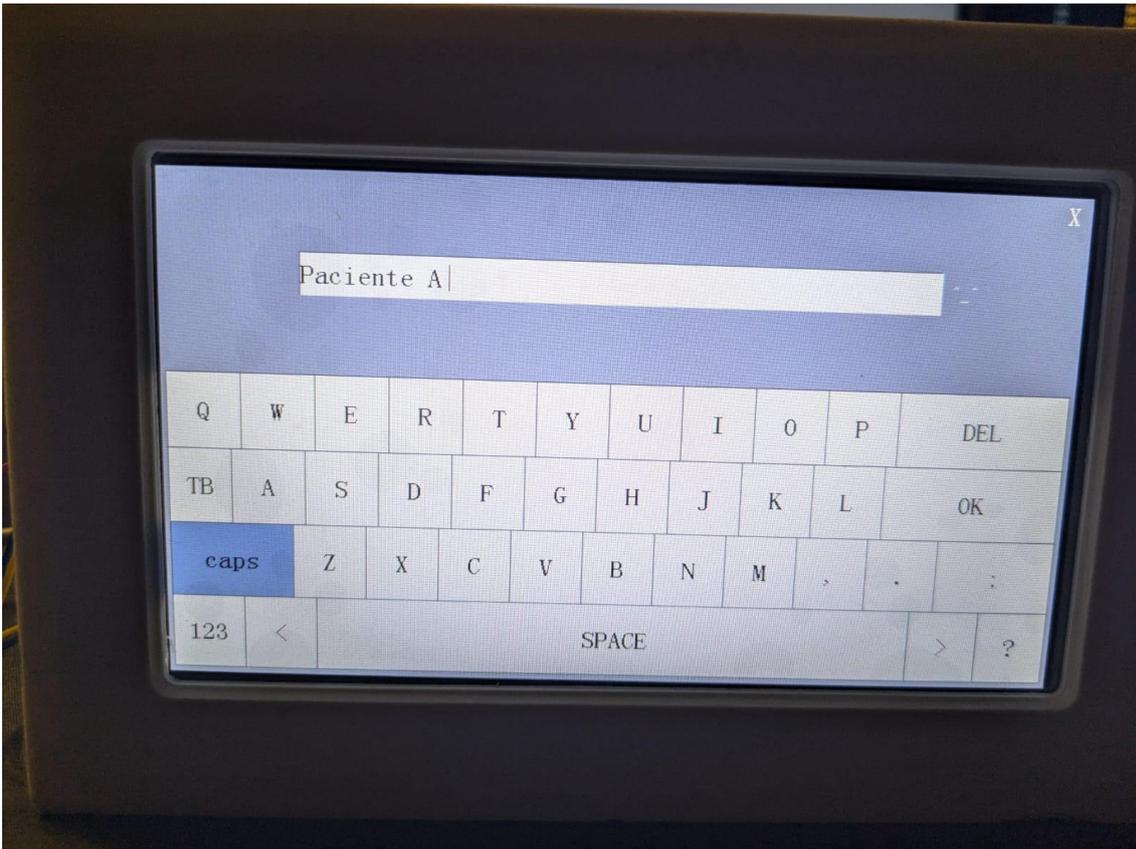


Figura 34. Ingreso de nombre y edad

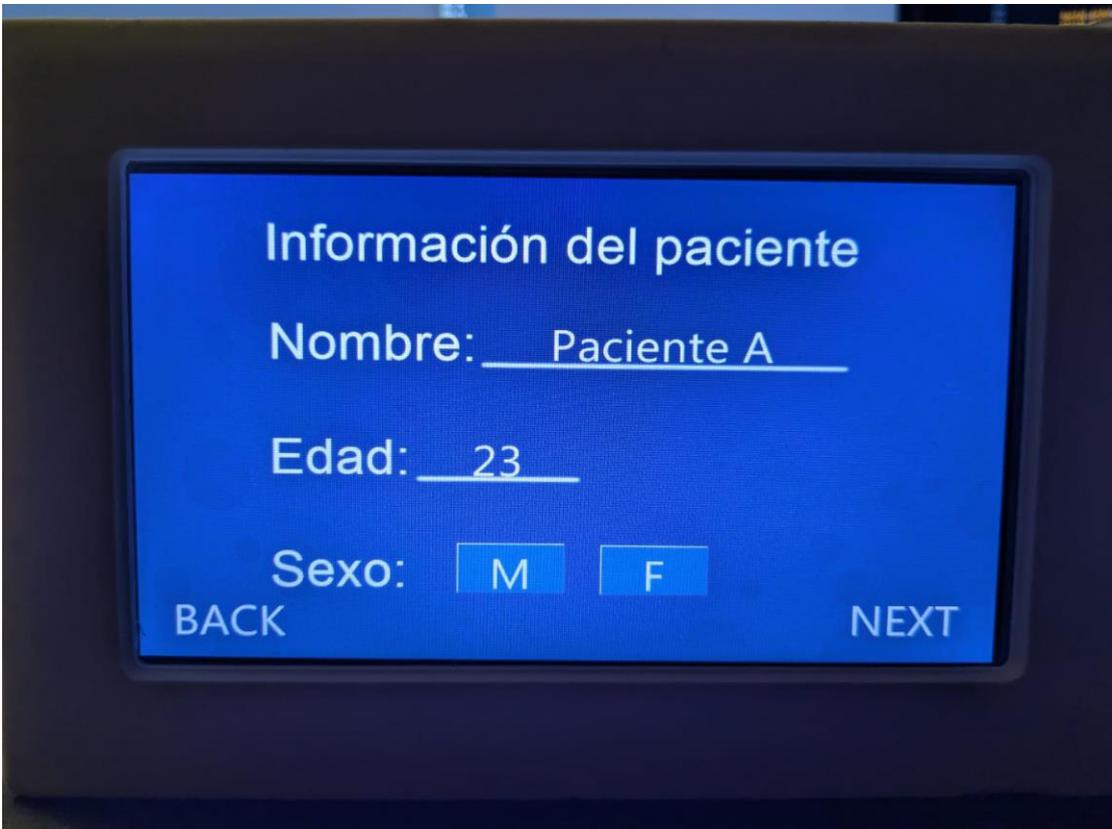


Figura 35. Pantalla de Información del Paciente.

1.2 Realización de la escala de Glasgow.

El siguiente paso es la selección de los valores en la escala de Glasgow en cada uno de sus valores: abertura ocular, respuesta verbal y respuesta motora. Una vez realizada la valoración se debe dar click en el checkbox correspondiente.

Categoría	Respuesta	Valor	Seleccionado
ABERTURA OCULAR	Espontánea	4	<input checked="" type="checkbox"/>
	Al estímulo verbal	3	<input type="checkbox"/>
	Al estímulo doloroso	2	<input type="checkbox"/>
	Ninguna	1	<input type="checkbox"/>
RESPUESTA VERBAL	Orientada	5	<input checked="" type="checkbox"/>
	Confusa	4	<input type="checkbox"/>
	Palabras inadecuadas	3	<input type="checkbox"/>
	Sonidos Incomprensibles	2	<input type="checkbox"/>
	Ninguna	1	<input type="checkbox"/>
RESPUESTA MOTORA	Obedece órdenes	6	<input checked="" type="checkbox"/>
	Localiza el dolor	5	<input type="checkbox"/>
	Movimiento de rotación	4	<input type="checkbox"/>
	Flexión hipertónica	3	<input type="checkbox"/>
	Extensión hipertónica	2	<input type="checkbox"/>
	Ninguna	1	<input type="checkbox"/>

Resultado: **15**

CALCULAR

BACK NEXT

Figura 36. Realización de la escala de Glasgow

1.3 Toma de signos vitales.

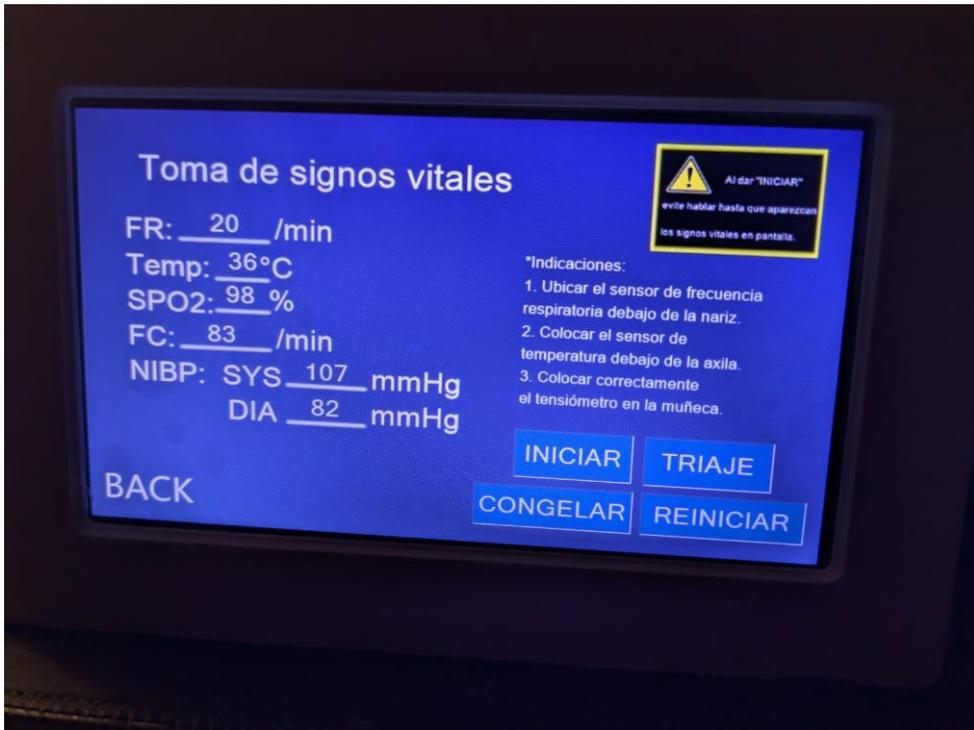


Figura 37. Toma de signos vitales (a)



Figura 38. Toma de signos vitales (b)



Figura 39. Toma de signos vitales (c)

1.4 Realización del Triage



Figura 40. Realización del Triage

2. Pruebas de verificación.

Para realizar pruebas de verificación, se utilizó el prototipo en un centro de salud de primer nivel del Guasmo Central que afortunadamente contaba con un monitor multiparámetros que permitió realizar las comparaciones pertinentes. Las comparaciones se realizaron en 3 pacientes. En base a las mediciones tomadas y observadas es posible definir que el prototipo se encuentra dentro de un margen de error adecuado.

Tabla 4. Signos vitales del prototipo

Pacientes	Paciente 1	Paciente 2	Paciente 3
Presión sistólica - Prototipo	100	130	116
Presión diastólica - Prototipo	75	101	73
Frecuencia Cardíaca - Prototipo	89	73	124
Temperatura - Prototipo	36	36	36

Tabla 5. Signos vitales del monitor multiparámetros

Pacientes	Paciente 1	Paciente 2	Paciente 3
Presión sistólica - Monitor	107	143	121
Presión diastólica - Monitor	67	99	93
Frecuencia Cardíaca - Monitor	89	83	130
Temperatura - Monitor	36.5	36.7	36.1

Cabe recalcar que las mediciones fueron realizadas en ambos brazos al mismo tiempo, y generalmente puede existir un margen de error aceptable de 10 mmHg entre un brazo y el otro.



Figura 41. Paciente 1 signos vitales – Prototipo



Figura 42. Paciente 1 signos vitales - Monitor



Figura 43. Paciente 3 signos vitales - Prototipo y Monitor Multiparámetros

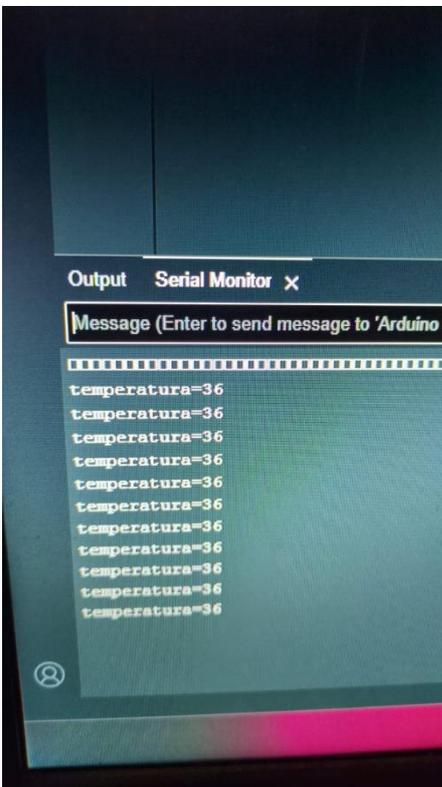


Figura 44. Paciente 3 Temperatura – Prototipo



Figura 45. Paciente 3 Temperatura - Termómetro infrarrojo

CONCLUSIONES

Se lograron identificar y recopilar todos los datos esenciales necesarios para el registro de pacientes en el sistema automático de triaje. Esta recopilación de datos garantiza un proceso de triaje más eficiente y organizado, permitiendo a los profesionales de la salud acceder rápidamente a la información crítica del paciente. La correcta identificación de estos datos contribuye a mejorar la calidad de atención en los centros de salud.

Se desarrolló con éxito un sistema de clasificación de gravedad que permite evaluar de manera precisa los síntomas de los pacientes para su correcta categorización en el proceso de triaje. Este sistema facilita una respuesta más rápida y adecuada del personal médico ante los diferentes niveles de urgencia, mejorando así la priorización en la atención y optimizando el uso de los recursos en emergencias.

Así mismo, mediante este proyecto ha sido posible recalcar la importancia de los sistemas de triaje no solo en el área de emergencias en hospitales grandes, sino también en los pequeños centros de salud que manejan urgencia, puesto que es desde ese punto donde las emergencias pueden ser evitadas.

Por otro lado, se elaboró un sistema automático que cumple con los criterios de ser intuitivo y fácil de usar, permitiendo que los pacientes registren su información de manera rápida y precisa. Este enfoque ha reducido significativamente los tiempos de registro y ha minimizado los errores asociados con la entrada de datos manual, mejorando la experiencia del paciente y la eficiencia del proceso de triaje.

Un equipo capaz de automatizar y agilizar el proceso de triaje representa un gran soporte en la atención de urgencias, ya que agiliza al personal médico en su capacidad de tratar las urgencias médicas. Se diseñó un prototipo funcional de un sistema automático

para la toma de signos vitales, que ha mostrado ser efectivo y preciso en su operación. Este prototipo puede ser fácilmente implementado en entornos clínicos, proporcionando datos vitales en tiempo real y apoyando la toma de decisiones médicas rápidas y bien informadas. El éxito del diseño del prototipo abre la puerta para su futura producción a gran escala y su integración en sistemas de salud a nivel nacional.

RECOMENDACIONES

Se recomienda conocer de antemano los sensores con los que se pretende trabajar, esto disminuye los costos y agiliza el proceso de trabajo para la elaboración de cualquier prototipo.

Para obtener mejores datos de los sensores se recomienda realizar las adaptaciones necesarias, de esta forma es posible disminuir el error durante la obtención de información, aún más en proceso como lo es la toma de signos vitales.

Se recomienda realizar una valoración previa al momento de decidir utilizar una pantalla en específica, a pesar de que el programa de edición Nextion Editor facilita el diseño visual de la pantalla, no se encuentra información clara y precisa sobre el lenguaje de programación utilizado por la pantalla, ni los comandos necesarios para un óptimo funcionamiento. Así mismo, se recomienda utilizar diversas herramientas de diseño gráfico para lograr un acabado intuitivo y agradable a la vista.

BIBLIOGRAFÍA

Aguirre A, Baily E, Zusy ML, Corpas A, Llimona A. Enciclopedia de la Enfermería. Vol. 3. Barcelona: Editorial Océano; 1998.

Ajani K. (2012). Triage; a literature review of key concepts. JPMA. The Journal of the Pakistan Medical Association, 62(5), 487–489.

Ali, F. (2018). An overview of the history and development of triage. EC Emergency Medicine and Critical Care 2.3 American University of Antigua, College of Medicine.

Arbour, C., & Gélinas, C. (2010). Are vital signs valid indicators for the assessment of pain in postoperative cardiac surgery ICU adults?. Intensive & critical care nursing, 26(2), 83–90. <https://doi.org/10.1016/j.iccn.2009.11.003>

Arduino. (n.d.). *Arduino UNO Rev3*. <https://store.arduino.cc/products/arduino-uno-rev3>

Arduino. (n.d.). *Arduino Nano*. <https://store.arduino.cc/products/arduino-nano>

Arduino. (n.d.). *Arduino Mega 2560 Rev3*.
<https://store.arduino.cc/products/arduino-mega-2560-rev3>

Arduino Core Team. (2024). *Wire Library for Arduino*. Arduino.
<https://www.arduino.cc/en/Reference/Wire>

Arduino Mega 2560 Rev3 — Arduino Official Store. (2024).
<https://store.arduino.cc/products/arduino-mega-2560-rev3?srsltid=AfmBOopkCMBcneFKy3KzkeY4tZpQYEy3KwQd4vHkYedI9RrgD6s>

Arduino Nano — Arduino Official Store. (2024).
<https://store.arduino.cc/products/arduino-nano>

Arduino Uno Rev3 — Arduino Official Store. (2024).

<https://store.arduino.cc/products/arduino-uno-rev3>

Ávila-Cárdenas, L., & De la Rosa-Ferrera, J. M. (2022). Triage en el servicio de emergencia en el Hospital del Sur de Esmeraldas, Ecuador [Triage in the emergency service at the Hospital del Sur de Esmeraldas, Ecuador]. *Archivos Médicos de Camagüey*, 26. <https://revistaamc.sld.cu/index.php/amc/issue/view/196>

Ávila Moreno, O. D., Vergara Centeno, J. L., & Franco Coffré, J. A. (2022). Un desafío sanitario en la gestión del servicio de medicina crítica de un hospital del Ecuador: Vivencias en la pandemia COVID-19. *Boletín de Malariología y Salud Ambiental*, 62(1). <https://doi.org/10.52808/bmsa.7e6.621.001>

Azeredo, T. R., Guedes, H. M., Rebelo de Almeida, R. A., Chianca, T. C., & Martins, J. C. (2015). Efficacy of the Manchester Triage System: a systematic review. *International emergency nursing*, 23(2), 47–52. <https://doi.org/10.1016/j.ienj.2014.06.001>

Bersosa Webster, A. C. (2017). Propuesta de implementación de un consultorio de triaje como estrategia para mejorar la calidad y cobertura ante la demanda de pacientes de un subcentro de salud de la parroquia Hermano Miguel, Cuenca - Ecuador [*Trabajo de titulación, Universidad*]. Facultad de Posgrados.

Burton, M. (2015). *DallasTemperature Library for Arduino*. <https://github.com/milesburton/Arduino-Temperature-Control-Library>

Cervantes, J., Reyes, J., & Bracho, G. (2017). DESARROLLO DE UN PROTOTIPO PARA SU USO COMO HERRAMIENTA DE APOYO AL PERSONAL MÉDICO EN LA GRADACIÓN DE TRIAJE PARA PACIENTES EN LOS SERVICIOS DE URGENCIAS BASADO EN SUS SIGNOS VITALES. *Investigaciones Andina*, 19(34),1829-1843. <https://www.redalyc.org/articulo.oa?id=239057355006>

Chérrez-Anguizaca, J. E., & León-Micheli, E. X. (2021). La aplicación del Triage en la prestación del servicio de salud en el Ecuador. *Revista Interdisciplinaria de Humanidades, Educación, Ciencia y Tecnología*, 7(3), Edición Especial III. Universidad Nacional Experimental Francisco de Miranda (UNEFM).
<https://doi.org/10.35381/cm.v7i3.572>

Christ, M., Grossmann, F., Winter, D., Bingisser, R., & Platz, E. (2010). Modern triage in the emergency department. *Deutsches Arzteblatt international*, 107(50), 892–898. <https://doi.org/10.3238/arztebl.2010.0892>

Dallas Semiconductor. (2004). *DS18B20: Digital thermometer*. Retrieved from <https://www.maximintegrated.com/en/products/sensors/DS18B20.html>

Evans, D., Hodgkinson, B., & Berry, J. (2001). Vital signs in hospital patients: a systematic review. *International journal of nursing studies*, 38(6), 643–650.
[https://doi.org/10.1016/s0020-7489\(00\)00119-x](https://doi.org/10.1016/s0020-7489(00)00119-x)

González, J. V., Arenas, O. A. V., & González, V. V. (2012). Semiología de los signos vitales: Una mirada novedosa a un problema vigente:/Vitals sign semiology: the new look to an actual problem. *Archivos de Medicina (Manizales)*, 12(2), 221–240.
<https://doi.org/10.30554/ARCHMED.12.2.10.2012>

Lähdet, E. F., Suserud, B. O., Jonsson, A., & Lundberg, L. (2009). Analysis of triage worldwide. *Emergency nurse: the journal of the RCN Accident and Emergency Nursing Association*, 17(4), 16–19. <https://doi.org/10.7748/en2009.07.17.4.16.c7122>

Maxim Integrated. (n.d.). *MAX30102: Pulse oximeter and heart-rate sensor*. Retrieved from <https://www.maximintegrated.com/en/products/sensors/MAX30102.html>

Mitchell, G. W. (2008). A Brief History of Triage. *Disaster Medicine and Public Health Preparedness*, 2(S1), S4–S7. doi:10.1097/DMP.0b013e3181844d43

Miller, I. (2019, Septiembre 12). What the heck y going on in this Emergency Department. <https://shojiwax.com/2019/09/12/what-the-heck-is-going-on-in-this-emergency-department/>

Narváz Jaramillo, M., Nazate Chuga, Z., Pozo Hernández, C., & Tavera Lits, R. (2022). Análisis multicriterio en el ámbito sanitario: Selección del sistema de triaje más adecuado para las unidades de atención de urgencias en Ecuador. *Revista Investigación Operacional*, 43(3), 316-324. Universidad Regional Autónoma de los Andes.

Nextion. (n.d.). *Basic Series Introduction*. <https://nextion.tech/basic-series-introduction/>

Penagos, S. P., & De Urgencias, E. 2005. (2005). Control de signos vitales.

Picallo Fernández, B. (2019). Eficacia y confiabilidad del Sistema de Triage Manchester: Revisión bibliográfica [Trabajo de fin de grado, Universidade da Coruña]. Escola Universitaria de Enfermaría A Coruña.
https://ruc.udc.es/dspace/bitstream/handle/2183/25497/PicalloFernandez_Belen_TFG_2019.pdf

Robertson-Steel I. (2006). Evolution of triage systems. *Emergency medicine journal: EMJ*, 23(2), 154–155. <https://doi.org/10.1136/emj.2005.030270>

Seed Studio. (n.d.). *KY-037: Sound sensor module*. Retrieved from <https://www.seedstudio.com/KY-037-Sound-Sensor-Module-p-1743.html>

Soler, W., Gómez Muñoz, M., Bragulat, E., & Álvarez, A..(2010). El triaje: herramienta fundamental en urgencias y emergencias. *Anales del Sistema Sanitario de Navarra*, 33(1), 55-68. https://scielo.isciii.es/scielo.php?script=sci_arttext&pid=S1137-66272010000200008

Studt, J. (2007). *OneWire Library for Arduino*.
https://www.pjrc.com/teensy/td_libs_OneWire.html

Villegas González, J., Villegas Arenas, A., & Villegas González, V. (2012). Signos vitales, presión arterial, temperatura corporal, frecuencia del pulso, frecuencia respiratoria, oximetría. *Arch Med (Manizales)*, 12(2), 221–240

What is Arduino? | Arduino. (2018).
<https://www.arduino.cc/en/Guide/Introduction>

Wuerz, R. C., Travers, D., Gilboy, N., Eitel, D. R., Rosenau, A., & Yazhari, R. (2001). Implementation and refinement of the emergency severity index. *Academic emergency medicine : official journal of the Society for Academic Emergency Medicine*, 8(2), 170–176. <https://doi.org/10.1111/j.1553-2712.2001.tb01283.x>

Gilboy, N., Tanabe, P., Travers, D., & Rosenau, A. M. (2012). *Emergency Severity Index (ESI) handbook: A triage tool for emergency department care, version 4*. Agency for Healthcare Research and Quality.

ANEXOS

Se anexan pruebas realizadas en el centro de salud del guasmo central.



Prueba de frecuencia respiratoria.



Prueba de presión arterial.