

UNIVERSIDAD TÉCNICA DE AMBATO

FACULTAD DE INGENIERÍA EN SISTEMAS, ELECTRÓNICA E INDUSTRIAL

CARRERA DE TELECOMUNICACIONES

Tema:

SISTEMA INTEGRAL DE CONTROL Y SEGUIMIENTO PERSONALIZADO PARA GIMNASIOS, BASADO EN INTELIGENCIA ARTICIAL

Trabajo de titulación modalidad Proyecto de Investigación, presentado previo a la obtención del título de Ingeniero en Telecomunicaciones.

ÁREA: Programación y redes

LÍNEA DE INVESTIGACIÓN: Electrónica, Telecomunicaciones y Sistemas de control.

AUTOR: Darío Armando Guamán Coello

TUTOR: Ing. Víctor Santiago Manzano Villafuerte, Mg.

Ambato - Ecuador

julio – 2025

APROBACIÓN DEL TUTOR

En calidad de tutor del trabajo de titulación con el tema: SISTEMA INTEGRAL DE CONTROL Y SEGUIMIENTO PERSONALIZADO PARA GIMNASIOS, BASADO EN INTELIGENCIA ARTICIAL , desarrollado bajo la modalidad Proyecto de Investigación por el señor Darío Armando Guamán Coello, estudiante de la Carrera de Telecomunicaciones, de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial, de la Universidad Técnica de Ambato, me permito indicar que el estudiante ha sido tutorado durante todo el desarrollo del trabajo hasta su conclusión, de acuerdo a lo dispuesto en el Artículo 17 del Reglamento para la Titulación de Grado en la Universidad Técnica de Ambato y el numeral 6.3 del instructivo del reglamento referido.

Ambato, julio 2025.

Firmado electrónicamente por:
VICTOR SANTIAGO MANZANO VILLAFUERTE
Validar únicamente con FirmaEC

Ing. Víctor Santiago Manzano Villafuerte, Mg.
TUTOR

AUTORÍA

El presente trabajo de titulación con el tema: SISTEMA INTEGRAL DE CONTROL Y SEGUIMIENTO PERSONALIZADO PARA GIMNASIOS, BASADO EN INTELIGENCIA ARTICIAL es absolutamente original, auténtico y personal y ha observado los preceptos establecidos en la Disposición General Quinta del Reglamento para la Titulación de Grado en la Universidad Técnica de Ambato. En tal virtud, el contenido, efectos legales y académicos que se desprenden del mismo son de exclusiva responsabilidad del autor.

Ambato, julio 2025.



Darío Armando Guamán Coello

C.C 1805288261

AUTOR

DERECHOS DE AUTOR

Autorizo a la Universidad Técnica de Ambato para que reproduzca total o parcialmente este trabajo de titulación dentro de las regulaciones legales e institucionales correspondientes. Además, cedo todos mis derechos de autor a favor de la institución con el propósito de su difusión pública, por lo tanto, autorizo su publicación en el repositorio virtual institucional como un documento disponible para la lectura y uso con fines académicos e investigativos de acuerdo con la Disposición General Cuarta del Reglamento para la Titulación de Grado en la Universidad Técnica de Ambato.

Ambato, julio 2025.



Darío Armando Guamán Coello

C.C. 1805288261

AUTOR

APROBACIÓN DEL TRIBUNAL DE GRADO

En calidad de par calificador del informe final del trabajo de titulación presentado por el señor Darío Armando Guamán Coello, estudiante de la Carrera de Telecomunicaciones, de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial, bajo la Modalidad Proyecto de Investigación, titulado SISTEMA INTEGRAL DE CONTROL Y SEGUIMIENTO PERSONALIZADO PARA GIMNASIOS, BASADO EN INTELIGENCIA ARTICIAL , nos permitimos informar que el trabajo ha sido revisado y calificado de acuerdo al Artículo 19 del Reglamento para la Titulación de Grado en la Universidad Técnica de Ambato y el numeral 6.4 del instructivo del reglamento referido. Para cuya constancia suscribimos, conjuntamente con la señora Presidente del Tribunal.

Ambato, julio 2025.



Ing. Franklin Oswaldo Mayorga Mayorga, Mg.
PRESIDENTE DEL TRIBUNAL



Ing. Juan Pablo Pallo Noroña, Mg.
PROFESOR CALIFICADOR



Dr. Carlos Diego Gordon Gallegos.
PROFESOR CALIFICADOR

DEDICATORIA

El presente proyecto va dedicado mi familia, quienes han sido mi pilar fundamental a lo largo de toda mi etapa universitaria, su apoyo y fe en mi han sido esenciales en este largo proceso.

A mi madre y hermanos ejemplo de fortaleza y resiliencia, les agradezco por su respaldo incondicional y por enseñarme que con esfuerzo y perseverancia todo es posible.

Mis docentes y padrinos que han sido unos padres ajenos para mí, pero sus consejos de vida y experiencia me motivaron a terminar mi carrera universitaria.

A mis amigos por su compañía, aliento y motivación en los momentos difíciles, gracias por caminar conmigo hasta el final de esta meta que hoy es una realidad.

AGRADECIMIENTO

En primer lugar, quiero agradecer a Dios por haberme cuidado en cada paso que di a lo largo de mi vida universitaria.

A mi tutor, docente y amigo, Ing. Víctor Manzano, por compartir conmigo su vasto conocimiento, sabiduría y, sobre todo, su paciencia durante todo el proceso de titulación. Su guía y acompañamiento, tanto en este proyecto como en las asignaturas impartidas, han sido fundamentales para mi formación académica y el desarrollo de este trabajo.

Agradezco infinitamente a mi madre doña Carmen como le digo de cariño su apoyo incondicional, sus palabras de aliento y su fe en mi en todo momento.

A mis hermanos: Cristian, Tannia, Roberto, Luis, Daniel, Stalin y Alex, su apoyo infalible en mi me dieron más fuerza para seguir y conseguir esta meta.

A mi cuñado y cuñadas siempre dándome una palabra de aliento ante cualquier circunstancia espero ser un ejemplo para mis sobrinos la siguiente generación

A mis amigos: Jorge, Fernando, Juan, Jhon, Santiago, Eve, Nathy, Kevin, Jonathan, Carlos, Erick, Danna y Daysi por ser ese brazo de apoyo en momentos difíciles los llevare siempre en mi corazón.

ÍNDICE GENERAL DE CONTENIDO

PORTADA ..... **i**

APROBACIÓN DEL TUTOR ..... **ii**

AUTORÍA ..... **iii**

DERECHOS DE AUTOR ..... **iv**

APROBACIÓN DEL TRIBUNAL DE GRADO ..... **v**

DEDICATORIA ..... **vi**

AGRADECIMIENTO ..... **vii**

ÍNDICE GENERAL DE CONTENIDO ..... **viii**

ÍNDICE DE TABLAS ..... **xii**

ÍNDICE DE FIGURAS ..... **xiii**

ÍNDICE DE ANEXOS ..... **xvi**

RESUMEN EJECUTIVO ..... **xviii**

ABSTRACT ..... **xix**

CAPÍTULO I. MARCO TEÓRICO **1**

1.1 Tema de investigación 1

1.1.1 Planteamiento del problema 1

1.2 Antecedentes investigativos 3

1.3 Fundamentación teórica 6

1.3.1 Inteligencia artificial 6

1.3.2 Métodos de aprendizaje 9

1.3.3 Modelos de detección 10

1.3.4 Softwares de etiquetado 13

1.3.5 Microcomputador 14

1.3.6 Plataforma de entrenamiento 17

1.3.7 Software de programación 18

1.3.8 Software para el desarrollo de aplicaciones 21

1.3.9 Sensores 21

1.3.10 Cámara Digital 23

1.3.11 Control de acceso 24

1.4 Objetivos 24

1.4.1 Objetivo general 24

1.4.2 Objetivos específicos 24

CAPÍTULO II. METODOLOGÍA 25

2.1 Materiales 25

2.2 Métodos 26

2.2.1 Modalidad de la investigación 26

2.2.2 Población y muestra 27

2.2.3 Recolección de información 27

2.2.4 Procesamiento y análisis de datos 28

CAPÍTULO III. RESULTADOS Y DISCUSIÓN 30

3.1 Desarrollo de la propuesta 30

3.1.1 Definición de los elementos para la propuesta 30

3.1.2 Elementos por detectar dentro del sistema 31

3.1.3 Esquema general del proyecto 34

3.1.4 Selección de tecnologías.....37

3.1.5 Descripción del sistema de control de acceso.....40

3.1.6 Entrenamiento del sistema 41

3.1.7 Detección de rostros mediante imágenes de video.....43

3.1.8 Descripción de la aplicación de seguimiento personalizado 50

3.1.9 Estructura de la aplicación de seguimiento personalizado 50

3.1.10 Descripción del dispositivo portátil.....53

3.1.11 Estructura del dispositivo portátil 54

3.2 Resultados.....55

3.2.1 Resultados de entrenamiento 56

3.2.2 Pruebas de funcionamiento 65

3.2.3 Prototipo final 74

3.2 Costos 75

3.3.1 Costos de insumos del proyecto 76

3.2.4 Costos del desarrollo técnico 76

3.2.5 Costo total 77

CAPÍTULO IV. CONCLUSIONES Y RECOMENDACIONES 78

4.1 Conclusiones 78

4.2 Recomendaciones 79

REFERENCIAS BIBLIOGRÁFICAS 81

ANEXOS 85

ÍNDICE DE TABLAS

Tabla 1 Modelos de detección de objetos 11

Tabla 2 Evolución de YOLO a través de los años 12

Tabla 3. Características de microcomputadores 16

Tabla 4 Plataformas de entrenamiento 18

Tabla 5 Características de los softwares de programación 20

Tabla 6 Comparación de software de desarrollo de aplicaciones 21

Tabla 7 Características técnicas de la cámara 23

Tabla 8 Criterios de recolección de información.....28

Tabla 9 Métodos de ingreso a centros de entrenamiento.....31

Tabla 10 Tipos y clasificación de rostros 32

Tabla 11 Actividades físicas en un centro de entrenamiento 34

Tabla 12 Parámetros de entrenamiento.....41

Tabla 13 Cálculos de métricas de los usuarios.....49

Tabla 14 Predicción de resultados obtenidos 56

Tabla 15 Matriz de confusión normalizada.....64

Tabla 16 Costos de insumos del proyecto.....76

Tabla 17 Descripción de un salario de un Ingeniero en Telecomunicaciones 77

ÍNDICE DE FIGURAS

Figura 1 Inteligencia artificial 6

Figura 2 Diagrama del procesamiento de imágenes 7

Figura 3 Capas del modelo de detección.....11

Figura 4 NVIDIA Jetson Nano.....15

Figura 5 Coral Dev Board 16

Figura 6 Clasificación de los sensores 22

Figura 7 Esquema general del sistema.....35

Figura 8 Sistema de Control.....40

Figura 9 Instalación de Yolo 42

Figura 10 Parámetros del Dataset.....42

Figura 11 Sistema a detector de rostros 44

Figura 12 Librerías utilizadas Python 45

Figura 13 Carga del modelo 45

Figura 14 Algoritmo de detección de rostros 46

Figura 15 Registro de rostros 47

Figura 16 Codificación de reproducción de audio.....47

Figura 17 Registro de usuarios.....48

Figura 18 Almacenamiento de los datos biométricos.....49

Figura 19 Importación de librerías 51

Figura 20 Primera ventana de navegación 51

Figura 21 Ventana de navegación.....52

Figura 22 Broker MQTT.....52

Figura 23 Protocolo MQTT.....53

Figura 24 Diagrama esquemático del sistema portátil.....55

Figura 25 Matriz de confusión 57

Figura 26 Métrica de precisión.....58

Figura 27 Curva de confianza -F1 58

Figura 28 Métricas recall 59

Figura 29 Curva de sensibilidad 60

Figura 30 Métrica mAP_50.....61

Figura 31 Métrica mAP_50:95.....62

Figura 32 Resultados generales.....63

Figura 33 Interfaz principal del control de acceso.....66

Figura 34 Codificación de registro 66

Figura 35 Interfaz de usuario registrado 67

Figura 36 Usuario no registrado.....68

Figura 37 Dispositivo portátil.....68

Figura 38 Lectura del sensor ECG.....69

Figura 39 Almacenamiento de datos 70

Figura 40 Icono de la aplicación.....71

Figura 41 Inicio de sesión 71

Figura 42 Menú Principal de la aplicación.....72

Figura 43 Progreso registrado en la interfaz de la app.....73

Figura 44 Recomendación de la app y el botón salir.....74

Figura 45 Prototipo final.....75

ÍNDICE DE ANEXOS

Anexo A. Especificaciones técnicas de la NVIDIA Jetson Nano 85

Anexo 2 Especificaciones técnicas de la NVIDIA Jetson 85

Anexo 3 Especificaciones tecnicas de la ESP32 86

Anexo 4 Especificaciones técnicas de la cámara Dahua.....87

Anexo 5 Diagrama esquemático de YOLOv8.....88

Anexo 6 Diagrama esquemático de la placa 89

Anexo 7 Visión 3D de la placa. 90

Anexo 8 Recomendaciones emitidas por la app al usuario. 91

Anexo 9 Recordatorio emitido por la app. 92

Anexo 10 Flutter con Android Studio.....93

Anexo 11 Características del prototipo final.....94

Anexo 12 Hardware y Software 95

Anexo 13 Validación de Sistema.....96

Anexo 14 Código de ejecución.....106

RESUMEN EJECUTIVO

El crecimiento sostenido de centros de entrenamiento en Ecuador ha motivado el diseño de un sistema inteligente de control y seguimiento personalizado para usuarios de gimnasio. Este sistema integra una placa ESP32 como núcleo de procesamiento y comunicación inalámbrica, junto con un sensor ECG encargado de capturar señales cardíacas durante la actividad física. La alimentación se gestiona mediante una batería Li-Po de 7.4 V, estabilizada por un regulador LM7805 para garantizar el suministro constante a los componentes. Paralelamente, se incorpora una cámara Dahua HC4 que permite realizar la captura de video en tiempo real para validar el ingreso del usuario mediante reconocimiento facial, lo cual garantiza un acceso seguro y automatizado.

El procesamiento avanzado de imágenes se ejecuta en una NVIDIA Jetson Nano, seleccionada por su alto rendimiento en tareas de visión por computadora. Los datos recolectados son transmitidos vía Wi-Fi hacia una aplicación móvil desarrollada en Flutter, que permite al usuario visualizar métricas personalizadas, registrar rutinas y acceder a recomendaciones basadas en su desempeño físico. Durante el desarrollo, se utilizó Google Colab como plataforma de entrenamiento para el modelo basado en YOLOv8, y Roboflow para la gestión y etiquetado de datos visuales.

Esta solución tecnológica representa un avance en la automatización y personalización de entornos deportivos, ofreciendo una experiencia inteligente y adaptativa que mejora el rendimiento, la seguridad y la motivación de los usuarios. El sistema se plantea como una herramienta innovadora en el contexto del bienestar y acondicionamiento físico basado en datos.

Palabras clave: Centros de entrenamiento, visión por computadora, reconocimiento facial, YOLOv8, inteligencia artificial.

ABSTRACT

The sustained growth of fitness centers in Ecuador has motivated the design of a customized intelligent control and monitoring system for gym users. This system integrates an ESP32 board as the processing and wireless communication core, along with an ECG sensor responsible for capturing cardiac signals during physical activity. Power is managed by a 7.4V Li-Po battery, stabilized by an LM7805 regulator to ensure a constant supply to the components. In addition, a Dahua HC4 camera is incorporated, enabling real-time video capture to validate user entry through facial recognition, ensuring secure and automated access.

Advanced image processing runs on an NVIDIA Jetson Nano, selected for its high performance in computer vision tasks. The collected data is transmitted via Wi-Fi to a mobile app developed in Flutter, which allows the user to view personalized metrics, log routines, and access recommendations based on their physical performance. During development, Google Colab was used as the training platform for the YOLOv8-based model, and Roboflow was used to manage and label the visual data.

This technological solution represents a breakthrough in the automation and personalization of sports environments, offering an intelligent and adaptive experience that improves user performance, safety, and motivation. The system is presented as an innovative tool in the context of data-driven wellness and fitness.

Keywords: Training centers, computer vision, facial recognition, YOLOv8, artificial intelligence.

CAPÍTULO I. MARCO TEÓRICO

1.1 Tema de investigación

SISTEMA INTEGRAL DE CONTROL Y SEGUIMIENTO PERSONALIZADO PARA GIMNASIOS, BASADO EN INTELIGENCIA ARTIFICIAL.

1.1.1 Planteamiento del problema

La creciente digitalización de los servicios ha impulsado la necesidad de desarrollar sistemas avanzados que garanticen la seguridad, la eficiencia y una experiencia optimizada para los usuarios. El control de ingreso ha dependido históricamente de métodos tradicionales, como el uso de tarjetas de acceso o registros manuales, que presentan vulnerabilidades significativas, incluyendo el riesgo de fraude y accesos no autorizados. Esta falta de un sistema automatizado limita la capacidad de supervisión en tiempo real, comprometiendo tanto la seguridad del recinto como la experiencia general de los usuarios.

Para enfrentar estos desafíos y mejorar la operatividad del gimnasio, resulta fundamental implementar un sistema de control integral que, mediante el uso de inteligencia artificial, automatice el acceso a las instalaciones y eleve la seguridad a un nuevo nivel. Este sistema no solo incluirá mecanismos de identificación biométrica, como el reconocimiento facial, sino que también permitirá monitorear patrones de acceso y detectar anomalías en tiempo real. Además, integrará un módulo de seguimiento personalizado del progreso físico de los usuarios, registrando parámetros como peso, musculatura y otras métricas clave. A través del análisis de estos datos, se proporcionarán recomendaciones personalizadas que contribuirán a optimizar el rendimiento físico de cada usuario, mejorando su experiencia y fomentando la adherencia a sus objetivos de entrenamiento.

El uso de la biometría se ha expandido globalmente, transformando el control de acceso en gimnasios, aeropuertos, empresas y centros educativos. Tecnologías como el reconocimiento facial y los lectores de huellas agilizan procesos y refuerzan la

seguridad al verificar identidades en tiempo real. A medida que la digitalización avanza, más países adoptan estos sistemas para reducir accesos no autorizados y mejorar la experiencia del usuario. La biometría se consolida como una herramienta clave en la modernización de infraestructuras a nivel mundial [1].

En la región, la implementación de tecnologías biométricas, como el reconocimiento facial, ha crecido significativamente, impulsada por la búsqueda de soluciones para la seguridad pública y la vigilancia. Sin embargo, este crecimiento ha estado acompañado de preocupaciones sobre la protección de datos personales y el respeto a los derechos humanos. Por ejemplo, empresas como Dahua y Hikvision, que han sido cuestionadas internacionalmente por su implicación en violaciones de derechos humanos, están activas en países como México, lo que resalta la necesidad de un marco regulatorio robusto para el uso de estas tecnologías [2].

En Ecuador, el uso de datos biométricos ha generado controversia, especialmente con la llegada de empresas como WorldCoin, que escanean iris a cambio de criptomonedas. Este proceso ha levantado preocupaciones sobre la legalidad de la recopilación de datos sin el consentimiento adecuado. A pesar de las advertencias de entidades como Arcotel y el Banco Central, la actividad de WorldCoin continuó siendo un tema de debate. El uso de tecnologías de reconocimiento biométrico por parte de empresas y entidades, como Corporación Favorita y el SNNA, resalta la creciente tendencia hacia la recolección de datos. Sin embargo, la falta de regulación adecuada deja a los ciudadanos vulnerables [3].

La información y la gestión de datos son aspectos esenciales tanto en el ámbito industrial como en entornos deportivos, donde el manejo adecuado de los datos de los usuarios es crucial para garantizar la seguridad y eficiencia de los servicios ofrecidos. En la actualidad, la era tecnológica ha facilitado el desarrollo de métodos de comunicación más rápidos y eficientes, potenciados por redes de internet y equipos móviles. La inversión en sistemas avanzados de gestión de datos es fundamental no solo para proteger la información personal de los abonados, como sus datos biométricos, sino también para realizar un seguimiento personalizado del progreso físico de cada usuario. Estos datos, como peso y musculatura, serán procesados para proporcionar recomendaciones personalizadas, optimizando la experiencia de

entrenamiento. Una mala administración de estos datos podría comprometer no solo la seguridad, sino también la efectividad del sistema y la satisfacción de los usuarios.

El uso de tecnologías avanzadas para manejar estos datos, especialmente los biométricos, requiere una atención especial en su protección debido a su naturaleza sensible. Garantizar la confidencialidad, integridad y disponibilidad de la información es esencial para el éxito del sistema propuesto, ya que cualquier fallo en la seguridad podría poner en riesgo tanto la privacidad de los usuarios como la eficacia del sistema de control de acceso y seguimiento físico. Esto resalta la necesidad de establecer un marco ético y legal que regule el tratamiento de los datos personales en el contexto de control de acceso y monitoreo del progreso físico de los usuarios.

1.2 Antecedentes investigativos

A través de la revisión de artículos científicos y trabajos de titulación disponibles en los repositorios principales de diversas universidades, se han identificado los siguientes antecedentes.

En 2023, Harrys Johao Pimentel Viera desarrolló la investigación titulada "Aplicación híbrida para el seguimiento progresivo de actividades físicas utilizando el framework Flutter" en la Universidad Nacional de Chimborazo. El proyecto tuvo como objetivo principal la creación de una aplicación móvil que permitiera registrar y monitorear el progreso físico de los usuarios del gimnasio American Mega Gym, ubicado en Riobamba. Esta aplicación optimizó la gestión de rutinas de ejercicio y recomendaciones alimentarias, centralizando la información en una plataforma digital accesible desde dispositivos móviles. La investigación empleó la metodología ágil SCRUM, permitiendo desarrollar la aplicación en fases iterativas que incorporaron mejoras como autenticación de usuarios, gestión de perfiles, almacenamiento de datos en Firebase y generación de reportes basados en la norma ISO/IEC 25022 para evaluar la satisfacción del usuario. Diseñada para gimnasios y centros deportivos, la aplicación AMG Fit proporcionó una solución eficiente para el control del entrenamiento de los usuarios, y tras su implementación y evaluación, obtuvo un nivel de satisfacción promedio del 96%, consolidándose como una herramienta tecnológica innovadora para la optimización del rendimiento físico [4].

En 2023, Gabriel Esaú Can López desarrolló la tesis "Diseño de un Sistema Biométrico para el Control de Asistencia del Personal Administrativo de la Escuela Superior de Arte de la Universidad de San Carlos de Guatemala", con el objetivo de automatizar el registro de asistencia mediante sensores de huellas dactilares y tecnología Arduino. El sistema integró un sensor óptico Adafruit, almacenamiento de datos en SQLite, una interfaz gráfica en Python y una cerradura eléctrica controlada por reconocimiento biométrico. Utilizando Arduino Mega 2560 como microcontrolador y simulaciones electrónicas en Proteus 8, se validó el diseño antes de la implementación. Los resultados mostraron una precisión del 98% en la identificación de huellas, con un tiempo de respuesta de 2.1 segundos por usuario, reduciendo en un 75% el tiempo de registro manual y disminuyendo errores administrativos en un 40%. Además, el sistema permitió detectar y reducir ausencias no justificadas en un 30% durante el primer mes de prueba, mejorando significativamente la eficiencia y transparencia en el control de asistencia de la institución [5].

En 2023, Jean Elías Manuel Reyes Campos, Christian Stephano Castañeda Rodríguez, Luis Daniel Alva Luján y Alberto Carlos Mendoza de los Santos desarrollaron el artículo titulado "Sistema de reconocimiento facial para el control de accesos mediante Inteligencia Artificial" en la Universidad La Salle, Perú. El objetivo fue crear un sistema de control de acceso basado en reconocimiento facial, utilizando redes neuronales convolucionales (CNN) como modelo de predicción y Python junto con librerías como Numpy, OpenCV y Imutils para su implementación. El sistema alcanzó una precisión del 88% en la identificación de individuos, utilizando un dataset de 450 imágenes por persona. Los autores concluyeron que el sistema mejora su eficacia al incrementar el tamaño de los datasets. El desarrollo se centró en la optimización de la seguridad y el acceso, permitiendo la autenticación biométrica de los usuarios de forma rápida y precisa, lo que refuerza la protección en instalaciones que requieren control riguroso [6].

En el año 2024, Francisco David Ahumada Reyes desarrolló la tesis titulada "Sistema de Control y Administración de un Centro de Entrenamiento (Gimnasio) desarrollado en .NET" en el Instituto Tecnológico Superior de Álamo Temapache. Esta investigación se centró en la creación de una aplicación integral que optimiza la gestión de servicios en gimnasios, abordando desafíos administrativos y operativos. El

sistema fue diseñado para resolver problemas de organización, tales como la gestión de membresías, programación de clases y control de clientes. Desarrollado en C# y la plataforma .NET 6, el software ofrece funcionalidades como la generación de informes y el seguimiento de usuarios, con el objetivo de mejorar la experiencia tanto de administradores como de clientes. El proyecto siguió una metodología de desarrollo incremental, permitiendo una implementación por fases y facilitando ajustes durante el proceso. El sistema fue probado en el gimnasio "Punto", logrando optimizar procesos y mejorar la eficiencia en la gestión diaria. El análisis de resultados demostró que la aplicación no solo cumplió con los objetivos propuestos, sino que también contribuyó a reducir errores operativos y mejorar la toma de decisiones basada en datos [7].

En 2024, Cristian Camilo Betancourt Rodríguez desarrolló la tesis titulada "Diseño de un sistema de automatización que permita la apertura de puertas de acceso para el ingreso de aeronaves en el hangar de la empresa CIAC LTDA" en la Universidad Cooperativa de Colombia. El proyecto tuvo como objetivo optimizar el control de acceso a hangares mediante la implementación de un sistema automatizado basado en tecnologías IoT (Internet de las Cosas) y comunicación por radiofrecuencia. El sistema diseñado permite la apertura y cierre remoto de puertas de hangar a través de una aplicación móvil, integrando módulos de control, sensores, relés y microcontroladores (ESP32). La metodología incluyó el análisis de requerimientos, diseño de circuitos electrónicos, desarrollo de prototipos y pruebas en entornos controlados. Se logró mejorar la eficiencia operativa del hangar, reduciendo los tiempos de espera para el ingreso de aeronaves y optimizando la seguridad. El proyecto se validó con pruebas piloto que demostraron un funcionamiento estable del sistema, destacando una reducción del 30% en el tiempo de apertura de puertas y una mejora significativa en el control de acceso del personal autorizado. Los resultados respaldaron la viabilidad de implementar sistemas IoT en infraestructuras críticas como hangares, optimizando recursos y mejorando la competitividad de la empresa CIAC LTDA en el sector aeroespacial [8].

1.3 Fundamentación teórica

1.3.1 Inteligencia artificial

La Inteligencia Artificial (IA) es la capacidad que tienen las máquinas como un ordenador o un robot para realizar tareas que normalmente requieren de la inteligencia humana. Estas tareas incluyen calcular, memorizar datos, comprender significados e incluso jugar al ajedrez. Los principales objetivos de la inteligencia artificial se centran en imitar la actividad cognitiva humana. Actualmente, la tecnología ha avanzado lo suficiente como para replicar procesos como el aprendizaje, el razonamiento, la percepción y la resolución de problemas. Además, gracias a la Inteligencia Artificial Generativa, también es posible desarrollar actividades creativas y de razonamiento avanzado [9].



Figura 1 Inteligencia artificial

a. Procesamiento Digital de Imágenes y Señales

El Procesamiento Digital de Señales e Imágenes (DSPI) se enfoca en el análisis, manejo y transformación de señales y datos visuales mediante el uso de técnicas y algoritmos computacionales. Al realizarse de manera digital, permite realizar tareas como filtrado, mejora de calidad, extracción de características y reconocimiento de patrones en señales de audio, imágenes y videos. Las aplicaciones del DSPI son vastas,

abarcando áreas como telecomunicaciones, medicina, procesamiento de imágenes médicas, seguridad y vigilancia, entre otras [10].

De esta manera se puede entender al procesamiento de imágenes y señales como se muestra en la Figura 2 en la cual se observa que pasa por algunas etapas.

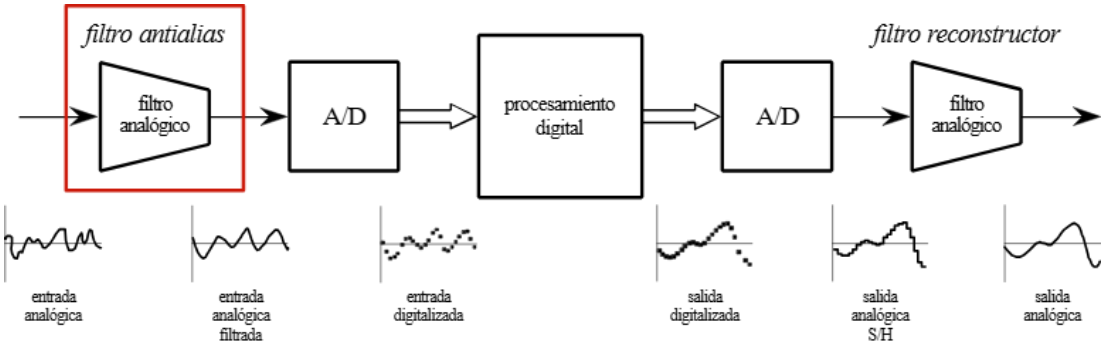


Figura 2 Diagrama del procesamiento de imágenes

b. Machine Learning

El Machine Learning, o Aprendizaje Automático, es una disciplina de la inteligencia artificial que se centra en el desarrollo de algoritmos y modelos computacionales capaces de aprender de datos previos para realizar tareas específicas sin una programación explícita. Estos algoritmos son entrenados utilizando conjuntos de datos para identificar patrones, relaciones y regularidades, lo que les permite generalizar y tomar decisiones o hacer predicciones sobre nuevos datos no vistos anteriormente. El Machine Learning tiene aplicaciones en una amplia gama de campos, desde reconocimiento de voz y visión por computadora hasta análisis de datos y sistemas de recomendación [11].

c. Técnicas de Machine Learning

Las técnicas de aprendizaje automático comprenden un conjunto diverso de herramientas y métodos diseñados para resolver una amplia variedad de problemas en este ámbito. Estas técnicas incluyen algoritmos especializados para tareas como

clasificación, regresión, agrupamiento y detección de anomalías, entre otras. Asimismo, pueden adaptarse a distintos tipos de datos, ya sean estructurados, no estructurados o semiestructurados, y encuentran aplicaciones en múltiples campos, como la medicina, las finanzas, el marketing y la robótica [12].

Supervisado

Esta es la modalidad más frecuente, donde los datos se presentan con etiquetas que guían al sistema para reconocer los patrones específicos que debe identificar [13].

No supervisado

En este enfoque, los datos carecen de etiquetas, lo que exige que la máquina identifique y analice patrones de manera autónoma. Aunque es menos frecuente, su aplicabilidad es más limitada en comparación con otras técnicas [13].

Por refuerzo

Un algoritmo adopta el enfoque de ensayo y error para aprender a cumplir un objetivo, evaluando diversas opciones y recibiendo recompensas o penalizaciones según qué tan bien se alineen sus decisiones con la meta establecida [13].

Supervisado

Esta es la modalidad más frecuente, donde los datos se presentan con etiquetas que guían al sistema para reconocer los patrones específicos que debe identificar [13].

d. Aprendizaje profundo

El Deep Learning permite a los modelos computacionales, compuestos por múltiples capas de procesamiento, adquirir representaciones de datos en distintos niveles de abstracción. Inspirado en la organización jerárquica del cerebro humano, este enfoque capacita a los modelos para identificar características complejas y llevar a cabo tareas avanzadas, como el reconocimiento de imágenes, el análisis de lenguaje natural y la predicción de patrones. Su habilidad para procesar grandes cantidades de datos lo ha posicionado como una tecnología fundamental en campos como la inteligencia artificial, el análisis de datos y la automatización [14].

e. Técnicas de Deep learning

Las técnicas de deep learning son increíblemente complejos y hay diferentes tipos de redes neuronales para abordar problemas o conjuntos de datos específicos. Cada uno tiene sus propias ventajas [15].

CNN

Las redes neuronales convolucionales (CNN) son modelos de inteligencia artificial usados principalmente en visión artificial y clasificación de imágenes. Detectan patrones en imágenes y videos mediante operaciones matemáticas como la multiplicación de matrices. Están compuestas por capas de nodos interconectados que se activan según ciertos umbrales para procesar y transmitir información a través de la red [15].

RNN

Las redes neuronales recurrentes (RNN) se usan para procesar datos secuenciales como lenguaje natural y reconocimiento de voz, aprovechando su capacidad de "memoria" para considerar entradas anteriores. Utilizan retro propagación en el tiempo para aprender de secuencias y predecir resultados futuros. Son útiles en traducción automática, asistentes virtuales y predicciones temporales. Las variantes como LSTM mejoran su rendimiento en dependencias a largo plazo. Sin embargo, enfrentan problemas como la desaparición o explosión de gradientes [15].

1.3.2 Métodos de aprendizaje

En el ámbito del aprendizaje automático, los métodos de aprendizaje abarcan algoritmos supervisados, no supervisados y de refuerzo. Entre las técnicas más comunes se encuentran los árboles de decisión, las redes neuronales, las máquinas de soporte vectorial, así como los métodos de agrupamiento y reducción de dimensionalidad [16].

a. Análisis del comportamiento del usuario

Se centra en entender la forma en como interactúan las personas con productos o servicios digitales, evaluando los patrones y tendencias en sus acciones [16].

Identificación de patrones

Implica la detección de regularidades y las estructuras en grandes volúmenes de datos, facilitando la predicción y comprensión de comportamientos futuros [16].

Algoritmos

Es un conjunto de instrucciones definidas y finitas, diseñadas para resolver un problema específico o realizar una tarea particular [16].

1.3.3 Modelos de detección

La detección de objetos es una pieza clave en la visión por computadora, ya que permite no solo identificar sino también localizar objetos específicos dentro de imágenes o videos. A diferencia de la clasificación de imágenes, que se limita a asignar una etiqueta a la imagen completa, la detección de objetos proporciona información precisa sobre la posición y las dimensiones de cada objeto mediante cuadros delimitadores [17].

A continuación, se presenta en la Figura 3 las capas por la que es sometida una imagen de entrada hasta llegar a la detección final del objeto.

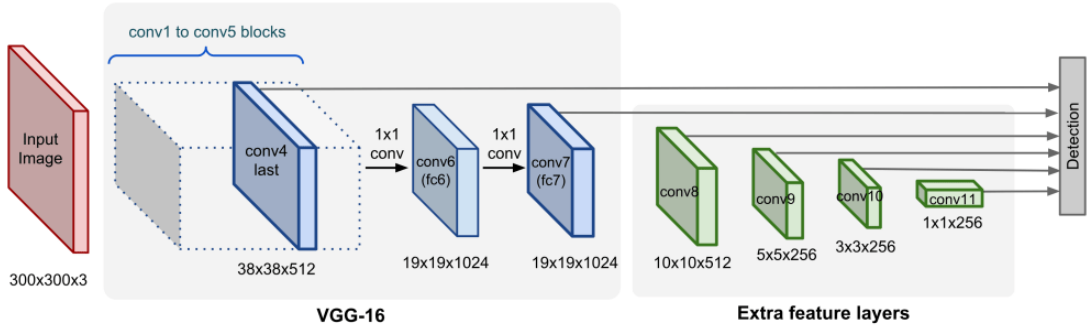


Figura 3 Capas del modelo de detección

En la Tabla 1 se muestra los modelos de detección de objetos con sus características más relevantes.

Tabla 1 Modelos de detección de objetos [17] [18]

Modelo	**Descripción**	**Precisión (mAP)**	**Velocidad FPS**	**Recursos**	**Versiones Existentes**
YOLO (You Only Look Once)	Realiza detección en una sola pasada sobre la imagen, prediciendo múltiples objetos a la vez.	~45 – 55 %	30-60	Bajo (puede correr en GPU y CPU)	YOLOv3, YOLOv4, YOLOv5, YOLOv7, YOLOv8
Faster R-CNN (Region-based Convolutional Neural Networks)	Genera propuestas de regiones y luego clasifica y ajusta las regiones para detectar objetos.	~50 – 60 %	5-7	Alto (requiere GPU potente)	Faster R-CNN (original), Mask R-CNN
SSD (Single Shot Multibox Detector)	Similar a YOLO, pero predice objetos en múltiples escalas, haciendo más eficiente la detección en un solo paso.	~45 – 55 %	20-30	Moderado (mejor en GPU)	SSD (original), SSD Mobilenet, SSD Inception
TensorFlow Object Detection API	Framework flexible para la detección de objetos que permite implementar diferentes arquitecturas (como SSD, Faster R-CNN, etc.).			Moderado a alto	Varios modelos como SSD, Faster R-CNN, EfficientDet, etc.

a. YOLO

YOLO es un modelo de detección de objetos en tiempo real que se basa en redes neuronales convolucionales (CNN). A diferencia de los métodos tradicionales, que segmentan la imagen en regiones o emplean múltiples etapas para identificar objetos, YOLO adopta un enfoque integral. Este modelo procesa la imagen de manera completa en una única pasada, lo que le permite detectar objetos y predecir su ubicación con gran velocidad y precisión [19].

La Tabla 2 presenta una comparación entre las distintas versiones de YOLO, abarcando desde YOLOv3 hasta las versiones más actuales, destacando sus principales características, ventajas y desventajas.

Tabla 2 Evolución de YOLO a través de los años [20]

Versión	**Año**	**Características**	**Precisión (mAP)**	**Velocidad FPS**	**Tamaño del modelo**	**Arquitectura principal**
YOLOv3	2018	Uso de FPN (Feature Pyramid Network). Detección multi-escala.	~57 % (COCO)	$\sim 30 - 50$	~250 MB	Darknet-53
YOLOv4	2020	Implementación optimizada para hardware.	~65.7 % (COCO)	~60	~250 MB	CSPDarknet-53
YOLOv5	2020	Modelo ligero y rápido. Facilidades para entrenar y exportar.	~67 % (COCO)	~140	4-90 MB Según la variante	PyTorch, enfoque modular
YOLOv7	2022	Introducción de "Extended Efficient Layer Aggregation".	~56.8 % (COCO)	~160	~74 MB	E-ELAN, basándose en PyTorch
YOLOv8	2023	Mejora en arquitectura, facilidad de uso con ultralytics.	~57.5 % (COCO)	~170	20-100 MB	Basado en PyTorch
YOLOv9	2024	Modelo mejorado de detección de objetos en tiempo real	~58.3 % (COCO)	~180	25-110 MB	Basado en PyTorch

1.3.4 Softwares de etiquetado

El proceso de etiquetado de imágenes es fundamental para construir conjuntos de datos destinados a algoritmos de detección de objetos (datasets), ya que suministra la información necesaria para que el modelo pueda aprender a identificar y ubicar objetos dentro de imágenes. Esta tarea implica señalar y marcar de forma manual las zonas relevantes en cada imagen, asignándoles una etiqueta que describa el objeto representado [21].

a. LabelImg

Herramienta de código abierto diseñada para el etiquetado manual de imágenes. Destaca por ser ligera, intuitiva y eficiente, lo que la convierte en una opción ideal para generar anotaciones en formatos compatibles con YOLO y Pascal VOC [22].

Principales características [22]:

- Compatibilidad con formatos XML (Pascal VOC) y TXT (YOLO).
- Interfaz gráfica sencilla y funcional.
- Disponible para Windows, macOS y Linux.

b. Roboflow

Plataforma basada en la nube que facilita la gestión integral de datos para proyectos de visión por computadora. Además de incluir herramientas para el etiquetado de imágenes, ofrece capacidades avanzadas de preprocesamiento, organización y exportación en una variedad de formatos [23]:

Principales características [23]:

- Soporte para formatos populares como YOLO, Pascal VOC y COCO.
- Posibilidad de etiquetado colaborativo en imágenes y videos.
- Funciones avanzadas de preprocesamiento, incluyendo ampliación de datos.

c. Labelme

Desarrollada por el MIT, es una herramienta de anotación diseñada para proyectos que requieren segmentación precisa de imágenes. Permite crear etiquetas poligonales que delimitan con exactitud las áreas de interés [24].

Principales características [24]:

- Generación de anotaciones poligonales para segmentación semántica.
- Compatibilidad con formatos como JSON.
- Ideal para proyectos que demandan alta precisión en la definición de los contornos de los objetos.

1.3.5 Microcomputador

Se trata de un equipo compacto que utiliza un solo microprocesador como su unidad principal de procesamiento (CPU). A diferencia de los mainframes o computadoras de gran escala, los microcomputadores son asequibles y adecuados para el uso personal o en pequeñas empresas. Gracias a su bajo costo, tamaño compacto y capacidad para ejecutar tareas complejas, son una excelente opción para actividades diarias como la navegación en internet, la edición de documentos, la programación y el desarrollo de sistemas embebidos [25].

a. NVIDIA Jetson Nano

Es un equipo informático especializado en aplicaciones de Inteligencia Artificial, optimizado para ejecutar modelos previamente entrenados de Deep Learning o Machine Learning enfocados en visión artificial. Cuenta con múltiples puertos de

conexión y una GPU integrada, lo que le permite operar de manera similar a una computadora personal, siendo especialmente adecuado para proyectos de visión computacional. No viene con un sistema operativo preinstalado, dando al usuario la libertad de seleccionar e instalar uno desde una tarjeta SD. Su diseño arquitectónico la convierte en una excelente opción para usos en automatización, domótica y otras áreas [26].

En la Figura 4 se observa a un microcomputador de forma pura sin elementos físicos.



Figura 4 NVIDIA Jetson Nano

b. Coral Dev Board

Creado por Google, este dispositivo está diseñado para correr modelos de aprendizaje automático, siendo especialmente adecuado para TensorFlow y otros entornos de inteligencia artificial. Incorpora un TPU optimizado para tareas de IA, junto con 1 GB de memoria RAM y 8 GB de almacenamiento interno, además de ofrecer conectividad WiFi y Bluetooth. Es compatible con varios entornos de desarrollo, principalmente los de Google, y se distingue por su bajo consumo energético y su capacidad para ejecutar modelos de alto rendimiento, lo que lo convierte en una excelente alternativa para proyectos de inteligencia artificial [27].

En la Figura 5 se observa su forma estética de la Coral Dev Board.



Figura 5 Coral Dev Board

c. Comparativa de microcomputadoras

En la Tabla 3 se expone las diferentes características de cada microcomputador que son efectivos para algoritmos de inteligencia artificial.

Tabla 3. Características de microcomputadores [28]

Característica	**NVIDIA Jetson Nano**	**Coral Dev Board**
CPU	Quad-core ARM Coportex-A57 1.43 GHz	Quad-core ARM Cortex-A53 1.5 GHz
GPU	NVIDIA Maxwell (128 núcleos CUDA)	Edge TPU integrado (procesador IA)
Acelerador de IA	No (solo GPU)	Sí (Edge TPU dedicado, 4 TOPS)
RAM	4 GB LPDDR4 (64-bit, 25.6 GB/s)	1 GB LPDDR4
Almacenamiento	Tarjeta microSD	Tarjeta microSD
Conectividad USB	4x USB 3.0 tipo A, 1x micro-USB	1x USB tipo C, 1x USB 3.0 tipo A
Ethernet	Gigabit Ethernet (10/100/1000 Mbps)	Gigabit Ethernet
Video Salida	HDMI 2.0 y DisplayPort 1.2	HDMI 2.0 (4K a 60 fps)
Video Procesamiento	H.264/H.265 (codificación y decodificación)	No especificado
I/O GPIO	40 Pines GPIO (compatible con Raspberry Pi)	40 Pines GPIO
Cámara	Interfaz CSI (MIPI-CSI, 15 pines)	1x Interfaz CSI
Aceleración de Visión	No	Sí (Edge TPU optimizado para visión)
Alimentación	5V (conector de barril o micro-USB)	5V USB-C
Dimensiones	100 x 80 x 29 mm	88 x 60 mm
Peso	~140 g	~45 g
Sistema Operativo	NVIDIA JetPack (Ubuntu Linux)	Mendel Linux (Debian-based)
IA y Visión por Computadora	GPU CUDA para TensorFlow, PyTorch	Edge TPU (4 TOPS, optimizado para TensorFlow Lite)
Uso Típico	IA, Visión por computadora, robótica	IA en dispositivos, visión, IoT

1.3.6 Plataforma de entrenamiento

Son servicios en línea que permiten a los usuarios almacenar, procesar y administrar datos sin requerir infraestructura física propia. Ofrecen recursos de computación, almacenamiento y redes a través de internet, facilitando el acceso a herramientas y aplicaciones avanzadas sin la necesidad de equipos de hardware específicos.

Estas plataformas eliminan las barreras físicas y los altos costos asociados al mantenimiento de una infraestructura local, fomentando el trabajo colaborativo entre equipos. Además, aprovechan las ventajas de la nube, como la escalabilidad, la seguridad y la eficiencia [29].

Tabla 4 Plataformas de entrenamiento [30]

Plataforma	Descripción	Características
Google Colaboratory	Llataforma en la nube de acceso gratuito que permite ejecutar código Python en notebooks sin necesidad de realizar configuraciones en el equipo local. Está pensada para simplificar el desarrollo y la ejecución de proyectos de aprendizaje automático, proporcionando acceso a GPUs y TPUs para acelerar los procesos.	- Es compatible con frameworks ampliamente utilizados como TensorFlow, Keras, PyTorch, entre otros. - Proporciona acceso a GPUs y TPUs para optimizar y acelerar el proceso de entrenamiento de modelos. - Facilita la ejecución de código en un entorno colaborativo, permitiendo que múltiples usuarios trabajen simultáneamente en el mismo notebook en tiempo real.
Kaggle	Plataforma en línea que proporciona recursos para el análisis de datos y el desarrollo de modelos de aprendizaje automático, respaldada por una comunidad activa de científicos de datos. Mediante Kaggle Kernels, los usuarios pueden ejecutar código en Python y R directamente en la nube, sin necesidad de instalar o configurar un entorno en sus dispositivos.	- Kaggle proporciona acceso sin costo a GPUs y TPUs para el entrenamiento de modelos, además de contar con una amplia variedad de datasets públicos. - Es compatible con marcos de trabajo como TensorFlow, PyTorch y Keras. - Facilita la colaboración en competencias de datos y permite compartir notebooks, integrando código y datasets dentro de un mismo entorno de trabajo.
AWS (Amazon Web Services)	AWS SageMaker es una plataforma totalmente administrada que simplifica la creación, el entrenamiento y la implementación de modelos de aprendizaje automático. Proporciona herramientas para entrenar modelos en la nube utilizando recursos computacionales de alto rendimiento, como GPUs, y permite una escalabilidad eficiente.	- Es compatible con marcos de trabajo conocidos como TensorFlow, PyTorch y MXNet, entre otros. - Optimiza el entrenamiento de modelos para la detección de objetos, con soporte para GPU. - Ofrece herramientas para la visualización, el preprocesamiento de datos y la gestión de modelos.

1.3.7 Software de programación

Un lenguaje de programación es un sistema organizado que posibilita a los desarrolladores escribir instrucciones y algoritmos que las computadoras pueden entender. Actúa como un intermediario entre los seres humanos y las máquinas,

facilitando la realización de tareas específicas y la resolución de problemas a través del procesamiento automático [31].

Es fundamental para el desarrollo y entrenamiento de modelos inteligentes. Permite implementar algoritmos de aprendizaje, gestionar grandes volúmenes de datos y realizar ajustes iterativos en los modelos. Este proceso abarca etapas clave como el preprocesamiento de datos, la selección de características, el entrenamiento, la validación y el despliegue, todo ello respaldado por las capacidades avanzadas de los lenguajes de programación modernos [31].

a. ***Características de los softwares de programación***

En la Tabla 5 se expone las principales características de los lenguajes de programación más utilizados como herramienta para el machine learning.

Tabla 5 Características de los softwares de programación [32]

Característica	Python	JavaScript	Matlab	Lisp
Sintaxis	Simple, clara, fácil de aprender; basada en indentación	Basada en C, requiere comprensión del DOM y manejo de eventos	Sintaxis matemática, orientada a matrices	Sintaxis basada en listas, orientada a recursión
Bibliotecas y frameworks	TensorFlow, Keras, PyTorch, Scikit-learn, XGBoost, etc.	TensorFlow.js, Brain.js, Synaptic (limitadas en comparación)	Toolboxes de Matlab como Statistics and Machine Learning Toolbox, Deep Learning Toolbox	Existen implementaciones como CLML, pero no tan populares
Facilidad de uso	Alta	Moderada	Moderada	Baja
Aplicaciones communes	Ciencia de datos, IA, análisis de datos, automatización, desarrollo web.	Desarrollo web, aplicaciones interactivas en navegador, ML para la web.	Investigación científica, simulaciones numéricas, ingeniería.	Inteligencia artificial simbólica, investigación en computación.
Integración de sistemas	Integración con otras tecnologías, sistemas, y bases de datos.	Integración sencilla con otros lenguajes a través de APIs web.	Se integra bien con sistemas de ingeniería y científicos, pero más cerrada.	Integración limitada fuera de aplicaciones especializadas
Rendimiento	Bueno, con bibliotecas optimizadas (TensorFlow, PyTorch).	Menor rendimiento para ML, pero adecuado para aplicaciones web.	Muy bueno en cálculos numéricos, optimizado para álgebra matricial.	Bajo rendimiento comparado con lenguajes.
Desempeño en ML	Ampliamente utilizado para modelos complejos.	Enfocado a proyectos en el navegador o de pequeña escala.	Menos flexible para deep learning	No es una opción común para el ML.
Implementación de modelos	Rápido con APIs de alto nivel y bibliotecas como Scikit-learn y Keras.	Requiere comprender cómo trabajar con ML en la web y bibliotecas limitadas.	Fácil en entornos científicos y académicos, pero limitado para modelos más generales	Requiere construir muchos algoritmos desde cero sin bibliotecas modernas.
Soporte para deep learning	Bibliotecas como TensorFlow, Keras, y PyTorch dominan el campo.	TensorFlow.js permite crear y ejecutar modelos en el navegador.	Con Deep Learning Toolbox, pero menos popular.	Limitado, no tiene soporte moderno para deep learning.

1.3.8 Software para el desarrollo de aplicaciones

Tabla 6 Comparación de software de desarrollo de aplicaciones [33]

Característica	**Mit App Inventor**	**Android Studio**	**Flutter**
Dificultad	Bajo	Alto	Medio - Alto
Interfaz de Desarrollo	Entorno grafico de bloques	Entorno profesional (IDE Nativo)	IDE´s Android Studio o VS Code con plugins
Lenguaje de programación	No es especifico	Java	Pyhton
Capacidad y flexibilidad	Limitada a funciones básicas	Muy alta	Alta con soporte multiplataforma
Rendimiento	Medio	Alta	Alta
Compatibilidad	Solo Android	Solo Android	Android, iOS, Web, Desktop
Facilidad de aprendizaje	Alta	Baja	Media
Uso	Prototipos, aplicaciones educativas	Aplicaciones educativas y empresariales	Aplicaciones educativas, empresariales, modernas con interfaz fluida.

1.3.9 Sensores

Un sensor es todo elemento o dispositivo que presenta una propiedad que es sensible hacía una determinada magnitud física, y cuando esta magnitud se modifica, la propiedad también presenta un cambio en sus características. Contrario a lo que la creencia popular dicta, un sensor puede convertir la variación de cualquier magnitud en la modificación de cualquier propiedad sensible, y no solamente convertir la magnitud física en una representación eléctrica o electrónica. Se tiene esta percepción debido a que mayoritariamente los sensores son utilizados en equipos eléctricos y electrónicos, y, por lo tanto, traducen la variación de la magnitud en una variación de voltaje o de corriente, sin embargo, no todos los sensores son eléctricos o electrónicos [34].

Para fines de interpretación y estandarización, para el diseño de sensores se generan métricas que permiten al usuario, o al dispositivo lector, poder interpretar escalas medibles y cuantificables [34].

a. Clasificación de los sensores

Los sensores se pueden clasificar de muchas maneras, pero las más relevantes se mencionan a continuación [35]:

- Su principio de funcionamiento
- El tipo de señal de salida
- El rango de valor de salida
- El tipo de variable medida

Para la actualidad existe una gran variedad de sensores los más relevantes se encuentran en la Figura 6.



Figura 6 Clasificación de los sensores

1.3.10 Cámara Digital

Una cámara digital es un dispositivo de hardware que captura imágenes instantáneas de alta calidad, sin necesidad de un proceso de revelado tradicional. A diferencia de las cámaras analógicas que utilizan película, una cámara digital emplea componentes electrónicos como chips CCD (dispositivo de carga acoplada) para registrar la intensidad de la luz y convertirla en datos digitales. La luz pasa a través del lente y llega al CCD, donde se transforma en cargas eléctricas que el procesador de la cámara modela en imágenes almacenadas digitalmente. Según el modelo, las cámaras digitales pueden incluir lentes fijos o intercambiables, visores ópticos o pantallas de visualización, sistemas de autoenfoque activos o pasivos, y controles avanzados que mejoran la exposición y el enfoque [36].

La información capturada se almacena en tarjetas de memoria desmontables o se transfiere directamente a una computadora mediante conexiones específicas, permitiendo su edición, archivo o envío inmediato a través de Internet. Inicialmente, el video en cinta era común para grabar imágenes, pero fue reemplazado por discos ópticos, discos duros y memorias flash. El formato más habitual de archivo de las imágenes digitales es JPG, lo que facilita su manejo y compatibilidad. Con el desarrollo de estas tecnologías, la fotografía ha evolucionado de un proceso físico-químico hacia un sistema electrónico que amplía las posibilidades de captura, transmisión y conservación de imágenes [36].

Tabla 7 Características técnicas de la cámara [37]

Modelo	Características principales	Tipo de conexión	Aspecto físico
Cámara DAHUA SD-H4C	- Resolución de hasta 4MP a 25 imágenes por segundo. - Óptica Fija de 4mm proporciona un ángulo de visión horizontal de 80.3°, vertical de 43.2°, y diagonal de 94.3°. - Almacenamiento Expandible. - Imágenes de alta calidad y detalle, captura cada aspecto de la escena.	Wifi	

1.3.11 Control de acceso

El control de acceso es un proceso de seguridad de datos que permite a las organizaciones gestionar quién está autorizado a acceder a los datos y recursos corporativos. El control de acceso seguro utiliza políticas que verifican que los usuarios sean quienes afirman ser y garantiza que se otorguen niveles de acceso de control adecuados a los usuarios.

Implementar el control de acceso es un componente crucial de la seguridad de aplicación web, lo que garantiza que solo los usuarios correctos tengan el nivel adecuado de acceso a los recursos adecuados. El proceso es fundamental para ayudar a las organizaciones a evitar filtraciones de información [38].

1.4 Objetivos

1.4.1 Objetivo general

- Implementar un sistema integral de control y seguimiento personalizado para gimnasios, basado en inteligencia artificial.

1.4.2 Objetivos específicos

- Determinar las tecnologías y herramientas para el control y seguimiento personalizado en gimnasios.
- Diseñar un sistema electrónico para el control de acceso al gimnasio utilizando inteligencia artificial.
- Desarrollar una aplicación intuitiva para el seguimiento del progreso personalizadas a los usuarios del gimnasio.

CAPÍTULO II. METODOLOGÍA

2.1 Materiales

Para el desarrollo del sistema integral de control y seguimiento personalizado para gimnasios, se han seleccionado materiales de hardware y software que permiten una implementación eficiente, segura y escalable. A nivel de hardware, se emplea una NVIDIA Jetson Nano, que proporciona la capacidad de procesamiento necesaria para ejecutar algoritmos de inteligencia artificial en tiempo real, especialmente para tareas como el reconocimiento facial y el análisis de video. También se utiliza una placa ESP32, que permite la conexión de sensores y la transmisión de datos mediante WiFi, facilitando la integración del sistema con una red local o la nube.

Se incluye un sensor ECG para capturar señales biométricas relevantes, y una batería Lipo como fuente de energía portátil y confiable, garantizando la operatividad del sistema ante posibles fallos eléctricos. Además, se incorpora una cámara Dahua HC4, que proporciona imágenes de alta resolución para la detección facial, y un altavoz o parlante para emitir alertas o mensajes de voz relacionados con el ingreso o eventos relevantes del usuario.

En cuanto al software, se utiliza Python como lenguaje principal para el desarrollo de algoritmos de visión artificial, reconocimiento biométrico y análisis de datos. Arduino IDE se emplea para programar la ESP32 y controlar sensores conectados. Sublime Text 3 se utiliza como editor ligero para codificación rápida. Para el desarrollo de la aplicación móvil del sistema, se usa Android Studio, permitiendo una interfaz intuitiva para usuarios y administradores. Además, se integra Google Colab como entorno colaborativo para pruebas experimentales y entrenamiento de modelos de machine learning, aprovechando su capacidad de procesamiento en la nube.

2.2 Métodos

2.2.1 Modalidad de la investigación

La presente investigación es aplicada, ya que se concentra en la aplicación práctica de técnicas de inteligencia artificial para el control de acceso al gimnasio, mediante el uso de datos biométricos. Esta modalidad se enfoca en la recolección y análisis de información relevante sobre algoritmos de aprendizaje automático Deep Learning, aplicados tanto para verificar la identidad de los usuarios en tiempo real como para el seguimiento personalizado de su progreso físico. La implementación de estos algoritmos permitirá monitorear indicadores como el peso y la musculatura de los usuarios, generando recomendaciones personalizadas para mejorar su rendimiento. Al integrar estos avances tecnológicos, se busca garantizar un acceso seguro y eficiente, optimizando la operatividad del gimnasio, mejorando la experiencia del usuario y promoviendo un seguimiento continuo y preciso de su desarrollo físico.

Se empleará la modalidad de investigación bibliográfica para revisar artículos científicos y recursos disponibles en repositorios universitarios, con el objetivo de establecer un fundamento teórico sólido sobre el uso de inteligencia artificial en sistemas de control de acceso. Esta revisión permitirá obtener datos relevantes sobre técnicas de Deep Learning aplicadas a la seguridad y al seguimiento personalizado del progreso físico de los usuarios, contribuyendo a la comprensión de enfoques actuales en la protección de instalaciones y la mejora de la experiencia del usuario. La información recopilada permitirá garantizar la efectividad del sistema propuesto para el control de acceso al gimnasio, optimizando tanto la seguridad como el rendimiento físico de los usuarios mediante la integración de tecnologías avanzadas.

Dado el enfoque práctico de la tesis, se emplea la investigación experimental, la cual consiste en la recolección de datos a través de la implementación y prueba del sistema de control de acceso mediante inteligencia artificial en entornos reales. Esta metodología permitirá evaluar la efectividad, precisión y viabilidad del sistema propuesto, así como su integración exitosa con las tecnologías existentes, como los sistemas biométricos para la identificación facial y el seguimiento físico de los

usuarios. A través de pruebas y validaciones, se asegurará que el sistema no solo mejore la seguridad en el gimnasio, sino que también optimice el control eficiente y personalizado del progreso físico de los usuarios, garantizando una experiencia mejorada y un acceso seguro y ordenado.

2.2.2 Población y muestra

La investigación se llevará a cabo con usuarios de cualquier gimnasio, que cuenta con una afluencia de muchos usuarios mensuales.

2.2.3 Recolección de información

Con el propósito de obtener los datos requeridos para la implementación del sistema de control de acceso mediante inteligencia artificial en el gimnasio, se buscará información proveniente de estudios previos y relacionados con el uso de tecnologías biométricas y la gestión de accesos. Además, bajo la orientación del tutor, se organizarán los datos obtenidos a través de simulaciones y pruebas del sistema en un entorno controlado, permitiendo evaluar su efectividad tanto en el monitoreo de accesos en tiempo real como en el seguimiento personalizado del progreso físico de los usuarios. La recolección de información también incluirá entrevistas con expertos en seguridad, análisis de incidentes previos y pruebas de campo realizadas durante el desarrollo y validación del sistema propuesto, con el fin de optimizar la seguridad y mejorar la experiencia de los usuarios.

Tabla 8 Criterios de recolección de información

Criterio	**Descripción**
¿Para qué?	Mejorar el control de acceso y seguimiento físico en el gimnasio mediante inteligencia artificial y biometría.
¿De qué personas u objetos?	Usuarios y visitantes al gimnasio.
¿Cómo?	Mediante la implementación de un sistema de control integral con inteligencia artificial.
¿Con qué?	Con herramientas de reconocimiento facial.
¿Cuándo?	Durante la investigación, se realizarán pruebas y simulaciones dentro de las instalaciones.
¿Dónde?	En un gimnasio o en un entorno simulado que replique su infraestructura de acceso.

2.2.4 Procesamiento y análisis de datos

Para el procesamiento y análisis de los datos en esta investigación, se considerarán las herramientas de software y hardware más adecuadas para la implementación efectiva de un sistema de monitoreo de control integral al gimnasio mediante inteligencia artificial, con el objetivo de mejorar la seguridad del sistema y garantizar el seguimiento personalizado del proceso físico. Este proceso incluirá los siguientes pasos:

- Revisión de información teórica obtenida.
- Determinación de materiales y tecnologías.
- Pruebas de funcionamiento del sistema de control integral.
- Puesta en funcionamiento del sistema.
- Almacenamiento de los registros de acceso y posibles incidencias.

- Análisis de los datos obtenidos del seguimiento físico de los usuarios
- Presentación de resultados de la investigación

CAPÍTULO III. RESULTADOS Y DISCUSIÓN

3.1 Desarrollo de la propuesta

Para el desarrollo del presente proyecto se implementa el algoritmo YOLOv8, para la detección en tiempo real de indicadores biométricos previos al entrenamiento físico. La solución integra de forma coordinada componentes de hardware y software que permiten el monitoreo continuo y simultáneo de parámetros fisiológicos. La etapa de etiquetado y preparación del dataset se lleva a cabo mediante la plataforma Roboflow, mientras que el entrenamiento del modelo se ejecuta en Google Colab, aprovechando la capacidad de procesamiento en la nube y la compatibilidad con YOLOv8. La arquitectura del sistema incorpora conectividad inalámbrica mediante tecnología WiFi, lo que garantiza la sincronización eficiente con dispositivos móviles. Adicionalmente, se desarrolla una aplicación móvil intuitiva que permite al usuario visualizar métricas de rendimiento, recibir alertas personalizadas y realizar un seguimiento integral de su progreso. Esta propuesta tiene como finalidad optimizar la experiencia del entrenamiento mediante un sistema de control inteligente, automatizado y adaptable a distintos perfiles físicos.

3.1.1 Definición de los elementos para la propuesta

Tras la pandemia, Ecuador ha experimentado un notable crecimiento en el sector fitness. En 2020, existían aproximadamente 1.765 gimnasios en el país para 2025, esta cifra aumentó a más de 2.500 representando un incremento del 41,6 % en cinco años. Este auge se atribuye a una mayor conciencia sobre la salud y el bienestar, así como al incremento del poder adquisitivo y la adopción de estilos de vida más activos. Además, la entrada de franquicias internacionales, como SmartFit, ha incrementado considerablemente la oferta en el país [39].

A partir de entrevistas realizadas a usuarios de diversos gimnasios, se identificó que la mayoría de estos establecimientos aún emplean métodos tradicionales para el registro de asistencia, principalmente mediante hojas físicas. Un porcentaje considerable ha

migrado a sistemas básicos, donde el registro se realiza en un ordenador a través de un código asignado por el administrador. Sin embargo, solo un número reducido de cadenas de gimnasios de mayor tamaño ha implementado tecnologías más avanzadas, como el ingreso mediante huella digital, representando esta una proporción mínima.

En la Tabla 9 nos permite demostrar la consecución del objetivo especifico 1, al mostrar las tecnologías utilizadas por los usuarios en los centros de entrenamiento a su ingreso.

Tabla 9 Métodos de ingreso a centros de entrenamiento

Tipo de registro	**Descripción**	**Porcentaje**
Hojas físicas.	Registro manual en hojas físicas.	60%
Sistemas básicos del ordenador.	Registro mediante un código asignado.	30%
Tecnologías avanzadas	Ingreso mediante biometría dactilar.	10%

Este contexto presenta una oportunidad ideal para implementar soluciones tecnológicas avanzadas que optimicen el control y seguimiento del rendimiento físico de los usuarios

3.1.2 Elementos por detectar dentro del sistema

En los últimos años, el crecimiento de los centros fitness en Ecuador ha sido notable, reflejando una tendencia nacional hacia el bienestar y la actividad física. La provincia de Tungurahua, como uno de los polos económicos más importantes del país, no ha sido ajena a esta expansión. Sin embargo, este incremento también evidencia la necesidad de implementar sistemas de control y seguimiento que optimicen el uso de estos espacios. En este contexto, la incorporación de tecnologías adaptativas y

accesibles se vuelve esencial para mejorar la gestión, el monitoreo personalizado y la eficiencia operativa de los centros de entrenamiento [39].

a. Rostro usuario

El presente proyecto se enfoca en la detección biométrica, específicamente en el reconocimiento facial del usuario, como mecanismo principal para el registro automatizado de asistencia al gimnasio. Este sistema se complementa con una aplicación móvil que proporciona recomendaciones personalizadas basadas en inteligencia artificial, orientadas a optimizar el rendimiento físico del usuario y mejorar su experiencia durante las sesiones de entrenamiento.

En la Tabla 10 se presentan los distintos tipos de rostros que serán reconocidos por el sistema, clasificados según sus características específicas.

Tabla 10 Tipos y clasificación de rostros [40]

Tipos de rostros	
Rostro ovalado	El rostro ovalado es ancho en la parte superior y más estrecho hacia el mentón.
Rostro redondo	El rostro redondo se caracteriza por tener más o menos igual al ancho y el largo del rostro, contornos redondeados.
Rostro cuadrado	El rostro cuadrado se caracteriza por tener la frente y la mandíbula con un ancho muy similar.
Rostro rectangular	El rostro| rectangular se caracteriza por ser similar al cuadrado solo que un poco más alargado.
Rostro alargado	El rostro alargado se caracteriza por que la cara es delegada y alargada hacia la zona del mentón.
Rostro corazón	El rostro corazón presenta una frente y pómulos anchos con mandíbula estrecha.
Rostro diamante	El rostro en forma de diamante son pómulos prominentes que resaltan sobre el ancho de la frente y la mandíbula.
Rostro triangulo	El rostro triangulo presenta una frente mucho más estrecha en comparación con una mandíbula especialmente ancha.
Rostro triangulo invertido	El rostro de triangulo invertido tiene más ancha la zona de la frente y tanto los pómulos como la mandíbula más estrechos.

En la Tabla 10 se detallan los distintos tipos de rostros según su forma, profundidad, rasgos faciales y anchura, especificando las características que permiten su clasificación.

b. Comportamientos y actividades físicas

Los usuarios valoran significativamente la manera en que sus entrenadores les asisten para mejorar su técnica dentro del centro de entrenamiento, ya que una correcta ejecución del ejercicio maximiza sus beneficios y reduce el riesgo de lesiones. Por otro lado, es común que los entrenadores sean permisivos con ciertos comportamientos de sus clientes, lo que los lleva a evaluar subjetivamente la fuerza de voluntad de cada usuario para mantener la constancia en su rutina diaria [41].

Un aspecto importante a considerar es que muchos usuarios esperan obtener resultados inmediatos; sin embargo, en aproximadamente el 98 % de los casos, los cambios físicos visibles no se manifiestan de forma inmediata. Este proceso está influenciado por diversos factores como la alimentación, el entorno familiar, el estado mental, la disponibilidad de tiempo y, principalmente, la disciplina del usuario.[41]

La Tabla 11 presenta una variedad de actividades físicas junto con los espacios específicos donde se desarrollan dentro de un centro de entrenamiento.

Tabla 11 Actividades físicas en un centro de entrenamiento [42]

Actividades físicas en un centro de entrenamiento	
Categoría	**Descripción**
Salas de entrenamiento	Espacios diseñados para diferentes tipos de actividad física como cardio, musculación, entrenamiento funcional y zonas de peso libre, que permiten desarrollar rutinas variadas según el objetivo del usuario.
Clases grupales dirigidas	Actividades colectivas lideradas por instructores/as, como spinning, aeróbicos, yoga, pilates, HIIT y danza fitness, enfocadas en mejorar la resistencia, coordinación, fuerza y flexibilidad.
Entrenamiento funcional	Ejercicios que imitan movimientos naturales del cuerpo, enfocados en el trabajo integral de fuerza, equilibrio, agilidad y coordinación, adaptables a distintos niveles de condición física.
Actividades de bajo impacto	Programas como yoga y pilates que fortalecen la musculatura profunda, mejoran la postura y aumentan la flexibilidad, siendo ideales para personas mayores o en rehabilitación.

c. Métricas de tiempo y presencia

En la mayoría de los centros de entrenamiento se ha evidenciado que muchos usuarios tienden a tomar intervalos de descanso excesivamente prolongados durante sus rutinas, lo que incrementa el desgaste físico y reduce la eficiencia del entrenamiento.

Además, las distracciones frecuentes, como el uso del teléfono móvil, afectan negativamente el ritmo y la continuidad de la sesión, conllevando incluso derivar en lesiones musculares por contracción inadecuada. Otro factor relevante es la frecuencia de asistencia, ya que se ha detectado una alta tasa de deserción durante las primeras dos semanas de entrenamiento. Esta problemática suele estar asociada a variables externas como la falta de tiempo, limitaciones económicas o situaciones personales.

3.1.3 Esquema general del proyecto

El diagrama que se muestra en la Figura 7 muestra la arquitectura general del sistema integral de control y seguimiento para gimnasios, el sistema está compuesto por cuatro

etapas principales: adquisición de datos, entrenamiento del modelo, procesamiento de información y detección e interpretación. Cada una de estas etapas incluye subprocesos específicos que permiten el funcionamiento coordinado del sistema, desde la captura de señales e imágenes hasta el análisis inteligente y la generación de alertas personalizadas.

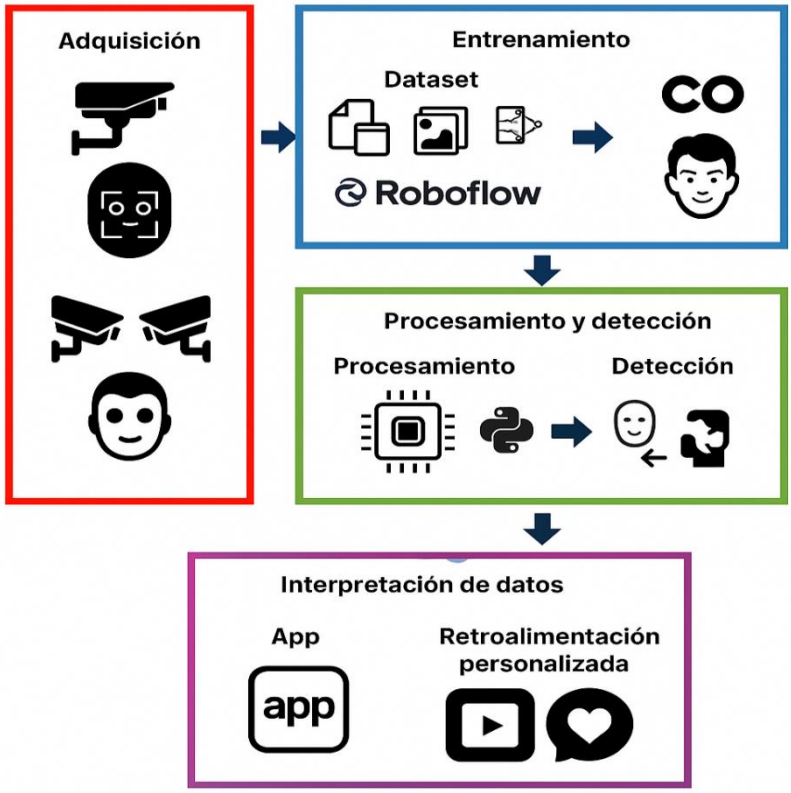


Figura 7 Esquema general del sistema

a. Etapa 1: Adquisición

En la etapa de adquisición, se lleva a cabo la captura de imágenes necesarias para el reconocimiento biométrico de cada usuario, utilizando una cámara Dahua HC4 con conectividad Wi-Fi y especificaciones adaptadas a los requerimientos del proyecto. Adicionalmente, se emplea un dispositivo electrónico equipado con un sensor ECG, encargado de registrar la actividad cardíaca durante los ejercicios de cardio, que involucran a la mayoría de los grupos musculares y aportan beneficios significativos a la salud cardiovascular. Finalmente, se dispone de una aplicación móvil que

proporciona a los usuarios, tanto nuevos como recurrentes, recomendaciones personalizadas diarias para sus rutinas de entrenamiento.

b. Etapa 2: Entrenamiento

Para la obtención del dataset adaptado al sistema, se recopilan y procesan datos que alimentan el módulo de control, previamente etiquetados con el fin de permitir la identificación precisa de cada usuario. Este procedimiento genera un conjunto de información estructurada y abundante, indispensable para el entrenamiento efectivo del modelo. Para acelerar dicho entrenamiento, se aprovecharon recursos computacionales en la nube, optimizando así los tiempos de procesamiento y la eficiencia del sistema de red neuronal.

En este proceso, la red neuronal mejora su rendimiento al ser entrenada con un conjunto de datos robusto y altamente representativo. Esto le permite identificar con mayor precisión patrones complejos que, de otro modo, serían difíciles de detectar, incrementando así la eficiencia y la confiabilidad del sistema.

c. Etapa 3: Procesamiento y detección

Procesamiento

Tras finalizar el entrenamiento, el modelo resultante se implementó en un microcomputador, también conocido como dispositivo de computación en el borde, especialmente diseñado para aplicaciones de inteligencia artificial. Este dispositivo, capaz de ejecutar algoritmos complejos de aprendizaje profundo, permite llevar a cabo tareas de visión por computadora de manera autónoma y eficiente. Así, se garantiza un procesamiento en tiempo real que facilita una respuesta rápida y dinámica en la detección de actividades delictivas, fortaleciendo la precisión y efectividad del sistema.

Detección

Cuando el modelo con el algoritmo en Python ya está entrenado e implementado en el microcomputador, procede a la identificación de aquellos objetos similares a los que se utilizaron en el entrenamiento, en este caso rostros, y con el registro los datos ya registrados, ya sean cortas o largas, de tal manera que al detectarlos resalta su comparativa de características, identificando las características obtenidas en las primeras detecciones de rostros y características faciales.

d. Etapa 4: Interpretación de datos

Mediante un dispositivo portátil conectado a sensores de ritmo cardíaco y pulsos por minuto, los datos se representan en una aplicación móvil que detalla los valores recibidos de los sensores mencionados. Además, se emite una alerta sonora que indica si el registro de algún usuario ya ha sido realizado o aún está pendiente.

3.1.4 Selección de tecnologías

a. Microprocesador

Para asegurar el desempeño óptimo del sistema de detección y registro facial, es imprescindible seleccionar un microcomputador cuyas especificaciones técnicas sean capaces de procesar video con la velocidad requerida, dado que la detección debe realizarse en tiempo real.

La Tabla 3 presenta las características y especificaciones técnicas de los distintos microcomputadores disponibles para aplicaciones de visión por computadora. En este contexto, la NVIDIA Jetson Nano se posiciona como la opción más adecuada para un sistema de detección y registro facial. Su GPU optimizada garantiza que el procesamiento se realice de manera eficiente, lo cual es crucial para un sistema de registro facial, al igual que su costo y facilidad de integración la posicionan como una opción económica y práctica.

b. Cámara

La cámara Dahua H4C es una opción clave para el registro y reconocimiento facial a su alta resolución, que captura imágenes nítidas y detalladas, garantizando una identificación precisa. Su procesamiento avanzado optimiza la calidad incluso en condiciones de iluminación variables, mientras que su visión nocturna infrarroja asegura un funcionamiento continuo, sin importar la hora. Además, incorpora inteligencia artificial para la detección facial, automatizando el registro de manera eficiente. Su compresión de video optimizada y compatibilidad con estándares de comunicación permiten una integración fluida y un sistema escalable, convirtiéndola en una solución confiable y versátil para aplicaciones de seguridad basadas en reconocimiento facial.

Las características de la cámara se presentan en la Tabla 7, en donde se observa la resolución, la cantidad de pixeles y el fps correspondientes.

c. Sensores

El sensor SEN053 se selecciona para la medición de ritmo cardíaco y pulsos por minuto debido a su alta precisión y capacidad para obtener datos en tiempo real, esenciales para un monitoreo confiable de la actividad cardiovascular. Su tecnología óptica avanzada permite detectar pulsaciones con sensibilidad y exactitud, incluso en condiciones de movimiento, garantizando así una captura consistente de señales vitales. Además, su diseño compacto y fácil integración con microcontroladores y sistemas embebidos facilita el desarrollo de dispositivos portátiles y soluciones inteligentes para la detección temprana y seguimiento continuo de la salud cardíaca, lo que lo convierte en una opción óptima para aplicaciones médicas en tiempo real.

d. Software de programación

La selección del software de programación adecuado se llevó a cabo mediante la comparación de las principales características de los lenguajes más empleados en el desarrollo de aplicaciones de inteligencia artificial, la evaluación de las características de estos lenguajes proporciona una base fundamental para medir sus capacidades y seleccionar el más adecuado según los requisitos específicos del proyecto. Este análisis

garantiza una elección estratégica y eficiente al integrar inteligencia artificial en el sistema.

Tras evaluar las opciones, se optó por Python para el desarrollo del proyecto debido a su vasta oferta de bibliotecas, su activa comunidad y su simplicidad de uso ejecutándose en todo tipo de entorno. Estos aspectos convierten a Python en una herramienta potente, accesible y gratuita para implementar proyectos de inteligencia artificial.

e. Modelo de entrenamiento

El modelo de entrenamiento implementado en este proyecto se basa en el algoritmo YOLOv8, seleccionado por su eficiencia en tareas de detección en tiempo real y su compatibilidad con dispositivos de bajo consumo como la NVIDIA Jetson Nano. El entrenamiento se realizó en la plataforma Google Colab, aprovechando sus recursos de GPU gratuitos, lo que permitió acelerar el proceso computacional y facilitar el acceso remoto a través de entornos de desarrollo colaborativos.

Para el entrenamiento, se utilizó un conjunto de imágenes faciales previamente capturadas y etiquetadas mediante la herramienta Roboflow, la cual permitió clasificar adecuadamente los datos y organizarlos para un modelo supervisado. Las imágenes fueron procesadas para cumplir con los formatos requeridos por YOLOv8 y se almacenaron en Google Drive para su integración directa con el entorno de Colab.

f. Software para el desarrollo de aplicaciones

La selección del software para el desarrollo de aplicaciones se llevó a cabo mediante una comparativa entre las principales características de los softwares más empleados para el diseño de apps.

En la Tabla 6 se observan las diferentes características que posee cada uno de los softwares más utilizados para el desarrollo de aplicaciones móviles, tanto para Android como para iOS, permitiendo además un enfoque adecuado en la comunicación con los dispositivos móviles que se deseen integrar.

Después de evaluar las opciones presentadas, se optó por el software Flutter junto con Android Studio debido a su capacidad para desarrollar aplicaciones nativas con una única base de código, su amplia comunidad de soporte, y la facilidad para realizar pruebas y depuración rápida gracias a funcionalidades, como se detalla en el Anexo 10.

3.1.5 Descripción del sistema de control de acceso

En la Figura 8 presenta la arquitectura del sistema de control de acceso basado en inteligencia artificial no obstante muestra la consecución del objetico especifico 2 donde se destaca sus sub-etapas con un enfoque centrado en los componentes físicos. Se ilustran las interconexiones del sistema, iniciando con una fuente de alimentación de 110V AC, responsable de suministrar energía a los módulos clave: cámara, servidor y altavoz. La comunicación entre estos componentes se establece mediante el protocolo RTSP, facilitando tanto la transmisión en tiempo real de las imágenes faciales de los usuarios como su envío al servidor para procesamiento.

Posteriormente, se ejecuta la comparación entre los rostros capturados y los registrados en la base de datos, determinando la autenticidad de la identidad del usuario. Finalmente, el sistema emite una notificación sonora que indica la autorización o denegación de acceso, optimizando la seguridad y eficiencia del proceso de verificación.

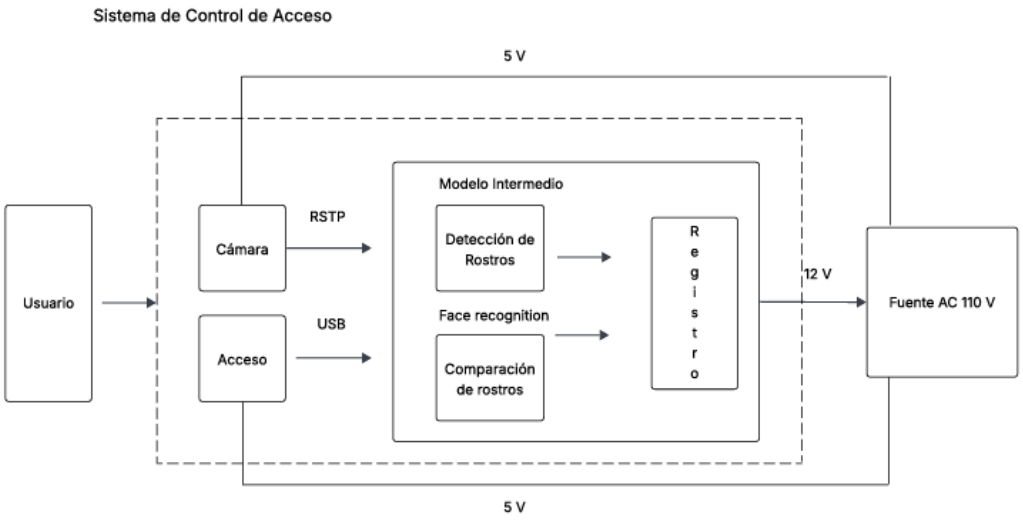


Figura 8 Sistema de Control

3.1.6 Entrenamiento del sistema

El proceso de entrenamiento del modelo requiere una evaluación cuidadosa de diversos factores fundamentales, los cuales se presentan detalladamente en la Tabla 12.

Tabla 12 Parámetros de entrenamiento

Parámetros del entrenamiento		
Parámetros	**Características**	**Detalle**
E. Train	1048	Etiquetas para entrenamiento
E. Valid	0	Etiquetas para validación
E. Test	0	Etiquetas para prueba
Cant. Imágenes	1048	Imágenes totales
Cant. Etiquetas	1049	Cantidad de etiquetas
Data	Data/Campo	Conjunto de datos
Épocas	60	Épocas para el entrenamiento
Batch	16	Tamaño de las muestras
Modelo	Yolov8 PyTorch	Modelo de entrenamiento
Clases	1	Etiquetas del dataset

Tras analizar estos datos, es posible proceder con el entrenamiento del modelo. Es importante señalar que, para su evaluación, se llevaron a cabo dos sesiones de entrenamiento con variaciones en ciertos parámetros, con el propósito de optimizar los resultados obtenidos.

Se habilitó un entorno de trabajo en Google Colab, aprovechando su capacidad para ejecutar procesos en GPU, lo que optimiza significativamente la velocidad y eficiencia del entrenamiento del modelo. Mediante repositorios alojados en GitHub, se ejecutaron las instrucciones necesarias para la descarga e instalación de YOLOv8, asegurando una configuración adecuada para el procesamiento de imágenes.

En la Figura 9, se muestra la línea de código utilizada para instalar YOLOv8 en Google Colab.

```
!pip install --upgrade --force-reinstall numpy
!pip install --upgrade ultralytics roboflow
```

Figura 9 Instalación de Yolo

Cabe recalcar que las versiones actuales de Yolov8 requieren actualizaciones de librerías como Numpy y Roboflow. Con dichos comandos se actualizarán automáticamente todos los paquetes necesarios para la instalación de Ultralytics.

Cuando ya se hayan actualizado y descargado todos los paquetes, continua la descarga del dataset a utilizar, en este caso, se usó uno ya existente que se encuentra en la plataforma online Roboflow.

En la Figura 10 se detallan varios parámetros correspondientes a la dirección de donde se obtuvo el dataset, entre ellos el más importante: la versión de YOLO que será utilizada.

```
[2]  !pip install roboflow

     from roboflow import Roboflow
     rf = Roboflow(api_key="lrFwCd27ADkMRI5eKW12")
     project = rf.workspace("karen-blpq5").project("fashionhub")
     version = project.version(8)
     dataset = version.download("yolov8")
```

Figura 10 Parámetros del Dataset

En la Figura 10 se muestran los parámetros principales empleados para el inicio del entrenamiento del modelo de detección. Considerando las limitaciones impuestas por Google Colab, el número de épocas se restringió a 60, con el fin de maximizar el aprovechamiento de los recursos disponibles en la plataforma. Esta configuración pretende lograr un balance óptimo entre el rendimiento del entrenamiento y las capacidades del entorno, asegurando resultados efectivos dentro de las restricciones de tiempo y procesamiento.

3.1.7 Detección de rostros mediante imágenes de video

La identificación de rostros en secuencias de video es una tecnología esencial en diversas aplicaciones de seguridad, vigilancia y sistemas inteligentes. Este proceso permite localizar y reconocer automáticamente caras humanas, ya sea en transmisiones en tiempo real o en grabaciones, facilitando análisis avanzados para tareas como control de acceso, supervisión de entornos y optimización de la interacción entre humanos y sistemas automatizados. Los distintos tipos de rostros se pueden observar en la Tabla 10, de tal manera que un modelo entrenado con una alta cantidad imágenes, identificara las características correspondientes a un rostro, mas no una única o tipo especie de rostro.

En la Figura 11 se muestra el funcionamiento principal del sistema detector de rostros, finalizando en la delimitación de un cuadro cuando detecte características similares a un rostro.

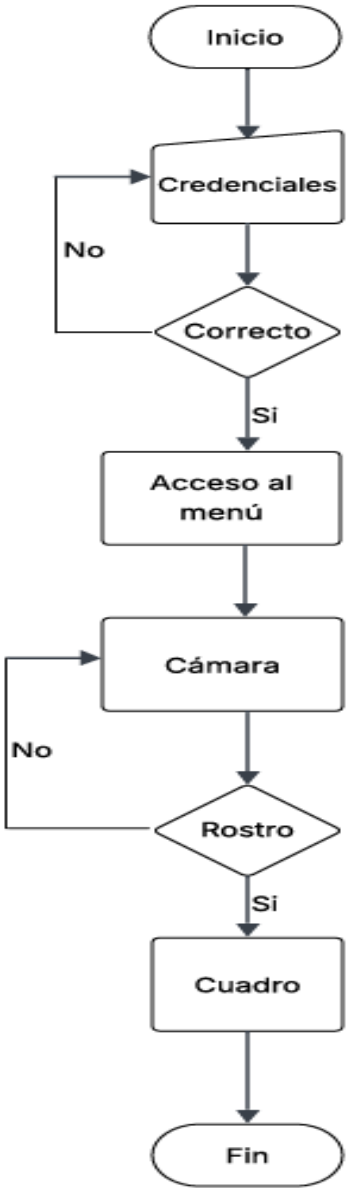


Figura 11 Sistema a detector de rostros

a. Importación de librerías

Para el desarrollo del sistema de reconocimiento facial y seguimiento inteligente en gimnasios, se utilizaron diversas librerías en Python. cv2 (OpenCV) permite capturar y procesar imágenes en tiempo real, mientras que face_recognition se encarga de la detección e identificación de rostros con alta precisión. numpy facilita el manejo de matrices y operaciones numéricas necesarias para el procesamiento de imágenes. Para la creación de la interfaz gráfica se empleó tkinter, complementado con messagebox y

simpledialog para la interacción dinámica con el usuario. La visualización de imágenes dentro de la GUI se realizó mediante PIL, a través de Image e ImageTk, asegurando compatibilidad con distintos formatos. Además, openpyxl permitió almacenar registros en hojas de cálculo Excel, y os se utilizó para gestionar rutas y archivos del sistema.

También se puede incluir datetime para registrar fechas y horas de ingreso. En conjunto, estas herramientas permiten construir un sistema eficiente, visual e interactivo, adaptado a las necesidades del entorno de entrenamiento.

```
1   import cv2
2   import mediapipe as mp
3   import threading
4   import time
5   import os
6   import csv
7   import tkinter as tk
8   from tkinter import messagebox
9   import numpy as np
10  import cv2
11  import torch
12  from ultralytics import YOLO
13
```

Figura 12 Librerías utilizadas Python

b. Aplicación del modelo entrenado

Una vez entrenado, el modelo basado en YOLOv8 es implementado en la microcomputadora Jetson Nano, permitiendo la detección facial en tiempo real dentro del entorno del gimnasio. Al identificar un rostro registrado, el sistema autoriza el acceso mediante una señal visual y registra el evento con hora y fecha. En caso de detectar un rostro no reconocido, se genera una alerta para el administrador. La aplicación permite el monitoreo constante de ingresos y salidas de los usuarios, mejorando la gestión del espacio y la seguridad.

```
# Cargar el modelo YOLOv8
model = YOLO("/home/dario/Descargas/best.pt")
```

Figura 13 Carga del modelo

c. Detección de rostros

La detección de rostros es una de las funciones principales del sistema, permitiendo identificar a cada usuario en tiempo real al ingresar al gimnasio. Mediante el uso del modelo YOLOv8, entrenado previamente con imágenes etiquetadas, el sistema localiza y encierra el rostro en un cuadro delimitador para su verificación. Esta detección se realiza de forma rápida y precisa, incluso en condiciones de iluminación variables. Al reconocer un rostro autorizado, se registra el acceso de forma automática. En caso de que el rostro no coincida con la base de datos, se activa una alerta visual para el administrador. Este proceso mejora la seguridad y automatiza el control de ingreso.

```
# Realizar la detección de rostros
results = model(frame)

# Dibujar las detecciones en el frame
for result in results:
    for bbox in result.boxes.xyxy:
        x1, y1, x2, y2 = map(int, bbox)
        cv2.rectangle(frame, (x1, y1), (x2, y2), (0, 255, 0), 2)
```

Figura 14 Algoritmo de detección de rostros

d. Registro de rostros

La detección de rostros en el sistema se inicia durante el proceso de registro del usuario, donde se capturan imágenes faciales en un periodo de tiempo determinado. Antes de iniciar la captura, el sistema verifica que los campos de nombre, apellido y cédula estén completos y que el usuario no haya sido previamente registrado, consultando un archivo CSV. Una vez validado, se crea una carpeta específica para el usuario, en la cual se almacenarán las imágenes capturadas.

Durante este proceso, la cámara identifica el rostro en tiempo real, y cada frame que contiene un rostro válido es almacenado con el fin de construir una base de datos sólida. Esta información servirá posteriormente para el entrenamiento del modelo de reconocimiento facial, garantizando precisión y organización en la detección y comparación de rostros durante el control de acceso.

```python
# Función para registrar rostro durante 10 segundos con máscara blanca
def registrar_usuario(nombre, apellido, cedula):
    if not nombre or not apellido or not cedula:
        messagebox.showerror("Error", "Completa todos los campos.")
        return

    carpeta_usuario = f"rostros_registrados/{cedula}_{nombre}_{apellido}"
    os.makedirs(carpeta_usuario, exist_ok=True)

    # Verificar si ya existe en el CSV
    if os.path.exists("usuarios.csv"):
        with open("usuarios.csv", "r", newline='', encoding='utf-8') as file:
            reader = csv.reader(file)
            for row in reader:
                if cedula in row:
                    messagebox.showwarning("Ya existe", "Este usuario ya fue registrado.")
                    return
```

Figura 15 Registro de rostros

e. Reproducción de audio

Durante las etapas finales del sistema, se implementa un módulo de reproducción de audio que permite emitir alertas sonoras configurables por el administrador. En este proyecto se utilizó un audio pregrabado como mensaje de advertencia cuando el usuario no se encuentra registrado o su información no es reconocida por la base de datos.

```python
if mejor_usuario is not None and mayor_similitud > 0.55:
    label_mensaje.configure(text=f"Usuario {mejor_usuario['Nombre']} {mejor_usuario['Apellido']} bienvenido.", text_color="green")
    reproducir_audio("/home/dario/Desktop/TesisFinal/UsuarioRegistrado.mp3")
    mostrar_datos_usuario(mejor_usuario["Cedula"])
else:
    label_mensaje.configure(text="Usuario desconocido, registrarse primero.", text_color="red")
    reproducir_audio("/home/dario/Desktop/TesisFinal/UsuarioNoRegistrado.mp3")
```

Figura 16 Codificación de reproducción de audio

f. Ingreso de usuarios al gimnasio con su registro

La Figura muestra el módulo de visualización de usuarios registrados dentro del sistema de control de acceso desarrollado. Esta interfaz permite consultar de forma estructurada los datos almacenados en la base de datos, incluyendo cédula y nombres completos. La información es gestionada mediante archivos Excel y organizada automáticamente tras el proceso de registro facial. Esta funcionalidad garantiza trazabilidad, control y verificación de identidad en tiempo real. Además, respalda la

validación del objetivo específico 2, relacionado con la implementación de un sistema electrónico basado en inteligencia artificial. Su integración mejora la eficiencia operativa y la seguridad del gimnasio.



Figura 17 Registro de usuarios

g. Almacenamiento de los datos biométricos faciales

En la Figura 17 muestra la estructura del sistema de almacenamiento de datos faciales implementado en el entorno Ubuntu. En la carpeta denominada “rostros” se organizan subdirectorios individuales para cada usuario registrado, identificados por su número de cédula y nombre. Esta organización permite una gestión eficiente de los datos biométricos capturados durante el proceso de registro. Cada subcarpeta contiene imágenes faciales y archivos de embeddings utilizados para el reconocimiento. Esta estructura respalda la trazabilidad y precisión del sistema de control de acceso. Su implementación evidencia la integración efectiva entre el módulo de detección facial y la base de datos del sistema.

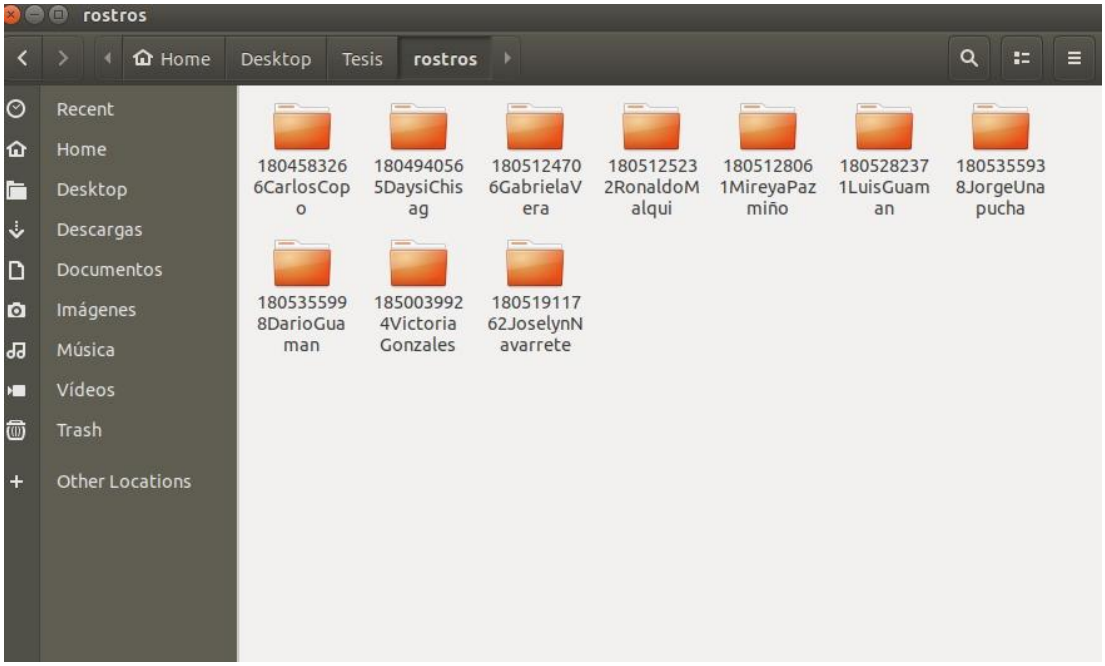


Figura 18 Almacenamiento de los datos biométricos

h. Cálculos de métricas en los usuarios

Tabla 13 Cálculos de métricas de los usuarios

Usuario	Precisión	Sensibilidad	F1Score	Exactitud	Eficiencia
Jorge Unapucha	0.980	0.900	0.938	0.905	0.931
Dario Guaman	0.988	0.882	0.932	0.909	0.928
Gabriela Vera	0.987	0.898	0.940	0.909	0.933
Victoria Gonzales	0.990	0.888	0.936	0.901	0.929
Ronal Malqui	0.977	0.884	0.928	0.896	0.921
Gabriela Navarrete	0.990	0.898	0.939	0.906	0.932
Carlos Copo	0.977	0.911	0.944	0.901	0.934
Mireya Pazmiño	0.984	0.889	0.936	0.892	0.926
Daysi Chisag	0.979	0.890	0.931	0.898	0.924
Luis Guaman	0.988	0.913	0.947	0.895	0.935

En la Tabla 13 presentada resume el desempeño individual de los usuarios registrados en el sistema de reconocimiento facial, evaluando cinco métricas clave: precisión, sensibilidad, F1 Score, exactitud y eficiencia. Cada fila corresponde a un usuario, permitiendo comparar su rendimiento en el proceso de identificación biométrica. Los valores reflejan un sistema robusto, con precisiones cercanas al 98% y eficiencias superiores al 92%. Esta información es fundamental para validar la efectividad del

modelo YOLOv8 implementado. Además, permite identificar posibles variaciones en la detección según las características faciales de cada usuario.

3.1.8 Descripción de la aplicación de seguimiento personalizado

La aplicación móvil de seguimiento personalizado cuenta con una interfaz principal intuitiva y segura, diseñada para permitir el registro individual de cada usuario garantizando la confidencialidad de sus datos personales mediante autenticación segura y almacenamiento cifrado. Esta plataforma tiene como objetivo ofrecer recomendaciones adaptativas basadas en el análisis del desempeño físico del usuario, desde las fases de calentamiento hasta la ejecución de su rutina diaria. La aplicación genera consejos personalizados orientados a mejorar el rendimiento, prevenir lesiones y favorecer la adherencia a los objetivos planteados por el usuario.

3.1.9 Estructura de la aplicación de seguimiento personalizado

a. Importación de librerías

En el desarrollo de la aplicación móvil del sistema, se implementó el uso del framework Flutter junto con la biblioteca MQTT para habilitar la comunicación en tiempo real con el servidor o microcomputador. Mediante la importación de material.dart, se construye una interfaz visual amigable e intuitiva para el usuario. A través de los paquetes mqtt_client y mqtt_server_client, la aplicación se conecta a un broker MQTT, permitiendo recibir notificaciones, estados del sistema o alertas desde el entorno de control. Esta integración es fundamental para mantener una interacción constante y bidireccional entre el usuario y el sistema de monitoreo, garantizando una experiencia fluida y dinámica en la gestión de accesos.

```dart
import 'package:flutter/material.dart';
import 'package:mqtt_client/mqtt_client.dart';
import 'package:mqtt_client/mqtt_server_client.dart';
```

Figura 19 Importación de librerías

b. Ventanas de navegación

En la pantalla de inicio de sesión forma parte fundamental de la aplicación móvil desarrollada en Flutter, permitiendo gestionar el acceso de los usuarios al sistema. Esta interfaz dinámica permite capturar las credenciales ingresadas, validarlas y controlar el flujo de navegación según el estado del usuario. Su diseño responde a eventos en tiempo real, como la escritura en campos de texto o la confirmación de ingreso, asegurando una experiencia fluida. Además, permite integrar lógicas de autenticación y seguridad que restringen el acceso solo a usuarios registrados, reforzando el control sobre el sistema de monitoreo y seguimiento personalizado.

```dart
class LoginScreen extends StatefulWidget {
  const LoginScreen({super.key});
  @override
  State<LoginScreen> createState() => _LoginScreenState();
}
```

Figura 20 Primera ventana de navegación

La interfaz correspondiente a los datos de entrenamiento está diseñada para presentar información relevante al usuario de forma estructurada y estéticamente optimizada. Cuenta con una barra superior que identifica el módulo y una zona de contenido con un fondo de degradado lineal, el cual contribuye a una experiencia visual agradable sin comprometer la legibilidad. Esta pantalla tiene como propósito mostrar indicadores relacionados con el rendimiento, evolución o historial de actividades del usuario, provenientes del sistema de seguimiento. La disposición visual favorece la jerarquización de la información y permite su rápida interpretación, lo que facilita la toma de decisiones por parte del usuario. Este componente de la aplicación cumple una función informativa clave dentro del ecosistema digital del sistema.

```
class DatosEntrenamientoScreen extends StatelessWidget {
  const DatosEntrenamientoScreen({super.key});
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(title: const Text('Datos de Entrenamiento')),
      body: Container(
        decoration: const BoxDecoration(
          gradient: LinearGradient(
            colors: [
              Color(0xFFcde6fb),
              Color(0xFF1c7ade),
```

Figura 21 Ventana de navegación

c. Comunicación para obtención de datos

Para permitir la comunicación en tiempo real entre la aplicación móvil y el sistema de monitoreo, se integró una librería compatible con el protocolo MQTT. Esta herramienta permite establecer una conexión eficiente con un broker MQTT, habilitando el envío y recepción de mensajes entre el dispositivo móvil y los componentes del sistema. Como resultado de esta integración, la aplicación puede recibir notificaciones sobre eventos como ingresos detectados, alertas de seguridad o recomendaciones personalizadas. La implementación de esta comunicación ligera y de baja latencia asegura una interacción continua, confiable y en tiempo real, optimizando el control de acceso y la experiencia del usuario.

```
29    # versions available, run `flutter pub outdated`.
30    dependencies:
31      flutter:
32        sdk: flutter
33      mqtt_client: ^9.7.2
```

Figura 22 Broker MQTT

Dentro del sistema, la configuración de la comunicación mediante el protocolo MQTT se realiza estableciendo los parámetros necesarios para conectarse a un broker, incluyendo la dirección del servidor y el identificador del cliente móvil. Esta configuración permite que la aplicación mantenga una conexión activa y segura con el sistema, facilitando el envío y recepción de mensajes entre los distintos componentes de la arquitectura. Como resultado de esta estructura, se logra una comunicación ligera, eficiente y en tiempo real, ideal para gestionar eventos relacionados con el control de acceso, monitoreo del sistema y actualización de estado del usuario dentro del entorno inteligente.

```
class MQTTService {
  final String broker = 'test.mosquitto.org'; // Cambia esto por tu broker MQTT
  final String clientId = 'flutter_client';
  final String topic = 'esp32/sensores';
```

Figura 23 Protocolo MQTT

3.1.10 Descripción del dispositivo portátil

El dispositivo electrónico central del sistema está compuesto por una placa de desarrollo ESP32, que proporciona conectividad inalámbrica WiFi y capacidad de procesamiento suficiente para ejecutar tareas de monitoreo en tiempo real. A esta placa se conecta un sensor ECG, que permite la captura continua de señales eléctricas del corazón del usuario durante su actividad física. Esta información biométrica es clave para la personalización del entrenamiento y la detección de posibles anomalías fisiológicas. La alimentación del sistema se realiza mediante una batería LiPo, que ofrece un equilibrio entre autonomía y portabilidad, permitiendo un uso prolongado sin comprometer el rendimiento. Todo este conjunto trabaja en conjunto con la aplicación móvil y el sistema de inteligencia artificial, que recibe los datos en tiempo real para generar alertas, recomendaciones personalizadas y reportes de salud y desempeño físico.

3.1.11 Estructura del dispositivo portátil

a. Codificación

La codificación del sistema electrónico se desarrolló en el entorno Arduino IDE, utilizando un ESP32 como unidad central de procesamiento. Se implementó una lectura continua de datos provenientes del sensor ECG, que son procesados y filtrados para su análisis en tiempo real. A través de una conexión WiFi, el sistema transmite la información recopilada a una plataforma de visualización remota. El programa también incorpora rutinas de ahorro energético, controlando la frecuencia de muestreo y el envío de datos. Para asegurar la estabilidad del sistema, se manejan interrupciones y validaciones de señal del sensor. La batería LiPo alimenta el circuito, regulada mediante un LM7805, mientras el código supervisa constantemente el voltaje de entrada. La lógica de control fue optimizada para garantizar un monitoreo preciso, estable y continuo del estado fisiológico del usuario.

b. Diagrama esquemático

En la Figura 22 se muestra el sistema electrónico está compuesto por una placa ESP32, un sensor ECG y una batería Li-Po, todos integrados mediante un esquema de alimentación regulada. La batería de 7.3 V entrega energía al sistema, pasando por un regulador LM7805 que convierte la tensión a 5 V para proteger los componentes. El sensor ECG se conecta a la entrada analógica de la ESP32 para la lectura de señales eléctricas del corazón en tiempo real. Las conexiones de tierra (GND) están unificadas para garantizar una referencia común y evitar diferencias de potencial. El esquema permite una adquisición estable de datos con bajo consumo energético. Esta configuración garantiza portabilidad y eficiencia en sistemas de monitoreo físico en tiempo real.

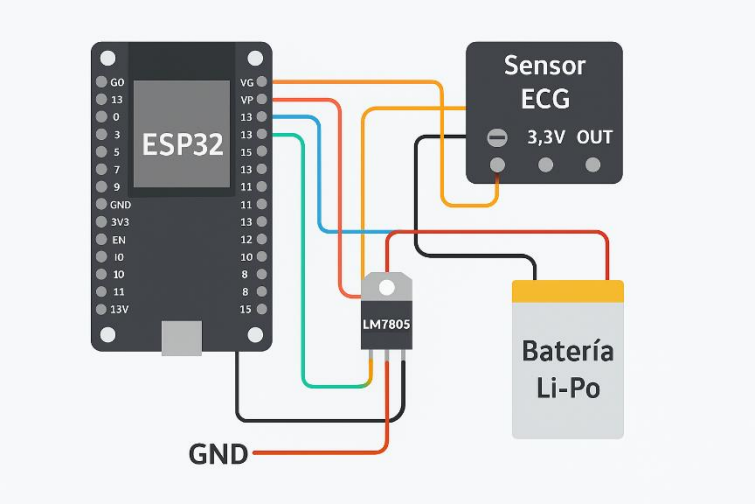


Figura 24 Diagrama esquemático del sistema portátil

c. Comunicación

El sistema implementa comunicación inalámbrica mediante la conectividad Wi-Fi integrada en la placa ESP32, lo que permite la transmisión en tiempo real de los datos adquiridos hacia la aplicación móvil. Esta interfaz inalámbrica elimina la necesidad de conexiones físicas adicionales, facilitando la portabilidad del dispositivo y su integración con plataformas de monitoreo. Gracias al protocolo TCP/IP, la ESP32 puede enviar datos procesados, como señales del sensor ECG permitiendo así el seguimiento personalizado del usuario desde cualquier ubicación con acceso a la red. Esta arquitectura es clave para garantizar la sincronización continua entre el dispositivo físico y el sistema de inteligencia artificial.

3.2 Resultados

A continuación, se presentan los resultados del entrenamiento del sistema de inteligencia artificial, desglosados capa por capa para facilitar una comprensión más clara de su funcionamiento.

3.2.1 Resultados de entrenamiento

a. Matriz de confusión normalizada

A partir de los archivos generados tras el entrenamiento del modelo YOLOv8, se obtuvo la matriz de confusión normalizada, la cual evalúa el desempeño del sistema en la clasificación de dos clases: "rostro" y "background". El análisis de esta matriz revela que el modelo logró una precisión destacada en la detección de rostros, con un valor del 99% en la predicción correcta de esta clase. Asimismo, la tasa de falsos positivos fue mínima, evidenciando un valor del 1% en la confusión con el fondo. Estos resultados reflejan un comportamiento altamente confiable del modelo para la tarea de reconocimiento facial, lo cual es fundamental en el contexto del sistema de control de acceso propuesto.

Tabla 14 Predicción de resultados obtenidos

Clase	**Valor**	**Porcentaje**
Rostro	0.99	99 %
Background	0.01	1 %

En la Figura 23 se presentan los resultados de las predicciones realizadas por el modelo en relación con las clases. La matriz de confusión evidencia una clasificación precisa entre ambas categorías.

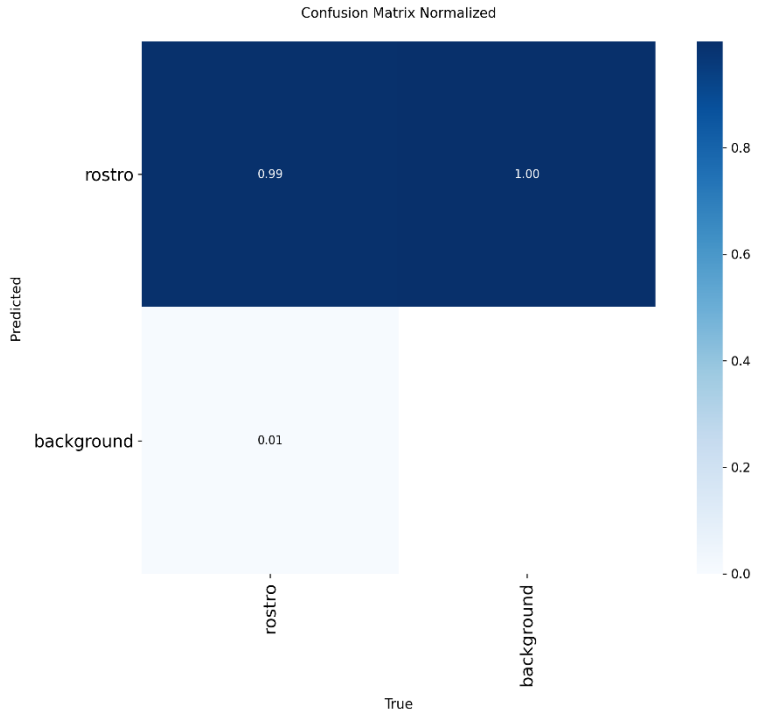


Figura 25 Matriz de confusión

b. Métricas de evaluación

- ***Precisión***

La precisión representa la proporción de predicciones correctas sobre el total de predicciones positivas realizadas por el modelo, evaluando su capacidad para evitar falsos positivos. En la Figura 24, correspondiente a la evolución de la precisión durante el entrenamiento, se observa una tendencia creciente y estable a lo largo de las épocas. El modelo alcanza un valor cercano al 99% de precisión en las últimas iteraciones, lo que indica un alto nivel de exactitud en la identificación de los casos positivos. Esto refleja un comportamiento confiable y coherente en su capacidad predictiva dentro del sistema.

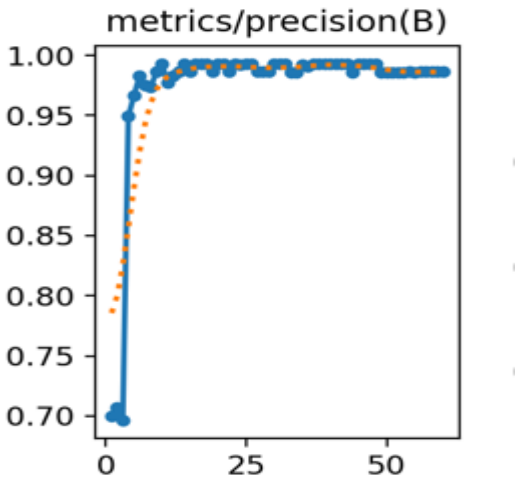


Figura 26 Métrica de precisión

- ***Confianza***

La puntuación F1 se evalúa en función de diferentes niveles de confianza, reflejando el equilibrio entre la precisión y la sensibilidad del modelo. En la Figura 25 se presenta la curva de confianza generada para la clase "rostro", donde se observa un valor máximo de F1 cercano a 0.99 alcanzado con un nivel de confianza de aproximadamente 0.673. Este comportamiento indica que el modelo mantiene un excelente desempeño predictivo cuando se encuentra dentro de ese umbral de decisión, consolidando su fiabilidad en tareas de detección facial.

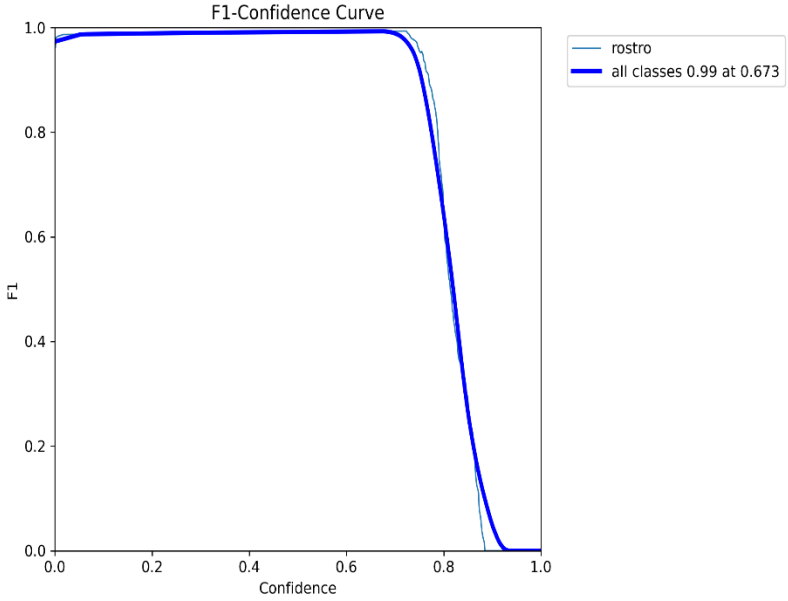


Figura 27 Curva de confianza -F1

- ***Sensibilidad - recall***

La sensibilidad representa la proporción de objetos reales correctamente identificados por el modelo frente al total de instancias existentes de una clase específica. En la Figura 26 se observa un incremento rápido durante las primeras épocas, estabilizándose cerca del valor de 1.0 hacia el final del entrenamiento. Esto indica que el modelo logra reconocer con alta eficacia casi la totalidad de los objetos de interés presentes en los datos de entrenamiento, mostrando un comportamiento robusto y consistente en la detección de la clase objetivo.

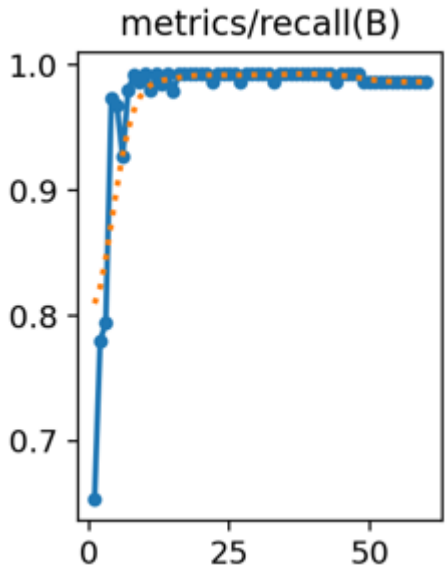


Figura 28 Métricas recall

En la Figura 27 se observa que el modelo alcanzó un valor de recall de 99% para todas las clases combinadas, evidenciado en el punto máximo de la curva. La forma plana de la gráfica indica que el modelo mantiene una alta sensibilidad en un amplio rango de predicciones.

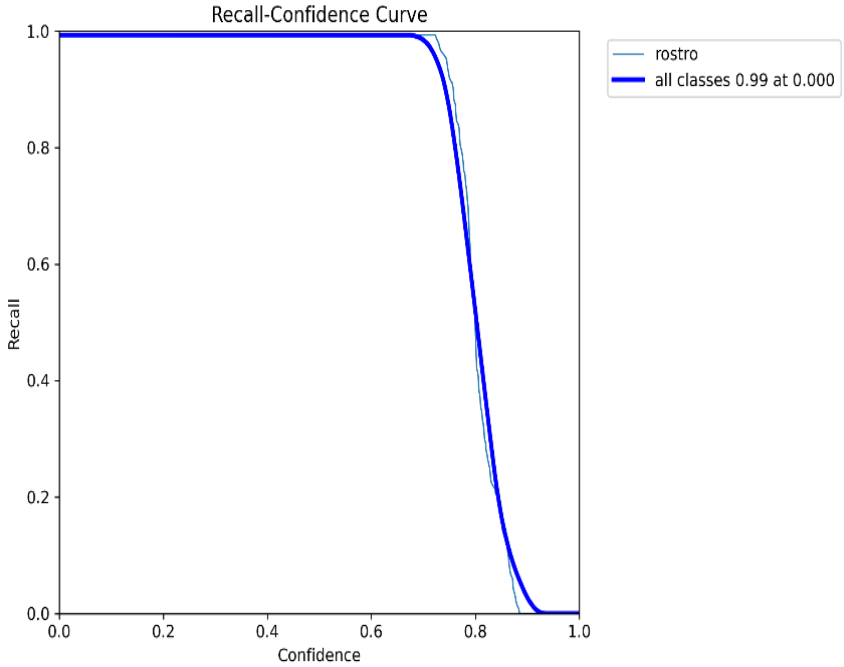


Figura 29 Curva de sensibilidad

- ***Métrica mAP_50***

La métrica mAP50 evalúa la precisión promedio del modelo al identificar correctamente los objetos dentro del conjunto de datos, considerando un umbral de intersección del 50% entre predicciones y realidad. En la Figura 28 se muestra la evolución de esta métrica a lo largo del entrenamiento, donde se observa un rápido ascenso durante las primeras épocas, estabilizándose posteriormente con un valor cercano al 99%. Este resultado indica que el modelo mantiene un alto rendimiento en la detección de los objetos, lo que respalda su eficacia en tareas de reconocimiento visual aplicadas al sistema de control de acceso.

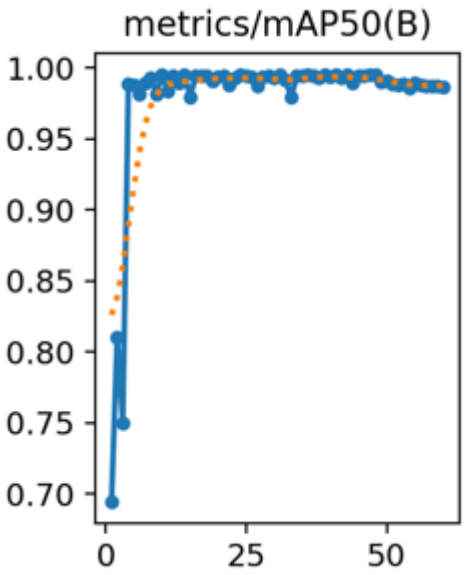


Figura 30 Métrica mAP_50

- ***Métrica mAP_50:95***

El mAP50:95 es una métrica más estricta que evalúa la precisión promedio del modelo considerando múltiples umbrales de intersección, desde 0.5 hasta 0.95, lo que permite medir su rendimiento bajo distintos niveles de exigencia. En la Figura 29 se aprecia que esta métrica inicia con valores bajos, pero muestra un crecimiento progresivo hasta estabilizarse cerca de 0.76 al finalizar las 60 épocas de entrenamiento. Este comportamiento refleja un desempeño sólido del modelo incluso bajo condiciones más rigurosas de evaluación, lo que indica una buena capacidad generalizada para detectar objetos con diferentes niveles de precisión.

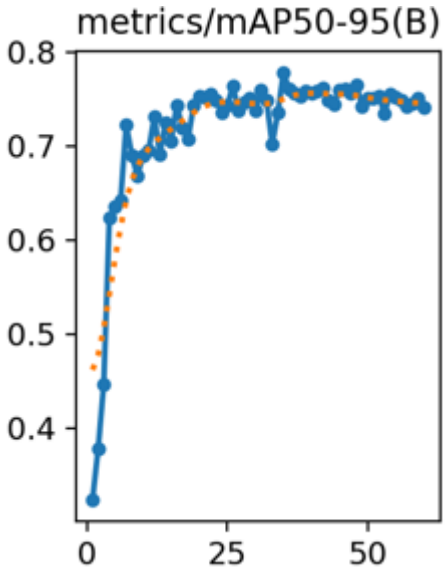


Figura 31 Métrica mAP_50:95

- ***Resultados Generales***

En la Figura 30 se visualizan los resultados generales del proceso de entrenamiento, representados a través de las curvas de pérdida correspondientes a las fases de entrenamiento (train) y validación (val).

Se evalúan tres tipos de pérdida: de caja delimitadora (box_loss), de clasificación (cls_loss) y de distribución de etiquetas (dfl_loss).

En las gráficas de entrenamiento, se aprecia una disminución constante y progresiva de las pérdidas a lo largo de las épocas, reflejando el aprendizaje efectivo del modelo. Por su parte, las curvas de validación muestran una tendencia similar, lo que indica una buena capacidad de generalización hacia datos no vistos. Este comportamiento confirma que el modelo converge correctamente, logrando estabilidad en las últimas etapas del entrenamiento.

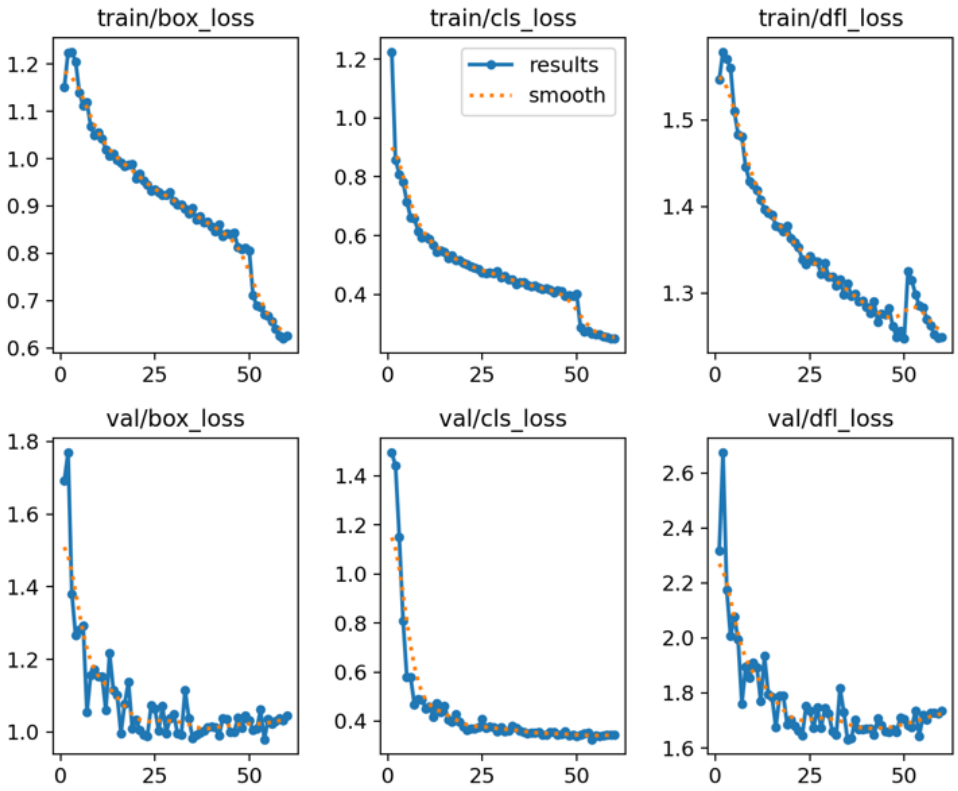


Figura 32 Resultados generales

c. Validación de resultados

Para validar los resultados, se recurre a la matriz de confusión normalizada como métrica clave de evaluación. Esta permite analizar el desempeño del modelo en la clasificación de las clases. Además, la cantidad de datos utilizados en el entrenamiento influye directamente en la fiabilidad de los resultados.

Ecuación 1

$$\boldsymbol{TP, TN, FP, FN = Valor\ (Mf_n) * Canti.Dataset}$$

Donde:

$\boldsymbol{TP}$ = Verdaderos positivos

$\boldsymbol{TN}$ = Verdaderos negativos

$\boldsymbol{FP}$ = Falsos Positivos

$\boldsymbol{FN}$ = Falsos Negativos

$\boldsymbol{Mf_n}$ = Matriz de confusión normalizada

Ecuación 2 es mediante la cual se tiene el valor de Precisión

$$\boldsymbol{Precisión} = \frac{\boldsymbol{TP}}{\boldsymbol{TP+FN}} \quad (1)$$

Ecuación 3 valor de Sensibilidad

$$\boldsymbol{Sensibilidad} = \frac{\boldsymbol{TP}}{\boldsymbol{TP+FN}} \quad (2)$$

Ecuación 4 valor de F1_Score

$$\boldsymbol{F1_{Score}} = \mathbf{2} * \frac{\boldsymbol{Precisión * Recall}}{\boldsymbol{Precisión + Recall}} \quad (3)$$

Ecuación 5 valor de exactitud

$$\boldsymbol{Exactitud} = \mathbf{2} * \frac{\boldsymbol{TP+VN}}{\boldsymbol{TP+FP+TN+FN}} \quad (4)$$

Una vez mostrado las ecuaciones se reemplaza los valores de la matriz de confusión no normalizada.

Tabla 15 Matriz de confusión normalizada

	TP	**FN**	**FP**	**VN**
Rostros	943	94	11	0

Para la obtención de los parámetros del sistema aplicamos las ecuaciones:

$$\boldsymbol{Precisión} = \frac{\mathbf{943}}{\mathbf{943+11}} = \mathbf{0.988}$$

$$\boldsymbol{Sensibilidad} = \frac{\mathbf{943}}{\mathbf{943+94}} = \mathbf{0.909}$$

$$\boldsymbol{F1_{Score}} = \mathbf{2} * \frac{\mathbf{0.988*0.909}}{\mathbf{0.988+0.909}} = \mathbf{0.946}$$

$$\boldsymbol{Exactitud} = \mathbf{2} * \frac{\mathbf{943+0}}{\mathbf{943+11+0+94}} = \mathbf{0.899}$$

$$\boldsymbol{\text{Eficiencia del sistema}} = \left(\frac{\boldsymbol{Precisión + Sensibilidad + F1 + Exactitud}}{\mathbf{4}}\right) * \mathbf{100\%} \quad (5)$$

$$Eficiencia\ del\ sistema = \left(\frac{0.988 + 0.909 + 0.946 + 0.899}{4}\right) * 100\%$$

$$Eficiencia\ del\ sistema = \left(\frac{3.742}{4}\right) * 100\%$$

$$Eficiencia\ del\ sistema = (0.9355) * 100\%$$

$$Eficiencia\ del\ sistema = 93.55\% \approx 94\%$$

$$Eficiencia\ del\ sistema = 94\%$$

3.2.2 Pruebas de funcionamiento

a. Control de acceso

- ***Interfaz principal***

La Figura 31 muestra la interfaz gráfica del módulo de autenticación del sistema "Login - Gym", desarrollada para el acceso seguro de usuarios. El diseño contiene campos de entrada para el nombre de usuario y la contraseña, seguido de un botón funcional denominado "Iniciar sesión", integrándose con el backend para validar credenciales. Su diseño minimalista prioriza la usabilidad y la rápida interacción del usuario con el sistema.



Figura 33 Interfaz principal del control de acceso

- ***Registro de usuario***

En la Figura 32 se muestra un fragmento de código corresponde al proceso de verificación de usuarios dentro del sistema. Evalúa si existe una coincidencia válida y si la similitud supera un umbral de 0.55. Si se cumple esta condición, se muestra un mensaje de bienvenida con el nombre del usuario en color verde y se reproduce un audio indicando el registro exitoso. Además, se muestran los datos del usuario mediante su cédula. En caso contrario, se despliega un mensaje en rojo indicando que el usuario no está registrado y se reproduce un audio de advertencia.

```
if mejor_usuario is not None and mayor_similitud > 0.55:
    label_mensaje.configure(text=f"Usuario {mejor_usuario['Nombre']} {mejor_usuario['Apellido']} bienvenido.", text_color="green")
    reproducir_audio("/home/dario/Desktop/TesisFinal/UsuarioRegistrado.mp3")
    mostrar_datos_usuario(mejor_usuario["Cedula"])
else:
    label_mensaje.configure(text="Usuario desconocido, registrarse primero.", text_color="red")
    reproducir_audio("/home/dario/Desktop/TesisFinal/UsuarioNoRegistrado.mp3")
```

Figura 34 Codificación de registro

- ***Usuario registrado y no registrado***

La Figura 33 presenta la interfaz del sistema de control de acceso al gimnasio, donde se visualizan tres botones principales: “Registrar usuario”, “Acceso” y “Usuarios registrados”. En la parte inferior se muestra un mensaje en color verde que confirma

el ingreso exitoso del usuario Carlos Copo. Esta notificación indica que el sistema ha validado su identidad correctamente. La interfaz fue diseñada para ofrecer un manejo intuitivo y una respuesta inmediata al usuario. El sistema integra funcionalidades de reconocimiento y validación en tiempo real para mejorar la seguridad y trazabilidad del acceso.



Figura 35 Interfaz de usuario registrado

En la Figura 34 se muestra en la parte inferior se despliega un mensaje de advertencia en color rojo que indica: “No se detectó un rostro válido”. Esta notificación es resultado del proceso de reconocimiento facial, el cual no logró identificar una imagen compatible con los registros existentes. El sistema está diseñado para validar únicamente rostros previamente registrados, reforzando la seguridad del acceso. Este tipo de alerta permite una rápida retroalimentación para el usuario y garantiza la integridad del proceso de autenticación.



Figura 36 Usuario no registrado

b. Dispositivo portátil

El dispositivo diseñado integra una placa ESP32 como unidad central de procesamiento y conectividad Wi-Fi. Está equipado con un sensor ECG que permite la adquisición en tiempo real de señales eléctricas del corazón del usuario durante la actividad física. La energía es suministrada por una batería Li-Po de 7.4 V, estabilizada mediante un regulador LM7805 para garantizar el funcionamiento seguro del sistema. Este conjunto portátil permite monitorear parámetros fisiológicos críticos de forma continua. Los datos son enviados a una aplicación móvil para su análisis y visualización personalizada.

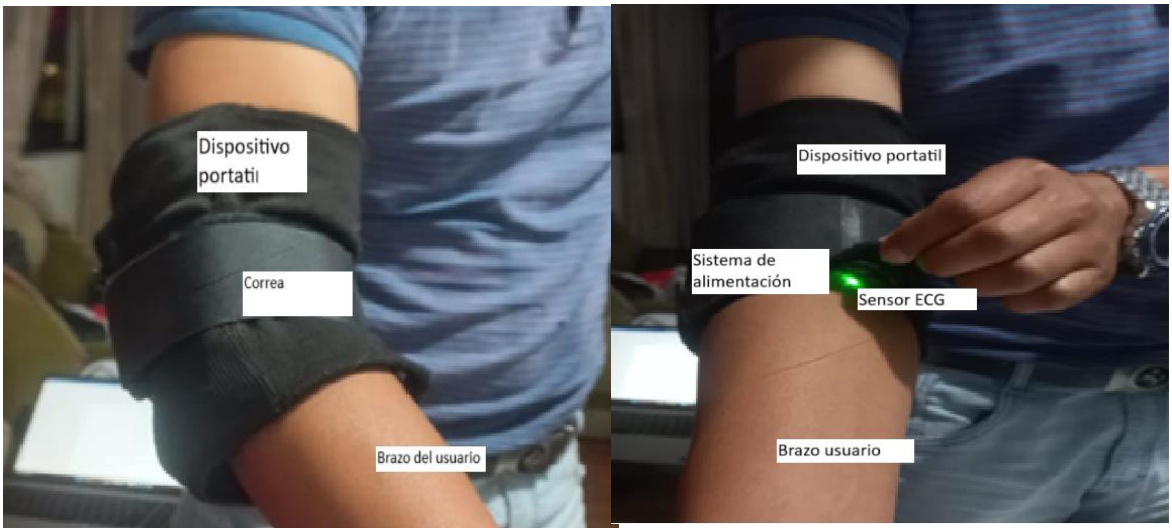


Figura 37 Dispositivo portátil

- ***Lectura de sensores***

La imagen muestra la lectura en tiempo real del sensor ECG, indicando los valores de frecuencia cardíaca en BPM (latidos por minuto) captados durante la actividad física. Los registros varían entre 110 y 118 BPM, lo que refleja una respuesta fisiológica activa y sostenida. Esta información es procesada por el microcontrolador ESP32 y enviada al sistema para su análisis. El monitoreo continuo permite generar alertas y recomendaciones personalizadas basadas en el esfuerzo físico del usuario.

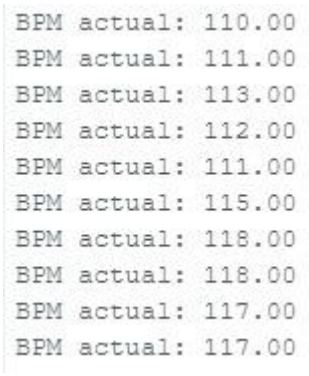


Figura 38 Lectura del sensor ECG

- ***Envió de datos***

El sistema realiza el envío de datos a través de la conectividad Wi-Fi integrada en la placa ESP32, transmitiendo en tiempo real las señales adquiridas por el sensor ECG. Estos datos son enviados hacia una aplicación móvil, donde son procesados y visualizados por el usuario. La comunicación se establece mediante protocolos ligeros que garantizan eficiencia y baja latencia. Esta transmisión continua permite una supervisión personalizada y oportuna del estado físico.

- ***Almacenamiento de datos***

El almacenamiento de datos del sistema se organiza localmente en el entorno de archivos de Ubuntu, dentro de un directorio denominado TesisFinal. En este repositorio se integran archivos Excel que contienen la información estructurada de los usuarios, tales como nombres, apellidos y cédulas, utilizados durante los procesos

de autenticación. Además, se incluye una carpeta llamada rostros, destinada a almacenar las imágenes faciales capturadas durante el registro. El sistema también incorpora archivos de audio en formato .mp3, los cuales se reproducen como respuesta auditiva según el resultado de la identificación. Toda la lógica de operación está gestionada por scripts en Python que acceden, procesan y cruzan los datos registrados. Esta arquitectura garantiza una gestión eficiente, modular y segura de la información del usuario.

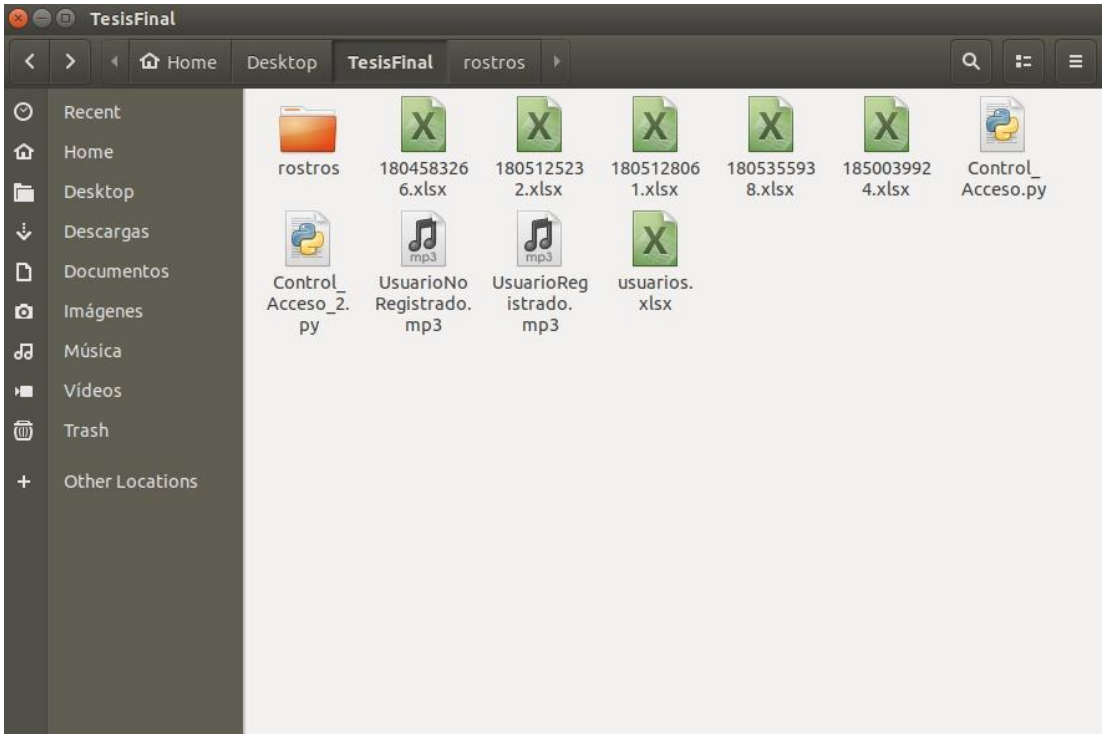


Figura 39 Almacenamiento de datos

b. Aplicación de control personalizado

- ***Icono de la aplicación***

En la Figura 38 muestra un logotipo acompañado del nombre "app_2_gym", que representa una aplicación móvil. Esta app está diseñada para el monitoreo en gimnasios. El diseño indica que fue desarrollada usando Flutter, enfocado en una interfaz intuitiva.

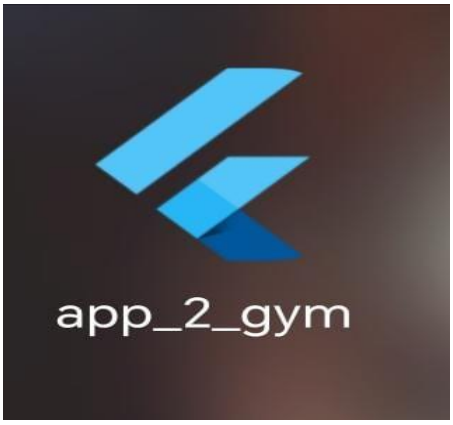


Figura 40 Icono de la aplicación

- ***Inicio de sesión***

En la figura 39 muestra la pantalla principal de la aplicación móvil desarrollada, donde el usuario puede seleccionar entre las opciones "Ingresar" y "Salir". Su diseño es minimalista y funcional, con botones centrados sobre un fondo azul degradado. Esta interfaz inicial permite acceder al sistema de seguimiento personalizado o cerrar la aplicación. Está diseñada para facilitar la navegación desde dispositivos Android.

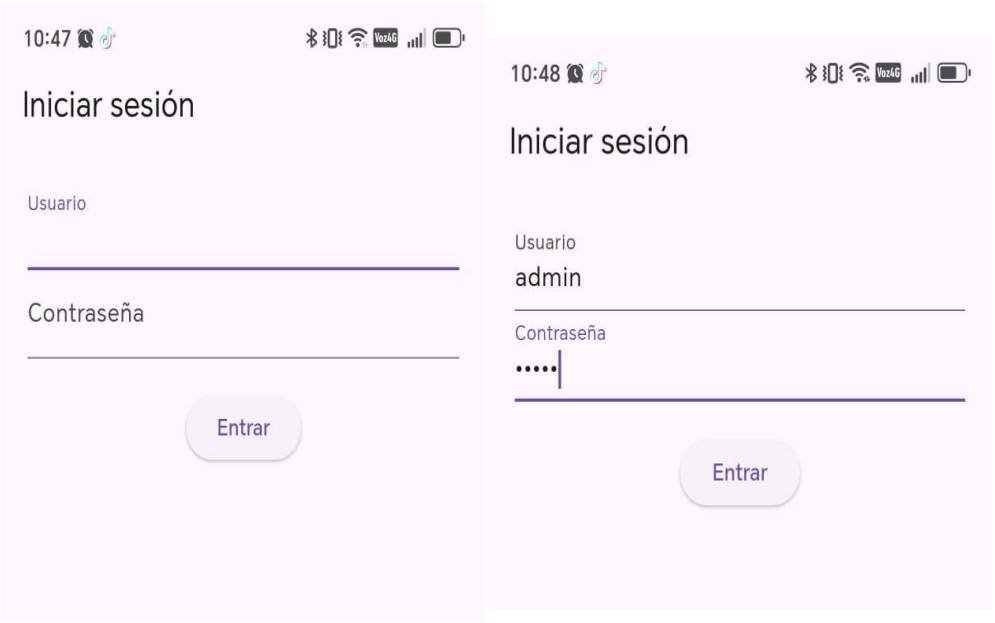


Figura 41 Inicio de sesión

- ***Menú principal***

En la Figura 40 presenta la interfaz gráfica del menú principal de la aplicación móvil. Este menú permite acceder a funciones clave del sistema como visualización de datos de entrenamiento, recomendaciones personalizadas, recordatorios y análisis inteligente. También incluye opciones para configurar la dirección IP del servidor y finalizar la sesión. La disposición de los botones sigue un diseño centrado en la usabilidad, facilitando la interacción del usuario con el sistema.



Figura 42 Menú Principal de la aplicación

- ***Progreso***

En la Figura 41 se muestra la interfaz de navegación interna de la aplicación móvil, donde el usuario accede a tres funciones principales: visualizar los datos de entrenamiento, consultar recomendaciones personalizadas y agregar recordatorios. El diseño mantiene una estética sencilla con botones azules sobre un fondo degradado, facilitando la interacción. Esta pantalla está pensada para ofrecer seguimiento, orientación y gestión de rutinas físicas de forma personalizada.

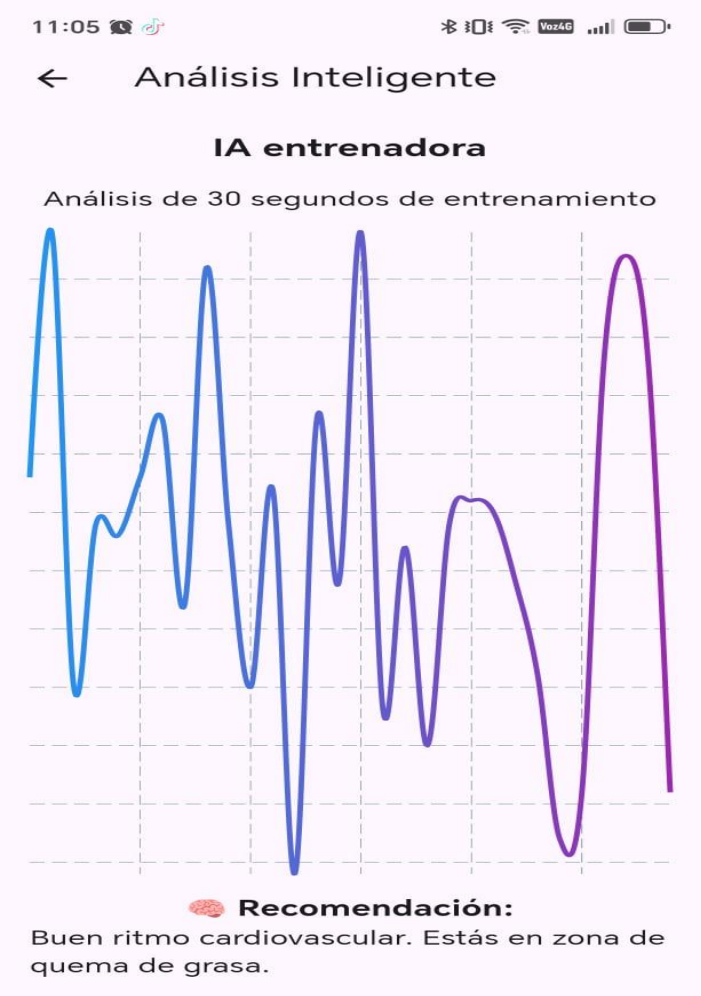


Figura 43 Progreso registrado en la interfaz de la app.

- ***Recomendaciones***

En la Figura 42 se muestra la pantalla de recomendaciones de la aplicación móvil, donde se visualizan sugerencias personalizadas para el usuario. Esta funcionalidad forma parte del sistema de seguimiento físico inteligente y se adapta al estado del usuario según los datos registrados.

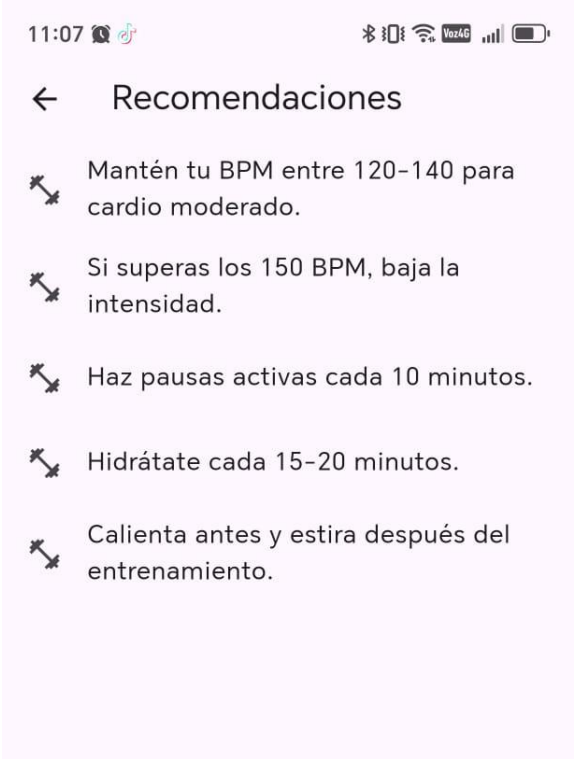


Figura 44 Recomendación de la app y el botón salir

3.2.3 Prototipo final

La imagen presenta el prototipo final del sistema de control de acceso basado en inteligencia artificial, implementado sobre una estación de trabajo funcional. El diseño incluye un ordenador portátil o monitor conectado por un cable HDMI conectado a una estructura elevada que sostiene una cámara DAHUA y en su interior un sistema auditivo para las alertas todo esto conectado a la NVIDIA Jetson NANO, permitiendo la captura de imágenes faciales en tiempo real y alertando al administrador sobre el ingreso a las instalaciones. Esta configuración facilita la detección biométrica desde

un ángulo óptimo, asegurando precisión en el reconocimiento. El entorno de trabajo está organizado con elementos de soporte como ventilación y herramientas de escritura, lo que evidencia un entorno de pruebas controlado. La disposición del hardware refleja una integración efectiva entre los componentes físicos y el software desarrollado. Este montaje fue clave para validar el funcionamiento del sistema en condiciones reales.



Figura 45 Prototipo final

3.2 Costos

En esta sección se detallan los costos asociados a cada uno de los elementos utilizados en la implementación del sistema.

3.3.1 Costos de insumos del proyecto

Tabla 16 Costos de insumos del proyecto

Material	Cantidad	Valor unitario	Valor total
NVIDIA JETSON NANO	1	$ 235.00	$ 235.00
Cámara Dahua	1	$ 65.00	$ 65.00
ESP 32	4	$ 8.00	$ 32.00
Sensor ECG	1	$ 41.52	$ 41.52
Batería de LiPo	1	$ 26.56	$ 26.56
Regulador de voltaje	2	$ 0.80	$ 1.60
Altavoces	1	$ 6.00	$ 6.00
Plataforma en la Nube	3	$ 45.00	$ 135.00
Conectores	10	$ 0.50	$ 5.00
Disco duro	1	$ 255.00	$ 255.00
Materiales de oficina	5	$ 2.50	$ 12.50
Total			$ 809.18

3.2.4 Costos del desarrollo técnico

Para estimar este valor, se consideró el número total de horas laborales invertidas durante las distintas fases del desarrollo del proyecto. Este cálculo incluye actividades como el diseño del sistema, la programación, las pruebas funcionales y la validación de resultados.

Descripción	Cantidad
Tiempo empleado en el proyecto fase de investigación, desarrollo y ejecución de pruebas de funcionamiento	25 días
Horas laborales	8 horas
Horas totales	200 horas

Según los datos oficiales del Ministerio de Trabajo correspondientes al año 2024, el salario básico mensual para un ingeniero en Telecomunicaciones, específicamente en el área de programación, es de $ 504.56 dólares. A partir de este valor, se calculó la tarifa horaria considerando una jornada laboral estándar de 8 horas diarias durante 20 días hábiles al mes. Este cálculo permitió obtener el valor económico por hora de trabajo profesional, como se detalla en la Tabla 16.

Tabla 17 Descripción de un salario de un Ingeniero en Telecomunicaciones

Descripción	**Cantidad**
Valor mensual	504.56
Valor diario	25.23
Valor hora	3,15

Por lo tanto, el costo total esta dado mediante la siguiente ecuación:

$$\boldsymbol{Costo\ total = Valor\ hora * Horas\ totales}$$

$$Costo\ valor\ tecnico = 3.15 * 200$$

$$Costo\ total = \$\ 630.75\ dolares$$

3.2.5 Costo total

Por lo tanto, utilizando la siguiente ecuación para el valor total del proyecto es:

$$\boldsymbol{Costo\ total = Costo\ desarrollo\ tecnico + Costo\ insumos}$$

$$Costo\ total = 630.75 + 809.18$$

$$Costo\ total = \$\ 1439.93\ dolares$$

Por lo tanto, se tiene que el proyecto tiene un costo de 1439.93 dólares americanos

CAPÍTULO IV. CONCLUSIONES Y RECOMENDACIONES

4.1 Conclusiones

- La incorporación de tecnologías avanzadas para el monitoreo y control personalizado en gimnasios ha emergido como un pilar clave para optimizar la experiencia del usuario y potenciar sus resultados físicos. La integración de sensores biométricos, dispositivos portátiles, algoritmos de inteligencia artificial y plataformas móviles posibilita la recopilación, análisis y visualización de datos en tiempo real, permitiendo un seguimiento dinámico y adaptativo. Estas innovaciones no solo favorecen la adherencia a programas de entrenamiento personalizados, sino que también proporcionan retroalimentación inmediata basada en el desempeño individual. En conjunto, estas herramientas impulsan la digitalización del ámbito deportivo, mejorando la eficiencia, motivación y seguridad en la práctica de actividad física.

- El desarrollo de un sistema electrónico basado en inteligencia artificial para el control de acceso en gimnasios ha permitido la integración de tecnologías avanzadas como visión por computadora, reconocimiento facial y procesamiento embebido, ofreciendo una solución segura, eficiente y sin contacto. La combinación de cámaras IP, algoritmos de reconocimiento facial entrenados con bases de datos personalizadas y microcontroladores como la Nvidia Jetson Nano garantiza un acceso automatizado y confiable. Asimismo, la incorporación de una interfaz gráfica mediante CustomTkinter facilita la gestión de registros y la identificación de usuarios en tiempo real. Este enfoque no solo optimiza la administración del gimnasio y refuerza el control de asistencia, sino que también mejora la experiencia de los socios al reducir errores humanos y proporcionar un sistema inteligente y escalable.

- El desarrollo de una aplicación intuitiva para el seguimiento del progreso personalizado de los usuarios del gimnasio ha permitido consolidar una herramienta móvil funcional, moderna y centrada en la experiencia del usuario. Utilizando el framework Flutter, se logró crear una interfaz multiplataforma

con alto rendimiento, compatible con dispositivos Android, facilitando así el acceso a métricas clave como ritmo cardíaco, nivel de actividad y avances físicos. La aplicación se conecta con sensores vía ESP32 utilizando el protocolo MQTT, permitiendo la recepción de datos de forma rápida y su representación gráfica de forma clara y amigable. Esta solución potencia la autonomía del usuario, promueve la constancia en el entrenamiento y convierte al smartphone en un aliado inteligente para alcanzar objetivos de salud y rendimiento físico.

4.2 Recomendaciones

- Para garantizar un rendimiento óptimo en el procesamiento de datos en tiempo real, es fundamental emplear procesadores con arquitectura eficiente y capacidad de cómputo adaptada a las exigencias del sistema. En este contexto, la Jetson Nano destaca por su bajo consumo energético, tamaño compacto y soporte para modelos de inteligencia artificial, lo que la convierte en una opción ideal. Este dispositivo permite ejecutar inferencias rápidas, gestionar múltiples sensores y mantener una conectividad estable con plataformas móviles. Además, su compatibilidad con entornos como Python facilita la integración de algoritmos personalizados, promoviendo un desarrollo ágil, económico y escalable de soluciones para el control y seguimiento inteligente en gimnasios.

- La selección de sensores biométricos de alta precisión y confiabilidad es clave para garantizar un monitoreo personalizado en tiempo real. Se recomienda el uso de dispositivos certificados para la medición de variables como la frecuencia cardíaca, como el Sensor de pulso, debido a su estabilidad, bajo consumo energético y compatibilidad con plataformas como ESP32. Para asegurar lecturas consistentes, es fundamental realizar una calibración inicial y pruebas de validación en condiciones reales de uso. Además, la implementación de algoritmos de filtrado y detección de errores permite

minimizar falsos positivos y adaptarse a distintos perfiles fisiológicos, optimizando la confiabilidad.

- Para una implementación física eficiente del sistema de control de acceso, es recomendable diseñar una estructura compacta y robusta que integre la cámara de reconocimiento facial, el servidor y el actuador de apertura. La unidad debe ubicarse estratégicamente en el ingreso del gimnasio, asegurando una alimentación eléctrica estable y una conexión de red confiable mediante WiFi. Para la comunicación con la aplicación desarrollada en Flutter, se sugiere establecer un canal MQTT seguro entre el dispositivo y la app, permitiendo el envío de notificaciones, actualizaciones del estado de acceso y sincronización con la base de datos del sistema. Este enfoque garantiza una interacción fluida y segura entre el prototipo físico y el entorno digital del usuario.

REFERENCIAS BIBLIOGRÁFICAS

[1] S. E. Schmeelk, M. Ali, and M. Qiu, "Access Control, Biometrics, and the Future," *Proceedings of the 2023 5th International Conference on Image, Video and Signal Processing*, 2023.

[2] J. K. y Andrew Park y J. Killoran y Jasmin Manseau, "Apoyo y humillación de la dignidad con tecnologías biométricas: una perspectiva de asequibilidad," *Journal of Business Ethics*, 2024.

[3] "WorldCoin abusó de nuestros datos biométricos, pero hay más - La Barra Espaciadora." .

[4] H. J. Pimentel Viera, "Aplicación híbrida para el seguimiento progresivo de actividades físicas utilizando el framework flutter," Riobamba, Universidad Nacional de Chimborazo, 2023.

[5] G. E. Can López, "Diseño de un sistema biométrico para el control de asistencia del personal administrativo de la Escuela Superior de Arte de la Universidad de San Carlos de Guatemala," Universidad de San Carlos de Guatemala, 2023.

[6] J. E. M. R. Campos, C. S. C. Rodríguez, L. D. A. Luján, and A. C. M. de los Santos, "Sistema de reconocimiento facial para el control de accesos mediante Inteligencia Artificial," *Innovación y Software*, vol. 4, no. 1, pp. 24–36, 2023.

[7] F. D. Ahumada Reyes, "SISTEMA DE CONTROL Y ADMINISTRACIÓN DE UN CENTRO DE ENTRENAMIENTO (GIMNASIO) DESARROLLADO EN. NET," 2024.

[8] C. C. Betancourt Rodriguez, "Diseño de un sistema de automatización que permita la apertura de puertas de acceso para el ingreso de aeronaves en el hangar de la empresa CIAC LTDA," 2024.

[9] P. P. Torralba, "Qué es la Inteligencia Artificial (IA): La guía definitiva para su comprensión," *Thinking for Innovation*, Mar. 2025.

[10] M. Aicha and A. E. Amina, "Técnicas y aplicaciones del procesamiento de imágenes y señales: un enfoque teórico," *2024 8th International Conference on Image and Signal Processing and their Applications (ISPA)*, pp. 1–8, 2024.

[11] "Aprendizaje automático: algoritmos, modelos y aplicaciones [Título provisional]," *Inteligencia artificial*, 2021.

[12] Q. L. y Mingxuan Zhang y Shiji Zhou y Fanyi Zhao, "Detección de anomalías del tráfico de seguridad de red basada en el método KNN de aprendizaje automático," *Journal of Artificial Intelligence General science (JAIGS) ISSN:3006-4023*, 2024.

[13] A. P. y C. Catal y Amr Abo-eleneen, "El rol del aprendizaje por refuerzo en las pruebas de software," *Inf. Softw. Technol.*, 2023.

[14] S. S. y V. Ravi y Sanchita Basak y Sayak Paul y P. Saikia y Frederick Ditliac Atiah y Vasilios Tsalavoutis y R. Peters, "Una revisión del aprendizaje profundo con especial énfasis en arquitecturas, aplicaciones y tendencias recientes," *ArXiv*, vol. abs/1905.13294, 2019.

[15] J. Sudhakar, V. V. Iyer, and S. T. Sharmila, "Generación de subtítulos de imágenes mediante redes neuronales profundas," *2022 International Conference for Advancement in Technology (ICONAT)*, pp. 1–3, 2022.

[16] H. W. y F. Gao y P. Xie y Longshui Zheng y Kaikai Deng, "Un algoritmo de identificación de usuarios basado en el análisis del comportamiento del usuario en redes sociales," *IEEE Access*, vol. 7, pp. 47114–47123, 2019.

[17] A. B. Amjoud and M. AMROUCH, "Detección de objetos mediante aprendizaje profundo, CNN y transformadores de visión: una revisión," *IEEE Access*, vol. 11, pp. 35479–35516, 2023.

[18] Z. Su, "Análisis comparativo de modelos de detección de objetos basados en CNN: R-CNN, SSD y YOLO más rápidos," *Highlights in Science, Engineering and Technology*, 2025.

[19] S. Bhattacharjee, "Una evaluación del rendimiento basada en la literatura del enfoque CNN YOLO (You Only Look Once) para la detección de objetos en tiempo real," *Computology: Journal of Applied Computer Science and Intelligent Technologies*, 2025.

[20] Y. Luo, "La evolución de YOLO: de YOLOv1 a YOLOv11 con un enfoque en las innovaciones de YOLOv7 en la detección de objetos," *Theoretical and Natural Science*, 2025.

[21] K.-H. Choi and J.-E. Ha, "Método de detección de objetos usando la imagen y el número de objetos en la imagen como etiqueta," *IEEE Access*, vol. 12, pp. 121915–121931, 2024.

[22] "Extracción automatizada de etiquetas de imágenes de informes de radiología - Una revisión."

[23] Q. Lin, J. Wang, H. Liu, and G. Ye, "RoboFlow: un sistema de gestión de flujo de trabajo centrado en datos para el desarrollo de robots mejorados con IA," pp. 1789–1794, 2021.

[24] S. H. y Martin Jägersand y Masood Dehghan y Xuebin Qin, "ByLabel: Una herramienta semiautomática de anotación de imágenes basada en límites," *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 1804–1813, 2018.

[25] "Microcomputadoras y sus aplicaciones para países en desarrollo," 2019.

[26] M. Toscano, S. Valladares, P. Morillo, R. Tufiño, and D. Vallejo-Huanga, "Evaluación del rendimiento del Nvidia Jetson Nano mediante una aplicación de aprendizaje automático en tiempo real," pp. 343–349, 2021.

[27] T. S. y Jonathan Winzig y Mauricio Guajardo Mendoza y Julio César Aldana Almanza, "Edge AI - Caso de uso en Google Coral Dev Board Mini," *2022 IET International Conference on Engineering Technologies and Applications (IET-ICETA)*, pp. 1–2, 2022.

[28] J. F.-A. y Javier Sánchez-Soriano y Sergio Bemposta Rosende y Sergio Ghisler, "Implementación de un sistema de visión de borde en computadoras de placa reducida integradas en vehículos aéreos no tripulados para la gestión inteligente del tráfico," *Drones*, 2023.

[29] "El futuro de la computación en la nube: beneficios y desafíos."

[30] C. B. y Piali Das y Tim Januschowski y D. Arpin y Baris Coskun y Valentin Flunkert y Alex Smola y Edo Liberty y Jan Gasthaus y Ramesh Nallapati y Laurence Rouesnel y Saswata Chakravarty y Sebastian Schelter y Zohar S. Karnin y Bernie Wang y Syama Sundar Rangapuram y Amir Sadoughi y Lorenzo Stella y Julio Delgado y Madhav Jha y David Salinas y Bing Xiang y Yury Astashonok y P. Gautier, "Algoritmos de aprendizaje automático elástico en Amazon SageMaker," *Actas de la Conferencia Internacional ACM SIGMOD 2020 sobre Gestión de Datos*, 2020.

[31] L. Scavuzzo, A. Lodi, N. Yorke-Smith, and K. Aardal, "Machine Learning Augmented Branch and Bound for Mixed Integer Linear Programming," *ArXiv*, vol. abs/2402.05501, 2024.

[32] D. Zaldivar, M. Pérez, and E. Cuevas, "Impacto de los lenguajes de programación en el rendimiento del aprendizaje," *International Journal of Information and Communication Technology Education*, 2025.

[33] "REVISIÓN COMPARATIVA DE FLUTTER Y EL DESARROLLO DE APLICACIONES TRADICIONALES DE ANDROID," *Revista Internacional de Investigación de Modernización en Tecnología y Ciencia de la Ingeniería*, 2023.

[34] E. A. Alsumait, "Tipos y aplicaciones de sensores," 2020.

[35] C.-Y. Yang, C.-L. Yang, and Z.-X. Chen, "Clasificación de sensores mediante una red neuronal convolucional mediante la codificación de series temporales multivariadas como imágenes coloreadas bidimensionales," *Sensors (Basel, Switzerland)*, vol. 20, 2019.

[36] J. Nakamura, "Sensores de imagen y procesamiento de señales para cámaras digitales," 2005.

[37] "Cámara WiFi Rotatoria Dahua. IPC-H4A," *Tienda Soultek*. .

[38] M. Uddin, S. Islam, and A. Al-Nemrat, "A Dynamic Access Control Model Using Authorising Workflow and Task-Role-Based Access Control," *IEEE Access*, vol. 7, pp. 166676–166689, 2019.

[39] Televistazo, “Ecuador experimenta un boom en el número de gimnasios y clientes tras la pandemia,” *www.ecuavisa.com*. May-2025.

[40] L. T. Castillo, “Tipos de Rostro | PDF | Autosuperación,” *Scribd*. Oct-2020.

[41] D. Pereira, L. Pereira, C. Sherrington, D. Felício, and A. Picorelli, “La adherencia a los programas de ejercicio para personas mayores está influenciada por las características del programa y factores personales: una revisión sistemática.,” *Journal of physiotherapy*, vol. 60 3, pp. 151–6, 2014.

[42] comunicacion, “Sala Fitness: Actividades y Equipamiento,” *Ciencias Deportivas*. May-2024.

ANEXOS

Anexo A. Especificaciones técnicas de la NVIDIA Jetson Nano



Descripción

NVIDIA Jetson Nano Developer Kit es una computadora pequeña y poderosa que le permite ejecutar múltiples redes neuronales en paralelo para aplicaciones como clasificación de imágenes, detección de objetos, segmentación y procesamiento de voz. Todo en una plataforma fácil de usar que funciona con tan solo 5 vatios.

Jetson Nano ofrece 472 GFLOPS para ejecutar algoritmos modernos de IA rápidamente, con una CPU ARM de 64 bits de cuatro núcleos, una GPU NVIDIA integrada de 128 núcleos y una memoria LPDDR4 de 4 GB. Ejecuta múltiples redes neuronales en paralelo y procesa varios sensores de alta resolución simultáneamente.

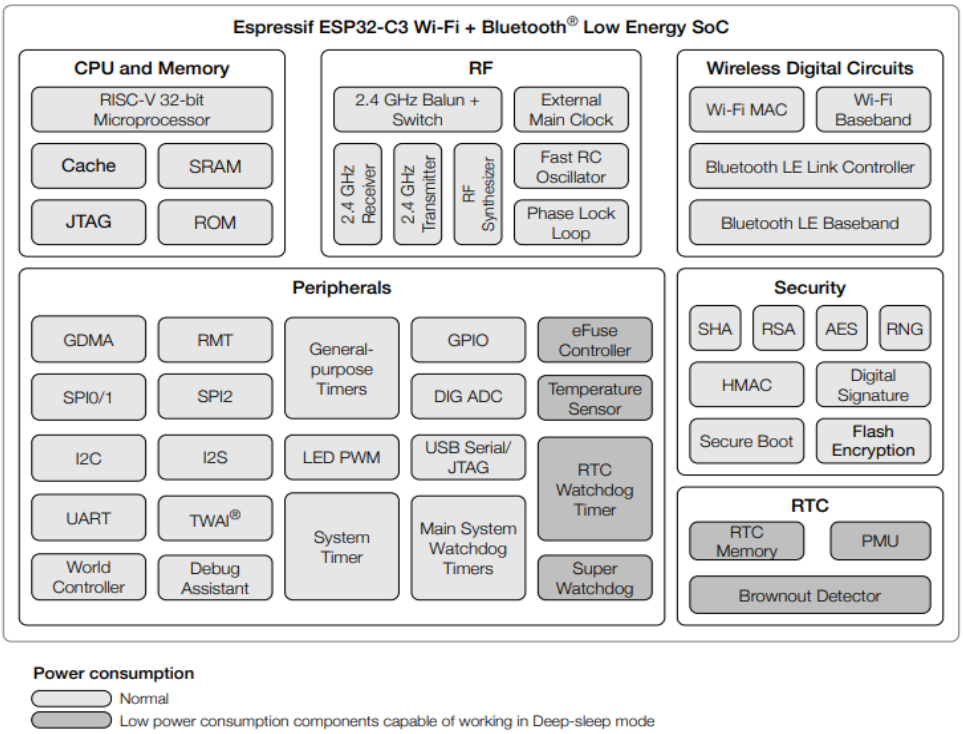
Características

- GPU: 128-CORE MAXWELL
- CPU: Quad-core ARM A457 @ 1.43 GHz
- Memoria: 4 GB 64-bit LPDDR4 25.6 GB/s
- Almacenamiento: tarjeta micro-SD (NO incluida. Recomendación: tarjeta Micro-SD 16GB)
- Codificador de video: 4K @ 30 | 4x 1080p @ 30 | 9x 720p @ 30 (H.264/H.265)
- Decodificador de video: 4K @ 60 | 2x 4K @ 30 | 8x 1080p @ 30 | 18x 720p @ 30 (H.264/H.265)
- Cámara: 2 carriles MIPI CSI-2 DPHY
- Conectividad: Gigabit Ethernet, conector de expansión M.2 Key E (Recomendación: NIC de modo dual AC8265).
- Monitor: HDMI y DP
- USB: 4x USB 3.0, USB 2.0 Micro-B
- Interfaces: GPIO, I2C, I2S, SPI, UART

Anexo 2 Especificaciones técnicas de la NVIDIA Jetson

Anexo B1. Especificaciones técnicas de la ESP32

The functional block diagram of the SoC is shown below.



ESP32-C3 Functional Block Diagram

Wi-Fi

- IEEE 802.11b/g/n-compliant
- Supports 20 MHz, 40 MHz bandwidth in 2.4 GHz band
- 1T1R mode with data rate up to 150 Mbps
- Wi-Fi Multimedia (WMM)
- TX/RX A-MPDU, TX/RX A-MSDU
- Immediate Block ACK
- Fragmentation and defragmentation
- Transmit opportunity (TXOP)
- Automatic Beacon monitoring (hardware TSF)
- Four virtual Wi-Fi interfaces
- Simultaneous support for Infrastructure BSS in Station mode, SoftAP mode, Station + SoftAP mode, and promiscuous mode
  Note that when ESP32-C3 scans in Station mode, the SoftAP channel will change along with the Station channel
- Antenna diversity
- 802.11mc FTM

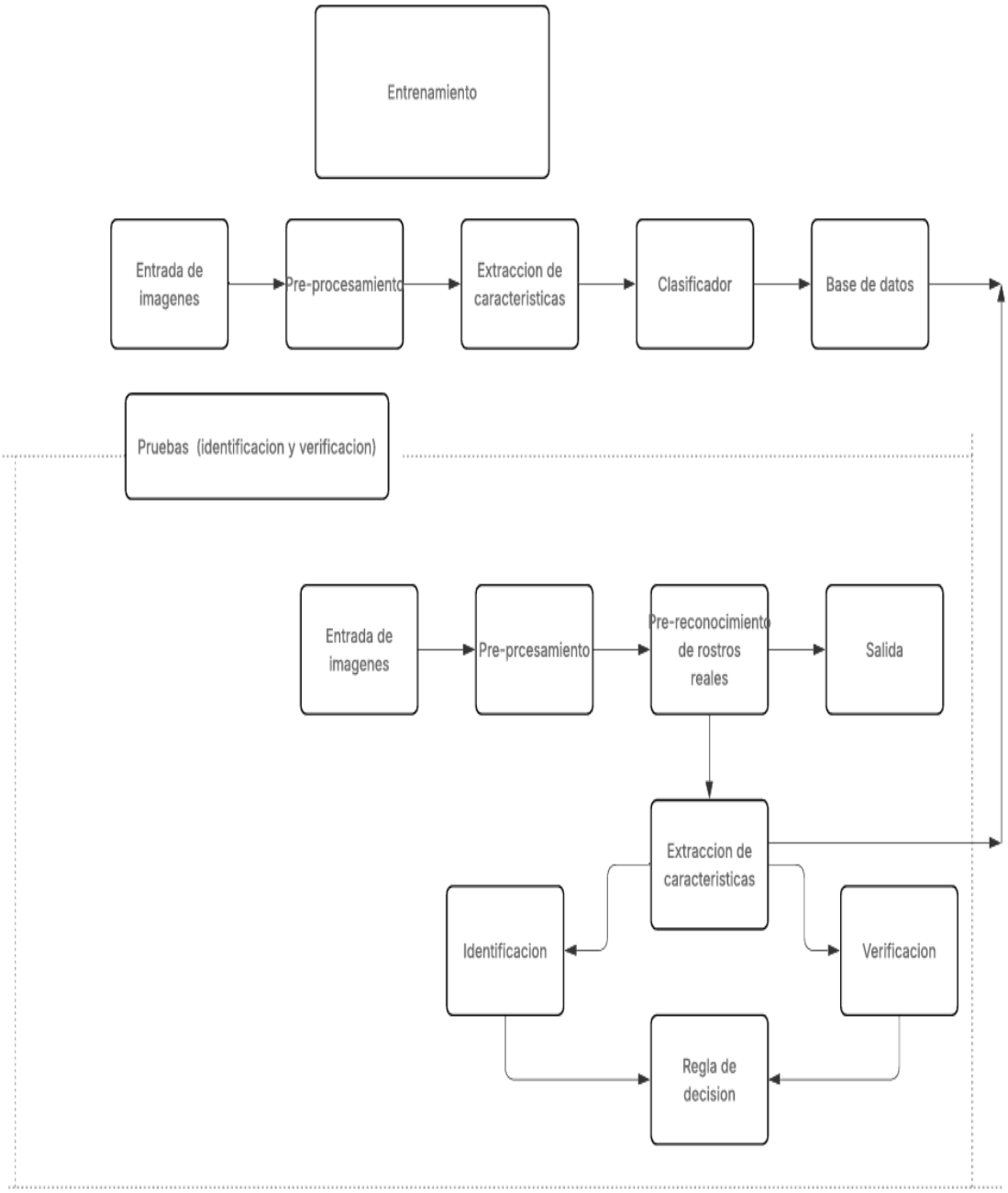
Anexo 3 Especificaciones tecnicas de la ESP32

Anexo C. Especificaciones tecnicas de la camara Dahua

Modelo	HCVR4104CNS3
Sistema	
Procesador principal	Procesador integrado
Sistema operativo	Linux embebido
Monitor	
Interfaz	1 HDMI, 1 VGA
Resolución	1920×1080, 1280×1024, 1280×720, 1024×768
Partición de pantalla de canales	1/4/6
OSD	Título de cámara, tiempo, pérdida de video, bloqueo de cámara, detección de movimiento, grabación
Detección y Alarma de video	
Activadores de eventos	Grabación, PTZ, Tour, salida de alarma, empuje de video, correo electrónico, FTP, Snapshot, Buzzer & Screen tips
Detección de video	Detección de movimiento, zonas MD: 396 (22 × 18), pérdida de video y manipulación
Entrada de alarma	N/A
Salida relay	N/A
Reproducción y copia de seguridad	
Reproducción	1/4
Modo de búsqueda	Hora/ fecha, alarma, MD y búsqueda exacta
Funcion de reproducción	Reproducir, pausa, detener, rebobinado, reproduccón rápida, reproducción lenta, siguiente archivo, archivo anterior, siguiente cámara, cámara anterior, pantalla completa, repetición, reproducción aleatoria, selección de copia de seguridad, zoom digital
Modo de copia de seguridad	Dispositivo USB / Red
Almacenamiento	
Disco duro interno	1 puertos SATA, con una capacidad de hasta 6 TB
Modo HDD	Individual
Interfaz auxiliar	
USB	2 puertos (USB 2.0)
RS232	N/A
Apoyo de terceros	
Marcas de cámaras soportadas	Dahua, Arecont Vision, AXIS, Bosch, Brickcom, Canon, CP Plus, Dynacolor, Honeywell, Panasonic, Pelco, Samsung, Sanyo, Sony, Videotec, Vivotek, y más
Audio y video	
Entrada de cámaras HDCVI	4 canales

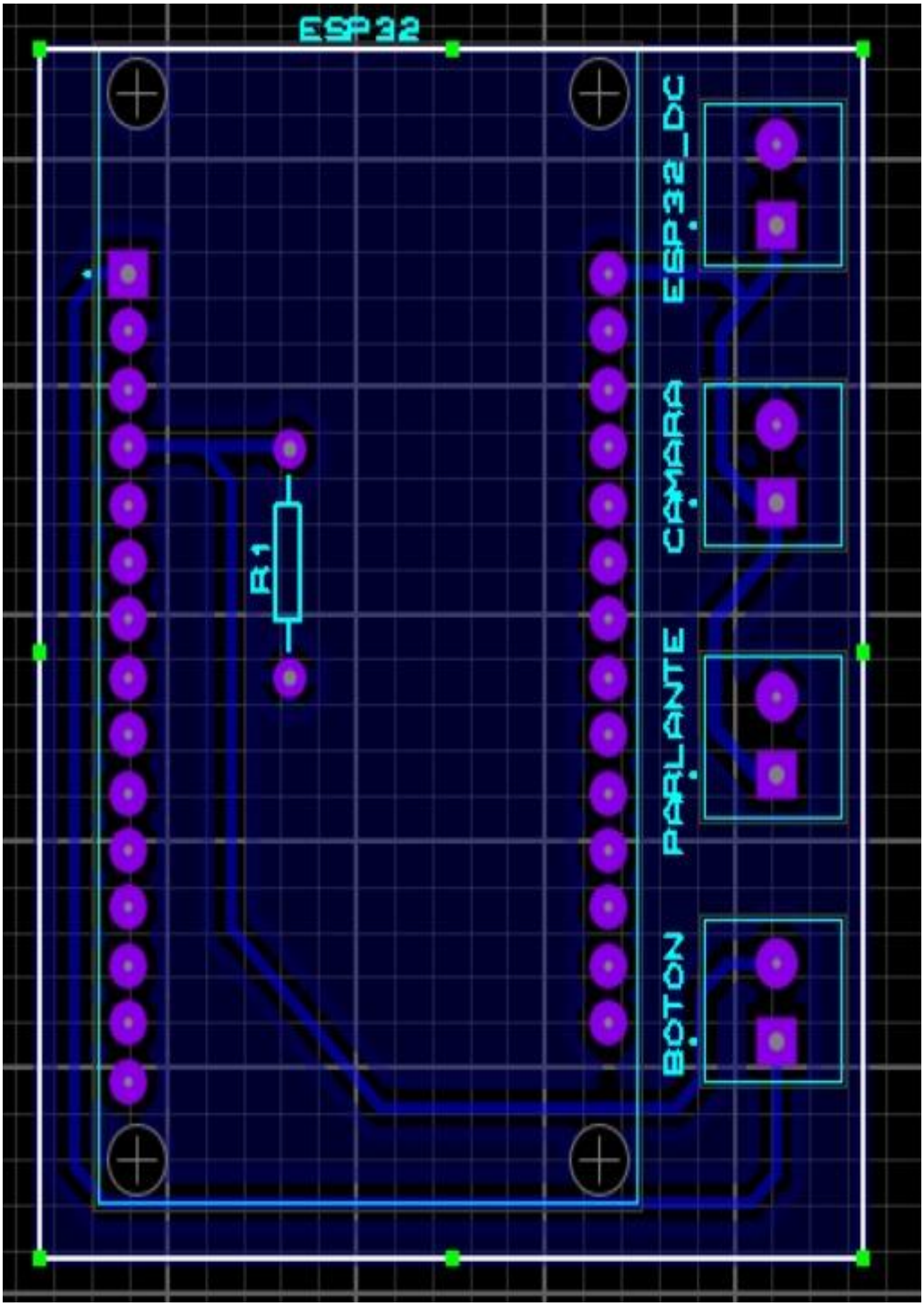
Anexo 4 Especificaciones técnicas de la cámara Dahua

Anexo D. Diagrama esquematico de YOLOv8



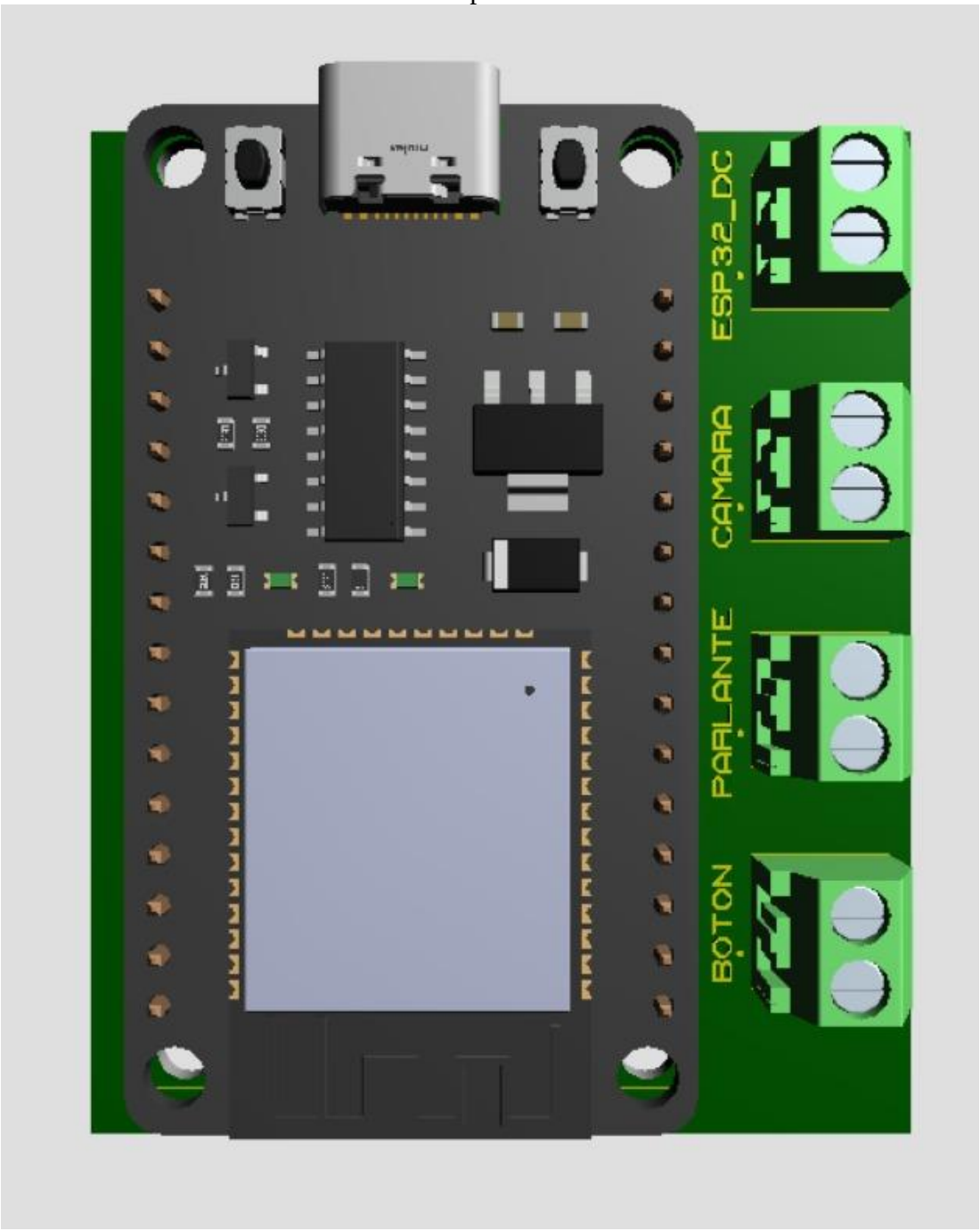
Anexo 5 Diagrama esquemático de YOLOv8

Anexo E. Diagrama Esquematico de la placa.



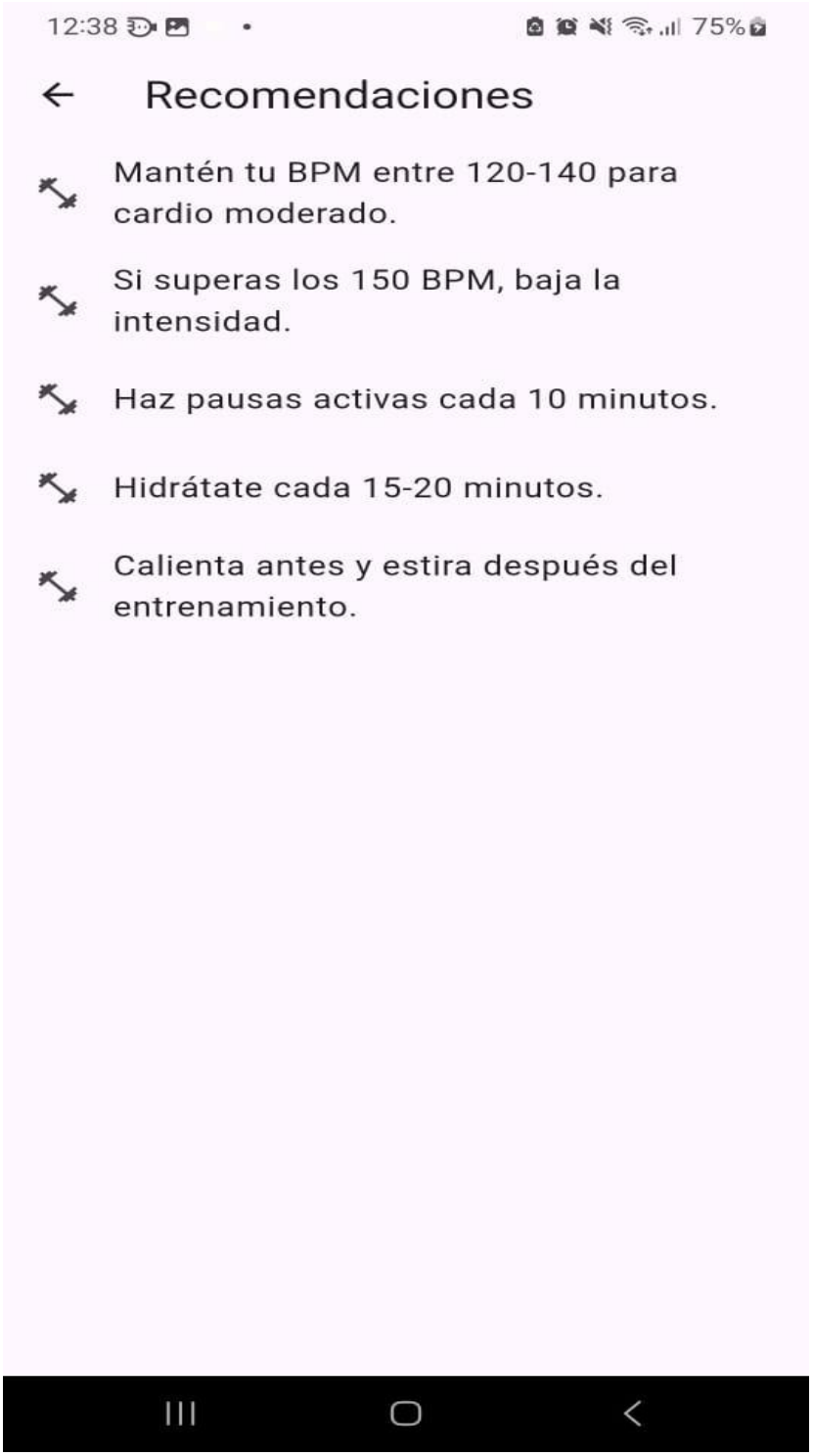
Anexo 6 Diagrama esquemático de la placa

Anexo F. Vision 3D de la placa montada sus elementos.



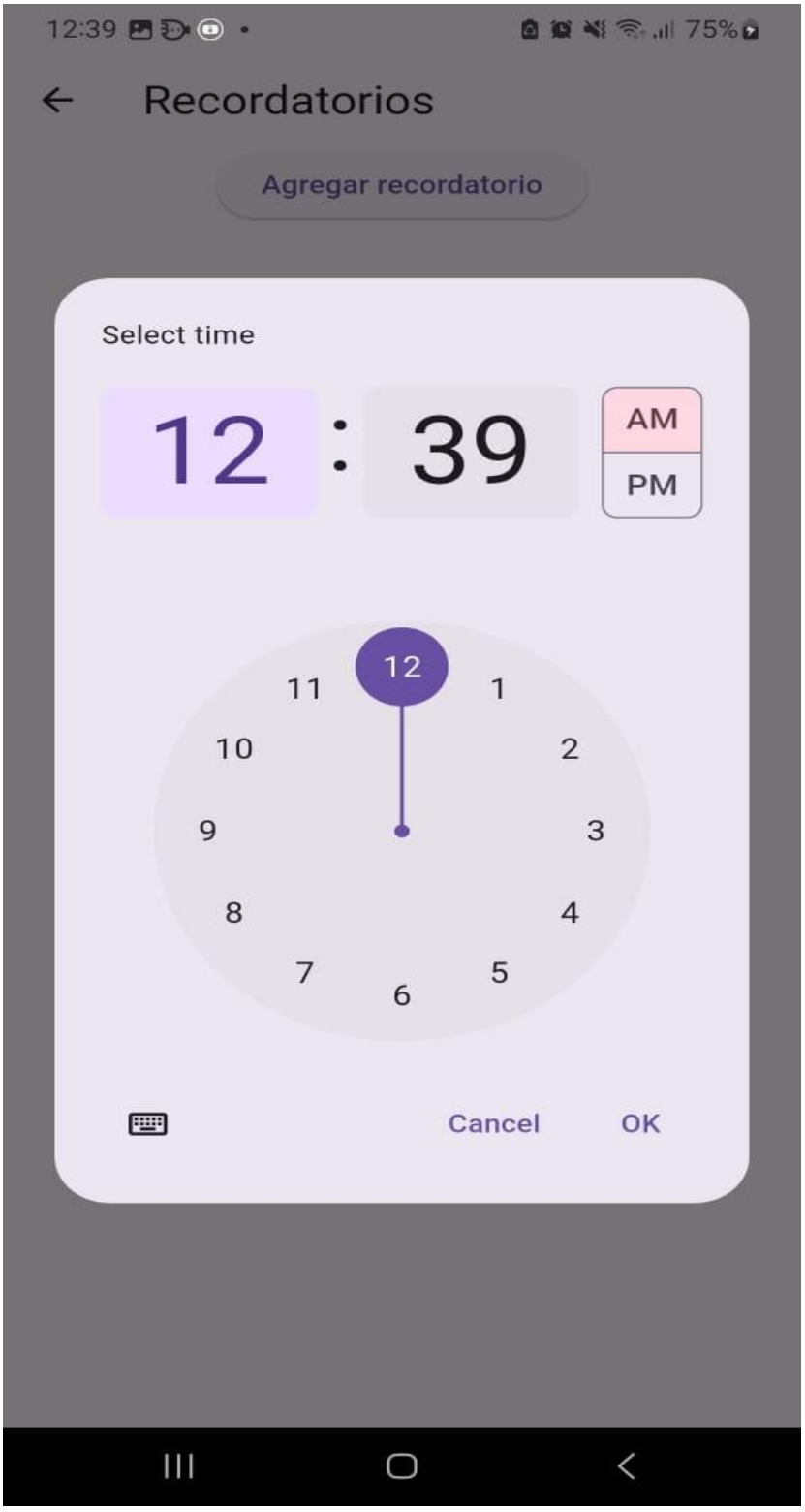
Anexo 7 Visión 3D de la placa.

Anexo G. Recomendaciones de la aplicación movil



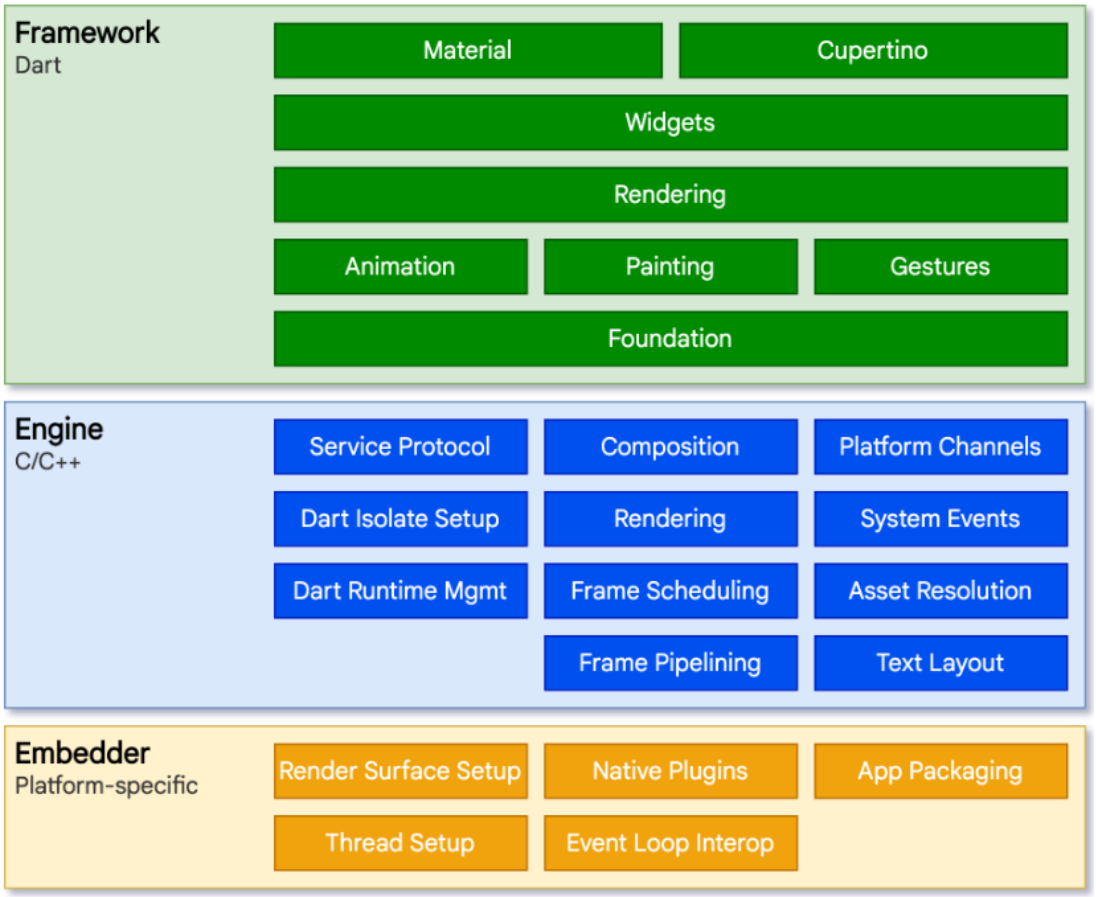
Anexo 8 Recomendaciones emitidas por la app al usuario.

Anexo H. Recordatorio dentro de la aplicación movil



Anexo 9 Recordatorio emitido por la app.

Anexo I. Arquitectura de Flutter con Android Studio.



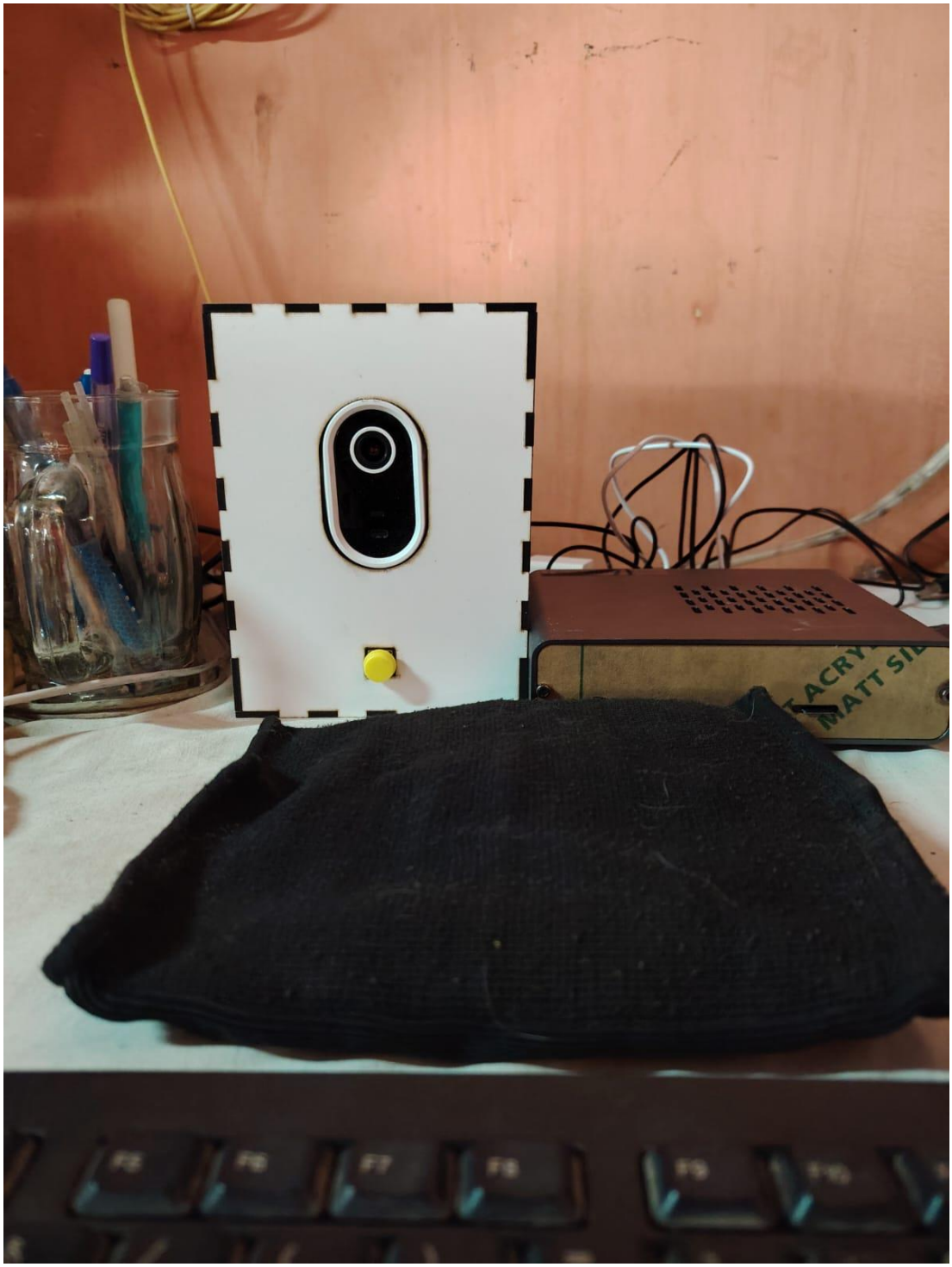
Anexo 10 Flutter con Android Studio

Prototipo Final con sus características

- Reconocimiento facial en tiempo real mediante el algoritmo YOLOv8, entrenado con imágenes etiquetadas para identificar usuarios al ingresar al gimnasio.
- Cámara Dahua HC4 de alta resolución, con visión nocturna y conectividad Wi-Fi, utilizada para capturar imágenes faciales con precisión.
- Procesamiento embebido con la microcomputadora NVIDIA Jetson Nano, que ejecuta los modelos de inteligencia artificial de forma local y eficiente.
- Aplicación móvil desarrollada en Flutter, que permite a los usuarios visualizar métricas de entrenamiento, recibir recomendaciones y registrar su progreso.
- Monitoreo fisiológico con un sensor ECG conectado a una placa ESP32, que mide la frecuencia cardíaca en tiempo real durante la actividad física.
- Sistema de alertas sonoras que notifica el estado del acceso (autorizado o denegado) mediante mensajes de audio personalizados.
- Interfaz gráfica en Python (CustomTkinter) para la gestión de usuarios, registro de datos y control de acceso desde una estación de trabajo.
- Entrenamiento del modelo en Google Colab y gestión de datasets con Roboflow, optimizando el rendimiento del sistema con recursos en la nube.



Anexo 11 Características del prototipo final



Anexo 12 Hardware y Software

Validación de la implementación del sistema



CERTIFICADO

Yo, Jonathan Misael Esparza Reyes en mi calidad de Gerente del GYM ATF FITNESS CENTER, me permito poner en su conocimiento la validación y respaldo del trabajo de titulación: "Sistema integral de control y seguimiento personalizado para gimnasios, basado en inteligencia artificial" propuesto por el estudiante Darío Armando Guamán Coello portador de la Cédula de Ciudadanía: 1805288261 estudiante de la carrera de Telecomunicaciones en la Facultad de Ingeniería en Sistemas Electrónica e Industrial de la Universidad Técnica de Ambato. A nombre de la Institución a la cual represento.

Atentamente.

Jonathan Misael Esparza Reyes

1804794863

Anexo 13 Validación de Sistema

Código de ejecución de Python

```
import cv2
import mediapipe as mp
import customtkinter as ctk
import pandas as pd
import numpy as np
import os
import time
from playsound import playsound
from insightface.app import FaceAnalysis
from sklearn.metrics.pairwise import cosine_similarity
import pygame
from flask import Flask
import threading

# Inicializar FaceNet
face_app = FaceAnalysis(name="buffalo_l")
face_app.prepare(ctx_id=0)

USUARIO_ADMIN = "admin"
CLAVE_ADMIN = "1234"

mp_face_detection = mp.solutions.face_detection
mp_drawing = mp.solutions.drawing_utils

# Archivo Excel de usuarios
archivo_excel = "usuarios.xlsx"
if os.path.exists(archivo_excel):
    df = pd.read_excel(archivo_excel)
else:
```

```
    df = pd.DataFrame(columns=["Nombre", "Apellido", "Cedula", "Foto",
"Embedding"])


def iniciar_login():
    login_window = ctk.CTk()
    login_window.geometry("300x180")
    login_window.title("Login - Gym")

    ctk.CTkLabel(login_window, text="Usuario").pack(pady=5)
    entry_usuario = ctk.CTkEntry(login_window)
    entry_usuario.pack()

    ctk.CTkLabel(login_window, text="Contraseña").pack(pady=5)
    entry_clave = ctk.CTkEntry(login_window, show="*")
    entry_clave.pack()

    label_login = ctk.CTkLabel(login_window, text="", font=("Arial", 11))
    label_login.pack()

    def verificar_credenciales():
        if entry_usuario.get() == USUARIO_ADMIN and entry_clave.get() ==
CLAVE_ADMIN:
            login_window.destroy()
            iniciar_aplicacion()
        else:
            label_login.configure(text="Credenciales incorrectas", text_color="red")

    ctk.CTkButton(login_window, text="Iniciar sesión",
command=verificar_credenciales).pack(pady=10)
    login_window.mainloop()
```

```
def mostrar_lista_usuarios():
    if not os.path.exists(archivo_excel):
        label_mensaje.configure(text="No hay usuarios registrados aún.", text_color="red")
        return

    df_usuarios = pd.read_excel(archivo_excel)
    lista_window = ctk.CTkToplevel(app)
    lista_window.geometry("400x300")
    lista_window.title("Usuarios Registrados")

    ctk.CTkLabel(lista_window, text="Usuarios Registrados", font=("Arial", 14, "bold")).pack(pady=10)

    for index, row in df_usuarios.iterrows():
        texto = f"{row['Cedula']} - {row['Nombre']} {row['Apellido']}"
        ctk.CTkLabel(lista_window, text=texto, font=("Arial", 12)).pack()

def reproducir_audio(ruta):
    try:
        pygame.mixer.init()
        pygame.mixer.music.load(ruta)
        pygame.mixer.music.play()
    except Exception as e:
        print(f"Error al reproducir audio: {e}")

def obtener_embedding(imagen):
    rostros = face_app.get(imagen)
    if rostros:
        return rostros[0].embedding
    return None
```

```
def registrar_usuario():
    registro_window = ctk.CTkToplevel(app)
    registro_window.geometry("400x400")
    registro_window.title("Registro de Usuario")

    entry_nombre = ctk.CTkEntry(registro_window, placeholder_text="Nombre")
    entry_nombre.pack(pady=5)
    entry_apellido = ctk.CTkEntry(registro_window, placeholder_text="Apellido")
    entry_apellido.pack(pady=5)
    entry_cedula = ctk.CTkEntry(registro_window, placeholder_text="Cédula")
    entry_cedula.pack(pady=5)
    entry_peso = ctk.CTkEntry(registro_window, placeholder_text="Peso (kg)")
    entry_peso.pack(pady=5)
    entry_altura = ctk.CTkEntry(registro_window, placeholder_text="Altura (m)")
    entry_altura.pack(pady=5)
    entry_masa = ctk.CTkEntry(registro_window, placeholder_text="Masa muscular
(%)")
    entry_masa.pack(pady=5)

    label_mensaje = ctk.CTkLabel(registro_window, text="", font=("Arial", 12))
    label_mensaje.pack()

    def capturar_rostros():
        nombre = entry_nombre.get()
        apellido = entry_apellido.get()
        cedula = entry_cedula.get()
        peso = entry_peso.get()
        altura = entry_altura.get()
        masa = entry_masa.get()
```

```
        if not nombre or not apellido or not cedula or not peso or not altura or not masa:
            label_mensaje.configure(text="Todos los campos son obligatorios.", text_color="red")
            return

        carpeta_usuario = f"rostros/{cedula}_{nombre}_{apellido}"
        os.makedirs(carpeta_usuario, exist_ok=True)

        # Abrir el stream RTSP
        cap = cv2.VideoCapture("rtsp://admin:L2EE2268@192.168.110.192:554/cam/realmonitor?channel=1&subtype=1")

        # Ajustar la resolución de la cámara
        cap.set(cv2.CAP_PROP_FRAME_WIDTH, 640)  # Resolución horizontal
        cap.set(cv2.CAP_PROP_FRAME_HEIGHT, 480)  # Resolución vertical

        # Descarta los primeros 30 frames para estabilizar la imagen
        for _ in range(30):
            ret, frame = cap.read()
            if not ret:
                print("Error al leer frame inicial")
                break

        tiempo_inicio = time.time()
        contador = 0
        embedding_final = None
        while time.time() - tiempo_inicio < 20:
            ret, frame = cap.read()
            if not ret:
```

```
            break

        frame_rgb = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)

        rostros = face_app.get(frame_rgb)

        if rostros:
            # Dibuja un rectángulo sobre el rostro detectado
            cv2.rectangle(frame, tuple(map(int, rostros[0].bbox[:2])), tuple(map(int, rostros[0].bbox[2:])), (0, 255, 0), 2)

            # Calcula el embedding promedio de los rostros detectados
            if embedding_final is None:
                embedding_final = rostros[0].embedding
            else:
                embedding_final = (embedding_final + rostros[0].embedding) / 2.0

            contador += 1

        # Guarda la imagen de cada rostro detectado
        ruta_imagen = os.path.join(carpeta_usuario, f"foto_{contador}.jpg")
        cv2.imwrite(ruta_imagen, frame)

        # Muestra la imagen
        cv2.imshow("Captura de Rostros", frame)

        if cv2.waitKey(1) & 0xFF == ord('q'):
            break

    cap.release()
```

```
        cv2.destroyAllWindows()

        # Guarda el embedding final calculado
        if embedding_final is not None:
            np.save(os.path.join(carpeta_usuario, "embedding.npy"), embedding_final)

        # Guarda los datos del usuario en el archivo Excel
        datos = {
            "Nombre": [nombre],
            "Apellido": [apellido],
            "Cedula": [cedula],
            "Foto": [os.path.join(carpeta_usuario, "foto_1.jpg")],
            "Embedding": [os.path.join(carpeta_usuario, "embedding.npy")]
        }

        global df
        df_nuevo = pd.DataFrame(datos)

        if os.path.exists(archivo_excel):
            df = pd.read_excel(archivo_excel)
            df = pd.concat([df, df_nuevo], ignore_index=True)
        else:
            df = df_nuevo

        df.to_excel(archivo_excel, index=False)

        # Guarda los datos físicos del usuario en un archivo Excel
        archivo_usuario = f"{cedula}.xlsx"
        datos_fisicos = {"Peso (kg)": [peso], "Altura (m)": [altura], "Masa muscular (%)": [masa]}
```

```
        df_usuario = pd.DataFrame(datos_fisicos)
        df_usuario.to_excel(archivo_usuario, index=False)

        label_mensaje.configure(text="Registro completado.", text_color="green")
        registro_window.destroy()



    btn_capturar = ctk.CTkButton(registro_window, text="Capturar Rostros",
command=capturar_rostros)
    btn_capturar.pack(pady=10)


def mostrar_datos_usuario(cedula):
    archivo_usuario = f"{cedula}.xlsx"
    if os.path.exists(archivo_usuario):
        df_usuario = pd.read_excel(archivo_usuario)
        datos_window = ctk.CTkToplevel(app)
        datos_window.geometry("300x200")
        datos_window.title("Datos del Usuario")

        label_titulo = ctk.CTkLabel(datos_window, text="Últimos datos de
entrenamiento", font=("Arial", 14, "bold"))
        label_titulo.pack(pady=10)

        for columna in df_usuario.columns:
            valor = df_usuario[columna].values[0]
            label_dato = ctk.CTkLabel(datos_window, text=f"{columna}: {valor}",
font=("Arial", 12))
            label_dato.pack()
    else:
        label_mensaje.configure(text="No hay datos registrados para este usuario.",
text_color="red")
```

```
def verificar_acceso():
    cap = cv2.VideoCapture("rtsp://admin:L2EE2268@192.168.110.192:554/cam/realmonitor?channel=1&subtype=1")
    ret, frame = cap.read()
    if not ret:
        label_mensaje.configure(text="Error al acceder a la cámara.", text_color="red")
        return

    frame_rgb = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
    rostro_actual = face_app.get(frame_rgb)
    if not rostro_actual:
        label_mensaje.configure(text="No se detectó un rostro válido.", text_color="red")
        reproducir_audio("C:\\Users\\DELL\\Desktop\\Tesis 2\\UsuarioNoRegistrado.mp3")
        return

    emb_actual = rostro_actual[0].embedding
    mejor_usuario = None
    mayor_similitud = 0.0

    for index, row in df.iterrows():
        emb_path = row["Embedding"]
        if os.path.exists(emb_path):
            emb_registrado = np.load(emb_path)
            similitud = cosine_similarity([emb_actual], [emb_registrado])[0][0]
            if similitud > mayor_similitud:
                mayor_similitud = similitud
                mejor_usuario = row
```

```
    cv2.destroyAllWindows()


    if mejor_usuario is not None and mayor_similitud > 0.55:
        label_mensaje.configure(text=f"Usuario {mejor_usuario['Nombre']}
{mejor_usuario['Apellido']} bienvenido.", text_color="green")
        reproducir_audio("C:\\Users\\DELL\\Desktop\\Tesis
2\\UsuarioRegistrado.mp3")
        mostrar_datos_usuario(mejor_usuario["Cedula"]
        label_mensaje.configure(text="Usuario desconocido, registrarse primero.",
text_color="red")
        reproducir_audio("C:\\Users\\DELL\\Desktop\\Tesis
2\\UsuarioNoRegistrado.mp3")


    global app, label_mensaje
    app = ctk.CTk()
    app.geometry("400x300")
    app.title("Control de acceso al Gym")


    ctk.CTkButton(app, text="Registrar usuario",
command=registrar_usuario).pack(pady=20)
    ctk.CTkButton(app, text="Acceso", command=verificar_acceso).pack(pady=20)
    ctk.CTkButton(app, text="Usuarios registrados",
command=mostrar_lista_usuarios).pack(pady=20)


    label_mensaje = ctk.CTkLabel(app, text="", font=("Arial", 12))
    label_mensaje.pack()
    app.mainloop()
# Ejecutar login
iniciar_login()
```

Anexo 14 Código de ejecución