ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

Facultad de Ingeniería en Electricidad y Computación

”Implementación de un sistema de seguridad física basado en IoT para el control de acceso y detención de posibles intrusos en el Centro de Datos de la FIEC”

Previo la obtención del Título de:

Ingeniero en Telemática

Presentado por:

Carlos Alexander Peñafiel Mera

Jaime José Véliz Ibarra

GUAYAQUIL - ECUADOR

Año: 2023

DEDICATORIA

CARLOS ALEXANDER PEÑAFIEL MERA

A mi familia y mis amigos, apoyo incodicional.

JAIME JOSÉ VÉLIZ IBARRA

A mi papá, mamá y hermano. Remanso de virtudes y voluntad.

DECLARACIÓN EXPRESA

”Los derechos de titularidad y explotación, nos corresponde conforme al reglamento de propiedad intelectual de la institución; Carlos Alexander Peñafiel Mera y Jaime José Véliz Ibarra, damos nuestro consentimiento para que la ESPOL realice la comunicación pública de la obra por cualquier medio con el fin de promover la consulta, difusión y uso público de la producción intelectual”

Firmado electrónicamente por:
CARLOS ALEXANDER PEÑAFIEL MERA

Carlos Alexander Peñafiel Mera

Firmado electrónicamente por:
JAIME JOSE VELIZ IBARRA
Razón: Firma Proyecto Integrador
Localización: En la línea
Fecha: 2024-02-06T11:34:19.492336-05:00

Jaime José Véliz Ibarra

EVALUADORES

Firmado electrónicamente por:
IGNACIO MARIN GARCIA

Ignacio Marín García

PROFESOR DE LA MATERIA

Firmado electrónicamente por:
DANNY ALFREDO TORRES MORAN

Danny Torres Morán

PROFESOR TUTOR

RESUMEN

Este proyecto tiene como fin la implementación de un sistema de seguridad física basado en IoT para el control de acceso y detención de posibles intrusos en el Centro de Datos de la FIEC. En la actualidad se realiza un control de acceso manual en una bitácora y, siendo un lugar neurálgico dentro de los servicios informáticos de la facultad, carece de la apropiada seguridad, es decir que cualquier persona que tenga la llave puede acceder sin previo aviso. Por lo que este proyecto busca mejorar el aspecto operativo de control de acceso y seguridad mediante un sistema que permita gestionar perfiles de usuarios con roles para crear y aprobar ordenes de trabajos. Mediante estas órdenes de trabajo es posible controlar el acceso al centro de datos mediante el uso pines de seguridad, además, controlar el aforo de la cantidad de personas permitidas para acceder. El proyecto hace uso de Software y Hardware de Código Abierto, y se comprende de tres módulos: Vigilancia (Control de aforo), Control de Acceso y Aplicación Web, cuyos procesos engloban las fases de Captura, Análisis, Procesamiento y Visualización de la arquitectura IoT. En la implementación se realizaron tres escenarios de pruebas: el primero, donde se definió qué sensor se iba a utilizar entre uno térmico y una cámara IP tradicional; el segundo, donde se definió si utilizar las librerías de análisis de imagen de OpenCV o de YOLOv8; y, el tercero, en donde se probó el sistema en ordenadores de distintas capacidades de RAM para ver su desempeño. Al lograrse la implementación se determinó que la combinación más apropiada para el procesamiento de imágenes en el control de aforo, requiere del uso de la librería YOLOv8 y mínimo 16GB de memoría RAM con una cámara basada en IP como la ESP 32 CAM.

Palabras Clave: Sistema de Seguridad, IoT, Control de acceso, Control de aforo, visión por computador, OpenCV, Yolov8

ABSTRACT

This document regards the implementation of a security system to manage the access and capacity control of the rack room of the Faculty of Electrical Engineering and Computer Science, with a design based on IoT architecture. Currently a manual access control is performed in a logbook and, being a neuralgic place within the computer services of the faculty, it lacks the appropriate security, i.e. anyone who has the key can access without prior notice. Therefore, this project seeks to improve the operational aspect of access control and security through a system that allows scheduling work orders, access and security through security pins, so that it can be accessed by unlocking an electronic lock. Additionally, the system controls the capacity of the number of people allowed in the work order. The project makes use of Open Source Software and Hardware, and is comprised of three modules: Surveillance (Capacity Control), Access Control and Web Application, whose processes encompass the phases of Capture, Analysis, Processing and Visualization of the IoT architecture. Three test scenarios were carried out in the implementation: the first one, where it was defined which sensor to use between a thermal sensor and a traditional IP camera; the second one, where it was defined whether to use OpenCV or YOLOv8 image analysis libraries; and the third one, where the system was tested in computers with different RAM capacities. Upon implementation, it was determined that the most appropriate combination requires the use of the YOLOv8 library, minimum resources of 16Gb of RAM with an IP camera connected to the ESP32 Cam, by saving steps in the processing stage.

Keywords: *IoT, Web Server, Computer Vision, OpenCV, YOLOv8*

ÍNDICE GENERAL

ABREVIATURAS

AP. Punto de Acceso (por sus siglas en inglés Access Point).

API. Interfaz de Programación de Aplicaciones (por sus siglas en inglés, Application Programming Interface).

ARM. Máquina RISC Avanzada (por sus siglas en inglés, Advanced RISC Machine).

BLE. Bluetooth de Bajo Consumo (por sus siglas en inglés, Bluetooth Low Energy).

CPU. Unidad Central de Procesamiento (por sus siglas en inglés, Central Processing Unit).

DBMS. Sistema de Manejo de Base de Datos (por sus siglas en inglés, Database Management System).

DOM. Modelos de Objetos del Documento (por sus siglas en inglés, Document Object Model).

ESPOL Escuela Superior Politécnica del Litoral

FIEC. Facultad de Ingeniería Eléctrica y Computación.

FK. Clave foránea (por sus siglas en inglés, Foreign Key).

HTML. Lenguaje de Marcado de Hipertexto (por sus siglas en inglés, Hypertext Markup Protocol.

HTTP. Protocolo de Transferencia de Hipertexto (por sus siglas en inglés, Hypertext Transfer Protocol).

I2C. Circuito Inter-Integrado (por sus siglas en inglés, Inter-Integrated Circuit).

IoT. Internet de las Cosas (por sus siglas en inglés, Internet of Things).

JSON. Notación de Objeto de JavaScript (por sus siglas en inglés, JavaScript Object Notation).

JWT. Identificador Web de JSON (por sus siglas en inglés, JSON Web Token).

PIN. Número de Identificación Personal (por sus siglas en inglés, Personal Identification Number).

PK. Clave Primaria (por sus siglas en inglés, Primary Key).

RAM. Memoria de Acceso Aleatorio (por sus siglas en inglés, Random Access Memory).

RISC. Computador con Conjunto de Instrucciones Reducidas (por sus siglas en inglés, Reduced Instruction Set Computer).

SCCB. Bus en Serie para Control de Camara (por sus siglas en inglés, Serial Camera Control Bus).

SQL. Lenguaje de Consulta Estructurada (por sus siglas en inglés, Structured Query Language).

YOLO. Sólo lo Ves una Vez, (por sus siglas en inglés, You Only Look Once).

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

ÍNDICE DE CODIGOS DE PROGRAMA

CAPÍTULO 1

1. INTRODUCCIÓN

La amenaza constante de ataques terroristas en el mundo y las actividades cada vez más sofisticadas de los grupos delictivos han aumentado la atención que se presta a la seguridad electrónica, por lo que existe la necesidad de inventar nuevos métodos para brindar una seguridad adecuada sitios públicos como universidades, centros comerciales, entre otros. El uso de tecnologías disruptivas da una respuesta innovadora a las necesidades que tiene la seguridad electrónica en América Latina y, a su vez, están modificando las tendencias actuales del mercado ([Agudelo, 2023]).

Las Tecnologías IoT permiten la comunicación y la transferencia de datos de diferentes dispositivos a través de Internet. Estos dispositivos pueden ser sensores que recopilan información sobre su entorno y las envían a través de Internet a otros dispositivos o sistemas [Campo, 2023].

En la industria de seguridad electrónica ha demostrado tener la capacidad de mejorar la seguridad de sus clientes y ofrecer servicios más personalizados [González, 2024]. Existen algunas maneras en que IoT tiene impacto en la industria de la seguridad electrónica tales como: Detección de intrusos, Sistemas de videovigilancia, Control de acceso, Protección de bienes y activos, Analítica de datos, entre otros.

Entre los dispositivos IoT que se utilizan en la seguridad electrónica se tienen cámaras de seguridad, sensores de movimiento, cerraduras electrónicas y sistemas de control de acceso, los cuales pueden detectar y responder automáticamente a cambios en su entorno, permitiéndoles a los sistemas de seguridad actuar rápidamente para proteger a las personas y propiedades [Arguello, 2023].

Los sistemas se seguridad electrónica tradicionales que cumplen funciones para controlar el acceso suelen ser simples, de baja eficiencia y con una capacidad de almacenamiento pequeña, impidiendo llevar un control óptimo que cumpla con los

requisitos de seguridad como los ofrecidos por tecnologías IoT [Wang, 2012].

Por otro lado, el bloqueo de puertas haciendo uso de llaves con cerraduras mecánicas a pesar de ser comúnmente utilizado y aceptado globalmente, suele tener problemas como la perdida de las llaves de las cerraduras, la copia no autorizada o hurto de las llaves, vulnerando la seguridad del lugar [Sowjanya, 2016].

Este proyecto está orientado a desarrollar una solución basada en IoT que permita monitorear y controlar desde un sitio web el acceso mediante la validación, registro y notificación vía correo electrónico del ingreso y salida de usuarios autorizados y el aforo máximo permitido dentro del centro de datos de la FIEC.

1.1 Problemática

El registro análogo es un método eficiente con respaldo físico usado en muchos campos, desde las bitácoras de campo en levantamientos topográficos hasta el control de acceso en aplicaciones de seguridad. Con el debido cuidado, estos archivos pueden durar décadas sin deterioro, quedando como material de consulta, de ser requerido.

La tendencia actual de migrar hacia la transformación digital, acrecentada por las medidas tomadas después de la pandemia, aceleró la adopción de herramientas tecnológicas para solventar diversas limitaciones tanto de asistencia como control. [García-Madurga, 2021] Esta tecnificación ha servido como motor de desarrollo para diversas empresas, ahorrando tiempo y dinero en procesos que, una vez automatizados, ha permitido la inversión de recursos económicos y humanos en otras partes de la cadena productiva, mejorando la adaptabilidad de estos.

El registro análogo, comparado con su contraparte de transformación digital, muestra que requiere de una mayor inversión de tiempo por parte del personal, en su uso y en el cuidado requerido para almacenar los documentos físicos que son su respaldo. Además, para ser consultada es requerido un acceso directo al archivo, lo que imposibilita cualquier acción remota de primera mano.

La digitalización de procesos de registro permite la tabulación rápida de información y su análisis, así como acceso remoto. De manera adicional, mejora la seguridad, puesto que cualquier cambio puede quedar registrado. Esta innovación conlleva una mejora de productividad, costos reducidos, y también aumenta la competitividad y el valor de marca.

1.2 Justificación

En el mercado existen sistemas de seguridad que funcionan de diversas maneras. Los más conocidos son aquellos que dan paso en base a tarjetas con tecnología RFID, las cuales suelen tener un identificador único asociado, y los sistemas de ingreso biométrico. El primero se encuentra supeditado a ser mostrado a un sensor para tener entrada, sin embargo puede extraviarse o robarse. Los sistemas de autenticación biométricos son más robustos y brindan más facilidades, pero su integración a un sistema preexistente suele ser costosa, adicional al hecho de que los sistemas de bitácoras que vienen integrados en los mismos no suelen ajustarse a los requerimientos empresariales, precisando de su modificación.

Adicionalmente, se ha observado que en los últimos 4 años no ha habido dentro del área un desarrollo investigativo en sistemas ciber-físicos orientados a seguridad que tengan utilidad y sean escalables. Por ejemplo, dentro de los últimos 4 años, no hay literatura que habla acerca de implementación IoT para accesos, control de bitácora, control de horarios y entrega de informes, lo que se refleja en la sección de Estado del Arte. Además, los desarrollos se han sentado en el uso de tarjetas magnéticas, lo cual no se puede llevar de un sistema propuesto a otro no sin antes haber hecho un trabajo de interconexión notable [Topic, 1993].

Por lo expresado, se ve la necesidad de implementar un sistema de seguridad que no solamente sea un control de acceso a un lugar, sino también que lleve un control de bitácora, para tener un detalle del tránsito de personas y que refleje en un archivo histórico los trabajos que dicho personal va a realizar, con fecha, hora y responsable. Si bien el control mediante tarjetas magnéticas es ubicuo y su implementación exitosa a todas luces, carece lo práctico de poseer un código de acceso, ya que al olvidar la tarjeta se imposibilita el acceso a un lugar.

Lo que se necesita, junto a la implementación de acceso por PIN, es que el sistema de seguridad aborde las posibles deficiencias en cuanto a un uso equivocado pudiesen haber, esto es, en vez de un PIN definido, que sea un PIN aleatorio, válido por una sola vez, para un trabajo en específico y, remitido vía correo al teléfono del requiriente, previamente aprobado por un administrador. Es necesario que el sistema de seguridad a su vez haga un control de cuántas personas han ingresado a un lugar mediante conteo,

para así elevar la barrera de protección frente a un uso indebido de credenciales, puesto que se imposibilita el hecho de que entren más personas de las permitidas al ingresar el código numérico y mantener la puerta abierta. El sistema de seguridad debe de poder trabajar en tiempo real o muy cerca de aquello para que sea efectivo, y tendrá que remitir las alamas de las irregularidades de acceso que vayan ocurriendo mediante un sistema de notificación.

Los usuarios deben de poder manipular el sistema y responder a los eventos que les sean notificados. Si bien un desarrollo robusto es la creación de una aplicación para teléfonos móviles, esto levanta problemáticas con respecto a qué teléfonos celulares podrán instalar dicho programa, ya que los teléfonos móviles vienen de distintas características que, en ciertos casos, deniega la instalación de los mismos desde la tienda nativa de aplicaciones, requiriendo pasos adicionales para el uso e impactando así la experiencia de usuario.

Otra consideración, mencionada por completitud, es el desarrollo de una aplicación para escritorio, que también demandaría idealmente el desarrollo de sus contrapartes para Linux y Mac aparte de Windows. La manera en que un sistema de seguridad propuesto pueda pasar por alto las consideraciones mencionadas, esto es, que se pueda utilizar en una gran mayoría de dispositivos móviles, sean teléfonos celulares, tablets y laptops, irrestricto si pertenecen a ecosistemas iOS, Android, Windows y demás, es la implementación de una aplicación web, que pueda accederse mediante un navegador.

Esto abre las puertas también para el uso de ordenadores convencionales. Un diseño modular bien definido permitirá que el sistema sea versátil, con diversas implementaciones no sólo dentro del ámbito del presente proyecto, que será instalado en el Centro de Datos de la FIEC de la ESPOL, sino que se pueda implementar en sitios que requiere control de acceso y de aforo parecido al del escenario propuesto. utilizar también para bodegas y otros lugares en donde haya inventario o celo en el tránsito.

1.3 Objetivos

Como **Objetivo General** del proyecto tenemos la implementación de un sistema de seguridad física basado en IoT para el control de acceso y detección posibles intrusos en el centro de datos de la FIEC.

Como Objetivos Específicos para lograr la consecución del proyecto se tiene:

Objetivo Específico 1. Implementar un módulo de control de acceso basado en hardware y software de código abierto que valide, registre y notifique vía correo electrónico el ingreso y salida de usuarios autorizados al centro de datos de la FIEC.

Objetivo Específico 2. Implementar una aplicación web que muestre un histórico y en tiempo real los datos obtenidos del módulo de control de acceso y de vigilancia para el control y monitoreo continuo.

Objetivo Específico 3. Implementar un módulo de vigilancia que detecte cuántas personas se encuentran en el centro de datos y envíe notificaciones.

1.4 Alcances y Limitaciones

Se implementa un sistema de seguridad física basado en IoT para el control de acceso y detención de posibles intrusos en el Centro de Datos de la FIEC, mediante el uso de Hardware y Software de Código Abierto, tales como una placa programable Raspberry Pi 3B, una placa programable ESP32 Cam, dos teclados numéricos, un servidor web, un servidor de imagenes, un servidor de base de datos, un pestillo inteligente y una cámara IP OV2460 ubicada en frente de la puerta de acceso del Centro de Datos. Es posible administrar el sistema desde una interfaz web que solo es posible acceder dentro de la red de datos institucional, la cual maneja perfiles de usuarios para crear y aprobar órdentes de trabajos.

En estas ordenes de trabajo se puede asignar un PIN de seguridad aleatorio el cual será digitado usando el teclado númerico a la entrada o salida de la puerta del Centro de Datos. Además, es posible indicar el número de personas que pueden acceder mientras dure el trabajo, esto con el fin de contabilizar el aforo dentro del Centro de Datos. En caso de que la cantidad de personas que ingresan sea mayor, se notificará esa novedad mediante correo electrónico a la persona encargada de la orden de trabajo. Cada evento

de entrada, salida y control de aforo es registrado como historial para posterior análisis.

El hardware utilizado para la implementación carece de recursos computacionales para almacenar permanentemente imagenes, por lo que se desarrolla una estrategia para almacenar temporalmente por 30 segundos las imagenes capturadas exclusivamente para procesarlas, esto hace que el análisis sea en tiempo real.

1.5 Estado del Arte

La importancia de la seguridad se refleja en la diversidad de trabajos que abordan este tema desde múltiples perspectivas y utilizando gran variedad de herramientas físicas y digitales para cumplir este propósito. Si bien la mayoría de los ataques en contra de centros de datos es alta desde el punto de vista virtual, la seguridad física suele operar en estándares que no se encuentran a nivel internacional, esto es, la mayoría cumplen con seguridad contra incendios pero la capacidad de respuesta de robo o daño físico no se suele tomar en cuenta. Hyun-Sun Kang [Kang, 2015] da como propuesta una lista de requisitos y recomendaciones que ayuda de manera eficiente el tener la seguridad física bajo control, entre ellas, la videovigilancia y control de sensores de IoT.

En el mercado actual, la disponibilidad de cerraduras inteligentes es notable, ofreciendo una amplia gama de métodos de comunicación y autenticación para los consumidores. Las cerraduras inteligentes comerciales, si bien son ampliamente adoptadas, frecuentemente presentan fallos en el diseño, la implementación y los enfoques de interacción [Ho et al., 2016] por lo que no son recomendadas para usos profesionales o institucionales.

Dentro de la ESPOL existen desarrollos que han buscado aprovechar la habilidad de dispositivos ubicuos, como teléfonos celulares, que tienen tanto WiFi como Bluetooth, para poder así asociarlos a una cerradura inteligente y permitir acceso de profesores o profesionales a sitios restringidos en la universidad. Esto, a diferencia de sistemas con tarjeta, facilita el tránsito de personas con las credenciales apropiadas por los sitios importantes del campus [Estrada, 2018b].

Del estado del arte tenemos, por ejemplo, el desarrollo de soluciones basado en arquitectura IoT, que aprovecha tecnologías como Bluetooth y Wifi, y se han implementado en plataformas móviles [Aluri, 2020].

El internet de las cosas permite una implementación de sensores muy amplia, lo que abre las puertas al prototipado de soluciones más completas, como uso biométrico y uso de tarjetas magnéticas, que se pensó para el tránsito de los laboratorios de la FIEC, ESPOL [Vásquez, 2023].

Algo similar se tiene con sistemas que implementan control de acceso, bitácoras y entrega de informes, basado en tecnología RFID, presente en tarjetas inteligentes que bien pudiesen ser ubicuas. Estos sistemas pueden estar enlazados con sensores y relés que controlen de diversas maneras los sistemas de temperatura e iluminación de un sitio, automatizando procesos para optimizar costos [Estrada, 2018a].

1.6 Marco Teórico

Un **API**, por siglas en inglés, significa Application Programming Interface. Es un programa cuyo código permite a dos aplicaciones de software comunicarse entre sí. En este caso, el sistema utiliza un API que se encarga de recibir la información de conteo de parte del Raspberry. Una vez teniendo dicha información, procede a remitirla para su almacenamiento en una base de datos. Esto permite liberar recursos en el microcontrolador, ya que dicho proceso se delega al ordenador que está ejecutando el API [Fernandez, 2019].

Axios es un cliente HTTP basado en promesas que trabaja tanto en un navegador como en un ambiente Node.Js. Esto facilita la interacción del envío de datos al backend [Padhyay, 2023].

El **AMG8833** es un sensor término infrarrojo conformado por una matriz de sensores térmicos de 8x8. Desarrollada por Panasonic, se comunica mediante un bus I2C, que es un tipo de comunicación serial desarrollada por Phillips [Gigi, 2021].

El **Backend** y **Frontend** son términos que, al ser utilizados dentro del ámbito de la programación de aplicaciones web, refieren a la programación que se realiza del lado de usuario, impactando directamente su experiencia (Frontend) o a la programación del lado del servidor, que es el desarrollo del funcionamiento del programa (Backend). Aquel programador que puede desarrollar una aplicación en ambos sitiales se denomina desarrollador ‘Fullstack’. En este proyecto, para el desarrollo de Frontend se utilizó en su gran mayoría el lenguaje HTML5; y para el backend se utilizó Python [Singhal, 2023].

Una **Base de Datos** es un conjunto organizado que contiene datos estructurados, almacenado de manera digital y que puede ser accedido por sistemas informáticos. Para esto, la base de datos se controla por un sistema de gestión de base de datos, o DBMS (Database Management System) por sus siglas en inglés. Una Base de Datos trabaja mediante tablas y su relación entre ellas para modelar un esquema práctico. Estas tablas contienen información detallada del objeto abstracto que pretenden modelar. Hay distintos sistemas de bases de datos, optimizadas para el uso que se les vaya a dar, para esto, hay bases contables, bases de datos para inventarios, etc. Estas bases de datos varían tanto en los recursos computacionales que necesitan para funcionar, léase memoria RAM y recursos del procesador, así como en tamaño y en velocidad de procesamiento de datos. Para este proyecto se utilizó la base de datos MariaDB, elegida por encima de SQLite [Melanie, 2023].

El **Framework**, dentro del mundo de la programación, es un conjunto de librerías y herramientas que se utilizan para desarrollar las aplicaciones de una manera mucho más eficiente, ya que estandariza muchos asuntos técnicos, permitiendo al programador enfocarse en los desafíos de su deber profesional. Así mismo, al definir las librerías a usarse durante el desarrollo, permite tener un estándar de calidad [Lucena, 2023].

Un **Algoritmo**, en el mundo de la programación, es un conjunto de instrucciones que son elaboradas de forma sistemática, con el objetivo de cumplir un fin determinado. Si el algoritmo no se encuentra elaborado en un lenguaje de programación convencional toma el nombre de pseudocódigo [Team, 2024].

La estructura **Docker** es una tecnología de Código Abierto que trabaja a tres niveles para lograr la virtualización de procesos y garantizar así su funcionamiento de forma contenida. La diferencia entre Docker y una máquina virtual es que las instancias del primero corren en el mismo sistema operativo, mientras que el segundo corre en una abstracción creada en la misma máquina virtual. Las tres fases de Docker son: dockerfile, que es un archivo que lleva instrucciones y configuración, incluyendo puertos, forma de comportarse ante caídas, etc.; el dockerimage, que lleva instrucciones para la creación del container, y por ello lleva las librerías, archivos de sistemas y similares; y docker container, que es una instancia creada por las imágenes [Hashemi,].

Un **DOM**, que por sus siglas en inglés significa Document Object Model, es una interfaz de programación que se utiliza mucho en documentos web. La utilidad de un DOM yace

en que transforma los diversos elementos de una página web, incluyendo la estructura, estilo, contenido y demás, como nodos y objetos. De esta manera, se facilita interactuar con los elementos de la página usando un lenguaje de programación [MozDevNet,].

OpenCV es una librería de código abierto que gira en torno a análisis de visión por computadora y aprendizaje automático (machine learning). La fortaleza de OpenCV es yace en disfrutar de una comunidad de desarrollo excepcionalmente grande, por lo que dentro de ella se ha desarrollado prestaciones en gran cantidad y eficiencia, siendo utilizada en proyectos comerciales y privados con gran éxito. Trabaja en interfaces de C, C++, Python y java [Kulhary, 2022].

YOLOv8 es una librería de código abierto desarrollada por Ultralytics, dentro del marco de Visión por Computadora y Reconocimiento de Patrones. Es un modelo basado en redes neuronales que contiene librerías para detección de objetos y clasificación; y se puede asignar a tareas de segmentación. La base se realiza a través de Python [Encord, 2023].

La **ESP32 Cam** es una tarjeta programable que tiene una cámara incluida. Cuenta con dos CPUs de 32 bits LX6 y la habilidad de funcionar como AP y programar servidores de red gracias a su antena Bluetooth, BLE y WiFi [Pascual, 2022].

La **Raspberry Pi** es el nombre de una serie de tarjetas programables con capacidad y uso que se presta para aplicaciones de IoT. La Raspberry usada para el presente proyecto es una Raspberry Pi 3 Modelo B, que cuenta con el microcontrolador BCM2837, que incluye un procesador ARMv8 de 64 bits y un procesador gráfico VideoCore IV. Dentro de ella tiene conectividad Bluetooth y Wifi [Bastiaansen, 2022].

IoT refiere al Internet de las Cosas en sus siglas en inglés. Es una red de dispositivos que se conectan entre ellos e intercambian datos, sea con otros dispositivos o con la nube. Mediante sistemas embebidos, una gran cantidad de dispositivos y máquinas puede tener esta funcionalidad aumentada. Los beneficios del IoT es que permite, mediante el análisis de la data generada por los sus distintos elementos constitutivos, aumentar la eficiencia de sus labores, mejorar el servicio al cliente y la toma de decisiones accionables, impactando positivamente el valor del negocio [Gillis, 2023].

El **Token de Acceso**, dentro del concepto de autenticación basada en tokens, es una forma de confirmar la identidad de un usuario o dispositivo. A grandes trazos, se encarga de comprobar si la entidad en cuestión posee un token previamente emitido. Con

respecto a gestionar accesos, que es el tema que compete al presente trabajo, un servidor usa autenticación con tokens para verificar la identidad de la entidad que pretende tener acceso, sea un usuario, una API, o algún ordenador común o servidor. Por principio, el token demanda confianza porque es expedido por una fuente confiable, por una autoridad. A diferencia de la autenticación por cookies, que basta sólo con una cookie almacenada en el ordenador, el servidor crea una sesión y asigna un token, cuya información se coloca en una base de datos. Este token y registro ocurre en cada solicitud HTTP que haga el usuario, ubicándose en el encabezado de dicha petición. Esto lo hace, con respecto a la seguridad, un protocolo más robusto [Blancarte, 2023].

Flask es un framework de Django cuyo uso principal es el desarrollo de aplicaciones web. Una de sus ventajas es que viene con un servidor implementado para desarrollo. Otra ventaja es que Flask es ligero, puesto que la gran mayoría de librerías que engloba están direccionadas al desarrolo web [Saini, 2023].

CAPÍTULO 2

2. METODOLOGÍA DE ARQUITECTURA IOT

Esta sección refiere la descripción de la metodología, los materiales, las fases de arquitectura IoT y los escenarios de pruebas previo a la implementación. Adicionalmente, se explica el funcionamiento de los diferentes módulos que componen el sistema de seguridad física, los cuales se relacionan con las diferentes fases de la arquitectura.

2.1 Descripción de Arquitectura IoT

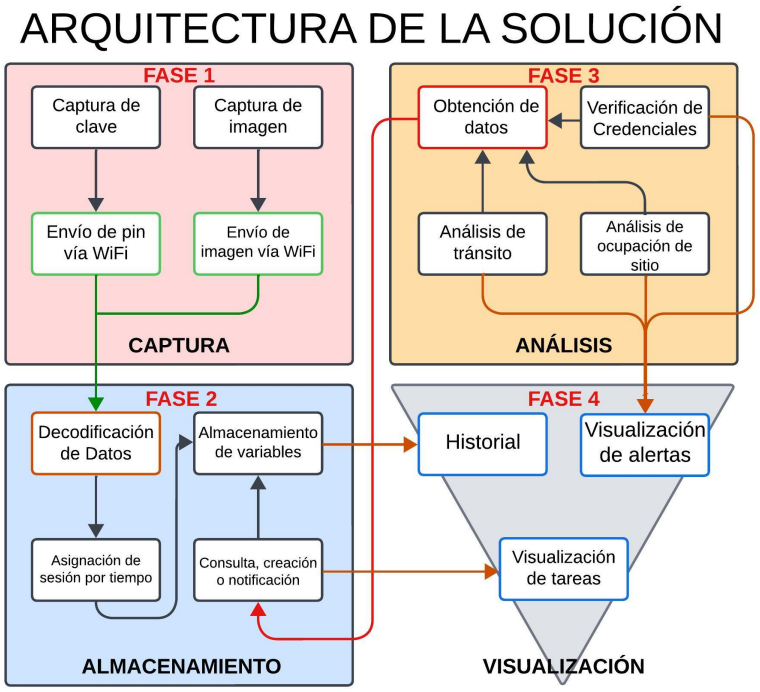


Figura 2.1: Fases de implementación basada en la Arquitectura IoT. Esta ilustración, dividida en cuadrantes, muestra los hitos más significativos de la presente implementación. Obra propia.

El enfoque principal dentro del proyecto es el uso de tecnologías Internet de las Cosas en el marco de seguridad física. La arquitectura convencional del IoT, por sus siglas en inglés, nos permite Capturar datos, Almacenarlos, Analizarlos y Visualizarlos. A continuación se detalla la arquitectura propuesta.

En la Figura 2.1 se observa cómo se definieron los hitos importantes de la implementación del sistema acorde a la Arquitectura IoT tradicional. Se definen como hitos a los procesos puntuales que realiza el sistema y se engloban dentro de las fases de la arquitectura IoT. Este ejercicio ayuda a identificar los elementos que se deben de considerar y la interacción entre ellos, representado aquí en flechas de diagrama de flujo. La figura se separa en cuadrantes, donde cada cuadrante representa una Fase de esta Arquitectura.

Fase 1: Captura de imagenes asociadas al control de aforo el cual se envian al Módulo de Control. Además, captura el PIN ingresado desde el teclado númerico para que desde el mismo servidor, se pueda validar que el ingreso ha sido exitoso o erróneo.

Fase 2: Almacenamiento, y está relacionada con la fase anterior, puesto que las imágenes capturadas permanecen almacenadas durante 30 segundos desde que se validó el PIN ingresado. En esta Fase tambien se considera la creación de perfiles de usuarios, la creación de las ordenes de trabajos y el registro de los eventos que generen cuando se cuenta n las personas dentro de la sala y cuando se entra o se sale del mismo. Esta Fase también almacena la información que usa y muestra el Módulo Servidor Web para sus procesos, que incluyen pero no se limitan a: creación de usuarios, modificación de órdenes de trabajo, ver información histórica, etc., todo acorde a los privilegios que tengan los usuarios dentro del sistema.

Fase 3: Análisis, guarda relación con la Fase 2, puesto que para analizar datos se precisa acceder a donde están almacenados estos datos. También se relaciona con la Fase 4, Visualización, puesto que el resultado de dicho análisis es el que se muestra al usuario. Aquí se realizan los análisis del Módulo Servidor Web, como análisis de conteo de aforo o la verificación de PIN. También se realizan análisis del Módulo de Vigilancia, ocurriendo en la máquina donde corre el servidor y parte en el Raspberry Pi 3B, que dentro de la implementación del sistema actúa como árbitro, informando al servidor cuándo debe de iniciar a analizar los datos de imagen.

Fase 4: se vale de las Fases 2 y 3 para funcionar, además contienen tres hitos

principales que son:

- Visualización del Historial: se muestra el historial de las Órdenes de Trabajo y el histórico de Usuarios.

- Visualización de alertas: aqui se lleva a cabo la remisión de correos hacia el administrador en caso de que se hayan detectado irregularidades.

- Visualización de tareas: desde aquí es posible (1) notificación mediante correo al usuario acerca de Órdenes de Trabajo asignadas/creadas, (2) la notificación a los usuarios del PIN de seguridad asignado para la Orden de Trabajo; y (3) la notificación de Activación de Usuarios u Órdenes de Trabajo.

2.2 Materiales y recursos

La siguiente tabla muestra el hardware y software que se utilizó en cada etapa de la implementación.

Tabla 2.1: Elementos de hardware y software utilizados en la implementación del sistema.

Captura		
1	ESP32 Cam	Hardware
1	Fuente de poder 5V	Hardware
1	OV2640	Hardware
2	Teclado Matricial	Hardware
1	Servidor de streaming (S:ESP32 Cam; C:Raspberry Pi 3B)*	Software
Almacenamiento		
1	Raspberry Pi 3B	Hardware
1	Fuente de poder 12V	Hardware
1	MicroSD 64Gb	Hardware
1	Servidor web (API) (S:Módulo Aplicación Web; C:Usuario, Raspberry)**	Software
Análisis		
1	Servomotor	Hardware
1	Servidor web (Flask, análisis de imagen) (S:;Módulo de Aplicación Web; S:Raspberry)	Software
Visualización		
1	Servidor Web ’Módulo Aplicación Web’	Software
*’S:’ Refiere a servidor; ’C:’ refiere a cliente. **Raspberry refiere a Raspberry Pi 3B.		

2.3 Estudio de casos

Se enumeran los escenarios en donde se comparan opciones de implementación tanto en hardware y software previo a la implementación final del sistema y su desarrollo se

detalla, respectivamente, desde la página 41, que refiere a los Análisis de Resultados; y, desde la página 47 que refiere a las Conclusiones y Recomendaciones..

Primer escenario: definición del sensor visual

Como se definió que el conteo de personas se haría mediante visión por computadora, se determinó que el sensor más apropiado para esto era el sensor térmico AMG3388, desarrollado por Samsung y de resolución 8x8. La consideración de un sensor térmico se razonó desde el argumento factual que un algoritmo de visión por computadora puede fallar si el ambiente no se encuentra bien iluminado. Un sensor término no capta fotones sino radiación térmica, obviando la necesidad de una correcta iluminación. La desventaja en su implementación yace en que para usarlo, los cuadros deben de ser primeramente procesados (interpolación, filtrado), causando un impacto computacional. Se eligió este sensor al ser, dentro del rango térmico, un sensor compuesto asequible. Al estar concientes de que la implementación no solo debía de ser eficiente con respecto a los recursos computacionales de las placas, sino que su implementación dentro del sistema tenía que tener un comportamiento funcional, se eligió como opción la cámara IP OV2460, cuya imagen puede ser usada para análisis de visión por computadora.

Segundo escenario: elección de las librerías de procesamiento de imagen entre OpenCV y YOLOv8

La librería Open Software de procesamiento de visión por computadora más conocida es OpenCV. Realiza el reconocimiento de imagen de la siguiente manera: cada cuadro lo procesa dos veces, el primero para determinar qué es el objeto, el segundo paso para determinar qué es el proceso. Es exacto pero consume muchos recursos. YOLOv8 es un conjunto de librerías de Código Abierto que en un solo paso logra identificar no sólo el objeto sino su posición. La desventaja yace en que, si bien es más ligero en cuanto a consumo de recursos, pierde exactitud con respecto a posición o a la identificación de muchos objetos que se encuentran muy cerca.

Tercer escenario: Prueba de recursos de sistema

Al finalizar la implementación del proyecto, se hace una prueba para observar el impacto que tiene la cantidad de RAM disponible que tiene el elemento que realiza el análisis visual. Para esto, se corre el algoritmo de conteo en: la Raspberry Pi 3B, que tiene 1 + 0.5Gb (Swap) de RAM; 1 PC windows con 16Gb de RAM; y 1 Macbook Pro de 32Gb de RAM.

2.4 Módulos del sistema de seguridad

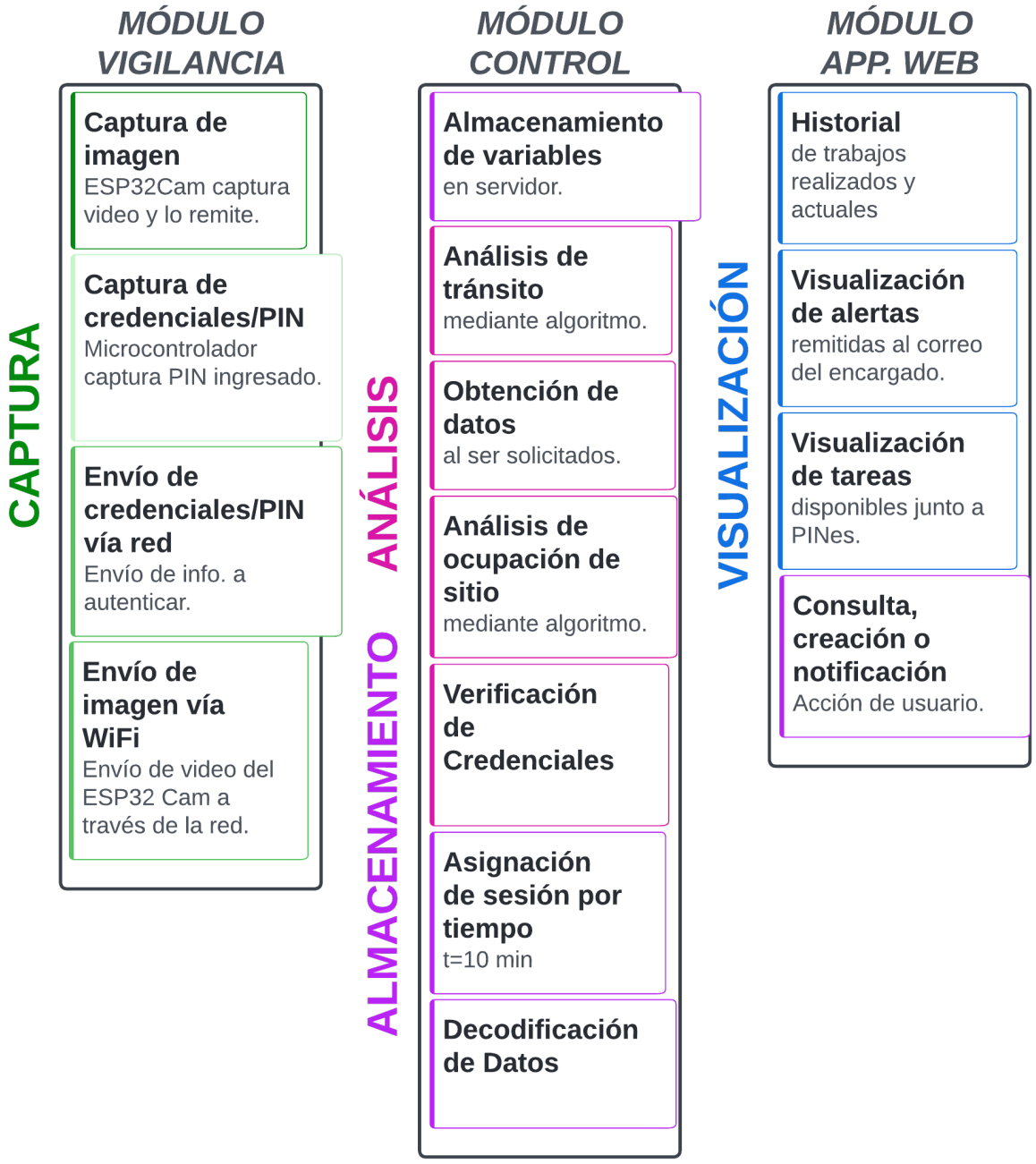


Figura 2.2: Relación de la Arquitectura IoT empleada con los módulos del sistema de seguridad. Obra propia.

Módulo de Vigilancia. El módulo de Vigilancia engloba al Raspberry Pi, que recibe de forma inalámbrica el video de parte del ESP32 Cam por medio de una conexión cliente-servidor. Es deber del Raspberry remitir dichas imágenes para que se cuente

el tránsito de las personas.

Para empezar, hay que alistar el Raspberry Pi 3B, que es el microcontrolador que está encargado de la remisión de las imágenes que serán analizadas para el conteo de personas. Para aquello, se procedió a instalar Raspbian para Raspberry en una tarjeta formateada de 64Gb. El sistema instalado fue la versión en 64 bits expedida el 5 de diciembre de 2023, versión de núcleo 6.1, y versión Debian 'Bookworm'.

Teniendo en cuenta que hay que hacer uso adecuado de los recursos de un microcontrolador, se definió una partición Swap de 4Gb, para usarse exclusivamente durante la partición, en caso de que algún proceso use más RAM del que tiene disponible la tarjeta. Esta precaución es importante, ya que la instalación de OpenCV demanda mucho tiempo. Con la plataforma lista, se procede a actualizar los repositorios de software e instalar tanto OpenCV como 'cmake'.

```
@article{sudo apt install -y build-essential cmake pkg-config libjpeg-dev
libtiff5-dev libpng-dev libavcodec-dev libavformat-dev libswscale-dev
libv4l-dev libxvidcore-dev libx264-dev libfontconfig1-dev libcairo2-dev
libgdk-pixbuf2.0-dev libpango1.0-dev libgtk2.0-dev libgtk-3-dev
libatlas-base-dev gfortran libhdf5-dev libhdf5-serial-dev libhdf5-103
libqt5gui5 libqt5webkit5 libqt5test5 python3-pyqt5 python3-dev
}
```

Código 2.1: Enunciado de paquetes a instalarse

Una vez finalizada la instalación y actualización se configura la ESP32, la cual aparte de conversar con el Raspberry se lo emplea como servidor de imágenes. La transmisión de estas imágenes se la hace a la base de datos para su conteo.

Módulo de Aplicación Web. El Módulo de Aplicación Web relaciona a la base de datos con la experiencia de usuario. De igual manera, implementa el envío de alertas al usuario Administrador, en caso de que el pin ingresado al teclado numérico no sea el adecuado, o, aún siendo el correcto, este haya sido ingresado a deshoras.

Para el desarrollo del Backend y del Frontend, se trabaja con tres secciones.

La ***primera sección*** es inherente pero no única a Django, y se basa en creación de modelos relacionales, en donde se definen las estructuras de datos de los modelos relacionales que se guardarán en la base de datos. Adicional, se crean y aprueban ordenes de trabajo y se registran eventos de puerta que se refiere al ingreso del PIN

para entrar o salir del cuarto.

Como ***segunda sección***, tenemos la creación de serializadores, los cuales están encargados de la manipulación de la información. Por ejemplo, es a través de estos que se efectúan las validaciones. La información procesada después es mostrada en las vistas o endpoints, que es como se definen a las páginas web con las que el usuario tiene contacto.

Como ***tercera sección*** o punto de interés tenemos a los métodos, que son las formas de obtener información, por lectura de puertos, de los periféricos o software a los que el microcontrolador tiene acceso.

Un ejemplo del funcionamiento correcto de estos tres principios es el siguiente: se tiene una vista cuya función es pedir al usuario la credencial para su ingreso. La información colocada será levantada mediante el método apropiado, que lo remitirá a uno o varios serializadores, en donde, después de ser verificados, van al modelo, el cual es el objeto con información destilada que es ingresada a la base de datos.

Aparte de las validaciones, la máquina en donde reside el Backend es donde se define el algoritmo de conteo, el cual tendrá un contador de ingresos, de egresos y el delta entre ambos para obtener el total de personas dentro del sitio.

Módulo de Control de Acceso.

El Módulo de Control de acceso engloba a la base de datos, el Raspberry Pi y los teclados numéricos. Aquí, el microcontrolador recibirá por medio de los teclados un PIN, el cual será enviado hacia la base de datos para analizar su validez.

Empezamos con la configuración de la Base de Datos que se relacionará directamente con la Aplicación. En el archivo de configuración del sistema se habilita la opción que permite la conexión remota desde otros equipos. Para prevenir ataques y errores de autenticación, se importó y usó la librería corseheaders. Para manejar el tratamiento y manejo de dichas autenticaciones, creación de usuarios, claves, etc., se usó la librería Djoser. También se habilitó el uso de tokens JWT para las transacciones, que sirve como identificador de usuario que está realizando transacciones. Anexo, se hace uso también de Axios, el cual es un cliente HTTP tanto para la aplicación como para el navegador. Entre otras cosas, se usa para enlazar el Raspberry Pi con la aplicación.

Para los endpoints se usa ReactJC, que es parte de una librería de JavaScript. ReactJC se especializa en crear páginas web sobre el DOM (Document Object Model)

de la página. Es decir, cada página tiene un DOM, que es una representación de objetos HTML de la página. ReactJC crea un DOM virtual encima de ese DOM, habilitando así la capacidad de crear una página dentro de una página, aprovechando recursos y beneficiándose el usuario así con mayor flexibilidad de edición, al poder editar mediante JavaScript aquellas estructuras de la página en HTML.

De igual manera, se usa la librería MaterialUI, que provee con los diversos elementos gráficos, sea tablas, imágenes, etc. Con esto se evita la creación de cada elemento gráfico desde cero, y aprovechando elementos gráficos con comportamientos complejos. Junto a MaterialUI está Formik, al cual está asociado. Formik es una librería para ReactJC que se especializa en manejar formularios de manera sencilla e intuitiva.

Para aumentar la seguridad, se coloca un tiempo de sesión o tiempo de vida al token de acceso, el cual después de 10 minutos de inactividad termina la sesión, para disminuir la probabilidad de que se haga un uso espurio del sistema al dejar la sesión olvidada.

Base de Datos. Para el desarrollo de la aplicación se procedió a identificar la necesidad de implementar un sistema de Base de Datos. Se decidió utilizar la base de datos MariaDB, elegida por encima de SQLite. El razonamiento detrás de la elección es que si bien SQLite es más ligera y rápida, MariaDB está optimizada para manejar una gran cantidad de datos y es más robusta para la aplicación de seguridad. De manera puntual, SQLite es una base de datos embebida, lo que significa que el motor de la base se relaciona directamente con la misma, sin necesidad de tener un servidor aparte. MariaDB es distinto en el punto que es una base de datos cliente-servidor, y el acceso se logra mediante una conexión de red.

SQLite es una base de datos que guarda la información en un archivo único. Con MariaDB, la información se guarda en distintos archivos, lo que facilita la flexibilidad de implementación y también de hacer respaldos de información, abriendo oportunidades futuras de mejora al sistema implementado.

Con respecto al tamaño de implementación, MariaDB fue diseñada y optimizada para uso empresarial y manejo de información a gran escala, y su implementación para uso en aplicaciones web, así como servidores, centros de datos y ambientes de la nube son variadas y bien documentadas. SQLite, en cambio, suele ser usada en implementaciones que necesiten de alcances más sencillos tanto en forma como en fondo.

Si bien tanto como MariaDB y SQLite fueron desarrollados a partir de MySQL, SQLite

opera con un subestándar de dicha base de datos. Como tal, MariaDB soporta el estándar completo de SQL, dando mayor poder y flexibilidad a la implementación de este proyecto, que denota en su concepción una complejidad incrementada.

CAPÍTULO 3

3. DISEÑO DEL SISTEMA E IMPLEMENTACIÓN

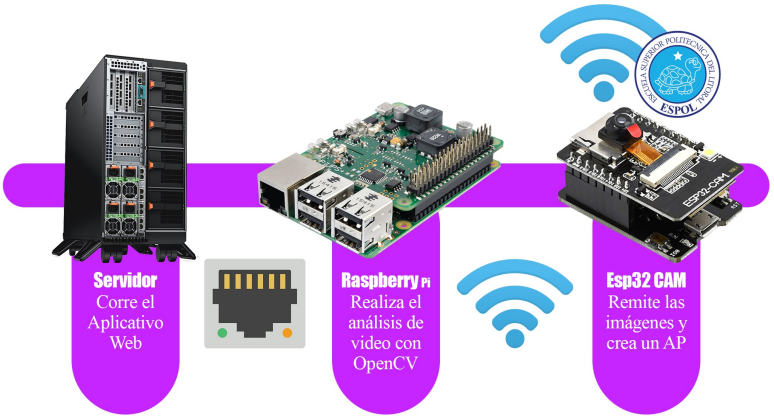


Figura 3.1: Diseño físico de la red que comunica servidor con Raspberry Pi y ESP 32 CAM. Obra propia.

Los tres componentes principales, ilustrados en la Figura 3.1, son el servidor, el Raspberry y el Esp32 Cam. Para que el sistema funcione todos deben de estar en una misma red para computar y gestionar la data recibida previo a la remisión a la base de datos. Para esto se aprovecha que el ESP32 Cam puede crear un punto de acceso (AP). De esta manera, los tres elementos más importantes quedan conectados en la red según los requerimientos de la implementación y la cámara podrá remitir las imágenes a analizar vía WiFi al Raspberry.

Para implementar el control de acceso físico, se emplea un pin de seguridad que es digitado mediante un teclado numérico matricial 4x4 que se conecta al Raspberry, la cual remitirá el número ingresado y lo cotejará con la base de datos. Este PIN es un número aleatorio, asignado por Ordenes de Trabajo, que se da a un solo colaborador y que es válido en un rango horario determinado.

Para lograr liberar recursos del Raspberry se implementa un API que estará a cargo de entregar al servidor información más destilada. Estas APIs también usan el framework Flask. Esto es necesario ya que las solicitudes de escritura a la base de datos lo maneja con cierta dificultad, lo que es notorio por la ralentización del sistema al cargarle la tarea de comunicarse con el servidor de manera íntegra. Para esto, el Raspberry funge como árbitro con respecto al ingreso del PIN, y mediante un identificador, notifica a la base de datos de actividad. El API se encarga de observar perennemente este identificador, el cual al cambiar su estado, realiza directamente los procesos de escritura al sistema de base de datos.

3.1 Módulo de Vigilancia

El análisis de imagen se realiza mediante el paquete de librerías para Python de OpenCV y YOLOv8, ambos siendo código abierto. Estos paquetes son un conglomerado de librerías de análisis de imagen por computadora y su implementación se puede revisar en la sección de Apéndice, apartados C y D (C y D respectivamente). Dentro de sus varias prestaciones, está la identificación de personas y el seguimiento de elementos dentro de la toma.

Funcionando de manera paralela tenemos al sensor visual, conectados al ESP32 Cam, que tomará la manifestación de un OV2640 conectada a través de la interfaz SCCB de la placa. La ubicación de esta se encuentra del lado de adentro del predio, a una altura de por lo menos dos metros y medio, y viendo hacia abajo para observar a los que entran y salen dentro de un área aproximada de un metro y medio [m2]. Este microcontrolador se comunica mediante WiFi con el Raspberry, y es el video que remite el cual es analizado por el algoritmo contador. Esta interacción se maneja en un servidor codificado en Flask, que es un framework de Python. Se hace uso de este framework puesto que a diferencia de Django es mucho más ligero, y hay menos configuraciones a alterar. Esto se hace deseable al ver los pocos recursos disponibles en las placas, y tener dicho servidor escuchando si hay algún cambio con respecto al conteo ayuda con los recursos. El ESP32 Cam controlará dos teclados numéricos, ubicadas a ambos lados de la puerta. De ser válido el pin ingresado en los teclados numéricos, se abrirá o cerrará un servomotor que hará las veces de pestillo.

Teniendo relación con ambos microcontroladores está la Aplicación Web, la cual tiene una base de datos que contiene las tareas a realizarse, recibirá la información de registro del contador de personas y cotejará si el PIN recibido por el teclado numérico anexado al Raspberry Pi es válido o no, enviando la orden de abrir o alertar.

3.2 Módulo de Aplicación Web

La base de datos contará con el Usuario Administrador, el cual podrá crear no solo otros usuarios para que los usen colaboradores de la FIEC, sino que también podrá registrar, revisar y eliminar ordenes de trabajo o tareas. Estas tareas refieren a las órdenes de trabajo levantadas por la persona pertinente en la FIEC, con el fin de realizar mantenimientos correctivos, preventivos, o alguna variación dentro del cuarto de Rack, todo esto por parte de prestadores externos.

El Usuario Colaborador de la FIEC no podrá sino ver la información que le compete a él, incluyendo el pin de ingreso al Centro de Datos.

De manera adicional, el Módulo de Aplicación Web, cotejará que el número que recibe por parte del contador coincida con la cantidad de personas que están autorizadas para estar en el Centro de Datos, según la orden de trabajo levantada, y podrá alertar al encargado en caso de que haya más gente de la necesaria, o haya un intento de ingreso fallido.

3.3 Módulo de Control de Acceso

Este Módulo, elaborado en Django y residente dentro de una máquina virtual en el Centro de Datos es el módulo que en principio reside entre Aplicación y Vigilancia, pero de forma práctica es el uso de datos generados por el código de Control de aforo e interpretado por el módulo de Aplicación.

De forma física se conecta con el Raspberry Pi y una de las funciones de control que tiene es cotejar si la contraseña ingresada es válida, y lo realiza al verificar con los PINes que estén almacenados dentro del sistema y que también esté acorde a la hora en la que se realizará el trabajo. Otra de las funciones es la recepción de las imágenes para su conteo.

El Control de Acceso también se encarga del análisis de ocupación, ya que toma las imágenes del Módulo de Vigilancia y las procesa, empezando a contar. La cantidad de personas que hay se coteja con la cantidad de personas que debería de haber y en base a eso remite una notificación al correo.

Con respecto de su interacción con el Módulo de Aplicación Web, no sólo es de proveer de la información requerida para su visualización y poder así colocarla en las páginas web que muestran datos históricos y operativos, sino que también es el de la verificación de credenciales ya que al momento de ingresar un usuario y contraseña, es aquí en donde se hacen las verificaciones pertinentes. Este módulo también es el que proporciona la información necesaria para que la aplicación genere las alertas y las remita vía correo.

3.4 Implementación

El sistema de seguridad se lo puede separar en dos partes. La primera refiere al Módulo de Aplicación Web, y la segunda es lo que engloba la analítica en tiempo real, esto es, el Módulo de Control y Módulo de Vigilancia.

La base de datos, que se relaciona con el Módulo de Aplicación Web, contiene toda la información que corresponde a la creación de tareas, vista de tareas existentes, control de usuarios con pines y cuántas personas deberían de estar presentes en el Centro de Datos de la FIEC. Esta parte de la implementación contará con dos tipos de Usuarios: el Administrador y el Colaborador. El Usuario Administrador tendrá la potestad de manipular la creación, destrucción y detalle de otros usuarios, así como determinar detalles para las tareas que se programen para su ejecución. Estas tareas tendrán información asociada: quién irá a realizar el mantenimiento, cuántas personas integrarán dicho equipo de mantenimiento, y qué colaborador estará a cargo de supervisar a aquel equipo.

DEL USUARIO 'ADMINISTRADOR'

El usuario Administrador tiene los privilegios que le permiten crear Órdenes de Trabajo y otros usuarios, así como designar como 'inactivos' a usuarios que se designe no deban acceder más al sistema. Por motivo de mantener un historial, este sistema no permite al usuario, sea Administrador o Colaborador, ejecute acción que borre del sistema tanto a Usuarios u Órdenes de Trabajo.

Figura 3.2: Captura de pantalla de la página web en donde se crean los usuarios, colores invertidos. Aquí se muestra la información requerida para la creación de nuevos usuarios.

El usuario Administrador tiene la potestad de dar privilegios de Administrador a otros usuarios, permitiendo así que varias personas puedan manejar la creación de órdenes y usuarios según la conveniencia de la organización.

Creación de Usuario. El Administrador tiene la capacidad de crear usuarios de tipo 'Colaborador'. Para aumentar la seguridad, cada usuario creado debe también de ser activado por el encargado. Como se ve en la Figura 3.2, al crear el usuario se pide el correo electrónico y las credenciales de acceso que va a usar. El correo electrónico ingresado es al correo donde se remitirán los pines de acceso que se crean de forma específica para cada Orden de Trabajo.

Activación e Inactivación de Usuario. Cada usuario Colaborador, para usar el sistema, debe de ser activado. El sistema mostrará la calidad de todos los usuarios presentes al Administrador, tal como se muestra en la Figura 3.3. Al ser activado, se notifica mediante un correo electrónico tanto al Administrador como al usuario Colaborador. Todo usuario que es creado en el sistema no puede ser eliminado. El Administrador puede desactivar el usuario, pero no lo puede borrar de la base de datos. Esto para que exista un histórico de usuarios creados, así como para tener la posibilidad

Figura 3.3: Captura de pantalla de la página web que muestran los usuarios y su estado activo o inactivo.

de desactivar ciertos usuarios después de que un trabajo se complete, de acuerdo con las políticas de seguridad que defina el encargado.

Modificación de Usuario. Un usuario puede ser modificado por el Administrador.

Figura 3.4: Captura de pantalla de la página en donde se crean las Ordenes de Trabajo.

Creación de Orden de Trabajo. Sólo el Administrador puede crear la Orden de Trabajo. Para esto, y como se muestra en la Figura 3.4, el programa solicita el ingreso

de la fecha en la que se va a realizar el trabajo, la hora en la que el trabajo va a principiar, el periodo de tiempo en el que se podrá realizar dicho trabajo, cuál será la actividad que se va a realizar, qué compañía es la que va a realizar la labor y la cantidad de personas, aparte del personal encargado, que va a estar dentro de las premisas.

Activación de Orden de Trabajo. Para que una Orden de Trabajo aparezca dentro del Sistema ante los ojos de los Colaboradores, primero debe de ser activada. La activación genera un PIN, el cual es remitido tanto al Administrador como al Colaborador que va a gestionar la actividad de dicha Orden de Trabajo. Si la Orden de Trabajo no se encuentra activa, no tendrá un PIN designado.

Para activar órdenes de trabajo, se accede al tab de navegación, que muestra la interfaz web cuya única tabla muestra el histórico total, en donde los elementos con menor tiempo de creación aparecen en la parte superior.

En esta tabla, al encontrar el elemento inactivo que se desee activar, se hace click en el ícono de inactivo para activar la Orden de Trabajo.

Modificación de Orden de Trabajo. Una Orden de Trabajo creada puede ser modificada en tiempo, en caso de que el arreglo programado haya tomado más de lo esperado o se haya informado la necesidad de variar la cantidad de personas que integrarán el equipo técnico que estará en las premisas.

Figura 3.5: Captura de pantalla de la página web que muestra el historial de usuarios registrados en el sistema.

Histórico de Usuarios. Sólo el Administrador puede ver los usuarios que se encuentran dentro del sistema. En esta interfaz web de administración de usuarios, que se puede apreciar en la Figura 3.5, se muestra una tabla con las columnas de ‘usuario’, ‘nombre’, ‘apellidos’, ‘correo electrónico’, ‘activo’, y, ‘Admin’.

Estas columnas, aparte de mostrar la información de los usuarios, muestra si dicho usuario tiene privilegios de administrador. También muestra si dicho usuario se encuentra activo. Adicional, la muestra es interactiva, puesto que, al presionarlo como botón, cambia el valor dentro de la celda. Esto permite cambiar el estado de activo a inactivo de manera fácil, así como también darle privilegios a un usuario de forma rápida.

Figura 3.6: Esta imagen muestra todas las Ordenes de Trabajo en el Sistema.

Histórico de Órdenes de Trabajo. El administrador tiene la capacidad de ver las Órdenes de Trabajo activas e inactivas. En esta interfaz web se muestra una tabla con ocho columnas, que son: Inicio de trabajo, duración, actividad, compañía, aforo, solicitante, PIN, y activo.

En estas se muestra información pertinente al tipo de trabajo que se va a realizar dentro de las premisas. En inicio de trabajo aparece la fecha, en duración, el tiempo que va a tomar realizar dicha tarea, en actividad, muestra el detalle de la labor que se va realizar, en compañía, la compañía a la que pertenecen los trabajadores, en aforo, la cantidad de personas que conforman el equipo técnico; en solicitante, el usuario que creó la orden, en PIN, el código de seguridad que debe de ingresarse para franquear la entrada, y en activo, el estado de la orden de trabajo, esto es, si dicha orden de trabajo, ya ha sido aprobado o si se encuentra en revisión.

Órdenes por Aprobar. Esta interfaz de la página web permite al administrador observar cuáles son las órdenes de trabajo que aún se encuentran pendientes por aprobar. En ella se muestra una tabla con las siguientes columnas: Solicitante, Fecha compañía, Duración, Actividad, y Aprobar.

La columna solicitante nuestra cuál es el usuario que creó el orden de trabajo. La

Figura 3.7: Página web que muestra las órdenes pendientes por aprobar. De colores invertidos, se muestra un listado de todas las órdenes que se encuentran pendientes, y se pueden activar con un botón.

columna de fecha muestra para cuándo dicha orden de trabajo se va a efectuar. La compañía muestra el nombre de la organización encargada a mandar el equipo de trabajo. La columna de duración refiere al tiempo que se estima necesario para realizar la prueba. La columna actividad contiene el detalle de las actividades profesionales que se han solicitado. La última columna, muestra la etiqueta de Aprobar. Ella muestra la información de la orden de trabajo. Si se encuentra activo, no, y también permite en un solo clic, activarla o desactivar.

DEL USUARIO 'COLABORADOR'

(a) Administrador (b) Colaborador

Figura 3.8: Se muestra la diferencia que hay para los usuarios con respecto a la vista de Ordenes de trabajo. A la izquierda se ve un listado no sólo de las órdenes activas, sino también de las órdenes inactivas, pendientes de aprobación. A la derecha sólo se ven las órdenes activas y son inmodificables.

A diferencia del usuario Administrador y tal como se muestra en la Figura 3.8, la cuenta de Colaborador es una cuenta que, al tener privilegios restringidos, puede solicitar nuevas Órdenes de Trabajo pero tiene que esperar hasta que éstas sean aprobadas. Adicional a esto, recibe el PIN asignado de las órdenes de trabajo que tiene activas.

DE LOS PROCESOS DEL APLICATIVO

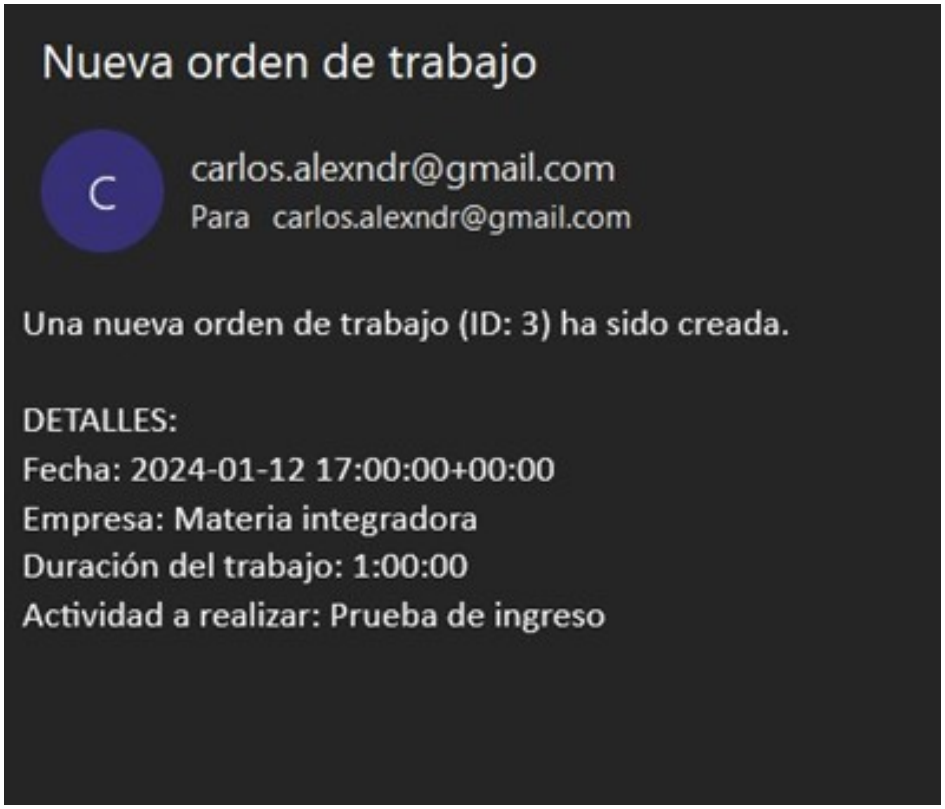


Figura 3.9: Muestra un correo enviado al momento en que se crea una nueva orden de trabajo.

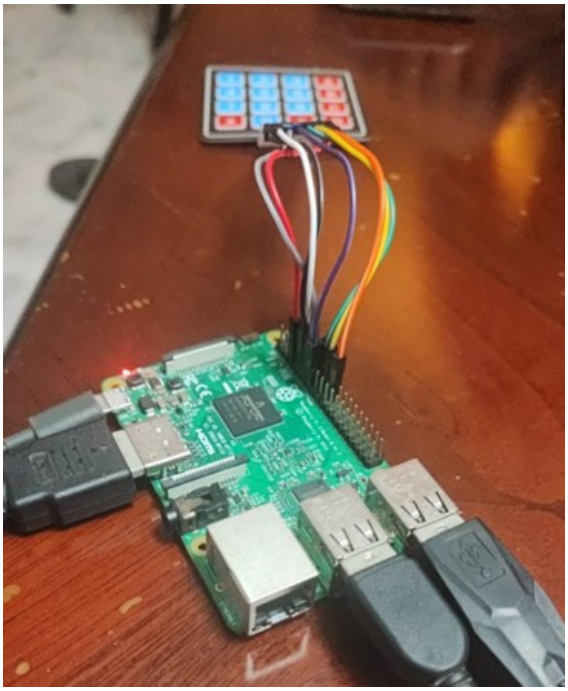
Correo de notificación de Orden de Trabajo creada. El aplicativo manda una alerta por correo al Administrador cada vez que se crea una nueva Orden de Trabajo. Esto se muestra en la Figura 3.9. El correo cuenta con los detalles de fecha en la que el trabajo va a ocurrir, qué empresa ha sido contratada para ejecutar la obra, cuál será la duración del trabajo y que actividad tendrá lugar dentro de las premisas. La duración del trabajo estipula también el tiempo por el cual el PIN será válido para ingresar.

Notificación de alerta por correo. El sistema remite notificaciones al correo cada vez que se ingresa un PIN incorrecto, o inclusive un PIN asignado a una Orden de Trabajo que no ha empezado aún, o que ya ha terminado.

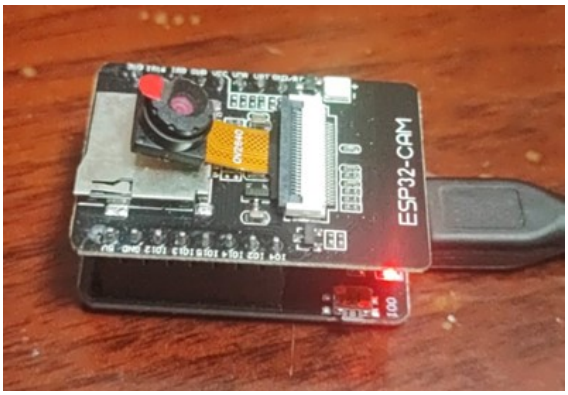
De manera adicional, se remite una alerta al correo de contacto del Administrador si hay más personas de las que debe de haber en el rack.

DE LOS PROCESOS DE LAS PLACAS

La parte de analítica en tiempo real engloba al control de ingreso y egreso, que se realiza por medio de análisis de video en tiempo real, y control de acceso por medio de pines. Tal como se muestra en la Figura 3.10, el Raspberry tiene conexión directa con el teclado numérico, tal como se muestra en la Figura 3.10, a la izquierda; y una conexión



(a) Raspberry y Teclado



(b) ESP32 Cam y OV2640

Figura 3.10: A la izquierda, se ve la imagen de la Raspberry Pi 3B conectada al teclado numérico. A la derecha se ve la placa ESP32 Cam con la cámara OV2640. La placa se encuentra sobre un shield de programación, que también lo energiza.

inalámbrica vía WiFi con la placa ESP32 Cam, que tiene una cámara IP modelo OV2640.

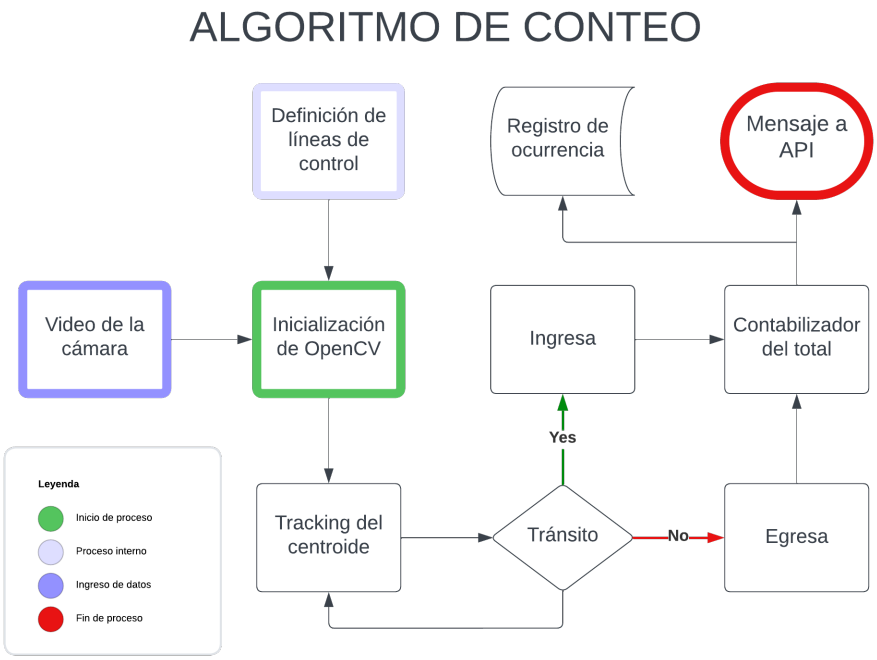


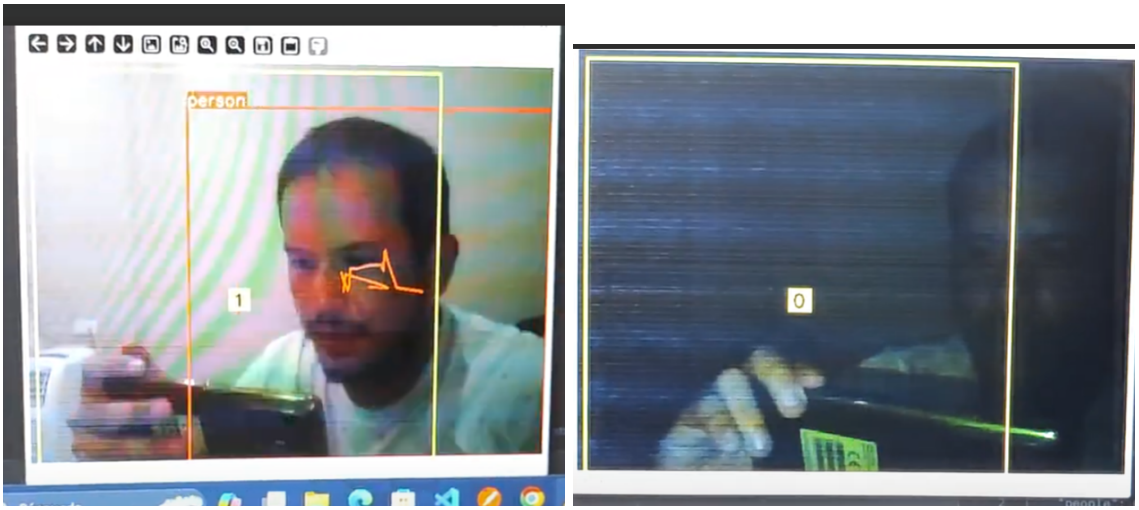
Figura 3.11: Diagrama de flujo del funcionamiento del algoritmo de conteo. Obra propia.

La Figura 3.11 detalla el funcionamiento del algoritmo. Primero, se recibe la

información en bruto de la cámara, esta va a ser analizada por el Raspberry. La cámara, conectada al ESP32 Cam, transmite la información al Raspberry Pi. Dentro de éste yace el algoritmo de conteo, que se puede observar en el Apéndice, sección E (E), que hace uso de una librería llamada YOLOv8, que refiere a análisis de imágenes por computadora.

Figura 3.12: Captura de pantalla del funcionamiento del programa de conteo. Se ve un rectángulo formado por dos líneas de control, sobre cuyo tránsito se define el ingreso o egreso.

Tal como se ilustró en la Figura 3.12, hay dos puntos importantes a considerar. En el primero, lo que hará el algoritmo es hacer uso de tracking de personas, para ver si dentro de la imagen podemos observar tránsito. Esto se muestra con el rectángulo rojo que llama la atención hacia la persona. A dicho rectángulo se le calculará el centroide. Este centroide es el elemento que usaremos para identificar si una persona ingresa al Centro de Datos. Para denotar el tránsito, se crea un 'histórico' que denota las coordenadas previas del centroide, lo que se ve como la línea verde. Estado esto listo, como segundo paso tenemos la definición de tres líneas de control, definidas perpendiculares a la línea de tránsito esperada. El funcionamiento de estas líneas es simple, dependiendo si el centroide se mueve de la línea media hacia algún extremo, se contabilizará acorde si es un ingreso o un egreso. En este caso, definimos que la línea 2 será la línea central. Cada acción estimulará una variable, en este caso, cuántos han ingresado, cuántos han salido y cuál es el total. Toda ocurrencia de este tipo es documentada en un log, que así mismo contiene los datos de fecha, hora, tipo de movimiento, y total.



(a) Iluminación apropiada. (b) Falta de iluminación.

Figura 3.13: A la izquierda se ve el reconocimiento de la persona, junto con el histórico de su movimiento. A la derecha se ve la aplicación funcionando con falta de iluminación, en la que no reconoce al sujeto.

Se determina también, y por completitud es necesario comentarlo, que para que el sistema funcione de forma correcta, el lugar a custodiar tiene que estar bien iluminado, ya que el común de las cámaras no funcionan de forma apropiada al haber poca luz, perdiendo detalle, aumentando granularidad e incrementando la borrosidad de las imágenes adquiridas. Esto da como resultado que a poca luz, el programa no va a detectar a personas correctamente, llevando a fallas operativas. Lo que está descrito en este párrafo lo podemos ver en la Figura 3.13. A la derecha, con la iluminación apropiada, se ve la detección de la persona, así como el histórico de su movimiento graficado en la línea irregular naranja. A la izquierda, con una iluminación no apropiada y aún con la luz de la pantalla iluminando la cara, el algoritmo detecta la existencia de algo, pero no lo califica como persona. Como tal, no define ningún histórico, no existiendo línea alguna.

El otro procedimiento que tiene que gobernar el Raspberry Pi es la recepción y remisión de un PIN mediante un par de teclados numéricos. Este PIN es creado por el mismo sistema y es asignado al colaborador. Cuando los equipos entren al Centro de Datos, cada ingreso se registra con dicho PIN. El teclado registra el PIN ingresado y lo remite a la base de datos, esperando una respuesta. Si la base de datos autoriza el PIN, mandará una señal que moverá un servomotor y permitirá la apertura de la puerta. Si no es reconocido, se registra una alerta. El proceso de funcionamiento es como sigue en la Figura 3.14. Nótese cómo ambos sistemas trabajan en conjunto.

Figura 3.14: Diagrama de flujo de asignación de pin. En morado se ve el inicio del flujo y en dorado.

La Figura 3.14 muestra el flujo de asignación del PIN. Esto ocurre al crear una tarea nueva. Cuando se levanta un requerimiento, se definen los detalles, esto es, qué colaborador estará a cargo de recibir al equipo de trabajo y a qué empresa pertenece dicho equipo de trabajo. También se define una hora en la cual se llevará a cabo la actividad planificada, sea la naturaleza que sea, y cuántas personas vendrán, sumado al encargado.

Una vez que todos esos detalles se han definido, el Administrador ingresa un PIN, el cual será válido para al acceso al Centro de Datos. El PIN creado es de 4 números y sólo está disponible durante la franja horaria en la que se hará el trabajo. En caso de que el trabajo se extienda, deberá ser comunicado acorde al administrador, para que modifique la extensión del trabajo y así no expire la validez del pin. De igual manera, dicho pin sólo es conocido al colaborador, quien es el encargado de hacer que todos los trabajadores ajenos a su actividad normal dentro del rack ingresen o salgan.

Si por fuerza mayor el colaborador se retira, dará una tarjeta de invitado con un pin designado, que será usado sólo por aquella persona y es de manera muy específica para salidas de fuerza mayor, sea ver material al automóvil o ir a la sala de baño.

El presente diagrama (Figura 3.15) muestra el proceso de validación del PIN, algo

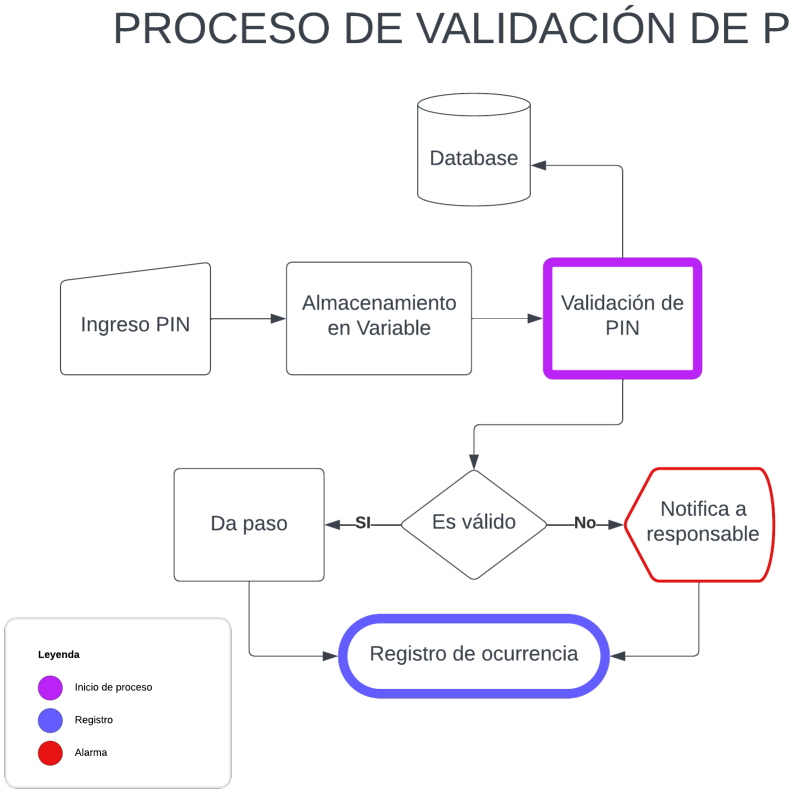


Figura 3.15: Diagrama de flujo de validación del PIN. La gráfica muestra como es un proceso dinámico en donde se recibe información del teclado numérico y se la consulta con la información en la base de datos. El resultado de esta información determinará si se da paso o se eleva una alerta. Obra propia.

muy notorio y correcto. Notorio por el hecho de ser necesario y correcto por la tácita tautológica del contrato social.

Primero, en la creación de tareas se define un pin para dicha tarea. Por lo tanto, la base de datos dentro de sí va a tender un conglomerado de pines, válidos todos pero para franjas horarias específicas. Fuera de ellos no accionarán el servo y no darán paso.

El ESP32 Cam, con los teclados numéricos asociados, recibirá un PIN, que será almacenado en un buffer antes de ser remitido a la terminal. El proceso de esta comunicación es mediante una relación servidor-cliente, donde el ESP32 Cam es el servidor, y remite el pin mediante una conexión Wifi.

La Aplicación recibe este mensaje, que lo coteja con los pines que se encuentran en la base de datos. Si el pin no existe, se levanta una alarma enviando un correo al responsable. Pero, si el pin existe, se precisará entonces verificar si el ingreso se ha

realizado durante la franja horaria para el que está disponible. En caso de que esté dentro de la franja horaria, el ESP32 Cam accionará un servomotor, el cual, liberando el pestillo, permitirá el paso.

Si el pin no está dentro de la franja horaria permitida, se levanta una alarma y se remite un correo al encargado.

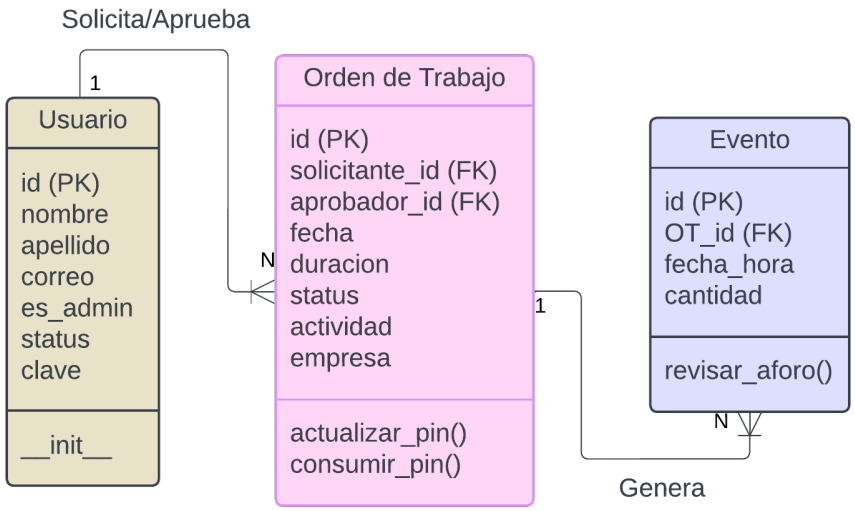


Figura 3.16: Diagrama de Relaciones del proyecto en donde se grafican las tres entidades, Usuario, Orden de Trabajo y Evento.

En un diseño relacional de base de datos, las tablas se relacionan con uno o varios valores únicos, denominados “llaves primarias o primary keys”, compartidos entre las tablas para de esta forma establecer una relación de entre los diferentes registros de nuestras tablas. Esto se ve en la Figura 3.16.

En la figura adjunta, se puede observar la nomenclatura de PK asociada a un atributo único en la tabla de usuarios, ordenes de trabajo y eventos a guardarse en nuestra base de datos, así como también la nomenclatura FK que hace referencia a una llave foránea o “Foreign key” el cual hace referencia a un atributo único perteneciente a la tabla de la que hace referencia.

La tabla de usuarios posee atributos con la información del personal dueño de la cuenta para su identificación y comunicación con el mismo, en caso de ser requerido por medio del atributo de correo electrónico. A su vez, también posee un campo para realizar borrados lógicos y su persistencia de datos en el sistema.

La tabla de ordenes de trabajo, consiste en los datos generales de las ordenes de

trabajo a realizarse, tales como duración para validación del pin de acceso, actividad para datos sobre el trabajo y, además los campos de solicitante_id y aprobador_id para ser relacionados al usuario que solicita el permiso de ingreso y quien aprueba esta orden de trabajo.

La tabla de eventos almacena la información enviada por el componente de hardware del proyecto y consiste en la información exacta de ingresos y salidas de personal, así como un campo de OT_id que hace referencia a la orden de trabajo asociada al evento en cuestión.

El sistema utiliza tecnología de autenticación basada en tokens JWT, los cuales consisten en un par de cadena de caracteres que son almacenados en base de datos y relacionados a la sesión de cada usuario y deben ser incluidos en el header de los requerimientos HTTP realizados al servidor.

En la Figura 3.17, se presenta el diagrama de flujo del inicio de sesión de un usuario y un requerimiento HTTP del tipo Get al servidor, se inicia realizando el ingreso de los datos de inicio de sesión por parte del usuario (username y password), el navegador incluye esta información en el cuerpo del mensaje y es enviado al endpoint de nuestro backend API. Luego de validar los datos y generar los tokens respectivos (token de acceso y refresco), estos son enviados de vuelta al navegar y son almacenados en memoria para ser utilizados en futuras solicitudes hacia el servidor backend. Estas solicitudes se realizan dentro de un marco de seguridad, ya que estos procedimientos son la parte visible de uso de tecnologías como JWT. La configuración de seguridad implementada en el backend se puede ver en la sección de Anexos, sección F (F).

Toda solicitud diferente a la de inicio de sesión, debe incluir el token de acceso en el encabezado de la solicitud HTTP, para ser validada la identidad del usuario y permisos asignados al mismo por el sistema. En caso de que el token se encuentre activo y el usuario posea permisos de acceso al endpoint solicitado, el sistema devuelve una respuesta con código 201 y es renderizado por parte del navegador.

En los casos en que la información entregada por parte del usuario es incorrecta, el servidor devuelve respuestas con código 400 y se renderizan mensajes de error por parte del navegador para el reintento de este por parte del usuario, según podemos observar en el diagrama de flujo siguiente.

En los casos en que el token de acceso se encuentra expirado, el sistema devuelve

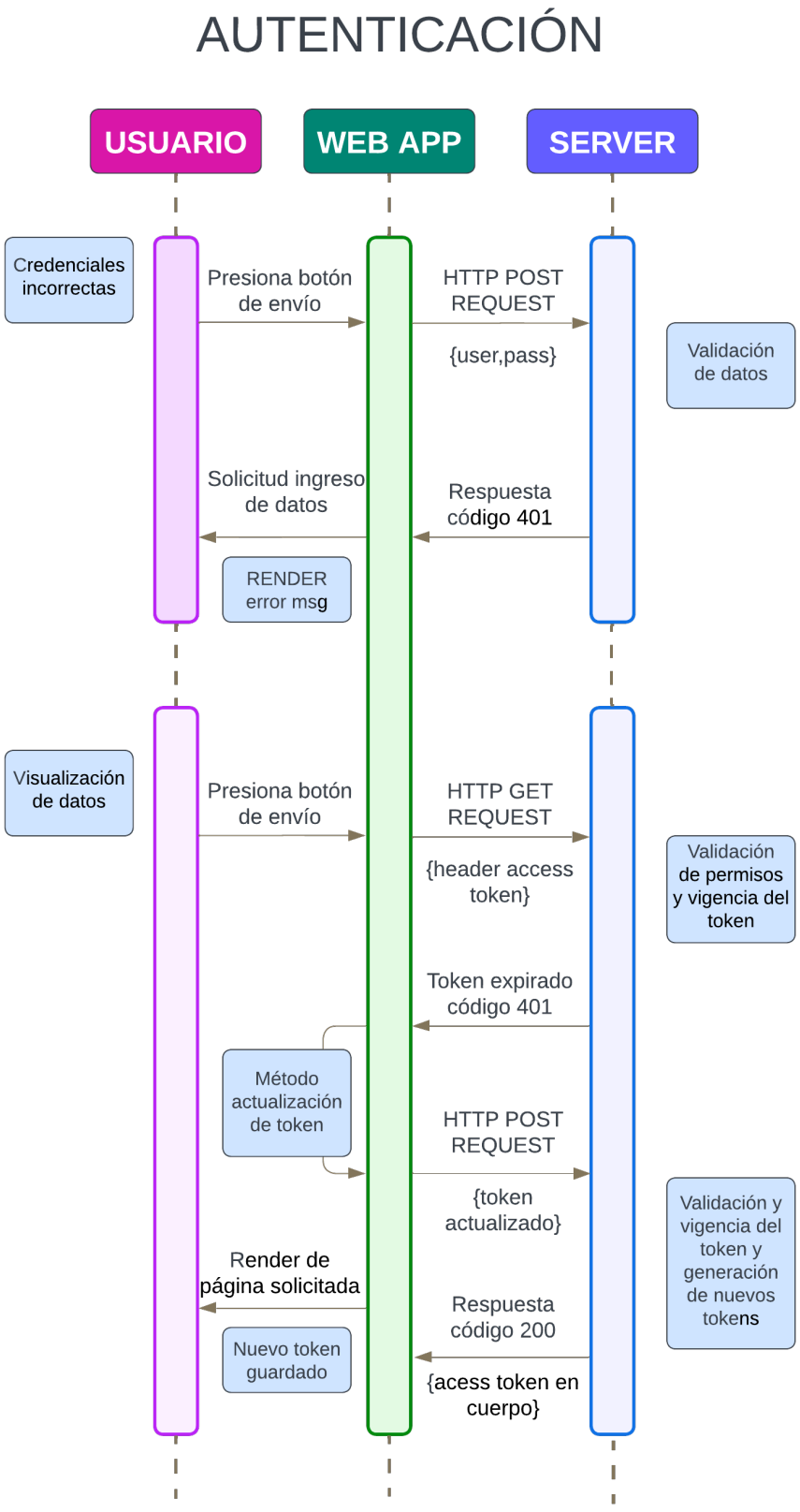


Figura 3.17: Diagrama de interacción entre entidades. Obra propia.

una respuesta con código 400 al frontend y este a su vez, realiza una solicitud de actualización de los tokens (se realiza una solicitud al endpoint de actualización y se incluye el token de refresco en el cuerpo del mensaje), todo esto de manera transparente para el usuario y almacena sus nuevos tokens de acceso en memoria del sistema (Figura 3.18).

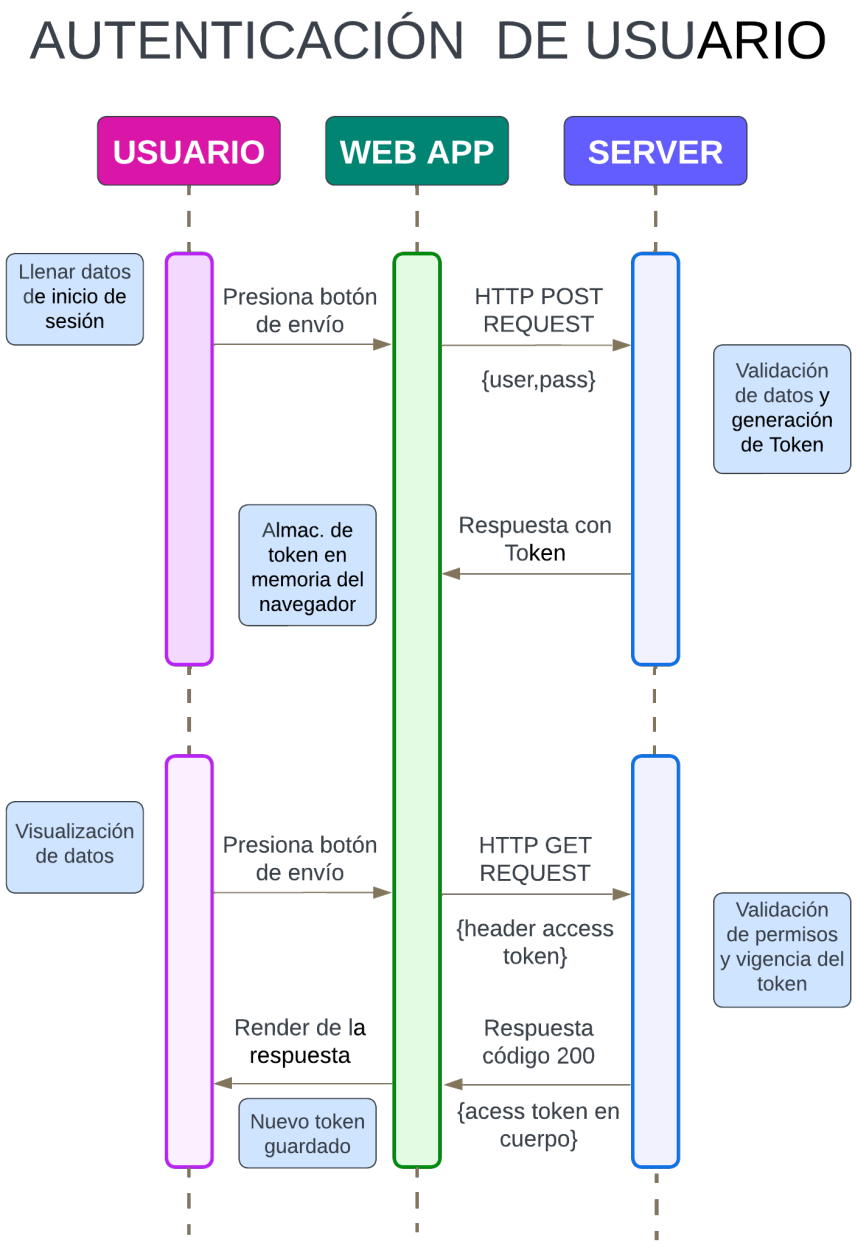


Figura 3.18: Diagrama de autenticación de información entre las entidades. Obra propia.

PROTOTIPO.

Con el código en funcionamiento, se procede a colocar la placa dentro de una caja

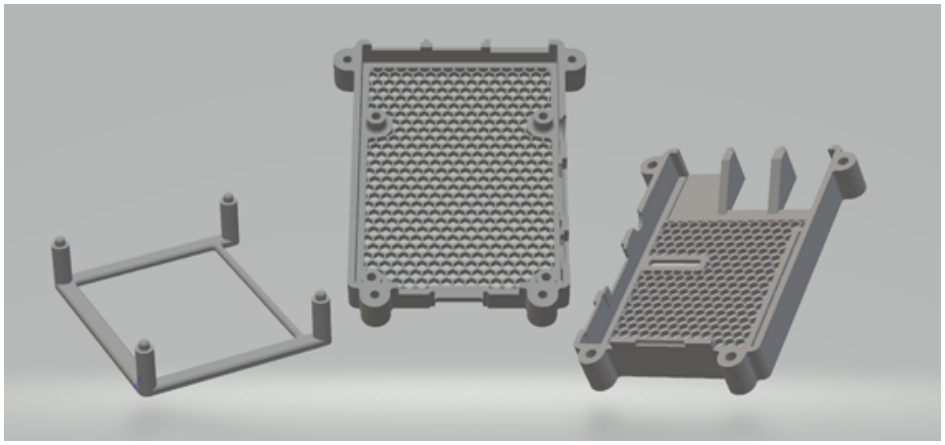


Figura 3.19: Modelo en tres dimensiones de la caja que protegerá al Raspberry Pi. Esta placa será impresa en 3D.

protectora, mostrada en la Figura 3.19, para facilitar su manipulación al instalarla, así como también para evitar que los elementos internos se encuentren a buen recaudo. Esta caja está compuesta por tres partes, un soporte; y las dos partes de la carcasa externa.

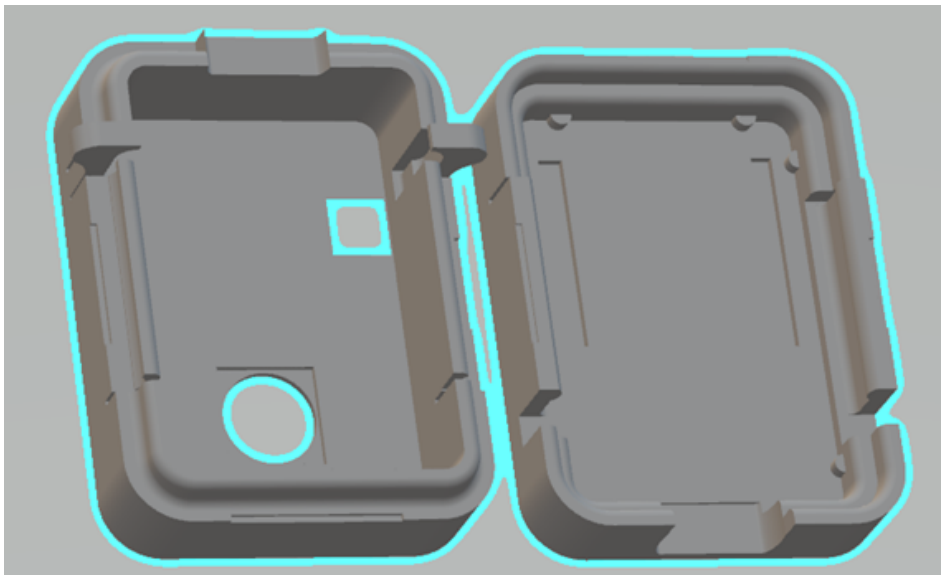


Figura 3.20: Modelo en tres dimensiones de la caja que protegerá el ESP32 Cam junto a su shield. Esta placa será impresa en 3D.

La placa ESP32 Cam también es colocada dentro de un reducto protector, cuyo diseño se muestra en la Figura 3.20. Ambas están hechas de plástico PLA e impresas en impresoras 3D. Ambos modelos son de código abierto.

CAPÍTULO 4

4. ANÁLISIS DE RESULTADOS

De la funcionalidad y eficiencia del Algoritmo. La primera prueba se la realiza con un video, que muestra el ingreso y egreso de personas. Se usó este video para probar el algoritmo en su primera versión, usando la librería OpenCV. Dicho video muestra tres personas que cruzan un portal en un sentido y dos que lo cruzan en sentido contrario. Preparando la placa de la manera descrita en el capítulo de Implementación, corriendo el algoritmo directamente desde la placa Raspberry Pi 3B, se realizaron 30 corridas. El registro de dichas 30 corridas muestra, de manera consistente y por 30 veces, tres ingresos y dos salidas, lo que fue considerado como aceptable a todas luces. Por la falta de recursos, se creó otro algoritmo implementando la librería YOLOv8, la cual permitió el análisis del video con mayor velocidad, consumiendo menos recursos de la placa. En este nuevo algoritmo, no se usó el video, sino que se utilizó el video directamente de la cámara IP, transmitido a través de la placa ESP32 Cam.

De la decisión de implementar un uso híbrido de librerías OpenCV+YOLOv8 en vez de sólo OpenCV. El uso híbrido de OpenCV con YOLOv8 fue la permutación que se utilizó para la implementación final. Antes de llegar a esta configuración se realizó una la siguiente prueba: Primero la instalación de OpenCV: En el Raspberry Pi 3B se contabilizó 2 horas y 52 minutos. En una computadora i7 de novena generación tardó 1 hora y media en instalarse.

Usando OpenCV, al crear el Punto de Acceso con la ESP32 Cam y enlazar con ello al Raspberry, se observó que el video se ralentizó, cayendo al umbral de diez fotogramas por segundo. Al iniciar el algoritmo se certificó su funcionamiento con el video, pero usando la cámara IP, el funcionamiento decayó. El movimiento de los sujetos que deben de ser contados tenía que ser lento, a menos de la mitad del paso normal de un adulto, que por convención se le da el valor de 10km por hora. Se observó también que al implementar la

conexión con la base de datos y los otros módulos causaba que los fotogramas bajasen a cinco fotogramas por segundo, cuando la placa no estaba siendo exigida. Se denotó una marcada inestabilidad en el sistema, siendo la gran mayoría de los casos el que correr el Algoritmo diese como resultado la caída del programa, tal como se ve en la Figura 4.1, lo cual no era aceptable. Se revisó el consumo de recursos al momento de correr el Algoritmo y el resultado fue un uso casi total de la memoria RAM. Adicional a esto, la comunicación con el servidor para la escritura de ocurrencias a la base de datos también fue un punto que impactó la ralentización del sistema, causando nuevamente su caída. Esto se observó ya que, al momento de solicitar escribir en la base de datos, el programa fallaba. Por esto se decidió implementar el sistema híbrido, combinando la librería general de OpenCV con el módulo particular y de código abierto denominado YOLOv8, ya que este último es más eficiente en el uso de los recursos.

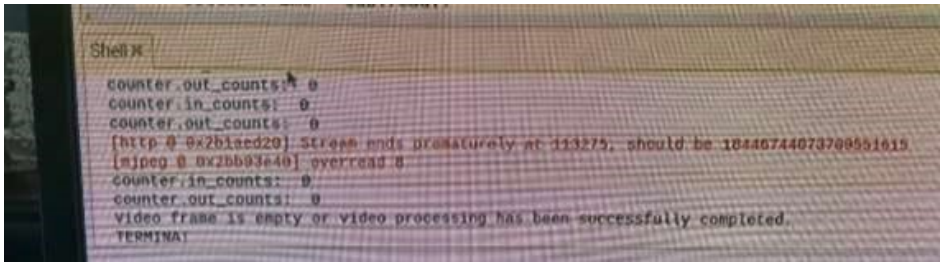


Figura 4.1: Imagen que muestra el error resultante referente a memoria.

La diferencia entre ambos yace en la manera en que se realizan los análisis. Las herramientas nativas de OpenCV analizan cada fotograma en dos pasos: el primero determinando la ubicación del objeto y el segundo identificando qué es, que ocurre aun cuando no se hace uso de dicha utilidad. Eliminar este segundo paso requiere una edición profunda de las librerías, lo que se obvió por tiempo. También hay que considerar que el hecho de graficar los artefactos visuales (caja de seguimiento, centroide, etc.) suman a los recursos consumidos. YOLOv8, en cambio, divide el fotograma en una red y realiza el análisis visual una sola vez. Esta manera de funcionar lo hace más ligero con respecto a consumo de recursos, a expensas de la exactitud al obtener la ubicación de objetos pequeños, ya que el modelo determina la posición en base a probabilidades de que dicho objeto se encuentre dentro de una cuadrícula, y a cuántos objetos caben dentro de la red en la que se dividió la imagen. Este cambio permitió que los fotogramas ascendiesen nuevamente a un promedio de 15 fotogramas por segundo. Con esta implementación híbrida se pudo correr el algoritmo, primero en la computadora, y se

pudo obtener resultados de forma aceptable a los requerimientos de la implementación, esto es, caminar lento.

Se realizó la prueba en tres sistemas:

Tabla 4.1: Comparativa de características de sistema.

	Raspberry Pi 3B	**PC**	**MacOS**
RAM	1Gb+0.5Gb (Swap)	16Gb	32Gb
Procesador	x64	i7 10th gen x64	i7 10th gen x64

Si bien el algoritmo corrió al realizarse sobre el video un análisis más gentil, por la falta de recursos y siendo la librería YOLOv8 basada en probabilidades, empezaron a tener lugar fallas de conteo en el Raspberry Pi. Estas fallas no se observaron al momento de correr el programa en una laptop con 16Gb de memoria RAM, siendo usados 14,76Gb. Hay que mencionar que aun así el programa se ejecutaba con notoria ralentización a 15 fotogramas por segundo. Sin embargo, el programa, corriendo en una laptop con más memoria, esto es, con 32Gb, fue la que mejor desempeño dio con respecto a fotogramas por segundo, subiendo a 27. El uso en esta máquina de memoria RAM se mantuvo en el valor de 15.23Gb de RAM.

De la elección del sensor OV2640 por encima del AMG8833. La primera iteración se eligió el sensor AMG8833. Este es un sensor térmico de resolución ocho por ocho para un total de 64 bits. Se lo consideró en un principio porque la cámara se había definido estrictamente para conteo, no para observar quién ingresaba. Para tener una mejor imagen, se aplicaba una extrapolación para duplicar de forma virtual la entrada a 16 por 16 bits. Este aumento sería pasado por un filtro bi-cúbico para suavizar los píxeles y poder así definir una mancha térmica. Este algoritmo de conteo tenía que pasar por dos procedimientos para facilitar el trabajo del algoritmo, mismo que se explica su funcionamiento con mayor detalle en la ilustración 15. Para esto se necesita tener una buena imagen, y para lograrlo se siguió el procedimiento siguiente: El primer paso era una interpolación, para aumentar de forma virtual la matriz de ocho por ocho píxeles a un valor más apropiado de 16 por 16 píxeles, mediante la librería scipy. El segundo procedimiento era aplicar a dicha imagen un filtro bi-cúbico para suavizar así la dureza del cambio de un píxel a otro. Este segundo paso permitía la creación de una ‘mancha’ térmica, y sería el seguimiento de esta imagen a través del predio el producto que finalmente

sería analizado por el algoritmo de conteo, cuadro a cuadro. En tiempo real, se obtuvo un promedio de cuatro cuadros por segundo en la mejor de las situaciones. En casi todas las instancias el programa dejaba de funcionar casi inmediatamente: Al momento de iniciar el algoritmo este se caía inmediatamente, el motivo siendo el consumo alto de recursos, específicamente la falta de memoria RAM de la Raspberry Pi (1Gb disponible), lo que ralentizaba todos los procesos de la placa, causando la inestabilidad del sistema. Esto ocurría debido a que debido al análisis previo y aplicación de filtros, que no permitía rastrear la mancha térmica.

Observando la marcada lentitud del sistema, se procedió a pasar dicho video por el algoritmo de conteo, el cual no fue exitoso sino al pasar muy lentamente delante del sensor, lo cual no es aceptable para la aplicación requerida. El error que tenía lugar ocurría por el salto del artefacto térmico de un fotograma a otro, lo que hacía al algoritmo pensar que no era el mismo objeto, sino dos objetos distintos. Al no haber manera de conciliar la posición cuadro a cuadro, el programa se colgaba. Siendo este el caso y, analizando que aumentar el paso de transmitir dicha imagen a través de una conexión inalámbrica ralentizaría aún más el proceso, contribuyendo al fallo del algoritmo de conteo, no se implementó la conexión del sensor AMG8833 a la placa ESP32 Cam, tal como se ve en la Ilustración 16.

La finalización de esta primera prueba principió con el mismo algoritmo, pero cambiando el sensor visual, eligiendo la cámara OV2640 que se conecta de forma cableada a la ESP32 Cam, lo cual fue positivo ya que, al procesar video a 15 cuadros por segundo el conteo se realizó sin problema alguno. Este éxito viene del hecho de que, a diferencia del sensor anterior, cuyo producto tenía que pasar, cuadro a cuadro, por dos procesos computacionales relativamente intensos (para la Raspberry Pi), los fotogramas del OV2640 se procesaban en el algoritmo directamente. Es importante notar que en esta prueba la ESP32 Cam estaba conectada directamente al Raspberry Pi.

Desechando la implementación del sensor térmico se optó por el uso de una cámara normal, para evitar el sobreconsumo de recursos. La placa elegida fue la ESP32 Cam, que es una tarjeta programable para desarrollo. Este modelo en particular viene con una cámara IP. Al realizar la implementación se pudo constatar que el algoritmo pudo funcionar sin problema alguno. Un elemento positivo que se aprovechó fue la capacidad de dicha placa de crear un punto de acceso (AP), que sería aprovechado para crear una

red interna, en donde fluiría la información entre todos los elementos más importantes previo a su envío hacia la red externa.

Esta implementación corrió sin problemas hasta que se agregó la conexión a la Base de Datos y la implementación del código del teclado numérico. Con la base de datos, el problema que surgió fue que la carga adicional ralentizaba la placa a un nivel en que la base creía que ésta simplemente no respondería. Para solucionar esto, primero se aumentó el tiempo de escucha, pero, para quitarle el peso de escritura de datos a la placa se creó un API que se encargaría de aquello. Con respecto a la implementación del código del teclado numérico, se notó que el procesamiento de imágenes mediante OpenCV se tornó muy lento, necesitando que las personas pasasen de uno a uno y lento, lo cual no era aplicable. Para resolver este problema se buscó e implementó un segundo algoritmo usando YOLOv8, que mejoró el tiempo de procesamiento de imágenes.

Conexión entre Raspberry y Base de Datos. Se logró comunicar la placa de Raspberry con el Servidor. Fue necesario realizar un cambio con respecto al tiempo de espera de los mensajes, ya que el valor por default es menor al requerido por la placa la cual, al estar procesando el video, necesita un valor ligeramente más alto para establecer la conexión, remisión y recepción de información. En todas las instancias el Raspberry comunicó al Servidor el número de personas que ingresó al lugar.

Conexión del teclado numérico con el Raspberry Pi y la Base de Datos. Se programó el teclado numérico y se la programó para su uso a través de la Raspberry Pi. Se logró hacer que, el PIN definido por la base de datos para las Órdenes de Trabajo se coteje con el ingresado por el teclado numérico, y así tomar la acción acorde a lo estipulado en el capítulo de Método y el capítulo de Implementación.

CAPÍTULO 5

5. CONCLUSIONES Y RECOMENDACIONES

5.1 Conclusiones

En relación con los resultados obtenidos del escenario de la prueba uno, el cambio de sensor térmico AMG883 al sensor OV2640 fue de suma importancia para la implementación del sistema, específicamente la etapa de funcionamiento del algoritmo, puesto que los fotogramas procesados ascendieron de 4 fotogramas por segundo a 15 fotogramas por segundo, un aumento del 275 por ciento por encima del valor original. Con este resultado se aporta tanto al Objetivo Específico 1, que engloba la implementación de un Módulo de Control de Acceso; así como del Objetivo Específico 3, que refiere a la implementación de un Módulo de Vigilancia.

En la prueba dos se observó que el cambio de una implementación pura de OpenCV hacia una implementación híbrida de OpenCV con YOLOv8 fue favorable para la implementación del sistema, específicamente en la etapa de funcionamiento del algoritmo, pero ya en la etapa de implementación final, ya que se pudo elevar un valor de 7 fotogramas por segundo a 15 fotogramas por segundo, representando un aumento del 114 por ciento por encima del valor original. Con este resultado se aporta en la consecución del Objetivo Específico 1, que engloba la implementación de un Módulo de Control de Acceso; así como del Objetivo Específico 3, que refiere a la implementación de un Módulo de Vigilancia.

Como se puede observar, el programa permite, mediante una aplicación web y desde la cuenta de un usuario administrador, la creación de nuevos usuarios y ordenes de trabajo. Tanto los usuarios, como las órdenes de trabajo, deben de pasar por un proceso de validación, y el sistema remite las notificaciones pertinentes sobre cada etapa del proceso. Compuesto de estas funciones, el presente Módulo de Control de Acceso valida,

registra y notifica vía correo electrónico el ingreso y salida de usuarios autorizados al Centro de Datos de la FIEC, tal como está registrado en el capítulo de Implementación y siguiendo la vía trazada en el capítulo de método, cumpliendo así en su totalidad con el Objetivo Específico No. 1.

Conforme a lo mostrado, se logró implementar una Aplicación Web en la que se denota un histórico tanto de las órdenes de trabajo como de los usuarios que han sido registrados en el sistema, y procesa la información obtenida del módulo de control y del módulo de vigilancia en tiempo real, facilitando un monitoreo continuo del sitio. Para esto, se hizo un desarrollo del Backend y Frontend del proyecto, como está registrado en el capítulo de investigación y acorde a lo planeado en el capítulo de Metodología, cumpliendo así en su totalidad con el Objetivo Específico No. 2.

Se logró implementar un módulo de Vigilancia que detecta cuántas personas se encuentran dentro del centro de datos, y se logró que dicho sistema remita notificaciones vía correo a las personas encargado del Data Center, esto según el respaldo que reposa en la unidad de Implementación del presente documento, cumpliendo así en su totalidad con el Objetivo Específico No. 3.

Concluimos que, habiendo cumplido con la implementación de los Módulos de Vigilancia, Control de Acceso y Aplicación Web y conseguido su correcta interacción, se logró la implementación de un sistema de seguridad ciber-física basada en IoT para el control de acceso y detención de posibles intrusos en el Centro de Datos de la FIEC.

5.2 Recomendaciones

Programación. Tener en cuenta que por limitante de recursos de procesador de la Raspberry Pi 3B no se puede realizar muchas actividades de alto nivel. Para el proyecto, esto fue notable cuando al mandar la notificación HTML, se colgaba el análisis de imagen. Esta caída también disparaba problemas con los servidores y causaba su caída: al tener problemas con la solicitud HTML paraba tanto el servidor del Backend y API y también la Raspberry Pi.

De la memoria del Raspberry Pi 3B. Aumentar el Swap permitirá que se ejecuten los requerimientos sin problemas, pero hay que tener el cuidado de disminuir dicho valor después del paso de instalación. Esto se debe a que la memoria Swap degradará

rápidamente la microSD, al tener dicha tarjeta límite de escritura y lectura de datos

Tener en cuenta el poder de procesamiento de la Raspberry Pi 3B. La primera iteración del proyecto consideró el uso de una cámara térmica tipo ABC de baja resolución para realizar el conteo. Esta cámara fue considerada para poder realizar el conteo por visión de manera anónima. Por la naturaleza de baja resolución, se precisaba pasar la data bruta de la cámara por un filtro bi-cúbico para duplicar de forma virtual la resolución, y luego usar un algoritmo para definir altos y bajos términos y quedarse con los altos, para finalmente realizar el conteo. Se tuvo que realizar una modificación con respecto a dicho sensor, ya que el proceso de filtrado tenía que ocurrir fotograma por fotograma, y demandaba un consumo de recursos mucho más alto del que era tolerado por la placa. Al ser demasiado lento el filtrado, ocurría que se empezaban a perder fotogramas. Esto impactó de forma muy negativa el proceso, imposibilitando su funcionamiento en tiempo real y estorbando el rastreo de las lecturas de calor, causando error fatal y la caída del programa, lo que obligó la modificación de los elementos a usarse en el proyecto.

Observaciones con respecto al uso del sistema. El funcionamiento del sensor OV2640 no es óptimo en condiciones de baja luz, lo que es normal. Por reducción al absurdo, el sistema no funcionará a poca luz o a carencia de luz. Se postula que para el correcto funcionamiento del sistema, el área debe de estar bien iluminada, y colocada de tal manera que se puedan ver los sujetos de forma completa. Esto se debe a que si se ve parte de una pesona, esto es, por ejemplo, parte de un brazo, el algoritmo no va a identificar el objeto hasta que vea detalles más determinantes.

Uso de Código Abierto. Tener en cuenta que tanto OpenCV como YOLOv8 son librerías grandes, con implementaciones variadas pero que no están optimizadas en consumo de recursos. Teniendo esto en cuenta, YOLOv8 funciona mejor con respecto a las necesidades de la implementación del presente proyecto, pero se recomienda, dentro de lo posible, que si se puede evitar que se realice cierto procesamiento en la placa y colocar esa carga en otra, como se hizo en el presente proyecto, con APIs y la placa ESP32 Cam, se haga dicho descargo. Así mismo, se recomienda se haga una lectura crítica del código, a fin de poder optimizar su ejecución. Al ser código abierto, se pueden revisar los constituyentes de las librerías, y probar obviando ciertos procesos que demandan recursos, a ver si con dichas variaciones la implementación sigue siendo válida y, en caso de que sea así, qué tanta es la mejora.

5.3 Trabajo a Futuro

Como trabajo a futuro se puede explorar la optimización de la librería OpenCV, específicamente en la parte de detección de personas, para poder obviar los elementos que consuman más recursos y pueda así usarse en plataformas de desarrollo ligeras.

Con respecto a la captura de imágenes, la presente implementación guarda 20 segundos, que empieza justo después de ingresar el PIN, para ser analizados. Se descarta ese archivo después. Se pretende que en un trabajo futuro se realice una implementación en donde se pueda recibir de forma indefinida el video, realizando un análisis de imagen en tiempo real, y así eliminar esa limitación del sistema.

De igual manera, se puede diseñar y ejecutar pruebas de desempeño más destiladas, para poder observar de forma inequívoca que impacta más al uso de recursos cuando se trata de visión por computadora. Las pruebas podrían enfocarse con la variación de memoria RAM disponible, así como el comportamiento del sistema contra varios procesadores. Esto, de la mano del desarrollo de código, podría dar como resultado implementaciones eficientes, con beneficios dentro del marco del Internet de las Cosas. Otra versión de esto puede ser una comparativa entre las placas más conocidas dentro del movimiento de Hardware y Software de Código Abierto, así como una comparativa entre el desempeño de las placas Raspberry Pi 3, 3B y 4. Se puede realizar pruebas de desempeño con respecto a cómo afecta la resolución del video de entrada a los algoritmos de visión por computadora, de forma directa, investigar acerca de resoluciones de 1MP, 2MP, 3MP, 4MP Y 5MP, que son las resoluciones comerciales más utilizadas.

Para mejorar el desempeño del sistema, se puede hacer la implementación con el cambio de la conexión entre la placa de desarrollo ESP32 Cam y los Módulos de Control y Vigilancia. Esto se da porque la velocidad en la que la cámara da la información a ser procesada se ve impactada por los pasos adicionales que se adosan a la conexión WiFi, esto es, la transmisión inalámbrica, codificación y decodificación para su posterior uso. Una conexión cableada se beneficiaría de una mayor velocidad de conexión y por ende un mejor rendimiento del sistema.

Con respecto al desarrollo del presente sistema, específicamente el acceso a la página web, queda pendiente habilitar el portal para que el personal pueda tener acceso desde el internet, ya que al presente se necesita estar conectado al intranet de la ESPOL. El

abrir este acceso permitiría la creación de órdenes de trabajo sin tener que estar dentro del predio de la universidad o prescindir de un VPN que conectase a la red. Un beneficio de este alcance sería el poder programar actividades de carácter urgente y de ocurrencia emergente.

BIBLIOGRAFÍA

[Agudelo, 2023] Agudelo, D. C. (2023). Tecnologías disruptivas para la seguridad electrónica en latinoamérica. https://www.ventasdeseguridad.com/2023081023523/articulos/analisis-tecnologico/tecnologias-disruptivas-para-la-seguridad-electronica-en-latinoamerica.html. (Accessed on 2024-01-19).

[Aluri, 2020] Aluri, D. C. (2020). Smart lock systems: An overview. *International Journal of Computer Applications*, 177:40–43.

[Arguello, 2023] Arguello, F. (2023). Tecnología iot en la industria de seguridad electrónica.

[Bastiaansen, 2022] Bastiaansen, R. (2022). What is a raspberry pi used for?: Techtarget.

[Blancarte, 2023] Blancarte, O. (2023). Que son los access token.

[Campo, 2023] Campo, J. (2023). Tecnologías en auge que marcarán la diferencia en 2023 para el mercado de la seguridad electrónica.

[Encord, 2023] Encord (2023). Yolov8 for object detection explained [practical example].

[Estrada, 2018a] Estrada, Villavicencio, L. (2018a). Diseño de un sistema basado en sensores para el control de acceso, temperatura e iluminación de laboratorios de la fiec.

[Estrada, 2018b] Estrada, Avalos, S. (2018b). Desarrollo de un sistema alternativo de acceso a las áreas restringidas de la fiec mediante el uso de dispositivos móviles.

[Fernandez, 2019] Fernandez, Y. (2019). Api: Qué es y para qué sirve.

[García-Madurga, 2021] García-Madurga, Grilló-Méndez, M.-N. (2021). La adaptación de las empresas a la realidad covid: una revisión sistemática.

[Gigi, 2021] Gigi (2021). What is an amg8833? features and demo.

[Gillis, 2023] Gillis, A. S. (2023). What is iot (internet of things) and how does it work?: Definition from techtarget.

[González, 2024] González, R. (2024). . por qué recurrir al iot para mejorar la seguridad en nuestra empresa y proteger nuestra salud.

[Hashemi,] Hashemi, C. What is docker.

[Ho et al., 2016] Ho, G., Leung, D., Mishra, P., Hosseini, A., Song, D. X., and Wagner, D. A. (2016). Smart locks: Lessons for securing commodity internet of things devices. *Proceedings of the 11th ACM on Asia Conference on Computer and Communications Security*.

[Kang, 2015] Kang, H.-S. (2015). An efficient and secure physical security method of data center. *Journal of Security Engineering*, 12:609–620.

[Kulhary, 2022] Kulhary, R. (2022). What is opencv: an overview.

[Lucena, 2023] Lucena, P. (2023). ¿qué es el framework?: 2024.

[Melanie, 2023] Melanie (2023). Database: What is it, and how does it work?

[MozDevNet,] MozDevNet. Introduction to the dom - web apis: Mdn.

[Padhyay, 2023] Padhyay, A. (2023). Axios in react: A guide for beginners.

[Pascual, 2022] Pascual, C. (2022). Esp32 cam introducción y primeros pasos.

[Saini, 2023] Saini, A. (2023). An easy introduction to flask framework for beginners. https://www.analyticsvidhya.com/blog/2021/10/flask-python/#:~:text=Flask%20is%20used%20for%20developing,Lightweight. (Accessed on 2024-01-19).

[Singhal, 2023] Singhal, P. (2023). Frontend versus backend.

[Sowjanya, 2016] Sowjanya, N. (2016). Design and implementation of door access control and security system based on iot.

[Team, 2024] Team, D. (2024). Algoritmo: Definición y usos - datascientest.

[Topic, 1993] Topic, T. (1993). Sistema de control de acceso de personal y de seguridad para el edificio de ing. eléctrica de la prosperina.

[Vásquez, 2023] Vásquez, García, M. (2023). Diseño e implementación de un sistema de control de acceso para dispositivos de seguridad basado en tecnología iot.

[Wang, 2012] Wang, Z. (2012). Research of rfid intelligent access control system in the internet of things.

APÉNDICES

A Estudio Económico

Este estudio presenta una proyección de ingresos y egresos, así como los valores que se precisan para adecuar una oficina con el menaje indispensable para su correcto funcionamiento. Determinación de inversores. El valor que se considera para solicitar una inversión está compuesto por los siguientes siete elementos:

1. Adquisición de stock para conformar el producto, Esto involucra la compra de las placas ESP32 Cam, Raspberry Pi 3B+, Botonera, y cables, tanto UPT como USB para energizar las placas. Estos valores suman un total de USD2,081.18.

2. Adquisición de computadora para realizar el trabajo, Este valor cubre el valor de la torre y periféricos, incluido monitor, por un valor de USD1,078.64.

3. Adquisición de silla ergonómica para programar, Valor que asciende a USD480.00.

4. Adquisición de mesa, La mesa de reuniones servirá de multipropósito, a un precio de USD2200.00.

5. Cubrir los valores de márquetin por tres meses, se elige el periodo de tiempo de 3 meses para que se defina una estrategia de mercadeo, proporcionada por una agencia externa, y se pueda proceder a su aplicación. Al tenor de 200 dólares por mes se termina precisando, por este rubro, un total de USD600.00.

6. Cubrir los valores de alquiler por tres meses; y, A un total de USD1350.00.

7. Cubrir los valores de personería jurídica.

8. La personería jurídica que se creará será una Sociedad de Acción Simplificada. Su proceso toma una semana, entre redacción de estatutos e inscripción, y los honorarios de un abogado por su creación ascienden a USD572.00. Estos siete elementos suman un total de USD8,361.82, para lo que un préstamo que se solicitase sería de USD10,000.00, pagadero a 24 meses con una tasa de interés no mayor al 20 por ciento anual, según lo solicitado por alguna entidad bancaria de la República del Ecuador.

Depreciación y amortización de activos. Al no tener préstamos, no se elabora una tabla de amortización. Con respecto a la depreciación, se considera que para todo bien electrónico se estipula una depreciación del 33 por ciento anual sobre el valor de compra reflejado en la factura, de acuerdo con la Ley de Régimen Tributario Interno en su artículo 10; y artículo 28 del Reglamento para la aplicación de la Ley de Régimen Tributario Interno de la República del Ecuador. Cada cuadro tiene el porcentaje anual al que se deprecia el bien.

Bienes operativos.

2 Computadoras para uso de oficina con las siguientes características a precio unitario de USD719.00:

Proc. Intel Core i7 10700 | 8 Núcleos + 16 Hilos | Memoria Ram 32GB | Unidad de Estado Sólido 500GB | Mainboard h510 | HDMI | VGA | 6xUSB | RJ45 | PCIe x16 | Teclado | Mouse | Parlantes

33%	Valor Inicial	Primer año	Segundo año	Tercer año
2 PCs	$1438.00	$963.46	$488.92	$14.38

2 monitores para PC, a precio unitario de USD359.64, con las siguientes características:

MONITOR DELL P2222H | 21.5Inch | IPS LED | DP | HDMI | VGA | DP | FULL-HD | 1080p | Negro

33%	Valor Inicial	Primer año	Segundo año	Tercer año
2 Monitores	$719.28	$481.92	$244.56	$7.19

1 subscripción office 365 por USD60 al año para usarse hasta en 5 dispositivos.

No es sujeto de depreciación al ser un software en la nube y servicio por subscripción.

1 Impresora para uso de oficina Epson L1250 a un valor de USD199.

Conexión Wifi | Impresora con tinta continua | Super alta capacidad y economía | Costo de impresión ultra bajo | Nuevo sistema fácil de llenar

33%	Valor Inicial	Primer año	Segundo año	Tercer año
1 Impresora	$199	$133.33	$67.66	$1.99

Escritorio/mesa de reuniones de 12 personas a USD1278, de cedro.

33%	Valor Inicial	Primer año	Segundo año	Tercer año
	$199	$133.33	$67.66	$1.99
	Cuarto año	**Quinto año**	**Sexto año**	**Séptimo año**
1 mesa	$766.80	$639.00	$511.20	$383.40
	Octavo año	**Noveno año**	**Décimo año**	
	$255.60	$127.80	$0.00	

4 metros de largo | 2 metros de ancho | Cedro

Silla ergonómica para programador a USD480.

Reclinable | Ergonómica | Altura ajustable

33%	Valor Inicial	Primer año	Segundo año	Tercer año
	$480.00	$432.00	$384.00	$336.00
	Cuarto año	**Quinto año**	**Sexto año**	**Séptimo año**
1 silla	$288.00	$240.00	$192.00	$144.00
	Octavo año	**Noveno año**	**Décimo año**	
	$96.00	$48.00	$0.00	

Sofá para oficina a USD2200.

Chesterfield | Cuero | Inglés

33%	Valor Inicial	Primer año	Segundo año	Tercer año
	$2200.00	$1980.00	$1760.00	$1540.00
	Cuarto año	**Quinto año**	**Sexto año**	**Séptimo año**
1 sofá Chesterfield	$1320.00	$1100.00	$880.00	$660.00
	Octavo año	**Noveno año**	**Décimo año**	
	$440.00	$220.00	$0.00	

Televisor 32” LG Smart TV modelo 32LQ630BPSA para el recibidor a USD245.

Marca LG | Modelo 32LQ630BPSA | Color negro | Voltaje 220V | Pantalla LED 32” | Resolución HD | HDR | Interfaz USB | HDMI | Wifi

33%	Valor Inicial	Primer año	Segundo año	Tercer año
1 TV 32”	$245.00	$164.15	$83.30	$2.45

Aire acondicionado LG Alta Eficiencia 18.000 BTU Ecológico a USD400.

Tecnología invertir | 18mil BTUs | Wifi | Tecnología inverter | 220V | Temporizador

33%	Valor Inicial	Primer año	Segundo año	Tercer año
1 TV 32”	$400.00	$268.00	$136.00	$4.00

Con estos elementos se calcula que la inversión en mobiliario operativo asciende a un gran total de USD5,850.64

Stock de partes y suministros.

Placa Raspberry Pi 3B+

	Valor Unit.	**1er Trim.**	**2do Trim.**	**3er Trim.**	**4to Trim.**
Trimestral (en dólares)	49.50	*30 unidades* 1485.00	*45 unidades* 2227.50	*75 unidades* 3712.50	*30 unidades* 1485.00

Placa ESP32 Cam

	Valor Unit.	**1er Trim.**	**2do Trim.**	**3er Trim.**	**4to Trim.**
Trimestral (en dólares)	9.99	*30 unidades* 299.90	*45 unidades* 449.85	*75 unidades* 749.75	*30 unidades* 299.90

Botonera 4x4

	Valor Unit.	**1er Trim.**	**2do Trim.**	**3er Trim.**	**4to Trim.**
Trimestral (en dólares)	1.38	*30 unidades* 41.28	*45 unidades* 61.92	*75 unidades* 103.20	*30 unidades* 41.28

Rollo cable UTP

	Valor Unit.	**1er Trim.**	**2do Trim.**	**3er Trim.**	**4to Trim.**
Trimestral (en dólares)	210.00	*30 unidades* 210.00	*45 unidades* 0.00	*75 unidades* 210.00	*30 unidades* 0.00

Conector USB de poder

Activos inmuebles.

El proyecto no contará con bienes inmuebles durante el primer año.

	Valor Unit.	1er Trim.	2do Trim.	3er Trim.	4to Trim.
Trimestral (en dólares)	1.50	*30 unidades* 45.00	*45 unidades* 67.50	*75 unidades* 112.50	*30 unidades* 45.00

Análisis de costos. En esta sección están referidos los valores totales considerados para el desarrollo del presente proyecto. Como detalle, se cuantifica el valor no sólo de los artículos físicos sino también el valor de hora de programación. Con estos costos se obtiene también un precio de venta propuesto para distribuidores.

Propiedad Intelectual. En la parte de hardware, no se utilizó software propietario alguno, favoreciendo el código abierto para mantener los costos bajos sin impactar de mala manera sobre el producto final. Es así como se eligió OpenCV como la librería principal para análisis de imágenes. Su uso está bajo la licencia de Apache 2, y es gratis para uso comercial, lo que siempre es de desear.

De igual manera, el desarrollo del frontend y backend fue realizado pura y exclusivamente por placer del programador, lo que permite obviar el pago de honorarios a terceros por su injerencia intelectual.

Es por lo expuesto que se enuncia que no habrán problemas legales de tipo alguno, ya que al no haber licencias por adquirir o licencias que se estén usando de forma mercenaria e ilegal, no hay retribución a temer.

Materiales y mano de obra. Se listan los materiales usados dentro del proyecto y el valor al que fueron adquiridos para el desarrollo del prototipo.

No se considera uso o adquisición de una computadora para la programación necesaria, puesto que es un objeto cuya existencia se da por sentada. De igual manera y por el mismo motivo no se considera ningún valor por inversión de infraestructura o servicios básicos.

Se lista el costo de mano de obra para el desarrollo del proyecto. Estos valores no están relacionados con la réplica.

Es imperativo mencionar que el valor por desarrollo de base de datos refiere a valores por hora, para un estimado total de 60 horas en promedio. Ya que, considerando que el objetivo del presente proyecto es vender en amplia forma; que lo que garantiza la venta de un producto es la calidad del mismo; que lo que estimula la decisión de compra es un valor que se perciba como apropiado o como una ganga para lo que se pretende adquirir;

	Valor inc. IVA* (USD)	Cantidad	Total (USD)
Raspberry Pi 3B	60.00	1	60.00
Tarjeta MicroSD 64Gb	4.80	1	4.40
Fuente de poder 12V	4.00	2	8.00
ESP32 Cam**	25.00	1	25.00
Teclado Arduino	2.00	2	4.00
Servomotor	2.00	1	2.00
Cableado***	8.00	1	8.00
Impresión 3D de caja	7.00	1	7.00
		Total	118.00

*** Donde corresponde.**

**** Viene con el adaptador USB para configuración y uso.**

***** Conglomerado: Cable UTP, 300 pies, cable de cobre AWG 10 simple, 5 metros.**

	Valor inc. IVA* (USD)	Cantidad	Total (USD)
Creación de base de datos	12.00 por hora	160	700.00
Creación página web	400.00	1	400.00
Programación de microcontroladores	400.00	1	
		Total	1500.00

*** Donde corresponde.**

que el retorno de la inversión se obtendrá por la venta rápida y grande del mismo; y, que el proyecto se lo pretende vender como objeto a instalar, no como proyecto a desarrollar, se beneficia mucho en tener una línea de costo baja.

De igual manera, el valor de creación de página web refiere al estándar de la industria para una página de utilidad simple.

La programación de microcontroladores también refiere al estándar de la industria, con un aumento del 27por ciento, basado en el hecho de que el encargado de dicha tarea cobraría un valor discrecional mayor al promedio.

Venta al público. Teniendo en cuenta que el precio para distribuidor de un sistema similar, sin instalación, es de 316 dólares, que a este valor se le debe de sumar el impuesto al valor agregado de 12 por ciento; y, que el promedio de ganancia para artículos electrónicos de seguridad es de 35 por ciento, el costo de un sistema de características

parecidas al que se ha desarrollado puede estar valorado en USD477,80.

Estando el prototipo listo, se da un precio de venta de USD312 dólares americanos. En la jerga, se tiene una base de datos, un controlador de puerta, un pestillo automático, un autenticador, un controlador de existencias, una cámara, y una interfaz para gestión de tareas.

Punto de equilibrio. Con los egresos mensuales ascendiendo al valor de USD3,394.00 y el valor unitario del sistema se encuentra en USD312.50, se debe de vender un total de 11 unidades (valor redondeado hacia arriba) para poder llegar al punto de equilibrio.

B Proformas

SISEGUSA
Sistemas de Seguridad S.A.

***PROD* Sistemas de Seguridad S.A. (SISEGUSA)**
GUAYAQUIL
Urdesa Central, Av. Las Lomas 403 entre Calle 4ta. y 5ta. PBX.:
(04)3712850 Telf.:(04)2389994 Cel.: 0939535051
CONTRIBUYENTE ESPECIAL No. 0048
v.gutierrez@sisegusa.om

COTIZACIÓN
N° 1100397488

INFORMACIÓN DEL CLIENTE:				FECHA EMISION	
RAZÓN SOCIAL:	VELIZ IBARRA JAIME JOSE			23/11/2023 0	12:01
DIRECCIÓN:	6 PJE. 3A NE SL. 18--GUAYAQUIL				
TELEFONOS:	.	RUC:	0941263873001	VENDEDOR	
E-MAIL:	Oteito @msn.com				
E-MAIL2:	Oteito @msn.com			Victor Gutierrez	

CANT	MODELO	DESCRIPCIÓN	P.UNIT.	PROMOCIÓN	TOTAL
1	ZK-MB560-VL/ID	LECTOR BIOMETRICO FACIAL HUELLA Y PROXIMIDAD	194,94	191,04	191,04
1	ZK-TPM005B	FUENTE PARA CONTROL DE ACCESO 5AMP+1AM CONTINUOS	25,39	24,88	24,88
1	PA-CAJAALARMABL	CAJA DE ALARMA METALICA BLANCA	13,42		13,42
1	ST-12V-4AMP	BATERIA RECARGABLE 12VDC 4AMP	7,96		7,96
1	ZK-LM-2802	CERRADURA ELECTROMAGNETICA 600LB ZK	16,74	16,41	16,41
2	ZK-LMB-280L	SOPORTE L PARA CERRADURA ELECTROMAGNETIC	5,06	4,96	9,92
1	ZK-TLEB102-R	BOTON DE SALIDA DON'T TOUCH CON CONTROL REMOTO	19,42	19,03	19,03

OBSERVACIÓN:	SUB.TOTAL:	282.66
	DCTO 0.00%:	
DISPONIBILIDAD:	SUBT.NETO:	282.66
VALIDEZ DE LA OFERTA: 30/11/2023	BASE 12%:	282.66
FORMA DE PAGO:	BASE 0%:	
	12% IVA:	33.92
	TOTAL:	316.58

Figura 1: Proforma de sistema de seguridad cuya función es similar a la parte de acceso del sistema desarrollado.

Ofertas similares las hay como ZKTeco, mostrada en la Figura 1, reconocida marca china que ha entrado en el mercado ecuatoriano a ofrecer productos análogos al desarrollado por un costo similar. Vale notar que para obtener mejores precios y elementos, habría que lograr llegar a una economía de escala que permita bajar los

PROYECCIÓN DE INGRESOS Y EGRESOS.

	ENERO	FEBRERO	MARZO	ABRIL	MAYO	JUNIO	JULIO	AGO	SEPT	OCT	NOV	DIC
Unidades	10	10	10	15	15	15	25	25	25	10	10	10
Valor unitario	312,5	312,5	312,5	312,5	312,5	312,5	312,5	312,5	312,5	312,5	312,5	312,5
Ingresos												
Caja	0	-269	-538	-807	-203,5	400	1003,5	3352	5700,5	8049	7780	7511
Ventas	3125	3125	3125	4687,5	4687,5	4687,5	7812,5	7812,5	7812,5	3125	3125	3125
T. Ingresos	3125	2856	2587	3880,5	4484	5087,5	8816	11164,5	13513	11174	10905	10636
Egresos												
Oficina												
Luz	25	25	25	25	25	25	25	25	25	25	25	25
Agua	18	18	18	18	18	18	18	18	18	18	18	18
Caja chica	300	300	300	300	300	300	300	300	300	300	300	300
Internet	21	21	21	21	21	21	21	21	21	21	21	21
Alquiler	450	450	450	450	450	450	450	450	450	450	450	450
T. Oficina	814	814	814	814	814	814	814	814	814	814	814	814
Recursos humanos												
Programador 1	500	500	500	500	500	500	500	500	500	500	500	500
Programador 2	500	500	500	500	500	500	500	500	500	500	500	500
T. RRHH	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000
Costo de producto												
Producto*	1380	1380	1380	2070	2070	2070	3450	3450	3450	1380	1380	1380
Marketing	200	200	200	200	200	200	200	200	200	200	200	200
T. Ct. Prod.	1580	1580	1580	2270	2270	2270	3650	3650	3650	1580	1580	1580
T. Egresos	3394	3394	3394	4084	4084	4084	5464	5464	5464	3394	3394	3394
Ingreso	$ 269,00	$ 538,00	$ 807,00	$ 203,50	$ 400,00	$ 1.003,50	$ 3.352,00	$ 5.700,50	$ 8.049,00	$ 7.780,00	$ 7.511,00	$ 7.242,00

Figura 2: Proyección de Ingresos y Egresos de un ejercicio comercial ficticio, con valles comerciales a inicio de año y cimas en los últimos meses del año.

precios aún más y elevar la calidad. Con dicho valor de venta, se necesitaría vender 39 unidades para poder obtener el retorno de lo invertido.

C Captura de imagen mediante OpenCV

```
import cv2
import datetime

ip_addr = '192.168.1.103'
stream_url = 'http://' + ip_addr + ':81/stream'
videocapture=cv2.VideoCapture(stream_url)

def open_video():
    ip_addr = '192.168.1.103'
```

```
    stream_url = 'http://' + ip_addr + ':81/stream'
    videocapture=cv2.VideoCapture(stream_url)

    endTime = datetime.datetime.now() + datetime.timedelta(seconds=15)
    video_frame_count=0

    while videocapture.isOpened() and datetime.datetime.now() <= endTime:
        success, frame = videocapture.read()
        if not success:
            break
        video_frame_count+=1
        if video_frame_count == 1:
            cv2.namedWindow("Ultralytics YOLOv8 Region Counter Movable")
        cv2.imshow("Ultralytics YOLOv8 Region Counter Movable", frame)
        if cv2.waitKey(1) & 0xFF == ord("q"):
            break
    videocapture.release()
    cv2.destroyAllWindows()
    print("end of script")
    return video_frame_count
```

D Definición de librerías para análisis de imagen de YOLOv8

```
from ultralytics import YOLO
from ultralytics.solutions import object_counter
import cv2
import datetime

model = YOLO("yolov8n.pt")
in_counts = 0
out_counts = 0
```

```
ip_addr = '192.168.1.102'
stream_url = 'http://' + ip_addr + ':81/stream'
cap=cv2.VideoCapture(stream_url)

assert cap.isOpened(), "Error reading video file"
w, h, fps = (int(cap.get(x)) for x in (cv2.CAP_PROP_FRAME_WIDTH, cv2.CAP_PROP_FRAME_HEI
# Define region points
# region_points = [(200, 100), (420, 100), (420, 280), (200, 280)]
region_points = [(300, 100), (300, 400)]

'''
# Video writer
video_writer = cv2.VideoWriter("/Users/saulmestanza/Documents/workspace/doraemon/object_
                        cv2.VideoWriter_fourcc(*'mp4v'),
                        fps,
                        (w, h))
'''

# Init Object Counter
counter = object_counter.ObjectCounter()
counter.set_args(
    view_img=True,
    reg_pts=region_points,
    classes_names={0: 'person'},
    draw_tracks=True,
)
endTime = datetime.datetime.now() + datetime.timedelta(minutes=1)

print("INICIO!")

while cap.isOpened() and datetime.datetime.now() <= endTime:
    success, im0 = cap.read()
```

```
    if not success:
        print("Video frame is empty or video processing has been successfully complet
        break
    tracks = model.track(im0, persist=True, show=True, verbose=False, classes=[0])

    try:
        # im0 = counter.start_counting(im0, tracks)
        counter.start_counting(im0, tracks)
        in_counts = counter.in_counts
        out_counts = counter.out_counts
        print("counter.in_counts: ", in_counts)
        print("counter.out_counts: ", out_counts)
    except Exception as e:
        print("************", e)
    # video_writer.write(im0)

print("TERMINA!")
print(in_counts, out_counts)
cap.release()
# video_writer.release()
cv2.destroyAllWindows()
```

E Algoritmo Contador de personas

```
from collections import defaultdict
from shapely.geometry import Polygon
from ultralytics import YOLO
import cv2
import datetime
from ultralytics.utils.plotting import Annotator, colors
import numpy as np
from shapely.geometry.point import Point
from ultralytics.utils.files import increment_path
```

```
from pathlib import Path

track_history = defaultdict(list)

current_region = None

#Region en la que realizara analisis
counting_regions = [
    {
        "name": "YOLOv8 Rectangle Region",
        "polygon": Polygon([(10, 10), (500, 10), (500, 550), (10, 550)]),  # Polygon poi
        "counts": 0,
        "dragging": False,
        "region_color": (37, 255, 225),  # BGR Value
        "text_color": (0, 0, 0),  # Region Text Color
    },
]

def run(
    weights="yolov8n.pt",
    source=None,
    device="cpu",
    view_img=True,
    save_img=False,
    exist_ok=False,
    classes=[0],
    line_thickness=2,
    track_thickness=2,
    region_thickness=2,
):
    vid_frame_count = 0
    final_count=0
```

```
#Yolo model
model = YOLO(f"{weights}")
model.to("cuda") if device == "0" else model.to("cpu")

# Extract classes names
names = model.model.names

# Video setup
ip_addr = '192.168.1.100'
stream_url = 'http://' + ip_addr + ':81/stream'
videocapture=cv2.VideoCapture(stream_url)

endTime = datetime.datetime.now() + datetime.timedelta(seconds=15)

# Iterate over video frames
while videocapture.isOpened() and datetime.datetime.now() <= endTime:
    success, frame = videocapture.read()
    if not success:
        break
    vid_frame_count += 1

    #Extract the results from the deteccion apply to the frame
    results = model.track(frame, persist=True, classes=classes)

    #Deconstruction of the results from the analysis
    if results[0].boxes.id is not None:
        boxes = results[0].boxes.xyxy.cpu()
        track_ids = results[0].boxes.id.int().cpu().tolist()
        clss = results[0].boxes.cls.cpu().tolist()

        annotator = Annotator(frame, line_width=line_thickness, example=str(names
```

```
    for box, track_id, cls in zip(boxes, track_ids, clss):
        annotator.box_label(box, str(names[cls]), color=colors(cls, True))
        bbox_center = (box[0] + box[2]) / 2, (box[1] + box[3]) / 2  # Bbox cente

        track = track_history[track_id]  # Tracking Lines plot
        track.append((float(bbox_center[0]), float(bbox_center[1])))
        if len(track) > 30:
            track.pop(0)
        points = np.hstack(track).astype(np.int32).reshape((-1, 1, 2))
        cv2.polylines(frame, [points], isClosed=False, color=colors(cls, True),

        # Check if detection inside region
        for region in counting_regions:
            if region["polygon"].contains(Point((bbox_center[0], bbox_center[1])
                region["counts"] += 1
    if region['counts']>final_count:
        final_count=region['counts']

# Draw regions (Polygons/Rectangles)
""" for region in counting_regions:
    region_label = str(region["counts"])
    region_color = region["region_color"]
    region_text_color = region["text_color"]

    polygon_coords = np.array(region["polygon"].exterior.coords, dtype=np.int32)
    centroid_x, centroid_y = int(region["polygon"].centroid.x), int(region["poly

    text_size, _ = cv2.getTextSize(
        region_label, cv2.FONT_HERSHEY_SIMPLEX, fontScale=0.7, thickness=line_th
    )
    text_x = centroid_x - text_size[0] // 2
```

```
            text_y = centroid_y + text_size[1] // 2
            cv2.rectangle(
                frame,
                (text_x - 5, text_y - text_size[1] - 5),
                (text_x + text_size[0] + 5, text_y + 5),
                region_color,
                -1,
            )
            cv2.putText(
                frame, region_label, (text_x, text_y), cv2.FONT_HERSHEY_SIMPLEX, 0.7,
            )
            cv2.polylines(frame, [polygon_coords], isClosed=True, color=region_color,

        if view_img:
            if vid_frame_count == 1:
                cv2.namedWindow("Ultralytics YOLOv8 Region Counter Movable")
            cv2.imshow("Ultralytics YOLOv8 Region Counter Movable", frame)

        #if save_img:
        #    video_writer.write(frame)

        for region in counting_regions:  # Reinitialize count for each region
            region["counts"] = 0

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

del vid_frame_count
#video_writer.release()
videocapture.release()
cv2.destroyAllWindows()
print("end of count")
```

```
    return final_count


def main():
    """Main function."""
    run()


if __name__ == "__main__":
    main()
```

F Configuración de seguridad del Backend

```
    INSTALLED_APPS = [
        'django.contrib.admin',
        'django.contrib.auth',
        'django.contrib.contenttypes',
        'django.contrib.sessions',
        'django.contrib.messages',
        'django.contrib.staticfiles',
        'rest_framework',
        'corsheaders',
        'djoser',
        'rest_framework_simplejwt',
        'accounts',
        'work_orders',
        'django_filters',
]

REST_FRAMEWORK = {
    # Use Django's standard `django.contrib.auth` permissions,
    # or allow read-only access for unauthenticated users.
```

```
    'DEFAULT_PERMISSION_CLASSES': [
        'rest_framework.permissions.IsAuthenticatedOrReadOnly',
    ],
    'DEFAULT_AUTHENTICATION_CLASSES': (
        'rest_framework_simplejwt.authentication.JWTAuthentication',
        #'rest_framework.authentication.SessionAuthentication',
    ),
    'DEFAULT_FILTER_BACKENDS': ['django_filters.rest_framework.DjangoFilterBackend'],
}

SIMPLE_JWT = {'AUTH_HEADER_TYPES': ('JWT',),
   "ACCESS_TOKEN_LIFETIME": timedelta(minutes=30),
    "REFRESH_TOKEN_LIFETIME": timedelta(hours=1),
    "ROTATE_REFRESH_TOKENS": True,
    "BLACKLIST_AFTER_ROTATION": True,
    'ALGORITHM': 'HS256',
}
```

FIN