ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA

DESARROLLO DE UN PROTOTIPO DE SISTEMA DE PREVENCIÓN DE INTRUSOS EN RESIDENCIAS, BASADO EN UNA PLATAFORMA RASPBERRY PI MEDIANTE PROCESAMIENTO DE IMÁGENES

TRABAJO DE TITULACIÓN PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERA EN ELECTRÓNICA Y TELECOMUNICACIONES

VANESSA ELIZABETH ESPINOZA AGUILAR

DIRECTOR: Ing. RAMIRO EDUARDO MOREJON TOBAR, MSc.

Quito, Mayo 2022

AVAL

Certifico que el presente trabajo fue desarrollado por Vanessa Elizabeth Espinoza Aguilar, bajo mi supervisión.

Ing. Ramiro Morejón, MSc.
DIRECTOR DEL TRABAJO DE TITULACIÓN

DECLARACIÓN DE AUTORÍA

Yo Vanessa Elizabeth Espinoza Aguilar, declaro bajo juramento que el trabajo aquí descrito es de mi autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración dejo constancia de que la Escuela Politécnica Nacional podrá hacer uso del presente trabajo según los términos estipulados en la Ley, Reglamentos y Normas vigentes.

Vanessa Elizabeth Espinoza Aguilar

DEDICATORIA

A mi mami y mis hermanos por su amor, confianza y apoyo incondicional, por ser la definición perfecta de que unidos en familia y con la bendición de Diosito siempre existe la esperanza y llegada de mejores días. Y a mí, porque contra toda lucha que solo Diosito y yo conocemos, pude lograr terminar el sueño de mi vida y saber que los anhelos del corazón de la mano con el esfuerzo, empeño y bendición de él, se pueden cumplir.

Vane.

AGRADECIMIENTO

Agradezco a Dios por haberme sostenido, ser fortaleza y refugio durante todo este proceso, poniendo en mi camino a mis ángeles terrenales.

A mi mamita Vilma que jamás me ha dejado de apoyar, que ha sido mi cable a tierra cuando en los momentos más difíciles de mi existencia necesitaba una voz que me devuelva a la vida.

A mis hermanos Davicito y Ericksito por ser quienes le dan alegría a mi existencia, por ser ese amor incondicional desde que estaban en el vientre de mi mami y en varias ocasiones ser mis maestros de vida.

A mi mejor amiga Maritcita que es mi persona, mi incondicional y mi bendición en los momentos felices y también los más difíciles, siendo luz en mi vida y la de mi familia.

A mi director de tesis, Ing. Ramiro Morejón por extenderme la mano, brindarme su apoyo, consejos, y por la confianza brindada desde el día uno en la realización este proyecto.

A mis amigos y familiares más cercanos por su cariño, preocupación y ayuda más aún en los momentos que más necesité.

ÍNDICE DE CONTENIDO

RESUMEN

El presente proyecto tiene como objetivo el diseño y construcción de un prototipo que contribuya en la prevención y detección de intrusos dentro de una residencia, basado en plataformas de hardware y software open source.

El sistema está compuesto por un nodo cuyo hardware está conformado por una tarjeta Raspberry y una cámara. Si bien la cámara enfoca el fondo total del escenario a monitorear, el sistema ha sido diseñado para establecer un área de interés específica, en la cual se lleva a cabo el proceso de detección de un intruso. Este proceso se realiza mediante la aplicación del lenguaje de programación Python y el uso de la librería de visión artificial de código abierto OpenCV.

El momento en que el nodo detecte la presencia de un intruso, procede a alertar al usuario vía email sobre el evento en curso. Lo cual conlleva a la ejecución del proceso de almacenamiento local de la imagen capturada, así como el registro del detalle del evento en el servidor en la nube con la respectiva evidencia visual.

Mediante el uso de una aplicación web el usuario puede tener acceso a la información alojada en el servidor, y tomar las acciones pertinentes que garanticen su integridad personal y posteriormente dar aviso a las unidades de socorro.

Finalmente, el prototipo implementado se somete a las respectivas pruebas de funcionamiento en los escenarios que considerados como principales zonas de acceso de una residencia.

PALABRAS CLAVE: Raspberry, open source, Python, OpenCV, evidencia visual, email.

ABSTRACT

The present project has as objective to design and build a prototype that contributes to the prevention and detection of intruders within a residence, based on open source hardware and software platforms.

The system it is composed of a node whose hardware is conformed of a Raspberry card and a camera. Although the camera focuses on the entire background of the scene to be monitored, the system has been designed to establish a specific area of interest, in which the intruder detection process is carried out. This process is carried out by applying the Python programming language and using the open source computer vision library OpenCV.

The moment the node detects the presence of an intruder, it proceeds to alert the user via email about the event in progress. Which leads to the execution of the local storage process of the captured image, as well as the recording of the event details in the cloud server with the respective visual evidence.

Through the use of a web application, the user can have access to the information hosted on the server, and take the pertinent actions that guarantee their personal integrity and subsequently notify the relief units.

Finally, the implemented prototype is subjected to the respective performance tests in the scenarios that are considered to be the main access areas of a residence.

KEYWORDS: Raspberry, open source, Python, OpenCV, visual evidence, email.

1. INTRODUCCIÓN

Los indicadores de seguridad ciudadana emitidos por el Ministerio del Interior registran que la incidencia a escala nacional referentes a robos a viviendas en el Ecuador en el promedio anual ha disminuido [1], sin embargo, continúa siendo un riesgo que se da día con día en diversas modalidades.

Actualmente la Institución Articulada encargada de la seguridad ciudadana es la Policía Nacional del Ecuador en coordinación con el Servicio Integrado de Seguridad ECU 911, dotado del sistema de video vigilancia [2], pero a pesar de sus esfuerzos aún es un problema que no se logra erradicar. Debido a que este servicio de video vigilancia depende únicamente del personal operativo, el cual está encargado de monitorear todas las actividades que puedan generar situaciones de riesgo a nivel nacional y a la mínima inserción de mecanismos tecnológicos de alerta temprana, razón por la cual, los índices de incidencia continúan siendo alarmantes en varios sectores del país [1], dificultando las labores de combate y neutralización.

Por medio del avance tecnológico hoy en día es posible la utilización de técnicas de sistemas de alarma con procesamiento de imágenes, lo cual permite brindar nuevos mecanismos de alerta temprana en caso de detección de intrusos en una residencia; notificando directamente al usuario, contribuyendo así con la preservación de la integridad de los ciudadanos y brindando la posibilidad de ejecutar una intervención oportuna por parte del personal encargado de la seguridad ciudadana.

Por todo lo antes descrito se puede evidenciar que un problema de la vigilancia y monitoreo está relacionado con los sistemas de video vigilancia comerciales ya que en su mayoría proponen soluciones basadas en la supervisión y control, tomando el sistema de alerta como algo opcional que podría resultar en una implementación adicional que generará costos operativos. Para este caso en particular la implementación del prototipo en una residencia generará un sistema de monitoreo, detección y prevención de intrusos generando alertas al usuario final. Utilizando procesamiento de imágenes para reconocimiento de objetos extraños dentro del área de cobertura definida.

1.1 OBJETIVOS

El objetivo general de este Proyecto Técnico es desarrollar un sistema de seguridad residencial enfocado en la prevención de intrusos, utilizando una Raspberry Pi, con el uso de barreras virtuales y un procedimiento de alerta al usuario final

Los objetivos específicos del Proyecto Técnico son:

- Estudiar procesamiento de imágenes y las características de las tarjetas Raspberry Pi [3] [4] para la aplicación de algoritmos de procesamiento.
- Estudiar procesamiento de imágenes y las características de las tarjetas Raspberry Pi [3] [4] para la aplicación de algoritmos de procesamiento.
- Diseñar el prototipo basado en plataformas de hardware y software open source y la aplicación web donde se podrá visualizar las imágenes correspondientes a los eventos.
- Implementar el prototipo que permita la configuración de las barreras virtuales, así como también, la comparación de las imágenes capturadas.
- Generar pruebas del sistema de alarma mediante el envío de correos electrónicos y el almacenamiento de las imágenes captadas durante el evento en el servidor.
- Analizar resultados obtenidos del prototipo final.

1.2 ALCANCE

El presente estudio técnico presentará un prototipo de sistema de prevención de intrusos utilizando procesamiento de imágenes, el sistema estará enfocado al uso de una cámara de video conectada a la Raspberry Pi.

La detección de intrusos se realizará mediante algoritmos de comparación de las imágenes utilizando la librería de visión artificial de código abierto OpenCV, permitiendo establecer la presencia de cambios que ciertamente modifican la imagen previa, permitiendo concluir que es un elemento ajeno al entorno monitoreado. La Figura 1.1 detalla los componentes de hardware del sistema (nodo) y la Figura 1.2 detalla los componentes de software del sistema.

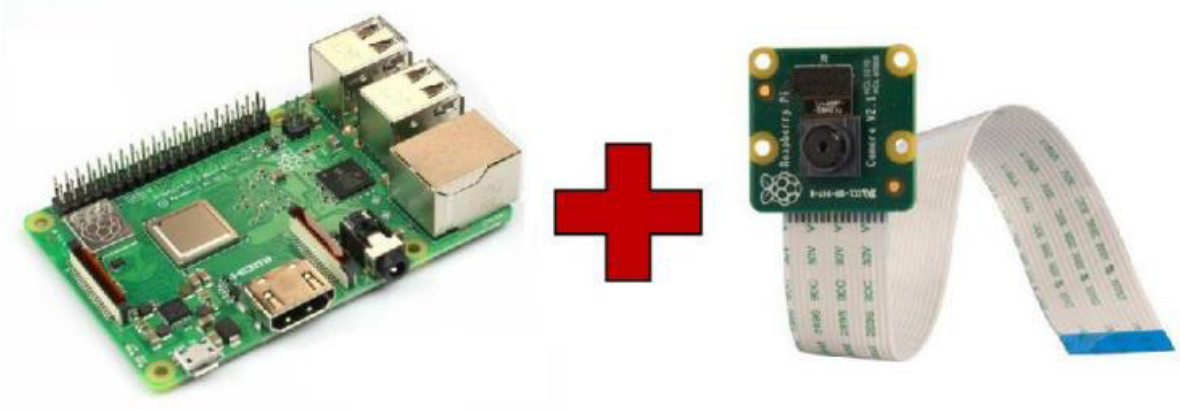

Figura 1.1 Componentes de Hardware del Nodo

Para la detección de intrusos en la imagen se prevé la creación de barreras virtuales, entendiéndose como una barrera virtual la posibilidad de definir áreas de interés en la imagen capturada. Estas áreas serán seleccionadas por el usuario en función del contenido de la imagen y la utilidad del usuario.

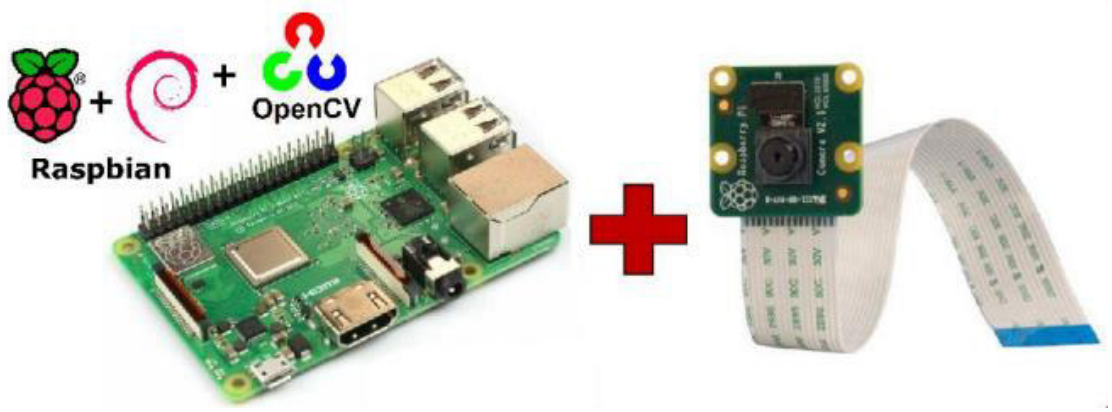

Figura 1.2 Componentes de software del nodo

La generación de una alarma desencadenará dos procesos, el primero consiste en el envío de un mensaje de correo electrónico al usuario registrado, con la notificación de que se detectó una intrusión en la zona monitoreada, el segundo proceso consiste en el almacenamiento del conjunto de imágenes registradas en el servidor. La Figura 1.3 describe los procesos del mecanismo de alerta. El administrador podrá acceder al servidor para consultar los eventos de alarma en virtud de la estampa de fecha y hora que se incluirá [5] en el sistema.

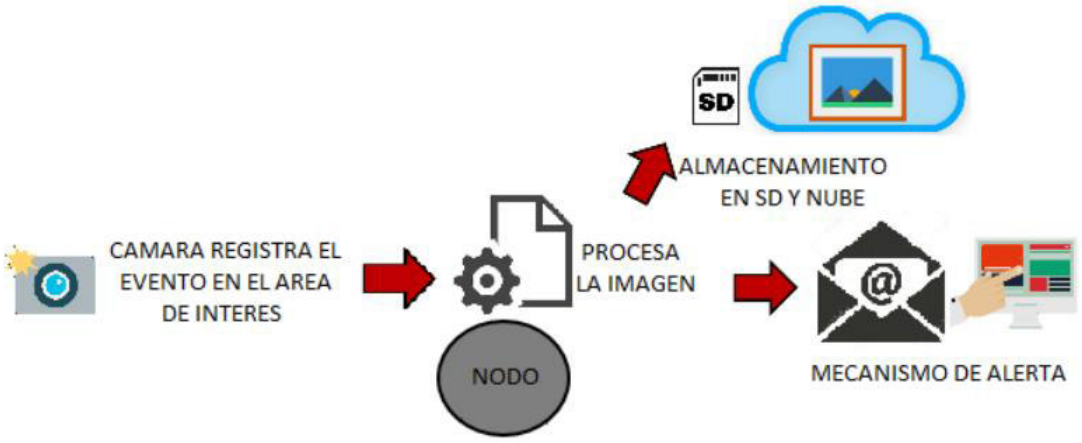

Figura 1.3 Mecanismo de alerta del sistema

Como mesa de prueba para este prototipo se escoge una residencia que cuente con parqueadero ya que se quiere monitorear el acceso al mismo, verificando si existe intrusiones durante el ingreso de un vehículo, además de otros ambientes de la residencia como lo es la entrada principal y trasera. Este proyecto tiene un producto demostrable que consiste en un prototipo de la cámara conectada a una plataforma Raspberry Pi y el servicio web.

1.3 MARCO TEÓRICO

En el presente trabajo de titulación se empleará la tecnología Raspberry Pi, en conjunto con OpenCV, la cual es una librería de código abierto de visión computarizada y machine learning [5], para el desarrollo de un sistema de seguridad residencial. El sistema de seguridad realizará la detección de amenazas y la alerta mediante correo electrónico al usuario.

1.3.1 SEGURIDAD CIUDADANA

En La Comisión Internacional de los Derechos Humanos [6] define a la seguridad ciudadana como las circunstancias propicias para que las personas gocen de una vida libre de las amenazas ocasionadas por la violencia y otros actos delictivos, frente a los cuales el Estado tiene la obligación de proteger los derechos humanos que pudiesen ser vulnerados ante estas coyunturas.

La seguridad ciudadana de un país puede ser cuantificada según la incidencia de diferentes eventos delictivos, tales como: homicidios intencionales, femicidios, maltratos físicos y psicológicos, delitos contra la libertad personal, como secuestros o trata de personas, delitos contra la propiedad, que abarcan robos a personas, domicilios, unidades

económicas, automóviles y bienes, delitos contra la integridad sexual y reproductiva, como abusos sexuales, violaciones y violencia de género, además de la delincuencia organizada y la corrupción

En el Ecuador se estableció una política pública integral denominada Plan Nacional de Seguridad Ciudadana y Convivencia Social Pacífica 2019-2030 con el objetivo de abordar la violencia estructural y cultural, además de las actuales amenazas a la seguridad ciudadana para fortalecer una convivencia pacífica y segura [6].

1.3.2 SITUACIÓN ACTUAL DE LOS MECANISMOS DE SEGURIDAD CIUDADANA EN EL ECUADOR

En el 2011 se puso en marcha el proyecto *"Sistema Nacional de Comando y Control para la seguridad ciudadana C4I2"*, cuyo enfoque se centra en disponer de un centro de crisis que cuente con una sala de comando y control que permita que las instituciones de atención de emergencias se interconecten mediante una plataforma tecnológica [2].

El Servicio Integrado de Seguridad ECU 911 está encargado de la atención de situaciones de emergencia de los ciudadanos, las cuales se reportan a través de la línea única de emergencias 911 y aquellas que se generen a través del sistema de video vigilancia y monitoreo de alarmas [2]. Actualmente el *"SIS ECU 911"* tiene dominio a nivel nacional y está conformado por: 7 Centros Zonales, 9 Centros Locales y 14 Salas Operativas.

1.3.3 INTRODUCCIÓN A LOS SISTEMAS DE VIDEO VIGILANCIA ACTUALES

Los sistemas de video vigilancia son sistemas de detección de amenazas en un área, ya sea dentro de un edificio, un conjunto residencial, o el completo espacio libre urbano o rural. [7]. Conformado por un conjunto de cámaras de vídeo con conexión a una grabadora digital y a un monitor central en el cual se tiene visibilidad de la imagen en tiempo real, que proviene de cada una de las cámaras.

Con el desarrollo tecnológico y económico se ha vuelto imprescindible el mejoramiento de los sistemas de videovigilancia para brindar seguridad tanto a la integridad de las personas como a sus bienes materiales, es por esto que en la actualidad es común contar con este tipo de sistemas en espacios públicos y privados que requieren un control eficiente.

Un sistema de video vigilancia permite controlar áreas estratégicas para evitar o controlar actos de vandalismo. Los estudios de seguridad realizados por la universidad de Florida en 2005 demuestran que el 47% de las pérdidas por robo en cualquier comercio provienen

de los empleados del propio local [8]. Al poseer una grabación constante los incidentes pueden revisarse y de esta manera tomar acciones frente a la situación.

1.3.3.1 CCTV

El circuito cerrado de televisión, por sus siglas CCTV, conocido comúnmente como video vigilancia [7], es un sistema que realiza la trasmisión de video a un número limitado de monitores, a diferencia de la televisión regular, mediante la instalación de cámaras en puntos estratégicos de un área. Este tipo de sistema es usado comúnmente para la detección de actividades criminales y su grabación. En la Figura 1.4 se puede observar los elementos principales de un CCTV.

El circuito cerrado de televisión puede ser de dos tipos: análogo o digital. En el sistema análogo se transmite el video continuo desde las cámaras a través de un cable coaxial de 75 Ohm hasta el DVR (Digital video recorder, Grabadora de video digital), este último es un dispositivo central pues también está conectado al monitor para la visualización.

Figura 1.4 Elementos de un sistema CCTV

Basado en los requerimientos de un sistema de CCTV y la necesidad de una conexión directa con una grabadora de video digital, por sus siglas en ingles DVR, el sistema puede resultar costoso de implementar y difícil de expandir [9]. Otra limitación que existe es la capacidad que tienen las DVR para procesar un número limitado de cámaras.

1.3.3.2 Cámaras IP

Las cámaras IP que funcionan en el Video Surveillance System, por sus siglas VSS, se encuentran en gran avance hacia el reemplazo de las cámaras análogas en el área de seguridad [9]. En la Figura 1.5 se puede observar un ejemplo de cámaras IP con antena WiFi.

Figura 1.5 Cámara IP con atena WiFi

Los sistemas de video IP son grabaciones que se realizan mediante una red, ya sea LAN (Local Area Network, Red de Área Local) o WAN (Wide Area Network, Red de Área Amplia). Lo que faculta la monitorización remota en tiempo real y el control y gestión de múltiples cámaras al mismo instante mediante una conexión de red. Por lo que la tecnología IP es más versátil y cómoda que el CCTV clásico, gracias a las siguientes ventajas [10] [11]:

- Accesibilidad remota: todos los usuarios autorizados pueden visualizar los videos en tiempo real y desde cualquier red en el mundo. Algo que en un sistema CCTV análogo también se logra mediante la instalación de servidores de video o DVR con conexión a la red.
- Mejor calidad de imagen: las cámaras de red con tecnología megapíxel tienen una calidad y resolución de imagen superior a las cámaras analógicas, además que estas últimas tienden a degradar las imágenes en el proceso de conversión de análogo a digital.

- Gestión de video inteligente: descarta las grabaciones de menor relevancia y permite respuestas programadas, mediante la gestión y notificación de alarmas cuando se detecta movimiento o audio.
- Estandarización e infraestructura de red: se integran con facilidad a sistemas de información de ordenadores y utiliza tecnología PoE (Power over Ethernet, Alimentación a través de Ethernet), por lo que es posible aprovechar la infraestructura existente.
- Flexibilidad y escalabilidad: los sistemas de video de red le facultan al usuario incorporar o modificar los equipos según se requiera, sin problemas de saturación, además las aplicaciones son prácticamente ilimitadas fuera y dentro del ámbito de la vigilancia y seguridad.
- Rentabilidad: el coste total es inferior al sistema tradicional CCTV análogo, con un ahorro significativo en la instalación e implementación de equipos para mejorar la fiabilidad del sistema.

Los sistemas VSS modernos tienen la capacidad de integrarse con otros sistemas como por ejemplo los sistemas de detección de fuego. Esta interoperabilidad crea un diseño de seguridad que puede reducir el error humano y los requerimientos de personal de seguridad.

1.3.4 FUNDAMENTOS SOBRE EL PROCESAMIENTO DE IMÁGENES

En la actualidad la tecnología digital permite la manipulación multi-dimensional de señales enfocado en sistemas que varían desde simples circuitos digitales hasta computadoras de avanzado procesamiento. La meta de esta manipulación se puede dividir en tres categorías [12]:

- Procesamiento de imágenes
- Análisis de imagen
- Entendimiento de imagen

Una imagen digital vista desde un espacio discreto compuesto de dos dimensiones, se derivada de una imagen analógica en un espacio continuo a través de un proceso de *sampling* o referida frecuentemente como digitalización. Los efectos de la digitalización se pueden observar en la Figura 1.6.

La imagen mostrada en la Figura 1.6 ha sido dividida en 16 filas y 16 columnas. A cada píxel se le asigna un valor equivalente al brillo promedio aproximado al valor entero más cercano [12]. Existen valores estándar para los parámetros encontrados en una imagen digital. Estos valores pueden ser causados por estándares de video, por requerimientos algorítmicos o por el deseo de mantener el circuito digital simple.

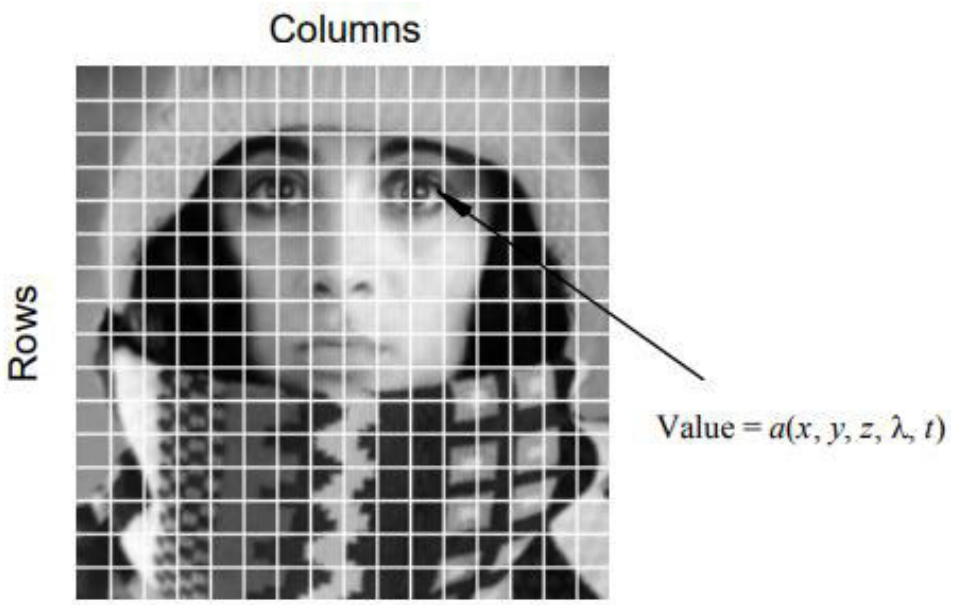

Figura 1.6 Digitalización de una imagen análoga

Una vez obtenida la representación digital de una imagen el procesamiento mediante programas de computación se convierte en un proceso mucho más sencillo. En el presente trabajo de titulación se utiliza la librería de procesamiento de imágenes es OpenCV.

1.3.5 TEORIA BASICA DE IMPLEMENTACION DE SERVIDORES DE DATOS Y SERVIDORES WEB

Un servidor es una computadora o sistema que provee de recursos, datos, servicios o programas a otros dispositivos, conocidos como clientes. En teoría cuando una computadora comparte dichos recursos con un cliente se convierte en un servidor. Un sistema individual puede proveer recursos.

1.3.6 REQUERIMIENTOS DE HARDWARE

1.3.6.1 Tarjetas Raspberry PI

Fue desarrollada por la Fundación Raspberry (Reino Unido), es una plataforma de hardware pequeña, potente y de bajo costo llamada Computadora de placa única que puede realizar tareas en una sola computadora.

Con una variedad de puertos de entrada y salida los cuales permiten la interacción con el mundo exterior, contribuye a la innovación de desarrollo de varios proyectos electrónicos.

Se requiere un sistema operativo sustentado en una distribución de Linux para que funcione la placa Raspberry Pi. Se almacena en la memoria de la MicroSD, esto significa que puede utilizar diferentes sistemas operativos simplemente cambiando la memoria. [13]

Las partes que componen las tarjetas Raspberry Pi generalmente son las siguientes, mismas que se pueden observar en la Figura 1.7:

- Tarjeta de Red WiFi y Bluetooth
- Conector de Video
- Puerto USB
- Puerto Ethernet
- Puerto para Cámara
- Salida de Audio
- Puerto HDMI
- Entrada Alimentación USB
- Puerto Display
- Procesador

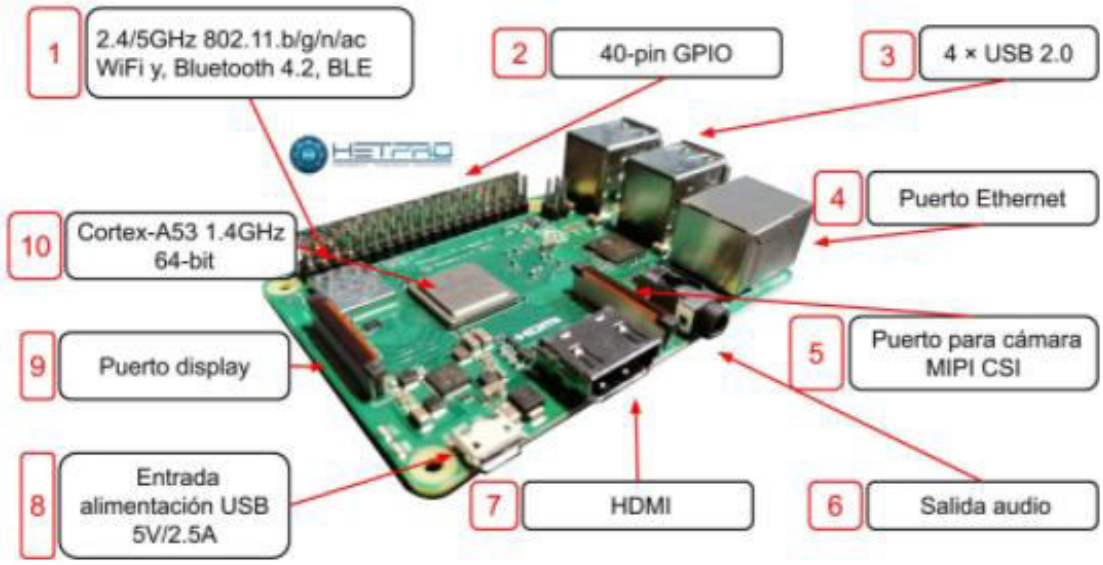

Figura 1.7 Elementos de Tarjeta Raspberry PI

La tarjeta está equipada con chips Broadcom BCM2835 / 2836/2837, según el modelo. El chip tiene un procesador ARM con varias frecuencias de operación siendo la frecuencia la frecuencia más alta de 1.2 GHz.

Mediante el pin de GPIO se logra establecer la comunicación entre la tarjeta Raspberry y el mundo exterior. Los modelos de placa A y B tienen 26 pines, mientras los modelos A, B, 2B, Zero, Zero v1.3, 3B tienen 40 pines.

1.3.6.2 Raspberry PI Modelo A y A+

Esta es la primera versión de una placa llamada Simple Board Computer, conformada por el chip Broadcom BCM2835 con un procesador ARM 700MHz, tiene 256 [MB] de RAM, ranuras de memoria SD en la cual se encuentra el sistema operativo. No cuenta con interfaz Ethernet, y posee únicamente un puerto USB y 26 pines GPIO de uso general.

1.3.6.3 Raspberry PI 3 Modelo A+/B+

La tarjeta Raspberry Pi 3 Model B+ al ser el último producto de la gama Raspberry Pi 3, dispone de un procesador de cuatro núcleos de 64 bits que funciona a 1,4 GHz, y LAN inalámbrica doble banda de 2,4 GHz y 5 GHz, Bluetooth 4.2 / BLE, Fast Ethernet y característica PoE.

La LAN inalámbrica dual band viene con certificación de cumplimiento modular, lo que permite el diseño del producto final donde se reduce significativamente las pruebas de funcionamiento, la mejora de costos y tiempo de comercialización.

La tarjeta Raspberry Pi 3 Model A+ en comparación con la modelo B+ cuenta con una memoria RAM 512 MB y no cuenta con capacidad PoE además esta tarjeta tiene un menor tamaño y costo [14] [15]. En la Figura 1.8 y Figura 1.9 se puede observar los dos tipos de tarjetas antes mencionadas.

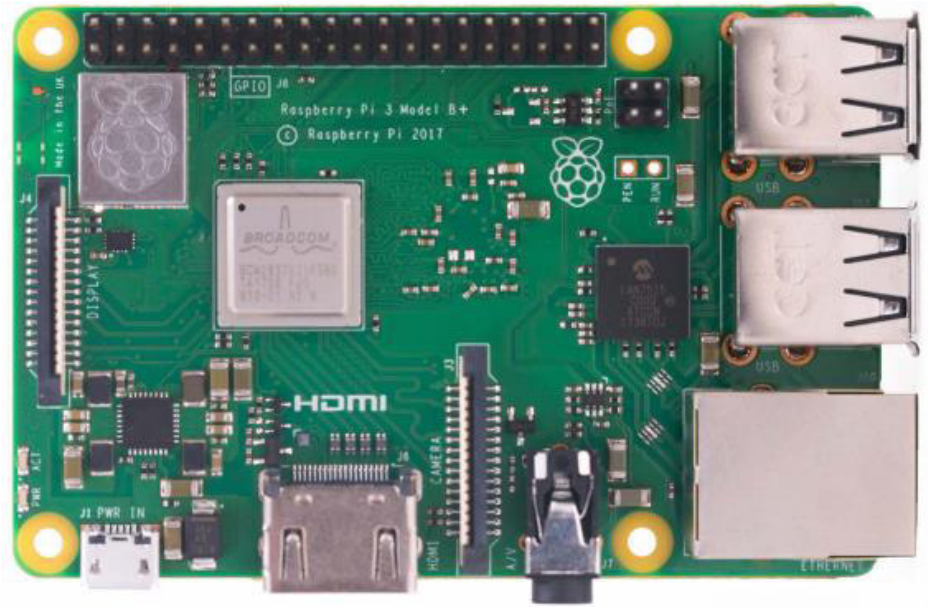

Figura 1.8 Tarjeta Raspberry Pi 3 B+

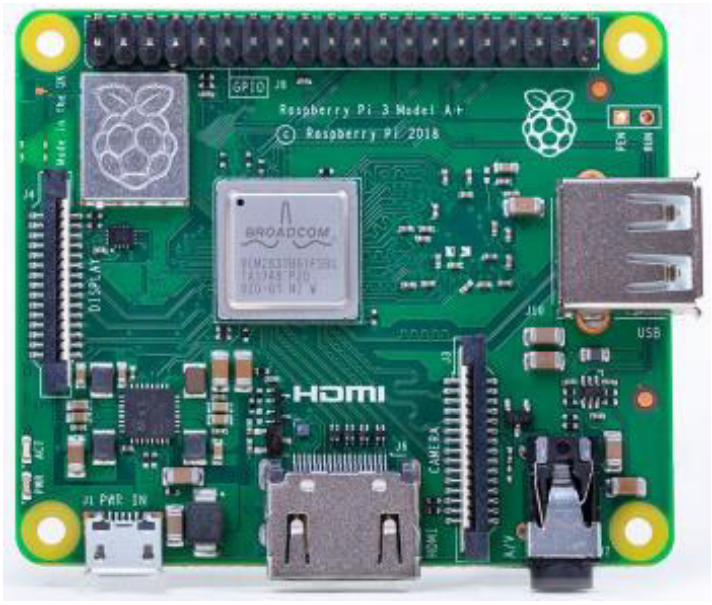

Figura 1.9 Tarjeta Raspberry Pi 3 A+

1.3.6.4 Raspberry PI 4 Modelo B

El último producto de Raspberry Pi es a Raspberry Pi 4 Model B. Cuenta con incrementos revolucionarios de velocidad del procesador, alto nivel de rendimiento multimedia, de memoria y conectividad en comparación a su predecesor Raspberry Pi 3B+, y es compatible con versiones anteriores.

Raspberry Pi 4 Model B cuenta con un rendimiento de escritorio similar al sistema de PC x86 de nivel de entrada. Entre las características principales de la tarjeta se tiene un procesador quad-core de 64 bits de alto rendimiento, capacidad para soportar doble pantalla a resoluciones máximo de 4K conectadas a un par de puertos micro-HDMI, decodificación de video por hardware hasta 4Kp60, con máximo de 8GB de RAM, LAN inalámbrica dual band (2,4/5,0 GHz), Bluetooth 5.0, Gigabit Ethernet, USB 3.0, y capacidad PoE (mediante un complemento PoE HAT separado). En la figura 1.10 se observa el tipo de tarjeta Raspberry Pi 4 B.

Figura 1.10 Tarjeta Raspberry Pi 4 Modelo B

La LAN inalámbrica dual band viene con certificación de cumplimiento modular, que permite el diseño del producto final reduciendo significativamente las pruebas de funcionamiento, la mejora de costos y tiempo de comercialización. [16].

1.3.6.5 Raspberry PI Pico

Raspberry Pi Pico es la primera placa de clase de microcontrolador de Raspberry Pi. Construido alrededor de una plataforma de silicio RP2040, Pico trae características de alto rendimiento, bajo costo y facilidad de uso para el espacio del microcontrolador.

Con una gran memoria en chip, un complejo de procesador simétrico de doble núcleo, tejido de bus determinista y un rico conjunto de periféricos aumentados con nuestro exclusivo

Subsistema de E / S programable (PIO), RP2040 proporciona a los usuarios profesionales con un poder y una flexibilidad inigualables. Con documentación detallada, un pulido Puerto MicroPython y un gestor de arranque UF2 en ROM, tiene la menor cantidad posible barrera de entrada para usuarios principiantes y aficionados. RP2040 se fabrica en un nodo de proceso moderno de 40 nm, que ofrece un alto rendimiento, bajo consumo de energía dinámica y baja fuga, con una variedad de modos de bajo consumo para admitir un funcionamiento de duración prolongada con la energía de la batería [17]. En la figura 1.11 se observa la tarjeta Raspberry Pi Pico.

Figura 1.11 Tarjeta Raspberry Pi Pico

1.3.7 Cámara V2 Raspberry Pi

El módulo de cámara Raspberry Pi v2 cuenta con un sensor Sony IMX219 de 8 megapíxeles (una mejora significativa en comparación a la cámara original que tiene un

sensor OmniVision OV5647 de 5 megapíxeles). En la figura 1.12 se observa el módulo de cámara V2.

A través del módulo de la cámara es posible la captura de videos en alta definición, al igual que la toma de fotografías fijas. Su manejo es apto a nivel de principiante, con características sofisticadas para usuarios avanzados permitiendo ampliar sus conocimientos. Entre las aplicaciones más usadas se encuentra videos de time-lapse, cámara lenta y otros. Cuenta con bibliotecas que se incluye con la cámara y permiten crear efectos.

Este módulo es compatible con todos los modelos de Raspberry Pi 1, 2, 3 y 4. Permite el acceso mediante las API MMAL y V4L, y existen varias bibliotecas de terceros creadas para su operación, entre las cuales se incluye la biblioteca de Python Picamera [18].

Figura 1.12 Cámara V2 Raspberry Pi

1.3.8 REQUERIMIENTOS DE SOFTWARE

1.3.8.1 Raspberry Pi OS

Es un sistema operativo open source basado en Debian, optimizado para su implementación a nivel de hardware con Raspberry Pi, y recomendado para la operación en una Raspberry Pi. Este sistema operativo cuenta con alrededor de 35,000 paquetes:

software precompilado contenido en un formato sencillo que permite una fácil instalación en la Raspberry Pi.

El sistema operativo Raspberry Pi se encuentra en desarrollo activo, en pro de mejorar la estabilidad y el rendimiento de los diferentes paquetes Debian en Raspberry Pi [19].

1.3.8.2 LAMP

LAMP es un acrónimo que describe un “Sistema de Infraestructura de Internet” basado en las siguientes herramientas:

- Linux
- Apache
- MySQL
- PHP

Mismas que en conjunto definen la infraestructura de un servidor web, utilizando un modelo de programación para el desarrollo del sistema.

a. LINUX

Linux es uno de los sistemas operativos más conocidos a nivel mundial. De define como sistema operativo a un software que administra y gestiona la comunicación entre los recursos de hardware asociados a su computadora (portátil o PC) y el software en cuestión.

El sistema operativo Linux se compone de lo siguiente:

Bootloader: el software que administra el proceso de inicio de su computadora. A nivel de usuario, esto se representa la pantalla de bienvenida con la que se inicia el sistema operativo.

Kernel: es la única parte del todo que en realidad se llama "Linux". Este es la parte central del sistema operativo que se encarga de administrar toda la comunicación segura entre el hardware y software. Conocido como el nivel más bajo y fundamental del sistema operativo.

Sistema de inicialización: es subsistema encargado de iniciar el espacio del usuario, es decir, gestiona el proceso de arranque después de que el arranque inicial se transfiere

desde el cargador de arranque GRUB (GRand Unified Bootloader) y así controlar los daemons.

Daemons: estos son servicios que se ejecutan en segundo plano, generalmente se inician durante el arranque o una vez se haya iniciado la sesión en el escritorio.

Servidor gráfico: este subsistema muestra los gráficos en su pantalla. Se le conoce como servidor X.

Entorno de escritorio: permite la interacción con el usuario. Entre los entornos que se puede elegir están: KDE, GNOME, Pantheon, Cinnamon, Mate, Xfce, Enlightenment, entre otros. En cada entorno de escritorio se incluye algunas aplicaciones integradas.

Aplicaciones: los sistemas Linux disponen de herramientas caracterizadas como “App Store” que permiten centralizar e instalar de forma sencilla una aplicación. Este tipo de herramientas se utilizan debido a que en el entorno de escritorio no se cuenta con aplicaciones variadas. Por ejemplo, Ubuntu Linux dispone del “Centro de Software de Ubuntu” que permite buscar las aplicaciones deseadas e instalarlas de forma centralizada [20].

b. Apache

El servidor HTTP Apache es un proyecto de “The Apache Software Foundation”.

El proyecto del servidor HTTP Apache ("httpd") es un software de servidor web de código abierto para sistemas operativos como Windows y UNIX. Como objetivo principal se busca proporcionar un servidor seguro, extensible y eficiente para proporcionar servicios HTTP basados en los estándares actuales de HTTP. [21]

c. MySQL

MySQL es un sistema desarrollado y distribuido por Oracle Corporation que permite la gestión de bases de datos basado en código abierto. Se conoce el término “base de datos” a una colección estructurada de datos, capaz contener una simple lista de nombres hasta administrar una gran cantidad de información de una corporación mundial.

La administración de datos (como acceder, agregar y procesar) se lleva a cabo mediante un servidor MySQL Server. Dado a que se manejan grandes cantidades de datos, los sistemas que administran bases de datos deben contar con una capacidad de procesamiento eficaz y robusta.

Las bases de datos MySQL son relacionales, lo que permite almacenar datos en tablas separadas. Las estructuras de la base de datos se basan en un modelo lógico, es decir con objetos como BDD, vistas, tablas, filas y columnas, que ofrece un entorno de programación flexible.

Se configuran reglas que rigen las relaciones entre los diferentes campos de datos, tales como: uno a uno, uno a muchos, único, obligatorio u opcional, y "punteros" entre las tablas existentes. La base de datos hace cumplir estas reglas, de tal modo que, únicamente precisa tener base de datos bien estructurada, para que las aplicaciones no tengan problemas con datos duplicados, inconsistentes, huérfanos, desactualizados o faltantes. [22]

d. PHP

MySQL PHP, un lenguaje de código abierto que significa preprocesador de hipertexto, pertenece a un lenguaje de desarrollo web que se puede integrar con HTML. En lugar de usar varios comandos para mostrar HTML, las páginas web PHP contienen código HTML que contiene código incrustado.

PHP se diferencia del PHP del lado del cliente como JavaScript, en que el código se ejecuta en el servidor y el HTML se genera y se envía al cliente. El código subyacente no está definido, pero el cliente es capaz de recibir el resultado de la ejecución del script. El servidor web también se puede configurar para manejar todos los archivos HTML con PHP, por lo que para el usuario es “transparente” el código contenido en la aplicación web. [23]

1.3.8.3 TEORIA BÁSICA PARA EL DISEÑO DE APLICACIONES WEB

Existen una gran cantidad de arquitecturas, frameworks, aplicaciones, y sistemas de publicación que ayudan al desarrollo de sitios web, incluyendo su administración alojamiento y técnicas que permiten la monitorización, escalabilidad y gestión de datos. [24] Para desarrollar una aplicación web se involucran diferentes tipos de diseño, entre los cuales se encuentran:

- Diseño funcional
- Diseño de arquitectura de software
- Diseño del proceso de negocio
- Diseño de interfaz de usuario

- Diseño de base de datos

Los diseños mencionados permiten un funcionamiento efectivo que requieren de la integración de prácticas de ingeniería de software y principios de diseño. La creación de aplicaciones que funcionan para un gran volumen de uso, son altamente complejas, teniendo como características una alta confiabilidad, alta disponibilidad, alta seguridad y una respuesta interactiva rápida. Para aplicaciones como Google, Amazon.com, eBay y Yahoo estas características son esenciales, y la razón por la cual son usadas por millones de usuarios [25]. Por lo tanto, podemos mencionar que la creación de una página web se fundamenta en el desarrollo de estas características en conjunto con la aplicación del diseño de cada uno de los componentes que la componen.

a. Diseño funcional

El diseño funcional de una aplicación comprende la creación de parámetros específicos de las tareas principales que realizará dicha aplicación, la definición de las tareas se la suele realizar mediante técnicas como por ejemplo las Historias de Usuario, las cuales facilitan la planeación del desarrollo al permitir dividir en pequeños módulos y asignar prioridades a los mismos, esto también permite dividir la aplicación y poder de esta manera crear productos entregables sin tener la aplicación completa desarrollada.

b. Diseño de arquitectura de software

La arquitectura de software indica el funcionamiento, interacción, y estructura entre las partes que componen a una aplicación de cualquier tipo, este es el nivel más alto de diseño de un sistema, consiste en un conjunto de abstracciones y patrones que poseen coherencia y proveen de un marco definido y claro para interactuar con el código fuente. Dicha arquitectura se selecciona con requisitos específicos y ciertas restricciones que se pueden definir en el diseño funcional [26].

c. Diseño del proceso de negocio

Una vez diseñado la funcionalidad y arquitectura del sistema el proceso de negocio requiere de un diseño de alto nivel por parte de una persona con conocimientos en el área en la cual se va a desempeñar la aplicación, es decir que aplica a tareas del mundo real sin intervenir en ellas las especificaciones técnicas.

d. Diseño de interfaz de usuario

La interfaz de usuario representa la parte visual que se le presenta a un usuario al momento de utilizar la aplicación, existen varios tipos de diseño que requieren generalmente del conocimiento de un diseñador gráfico, ya que permiten la interacción con cada uno de los componentes del sistema y por lo tanto su enfoque debe ser entendible para el usuario.

e. Diseño de base de datos

El diseño de una base de datos es requerido en la mayoría de las aplicaciones ya que permiten el almacenamiento de los datos recolectados o que se van a presentar al usuario. Dependiendo de la tecnología de almacenamiento de datos seleccionada se puede tener varios patrones de diseño para una base de datos, entre los más comunes se encuentran datos estructurados, semi-estructurados y no estructurados. [27]

2. METODOLOGÍA

En esta etapa se detalla la instalación la instalación de software necesario para llevar a cabo el proyecto e implementación de la interfaz web, lo cual se considera como parte de la fase de diseño.

2.1. SELECCIÓN DE HARDWARE

2.1.1 Raspberry Pi 3B+

Este micro-ordenador que posee un rediseño absoluto de la placa, con un procesador de mayor frecuencia, soporte WiFi de doble banda, conexión Ethernet más rápido; que son algunas de las mejores características que dispone este modelo. [15]

El procesador Broadcom BCM2837B0 SoC de 64 bits @ 1.4GHz para sus cuatro núcleos, ayuda a obtener un mejor rendimiento en las tareas que ejecute este micro-ordenador. Se dispone de un disipador situado encima del SoC, lo que ayuda a controlar las temperaturas generadas por el procesador al trabajar de forma intensiva.

El sistema operativo recomendado para Raspberry Pi 3B+ es Raspbian, que es una distribución del sistema operativo GNU/Linux basado en Debian. [28] [29]

2.1.2 Cámara v2

Es un módulo propio de las Raspberry que cuenta con un bus de tipo CSI, el cual permite que los datos de video e imágenes lleguen de forma más rápida al procesador de la tarjeta.

La Tabla 2.1 lista las principales características a nivel técnico de la cámara.

Tabla 2.1. Características del módulo de Cámara v2.

Propiedad	Característica
Dimensiones	25mm x 23 mm x 9mm.
Peso	3,4 g
Resolución	8 Megapixeles
Resolución de videos	1080p30,720p60, 640x480p90
Formato de Imágenes	YUV420,RGB888, BMP, PNG, JPEG, JPEG+RAW, GIF.

2.2. SELECCIÓN DE SOFTWARE

Para la realización del presente proyecto, se elige software open source y gratuito.

2.2.1 Raspbian

Se utiliza el sistema operativo Raspbian Stretch, el cual se descarga de forma gratuita en la página oficial de Raspberry Pi. Cuenta con un entorno gráfico práctico y sencillo.

2.2.2 MobaXterm

Es una herramienta de computación remota que dispone de una interfaz intuitiva y de fácil administración, misma que permite acceder de forma eficiente a servidores remotos.

2.2.3 RealVNC

Permite el acceso a la interfaz gráfica propia de cada sistema operativo, lo cual resulta en un entorno más amigable principalmente para el usuario final.

2.2.4 Cloud Computing

Se crea una instancia EC2 de AWS (Amazon Web Services), ya que al proporcionar capacidad de computación escalable en la nube de AWS forma parte del servicio de infraestructura para posteriormente crear el servidor LAMP.

2.3. DISEÑO DEL NODO

Se requiere realizar la configuración respectiva de los componentes del nodo, misma que se detalla a continuación.

2.3.1 INSTALACIÓN DEL SISTEMA OPERATIVO

Se procede a cargar el sistema operativo en una memoria MicroSD, el cual se obtiene en la página web del fabricante. En el presente proyecto se utiliza la versión completa que cuenta con entorno gráfico.

Una vez descargado Raspbian Stretch se descomprime el archivo en cuestión y se obtiene un archivo con extensión *".img"*, mismo que corresponde a la imagen que debe ser cargada en la memoria MicroSD. Esta memoria debe cumplir con características de tipo booteable y contar con al menos 8GB de capacidad.

Mediante el programa SDFormatter se puede configurar la característica booteable de una MicroSD, para lo cual, se selecciona la memoria respectiva, se marca la opción *"Overwrite format"* y se procede a formatear como se visualiza en la Figura 2.1.

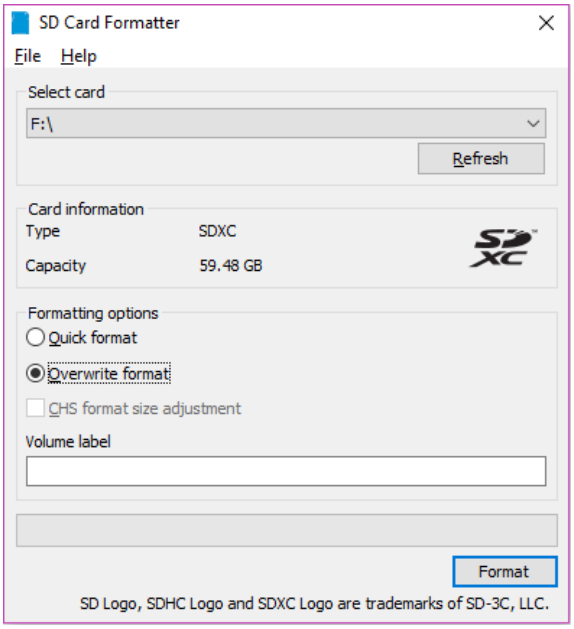

Figura 2.1 Formateo de Memoria

La escritura del sistema operativo en la memoria MicroSD se lleva a cabo mediante el programa Win32DiskImager, una vez se ejecuta el programa se despliega una ventana en la cual se ingresa la ubicación del archivo de extensión *".img"* y se procede a grabar la imagen al seleccionar la opción *"Write"* como se visualiza en la Figura 2.2.

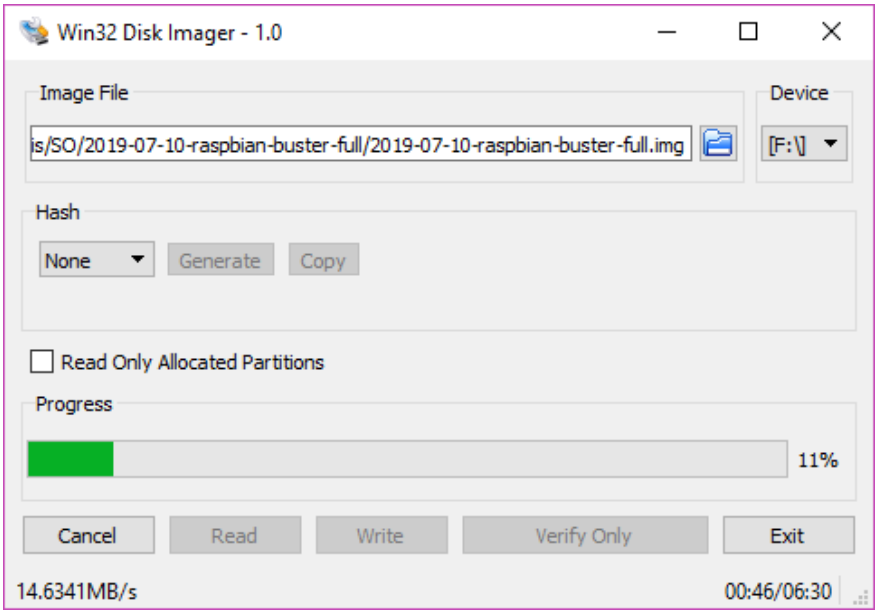

Figura 2.2 Escritura del Sistema Operativo

Al finalizar el proceso de escritura se despliega una ventana emergente que indica que el proceso ha finalizado de forma exitosa.

Una vez cargado el sistema operativo en la MicroSD, se procede a insertar la memoria en la ranura respectiva de la tarjeta Raspberry Pi 3B+ y se conecta la tarjeta a la fuente de alimentación.

Para comunicarse con la tarjeta, en el presente proyecto se optó por una conexión remota mediante SSH (Secure Shell, Interprete de órdenes Seguro) y protocolos de Escritorio Remoto.

2.3.1.1 Conexión Remota

Para la conexión de la tarjeta Raspberry a la red interna, se dispone de un cable Ethernet y del programa Advance IP Scanner facilita determinar la IP de la Raspberry como se indica en la Figura 2.3.

Una vez determinada la IP se puede utilizar algunos software como: RealVNC, MobaXterm, entre otros, para establecer la conexión SSH con la tarjeta.

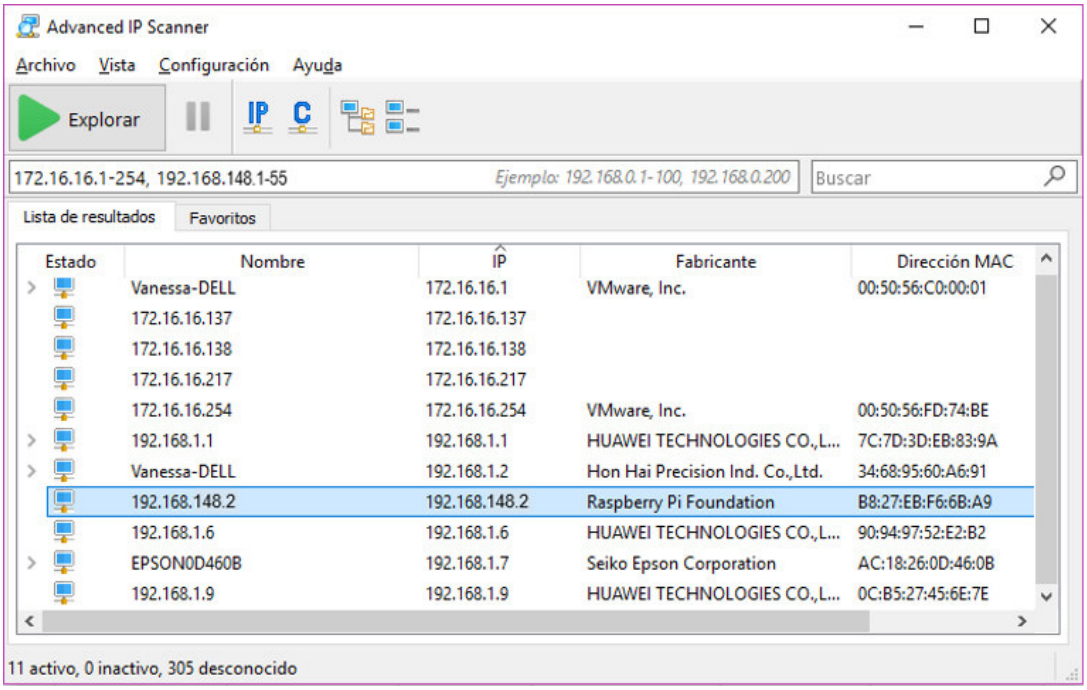

Figura 2.3 Identificación de IP

2.3.1.2 Conexión Remota mediante MobaXterm

Para establecer una sesión nueva en la herramienta MobaXterm de debe seleccionar la opción *"Session"* > *"New Session"*, seleccionar el protocolo SSH y colocar la IP de la Raspberry en el campo *"Remote Host"*. Previo a la comunicación se debe especificar el usuario *"pi"* y seguido de la contraseña "R4spB3rry". La Figura 2.4 indica la comunicación exitosa establecida entre la PC y la tarjeta Raspberry.

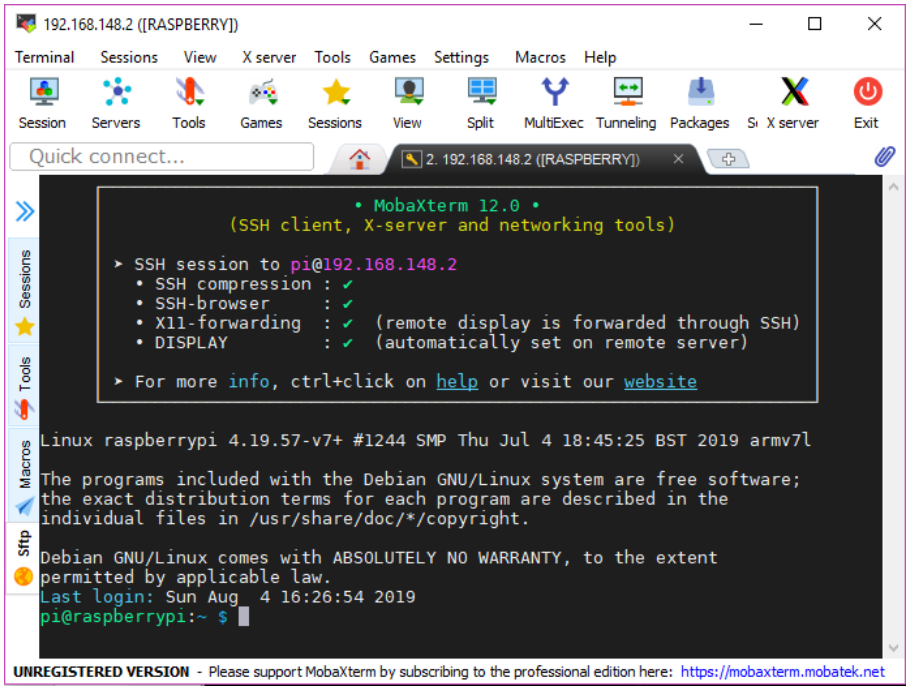

Figura 2.4 Conexión SSH entre Raspberry y PC

2.3.1.3 Conexión Remota con RealVNC

Para poder visualizar las fotografías captadas, se optó por establecer una conexión remota con RealVNC debido a que permite el acceso a la interfaz gráfica de la Raspberry.

RealVNC trabaja baja un modelo cliente-servidor, por lo tanto, es necesario instalar la aplicación tanto en la Raspberry como en la PC.

La instalación en la tarjeta Raspberry (servidor) se realiza mediante el comando: *"sudo apt-get install realvnc-vnc-server realvnc-vnc-viewer"* y habilitando el servicio *"VNC"* que se encuentra dentro de la opción *"Interface Options"*. En el cliente se instala la aplicación VNC Viewer.

Para establecer la comunicación entre Raspberry y PC, en VNC Server crea un escritorio virtual; para ingresar a este escritorio se ejecuta el comando *"vncserver :x"* siendo x el número de escritorio virtual que va a ser creado.(Figura 2.5)

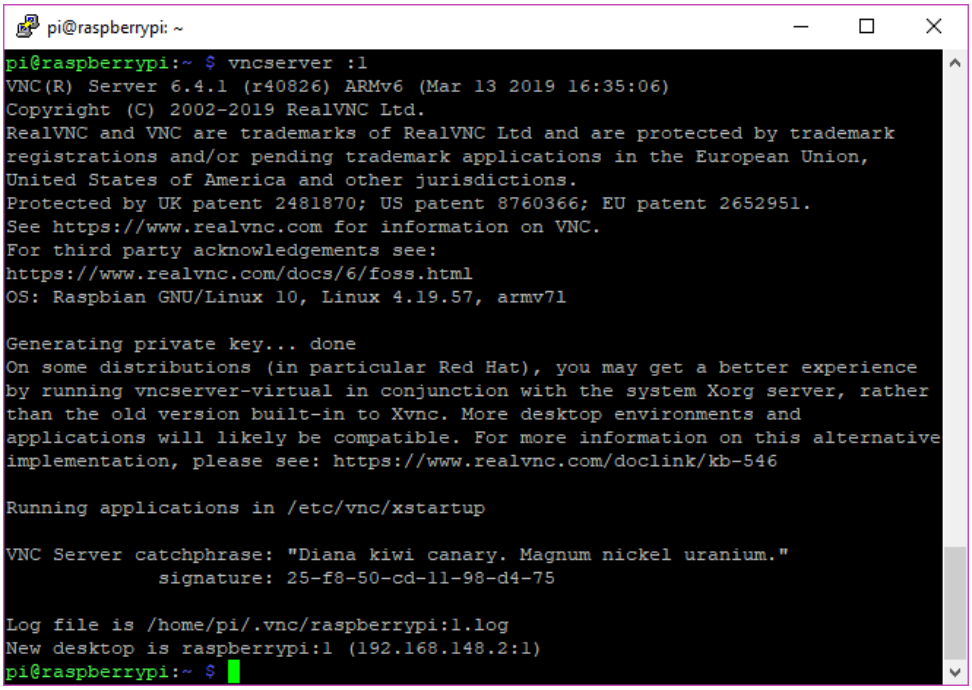

Figura 2.5 Comunicación SSH entre Raspberry y PC

En el VNC Viewer se ingresa la dirección IP de la Raspberry seguido del número del escritorio virtual *"192.168.148.2:1"* y en la ventana de autenticación se ingresa el usuario y contraseña anteriormente expuestos, como se visualiza en la Figura 2.6

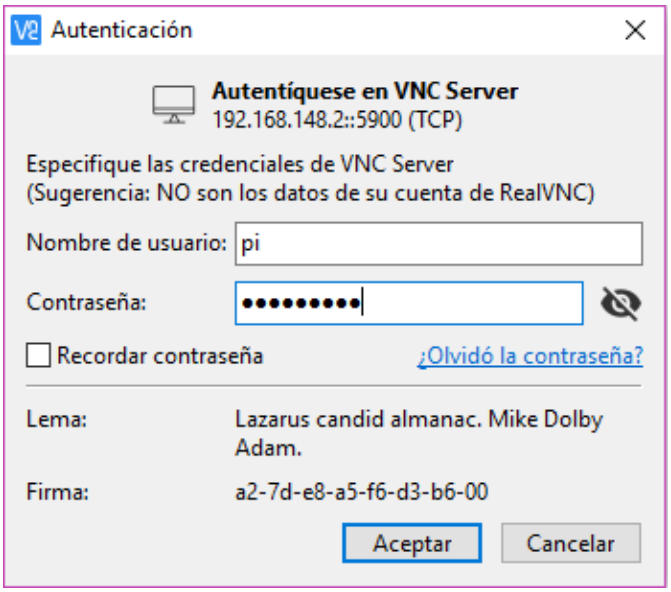

Figura 2.6 Autenticación VNC Viewer

Una vez aceptada la autenticación se puede observar la interfaz gráfica de la Raspberry. (Figura 2.7)

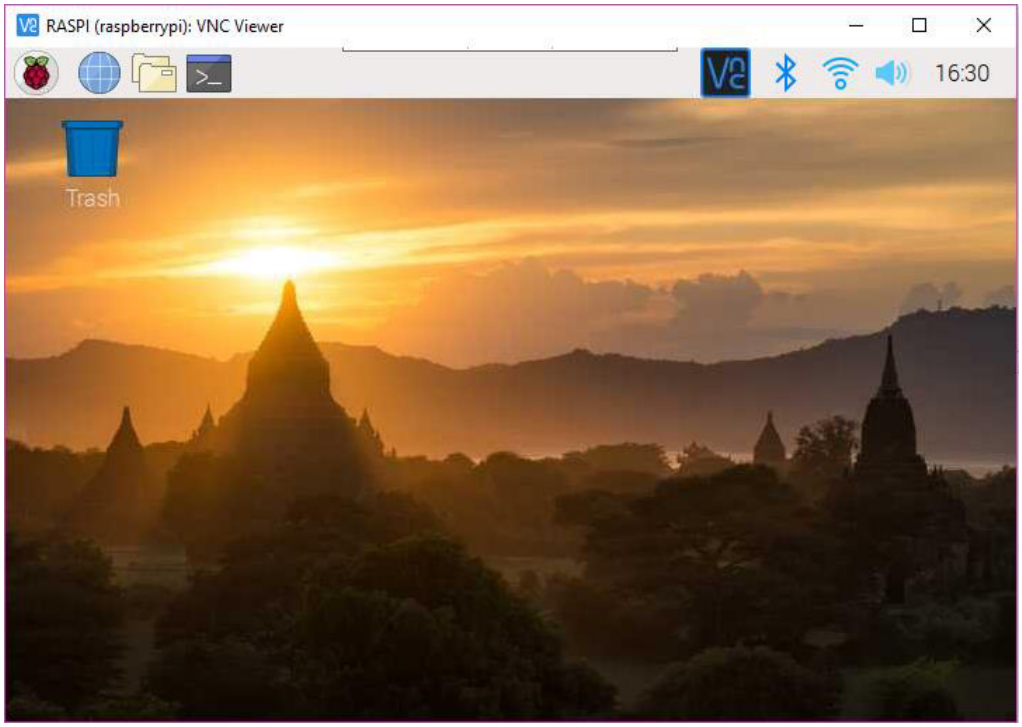

Figura 2.7 Escritorio Remoto

2.3.2 CONFIGURACIÓN DE LA CÁMARA

La instalación del hardware se inicia al momento de insertar el cable de datos en el puerto CSI de la Raspberry. Para la configuración de software se ingresa en el menú de configuración de la tarjeta Raspberry, en la opción *"Interface Options"* y se habilita la cámara mediante la opción *"P1 Camera"*.

Para comprobar el módulo de cámara esté funcionando de forma correcta, se ingresa en el terminal de la Raspberry el comando: *"raspistill –o imagen_prueba.jpg"*, siendo *"imagen_prueba"* el nombre de la imagen capturada y *".jpg"* identifica la extensión del archivo. Este archivo se almacena por defecto en el directorio *"/home/pi"*, como se visualiza en la Figura 2.8

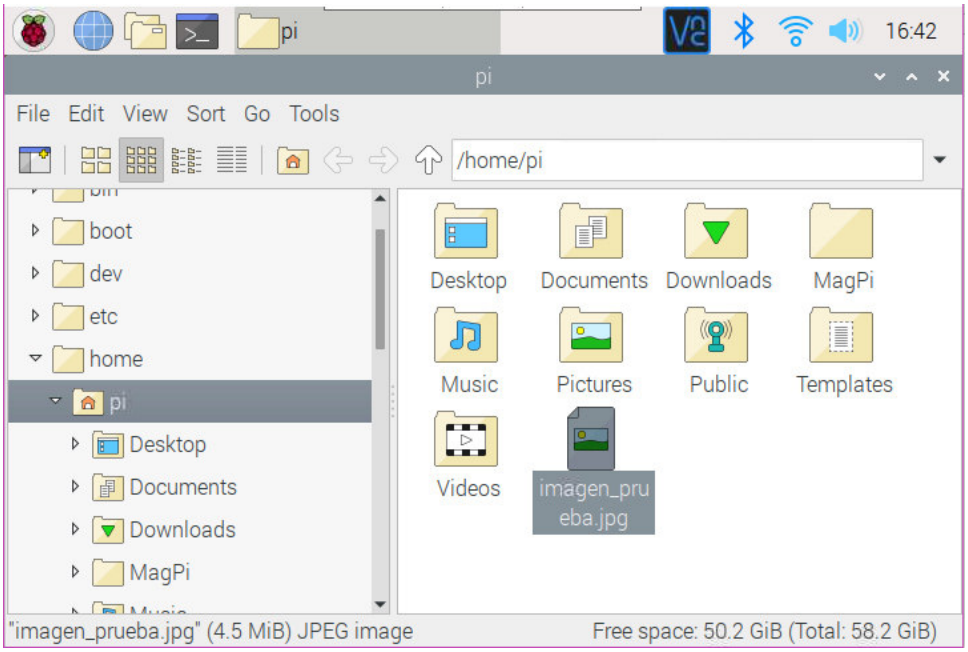

Figura 2.8 Prueba funcionamiento módulo cámara

En el presente proyecto se utiliza la librería *Picamera*, la cual nos permite controlar la cámara y está basada en lenguaje Python.

2.3.3 IMPLEMENTACIÓN SCRIPT DE FUNCIONAMIENTO

Se hace uso del lenguaje de programación Python versión 2.7 para llevar acabo la programación del prototipo.

La Figura 2.9 describe los procesos que ejecuta el nodo para: detectar la presencia de intrusos en el área definida, almacenar localmente la imagen procesada, envío de las imágenes al servidor, envío de alerta de intrusos al correo registrado e inserción de datos (fecha y hora de envío de alerta) en la base de datos.

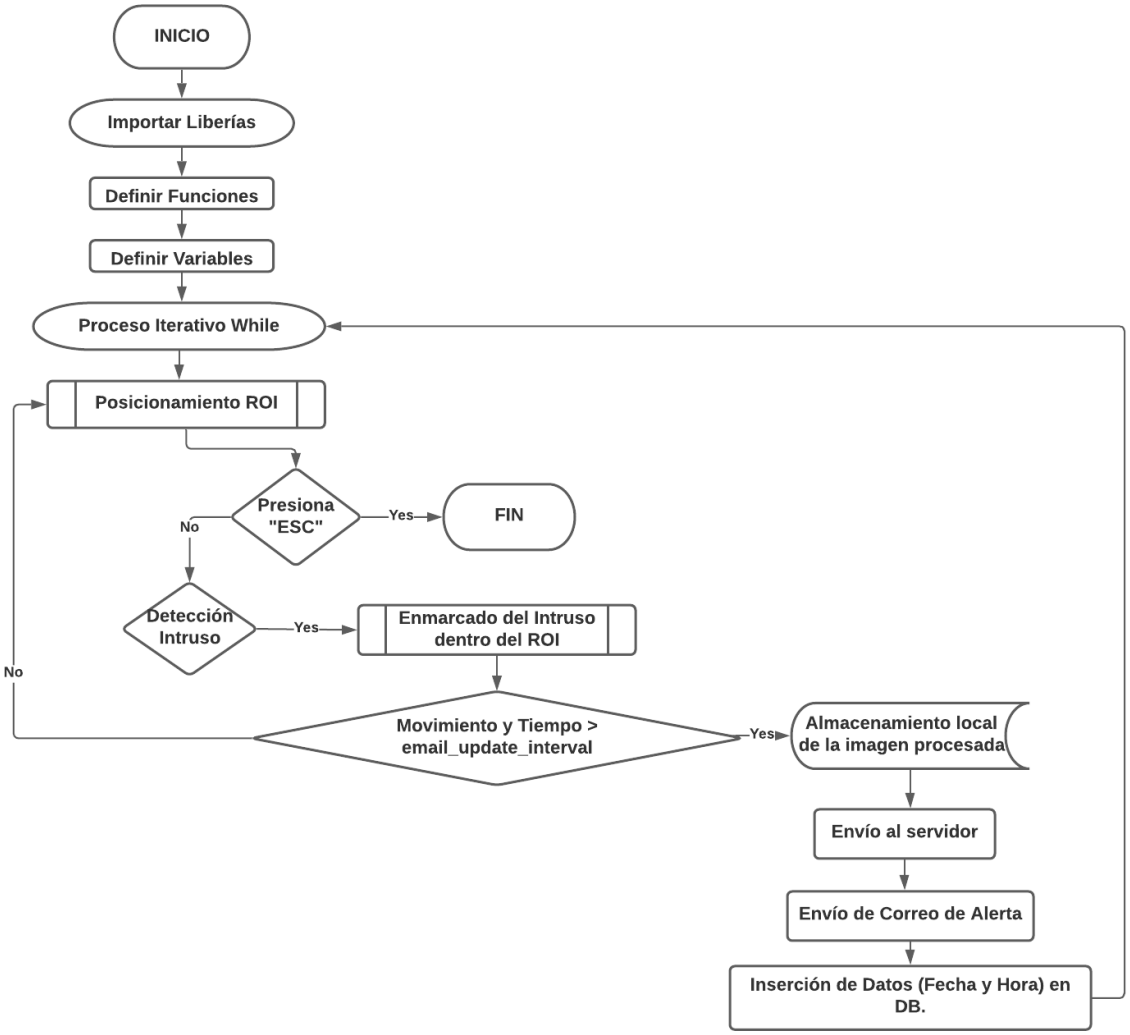

Figura 2.9 Diagrama de flujo de los procesos del nodo.

El script de funcionamiento inicia importando las librerías necesarias para la ejecución de las diferentes etapas que cumple el nodo, define las variables y funciones que intervienen en la ejecución como subprocesos.

El proceso iterativo “While” realiza la captura constante de frames en tiempo real, sobre los cuales se posiciona el área de interés predefinida. Dentro de esta área se realiza el proceso de detección de movimiento mediante clasificadores en cascada “Haar Cascade” los cuales están basados en el enfoque machine learning mismos que están estrechamente relacionados con la “visión por computadora” y el procesamiento de imágenes [30]. Al detectar movimiento dentro del área de interés, se dibuja un contorno rectangular alrededor del “objeto” detectado. (Figura 2.10)

```
# Proceso de detección de intrusos
while True:
    _, image_frame = cam_capture.read()

    # Posicionamieto del ROI en la imagen inicial
    r = cv2.rectangle(image_frame, upper_left, bottom_right, (100, 50, 200), 5)
    rect_img = image_frame[upper_left[1] : bottom_right[1], upper_left[0] : bottom_right[0]]

    # Para detección de movimiento, procesamiento de imagen
    found_objects = False
    rect_img = cv2.resize(rect_img, None, fx=scaling_factor, fy=scaling_factor, interpolation=cv2.INTER_AREA)
    gray = cv2.cvtColor(rect_img, cv2.COLOR_BGR2GRAY)

    objects = object_classifier.detectMultiScale(
            gray,
            scaleFactor=1.1,
            minNeighbors=5,
            minSize=(30, 30),
            flags=cv2.CASCADE_SCALE_IMAGE
    )

    if len(objects) > 0:
            found_objects = True

    # Establecimiento de un rectángulo alrededor del intruso detectado.
    for (x, y, w, h) in objects:
        cv2.rectangle(rect_img, (x, y), (x + w, y + h), (0, 255, 0), 3)
```

Figura 2.10 Segmento de código posicionamiento ROI, detección movimiento, procesamiento de imagen y enmarcado de intruso.

Se estableció un “identificador único” que sirve para asignar un nombre a la imagen en cuestión, una vez se haya detectado una posible intrusión. (Figura 2.11)

```
# Identificación y almacenacimento de la imagen procesada (cuando se detectó un intruso)
tife=time.strftime('%Y%m%d%H%M%S')
prefijo="image"
path="/var/www/html/Images"
n_local=path+"/"+prefijo+tife+".jpg"
```

Figura 2.11 Segmento de código identificador único.

Se optó por definir un intervalo de tiempo durante el cual se enviará un solo correo electrónico de alerta, con el fin de no saturar la bandeja de entrada del usuario y una posible incurrencia de correos catalogados como SPAM. Durante este intervalo de tiempo y con el antecedente de una posible intrusión, se procederá a almacenar la imagen capturada en un repositorio local y el respectivo envío al servidor alojado en la nube; seguido del envió de alerta de intrusión al correo electrónico registrado y se almacenan los datos de la alerta en virtud de la fecha y hora, en la base de datos. (Figura 2.12)

```
    # Definición para envío de alerta
    if found_objects and (time.time() - last_epoch) > email_update_interval:
        last_epoch = time.time()
        print ('Enviando email...')

        # Almacenamiento local de la imagen procesada
        cv2.imwrite(n_local, rect_img)

        # Envío de imagen al servidor AWS-EC2 mediante SCP
        print ('Envío a servidor')
        os.system('sudo scp -r -i /var/www/html/SVR_AWS/servidoraws2021.pem "%s" ubuntu@3.21.178.11:/var/www/html/Imagenes_eventos'%n_local)

        # Parámetros de envío
        msgRoot = MIMEMultipart('related')
        msgRoot['Subject'] = 'Alerta: Actualización de seguridad'
        msgRoot['From'] = fromEmail
        msgRoot['To'] = toEmail
        msgRoot.preamble = 'Actualización de la cámara de seguridad de la Raspberry Pi'

        msgAlternative = MIMEMultipart('alternative')
        msgRoot.attach(msgAlternative)
        msgText = MIMEText('Posible intruso detectado por cámara de seguridad')
        msgAlternative.attach(msgText)

        msgText = MIMEText('<img src="cid:image1">', 'html')
        msgAlternative.attach(msgText)

        image = open(n_local, 'rb').read()
        msgImage = MIMEImage(image)
        msgImage.add_header('Content-ID', '<image1>')
        msgRoot.attach(msgImage)

        smtp = smtplib.SMTP('smtp.gmail.com', 587)
        smtp.starttls()
        smtp.login(fromEmail, fromEmailPassword)
        smtp.sendmail(fromEmail, toEmail, msgRoot.as_string())
        smtp.quit()

        print ('Hecho!')

        # Inserción de Datos en la DB
        Fecha1=time.strftime('%Y-%m-%d')
        Hora1=time.strftime('%H:%M:%S')
        query=("INSERT INTO Eventos" "(Fecha,Hora)" "VALUES ('%s','%s' )")%(Fecha1,Hora1)
        run_query(query)
    if cv2.waitKey(1) == 13:
        break

cam_capture.release()
cv2.destroyAllWindows()
```

Figura 2.12 Segmento de código almacenamiento local, envió de imagen al servidor, envío de correo e inserción de datos en la DB.

2.3.3.1 CONFIGURACIÓN DEL PROTOCOLO SMTP

Para llevar a cabo el envío de correo electrónico al momento en que se genera la alerta de intrusión, se crea un correo electrónico que cumple la función de remitente. Se opta por utilizar el servidor de Gmail ya que cuenta con la opción de habilitar el *"Acceso de aplicaciones menos seguras"* como se indica en la Figura 2.13, lo cual es un requisito obligatorio para el uso del protocolo SMTP.

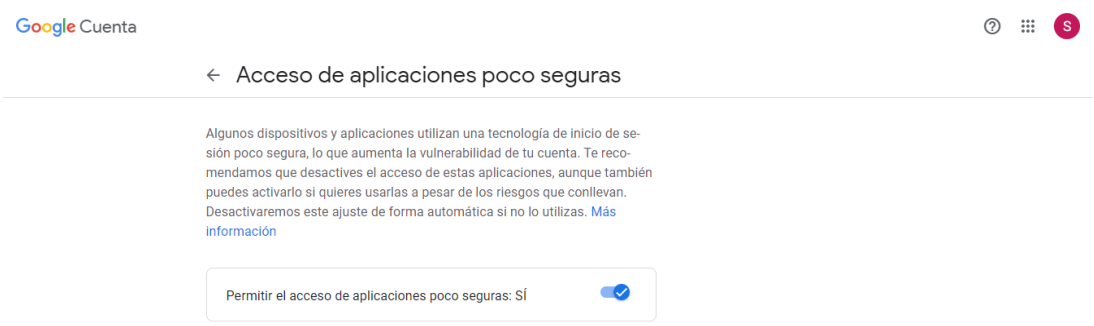

Figura 2.13 Acceso de aplicaciones menos seguras.

La cuenta creada y que cumple la función de remitente es *"secucam21@gmail.com"* y la cuenta a la cual se enviará el correo de alerta, es decir el destinatario, será la dirección de correo electrónica registrada por el usuario como parte de la configuración inicial. La Figura 2.14 muestra el segmento de código para el envío de alertas por correo electrónico, donde la variable *'fromEmail'* es la cuenta de correo electrónico creada.

```
# Parámetros de envío
msgRoot = MIMEMultipart('related')
msgRoot['Subject'] = 'Alerta: Actualización de seguridad'
msgRoot['From'] = fromEmail
msgRoot['To'] = toEmail
msgRoot.preamble = 'Actualización de la cámara de seguridad de la Raspberry Pi'

msgAlternative = MIMEMultipart('alternative')
msgRoot.attach(msgAlternative)
msgText = MIMEText('Posible intruso detectado por cámara de seguridad')
msgAlternative.attach(msgText)

msgText = MIMEText('<img src="cid:image1">', 'html')
msgAlternative.attach(msgText)

image = open(n_local, 'rb').read()
msgImage = MIMEImage(image)
msgImage.add_header('Content-ID', '<image1>')
msgRoot.attach(msgImage)

smtp = smtplib.SMTP('smtp.gmail.com', 587)
smtp.starttls()
smtp.login(fromEmail, fromEmailPassword)
smtp.sendmail(fromEmail, toEmail, msgRoot.as_string())
smtp.quit()
```

Figura 2.14 Implementación SMTP.

El registro del destinatario se lo realiza directamente en el módulo de configuración, que se explica más adelante.

2.4. DISEÑO DEL SUBSISTEMA DE VISUALIZACIÓN Y ALMACENAMIENTO

Este subsistema permite visualizar, almacenar y administrar la información del nodo mediante una aplicación web de fácil administración e interpretación.

El servidor web se implementa en un entorno virtual de AWS *(Amazon Web Services)*, sobre una instancia EC2 *(Amazon Elastic Compute Cloud)* ya entre las características principales de la plataforma se tiene la escalabilidad de tamaño y la seguridad que ofrece. La primera instancia a la que se accede es gratuita durante el primer año y permite asignar una dirección IP pública. El Anexo D detalla la creación de la cuenta en AWS y configuración de la capa gratuita. [31]

Antes de instalar los requerimientos del servidor LAMP se asigna una dirección IP elástica.

2.4.1 ASOCIACIÓN IP ELÁSTICA

Este tipo de direcciones, son direcciones IPv4 caracterizadas por ser estáticas y públicas, las cuales están diseñadas para la informática en la nube dinámica. Permite enmascarar errores de las instancias reasignando las direcciones IP públicas a otra instancia de una cuenta.

Para la configuración se ingresa en el panel de administración de EC2, seleccionar la opción *"Direcciones IP Elásticas"* seguido de *"Asignar la dirección IP elástica"* con la opción por defecto *"Grupo de direcciones IPv4 de Amazon"* y por último *"Asignar"*. Una vez asignada la dirección, la asociación de la IP elástica se realiza al seleccionar la misma y en el menú *"Acciones"*, se escoge la opción *"Asociar dirección IP elástica"* seguido de la instancia y finalmente *"Asociar"*. En la Figura 2.15 se visualiza los parámetros de configuración para completar la asociación con la dirección IP elástica 3.21.178.11.

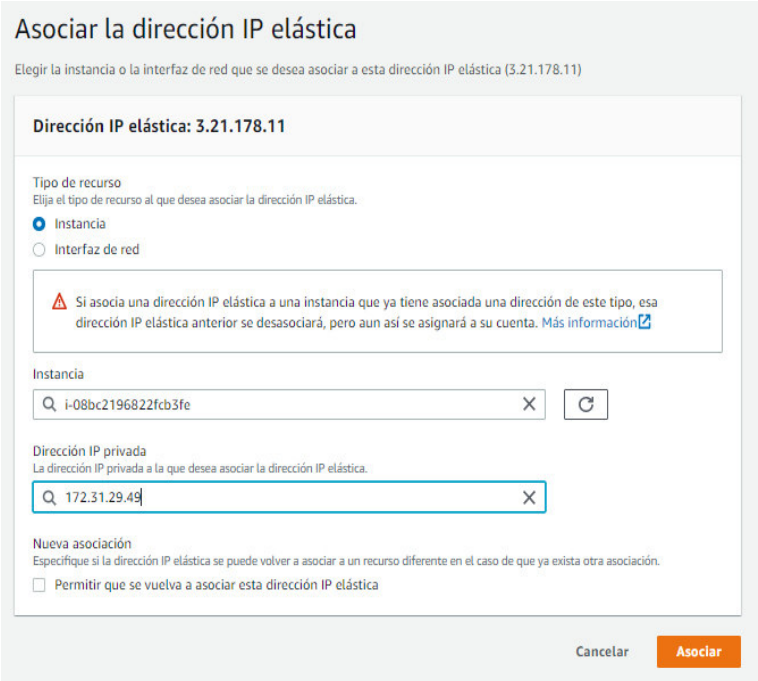

Figura 2.15 Asociación dirección IP elástica.

2.4.2 SERVIDOR WEB

Se realiza la instalación del servidor Apache en la instancia mediante el comando *"sudo apt-get install apache2"*, mismo que permite crear páginas y servicios web [32].

Una vez finalizada la instalación para verificar que el servidor web está corriendo correctamente, se ingresa en el navegador la dirección IP elástica de la instancia y el resultado debe ser igual al de la Figura 2.16

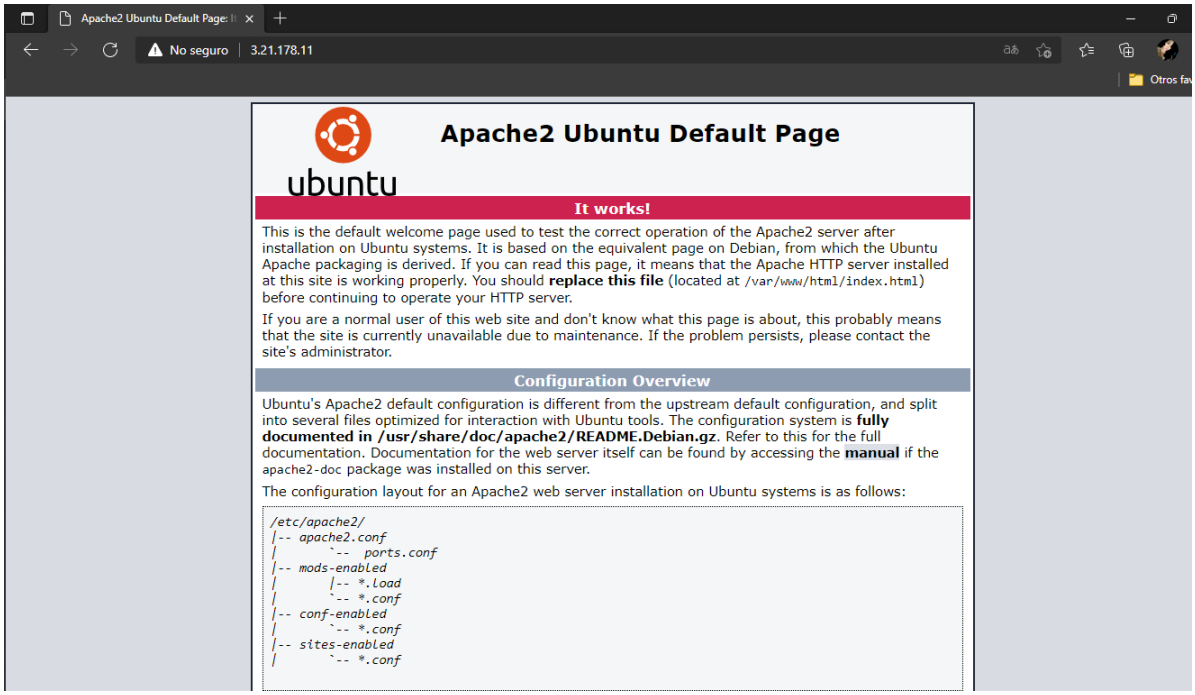

Figura 2.16 Servidor Web Apache verificación de funcionamiento

Para crear páginas web basadas en lenguaje php y poder establecer la conexión con la base de datos, se ingresa el siguiente comando *"sudo apt-get install php libapache2-mod-php php-mcrypt php-mysql"*.

2.4.3 ALMACENAMIENTO DE DATOS

En el presente proyecto el almacenamiento y gestión de datos se realiza mediante MySQL en conjunto con phpMyAdmin.

La instalación se realiza mediante el comando *"sudo apt-get install mysql-server"*, el cual pide especificar la contraseña del usuario root. La gestión de base de datos se realiza mediante phpMyAdmin, por lo que se ejecuta el comando *"sudo apt-get install phpmyadmin"* para la instalación; y durante el proceso de instalación se ingresa la clave el usuario root de MySQL.

Para ingresar a la base de datos colocamos la ruta *"http://3.21.178.11/phpmyadmin"* en el navegador, como se muestra en la Figura 2.17

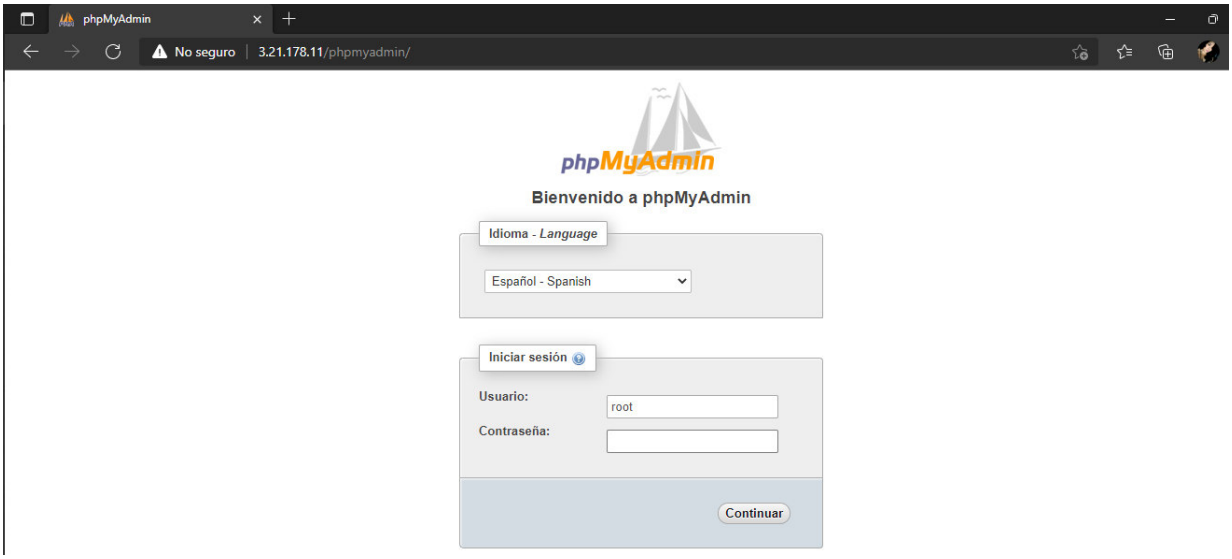

Figura 2.17 Login phpMyAdmin

Al iniciar sesión mediante el usuario root, se crea la base de datos *"Monitoreo"* la cual alberga la tabla *"Eventos"*, misma que está conformada por los campos: Fecha y Hora.

El campo *"Fecha"* contiene la fecha en formato año-mes-día de la emisión de la alerta y el campo *"Hora"* contiene la hora de ocurrencia de la alerta en formato HH:MM:SS. La figura 2.18 visualiza la estructura mencionada

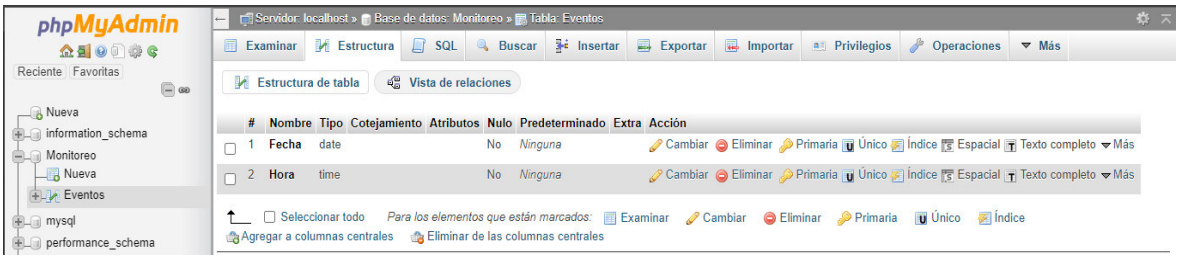

Figura 2.18 Estructura de la tabla Eventos

2.4.3.1 ACCESO REMOTO ALA BASE DE DATOS

Ya que el acceso a la base de datos MySQL solo se permite de forma local, se debe crear un usuario con los suficientes permisos para el acceso desde un equipo remoto.

La creación del usuario es mediante el comando *"CREATE USER 'user'@'%' IDENTIFIED BY 'password;"*, siendo user y password las credenciales creadas. [33]

Para asignar permisos y privilegios al usuario creado se ingresan los siguientes comandos [34]:

- *GRANT ALL PRIVILEGES ON *.* TO 'user'@'%' WITH GRANT OPTION;*
- *GRANT ALL PRIVILEGES ON *.* TO 'user'@'%' IDENTIFIED BY ' password' WITH GRANT OPTION;*
- *FLUSH PRIVILEGES;*

Se requiere modificar el archivo de configuración MySQL, seguido de la configuración los puertos en la instancia EC2. Para modificar el archivo se ingresa el comando *"sudo nano /etc/mysql/mysql.conf.d/mysqld.cnf"*, en la línea *"bind-address"* se cambia la dirección IP por *"0.0.0.0"*; lo cual permite el acceso remoto a cualquier dispositivo mediante las credenciales creadas.

En la instancia AWS, se configura los *"Security Groups"* al añadir una nueva regla *"Inbound"* que permita el tráfico de cualquier dirección IP mediante el puerto 3306 (MYSQL/Aurora) [35] como se indica en la Figura 2.19.

Name	ID de la regla del g...	Versión de IP	Tipo	Protocolo	Intervalo de puertos	Origen
–	sgr-02c8de2365b340a...	IPv6	HTTP	TCP	80	::/0
–	sgr-02227a2f6b215636c	IPv6	SSH	TCP	22	::/0
–	sgr-02050127a681d6...	IPv4	SSH	TCP	22	0.0.0.0/0
–	sgr-0cab421556d5529f9	IPv6	TCP personalizado	TCP	0	::/0
–	sgr-0dd05a1140a5ae6...	IPv4	MYSQL/Aurora	TCP	3306	0.0.0.0/0
–	sgr-06c7338c93dd0f83b	IPv4	TCP personalizado	TCP	0	0.0.0.0/0
–	sgr-02afe0e31ad3e950a	IPv4	HTTP	TCP	80	0.0.0.0/0

Figura 2.19 Regla Inbound MySQL/Aurora.

2.4.4 VISUALIZACIÓN DE DATOS

Al implementar una interfaz web se busca ofrecer una aplicación de fácil manejo e intuitiva, que permita la interacción del usuario y el prototipo.

En el presente proyecto el desarrollo de la aplicación web se realiza mediante el uso del servidor web, lenguajes de programación PHP, HTML y CSS. Se compone de:

- Index: contiene el menú de los módulos que componen la aplicación.
- Módulo de Configuración: se compone del registro de correo electrónico y la selección del área de interés. La figura 2.20 y la Figura 2.21 muestra el segmento de código utilizado para la obtención del correo ingresado por el usuario.

```
<td align=center>
<CENTER><h1><font color=gray>REGISTRO DE CORREO</font></h1></CENTER>
<form method="post" action="registro.php">
<table>
<tr>
<td><label>Correo Electrónico</td>
<td><input type="text" id="txtCorreo" name="txtCorreo"/></label></td>
</tr>
<tr>
<td align="center" colspan="2"><button type="submit">Registrar</button></td>
</tr>
</table>
</form>
</td>
```

Figura 2.20 Opción de ingreso del correo.

```
<html>
        <head>
                <?php
                        include('configuracion.php');
                        $correo = $_POST["txtCorreo"];
                        $archivo = fopen("regcorreo.txt","w");
                        fwrite($archivo,$correo);
                        fclose($archivo);
                ?>
        </head>
        <body>
                <div align="center">
                        <h2>Su registro ha sido exitoso</h2>
                </div>
        </body>
</html>
```

Figura 2.21 Registro de correo.

En este proyecto se definió tres áreas de interés, de las cuales se puede seleccionar solo una de las tres opciones. La Figura 2.22 muestra el segmento de código implementado.

```
<body>
<textarea id="textarea1" rows="1" cols="40">python /var/www/html/roi1.py</textarea><button id="copyBlock1">copiar</button>
</body>
<script language="JavaScript">
var textarea = document.getElementById("textarea1");
var answer = document.getElementById("copyAnswer1");
var copy = document.getElementById("copyBlock1");
copy.addEventListener('click', function(e) {
textarea1.select();
try {
var successful = document.execCommand('copy');
if(successful) answer.innerHTML = 'Copiado';
else answer.innerHTML = 'no se copio';
} catch (err) {
answer.innerHTML = 'Navegador no soportado';
}
});
</script>
```

Figura 2.22 Selección área de interés.

La Figura 2.23 muestra el front-end de lo expuesto.

Figura 2.23 Front-End Módulo de Configuración

- Módulo de Reportes: conformado por el detalle de los eventos que visualiza la información almacenada en la tabla Eventos y la galería de imágenes que contiene las imágenes generadas en las alertas.

  Mediante PHP se realiza la conexión y consulta de base de datos [36] como se visualiza en la Figura 2.24 y la Figura 2.25 muestra el front-end de lo expuesto.

```
<div class="clear"></div>
<CENTER><h1><font color=gray>DETALLE DE LOS EVENTOS</font></h1></CENTER>
<a href="galeria.php"> Haga clic aquí para visualizar la galería de imágenes de los eventos
</a>
<center>
<table border width="27%">
<tr bgcolor="black">
<th><font face=\"arial narrow\" color="magenta"><b>FECHA</b></font></th>
<th><font face=\"arial narrow\" color="magenta"><b>HORA</b></font></th>
</tr>
<?php
$datos=mysqli_connect("localhost","admin","RMpa.ad21") or die ("No se puede conectar a la BDD");
mysqli_select_db($datos,"Monitoreo") or die ("No se puede conectar a la BDD");
$query="SELECT * FROM Eventos";
$resultado=mysqli_query($datos,$query);
$numero=0;

while($filas=mysqli_fetch_assoc($resultado))
{
echo '<tr align=center bgcolor="gray"><td width=\"15%"><div align=\"center\"><font face=\"arial\" color="navy">'.$filas["Fecha"].'</font></div></td>';
echo '<td width=\"15%"><div align=\"center\"><font face=\"arial\" color="navy">'.$filas["Hora"].'</font></div></td></tr>';
$numero++;
}
echo '<tr align=center bgcolor="gray"><td colspan=2"><font face=\"arial\" color="navy"><b>N de Correos Recibidos: '.$numero.'</b></font></td></tr>';

mysqli_free_result($resultado);
mysqli_close($datos);

?>
</table>
</center>
```

Figura 2.24 Segmento de código para conexión con la base de datos y visualización de datos.

Figura 2.25 Front-End Módulo de Reportes – Detalle de los Eventos.

El desarrollo de la galería dinámica de imágenes se lleva a cabo mediante el envió de la imagen detectada desde el nodo al servidor, usando el protocolo SCP. La página web Galería de Imágenes mostrará el conjunto de imágenes almacenadas, la Figura 2.26 muestra el segmento de código implementado de lo expuesto.

```
<?php
$folder_path = 'Imagenes_eventos/';

$num_files = glob($folder_path . "*.{JPG,jpg}", GLOB_BRACE);

$folder = opendir($folder_path);

if($num_files > 0)
{
 while(false !== ($file = readdir($folder)))
 {
  $file_path = $folder_path.$file;
  $extension = strtolower(pathinfo($file ,PATHINFO_EXTENSION));
  if($extension=='jpg')
  {
   ?>

            <a href="<?php echo $file_path; ?>"><img src="<?php echo $file_path; ?>"  height="250" title="<?php echo $file_path; ?>" /></a>

            <?php
  }
 }
}
else
{
 echo "Carpeta vacia !";
}
closedir($folder);

?>
</body>
```

Figura 2.26 Segmento de código galería dinámica.

- Módulo de ayuda: contiene un archivo tipo pdf que en principio será entregado al usuario de forma física y el cual sirve como guía del usuario para la configuración del sistema. Este manual se adjunta en el Anexo E.

3. RESULTADOS Y DISCUSIÓN

En esta sección se presentan los resultados obtenidos por el prototipo en tres escenarios planteados en la mesa de pruebas, y posteriormente se analizan los resultados obtenidos.

3.1 PRUEBAS DE FUNCIONAMIENTO

Las pruebas de funcionamiento se ejecutaron en una residencia ubicada en el sector Norte de Quito, en la cual se planteó monitorear los tres accesos a la residencia: la entrada principal, la entrada del parqueadero y la entrada trasera. La Figura 3.1 muestra el escenario de pruebas, cabe recalcar que la entrada trasera no se logra visualizar en la misma.

Figura 3.1 Escenario de pruebas.

Los aspectos a considerar son:

- La conexión de la Raspberry Pi 3B+ a la red local, se optó por la conexión WiFi ya que permite movilidad.
- Un computador portátil o de escritorio conectada a la misma red local, en el cual se configurará como parte del trabajo inicial del personal técnico; un escritorio remoto mediante el software RealVNC. Mismo que permitirá al usuario registrar el correo electrónico al que se enviarán los correos de alerta y poner en ejecución el script desarrollado para el prototipo.

- Posicionamiento físico de la Raspberry pi 3B+ dependiendo el área a monitorear.

Una vez se hayan llevado a cabo los puntos anteriormente expuestos, se ejecuta el script en cada una de los escenarios a monitorear.

3.1.1 ENTRADA PRINCIPAL

Se seleccionó como ROI el *"Área de Interés 2"* y se ejecutó el script desde el terminal de la Raspberry. La Figura 3.2 se evidencia el script ejecutándose y en la Figura 3.3 se visualiza el posicionamiento del ROI en la imagen total que captura la cámara.

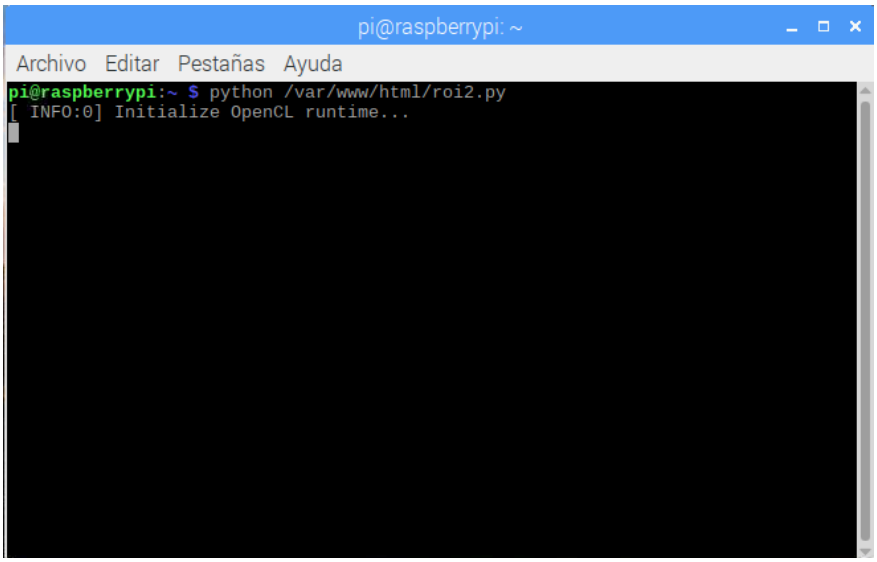

Figura 3.2 Ejecución del Script - Escenario 1.

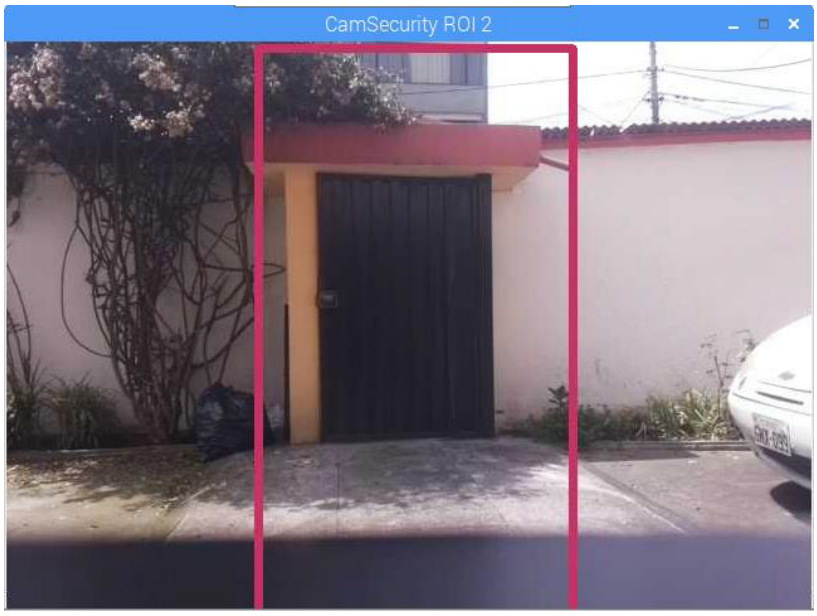

Figura 3.3 Entrada Principal.

Al momento de ejecutar el script, se ingresa en un continuo procesamiento de la imagen que está siendo capturada por la cámara en tiempo real. El momento en que se detecta un intruso (*"objeto extraño"*) dentro del área de interés de inmediato se enmarca este objeto dentro de un rectángulo de color verde y la imagen se guarda en el repositorio local de la Raspberry. Dicha imagen es enviada al email, al servidor y finalmente procede con la inserción de la "Fecha" y "Hora" de esta alerta en la base de datos. La Figura 3.4 muestra lo expuesto mediante prints en el terminal, de lo que va realizando el script.

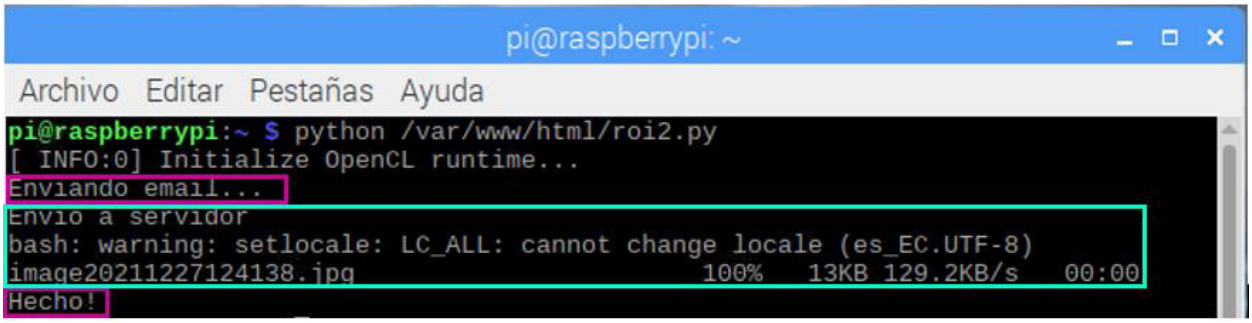

Figura 3.4 Detección de intruso - Entrada Principal.

La Figura 3.5 visualiza el almacenamiento local de la imagen y la Figura 3.6 muestra el correo de alerta enviado al correo electrónico registrado.

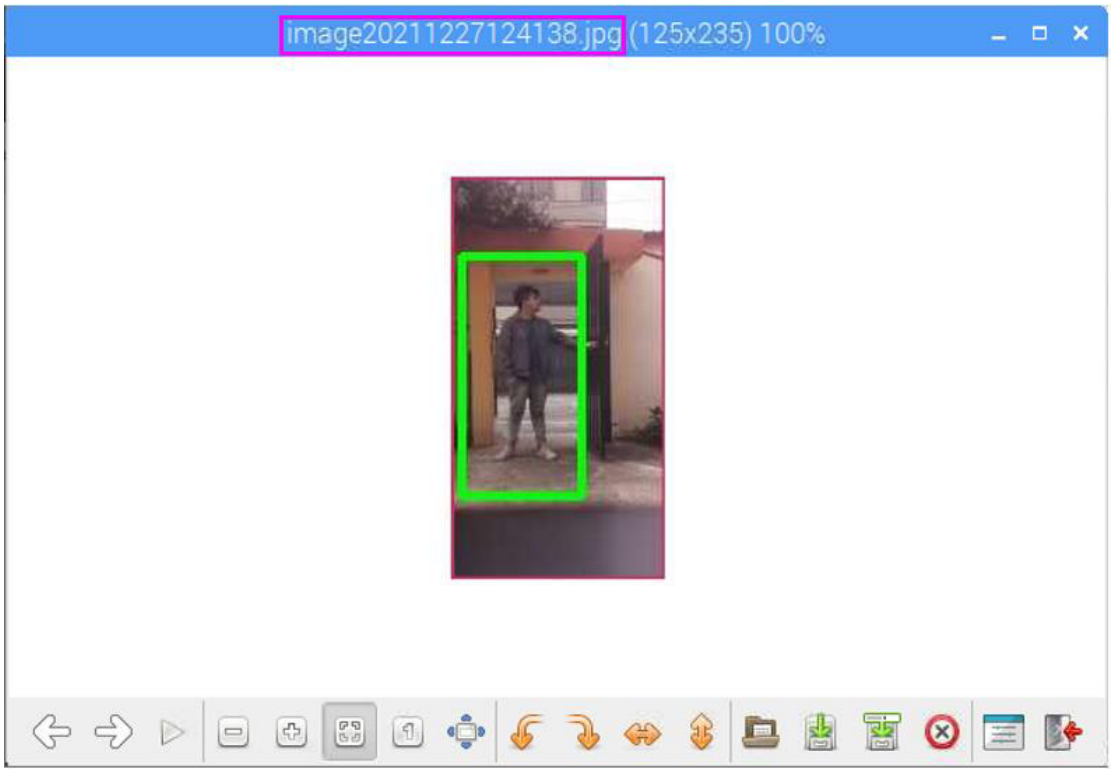

Figura 3.5 Almacenamiento local - Intruso Entrada Principal.

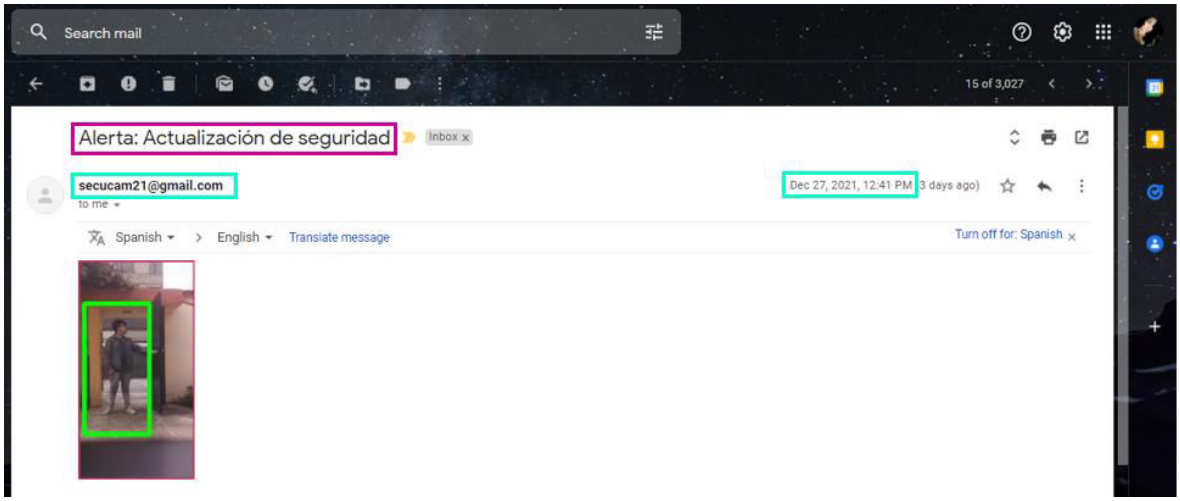

Figura 3.6 Correo de alerta – Entrada Principal.

Para consultar el detalle de estos eventos se ingresa el siguiente link *http://3.21.178.11/detevent.php* en cualquier navegador, ya sea desde un dispositivo móvil, laptop o PC de escritorio como se muestra en la Figura 3.7

Figura 3.7 Detalle de los Eventos - Entrada Principal.

La consulta de la galería que contiene las imágenes almacenas en el servidor, se puede realizar dando clic en el vínculo creado en la página web “Detalle de los Eventos” (Figura 3.8) o ingresando el siguiente link *http://3.21.178.11/galeria.php* en cualquier navegador (Figura 3.9)

Figura 3.8 Vínculo a la galería de imágenes.

Figura 3.9 Galería de imágenes - Entrada Principal.

Al pasar el cursor sobre la imagen se desplegará un texto flotante, el cual corresponde al nombre de la imagen (Figura 3.9) y se constata que es el mismo que el de la imagen almacenada localmente.

3.1.2 ENTRADA DEL PARQUEADERO

En este escenario se seleccionó como ROI el *"Área de Interés 3"* y se ejecutó el script desde el terminal de la Raspberry. La Figura 3.10 muestra el script ejecutándose y en la Figura 3.11 se verifica el posicionamiento del ROI en la imagen total que captura la cámara.

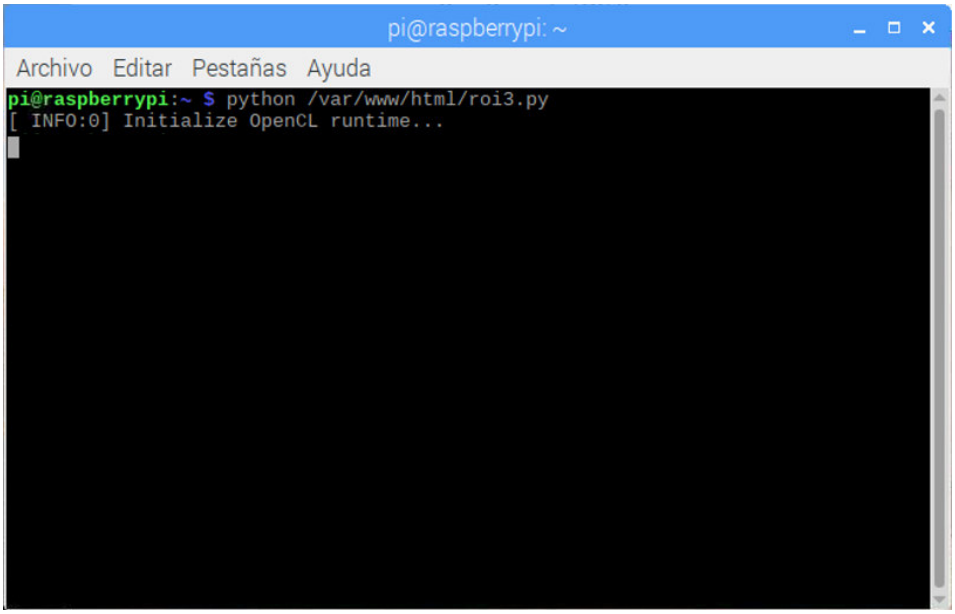

Figura 3.10 Ejecución del Script – Escenario 2.

Figura 3.11 Entrada del Parqueadero.

Al momento de ejecutar el script, el proceso es exactamente igual al expuesto anteriormente, difiriendo únicamente en el área de interés monitoreada. Inicia con un continuo procesamiento de la imagen capturada por la cámara en tiempo real. El momento en que se detecta un intruso (*"objeto extraño"*) dentro del área de interés se enmarca este objeto dentro de un rectángulo de color verde y la imagen se guarda localmente. Dicha imagen es enviada al email, al servidor y finalmente se inserta la "Fecha" y "Hora" de esta alerta en la base de datos. La Figura 3.12 muestra lo expuesto mediante prints en el terminal.

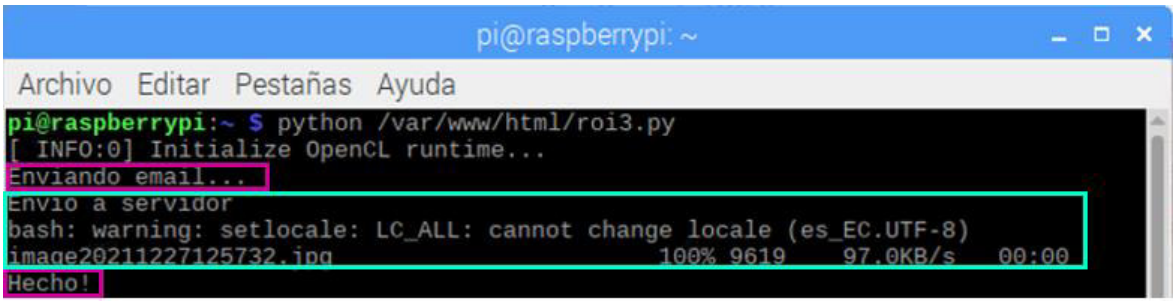

Figura 3.12 Detección de intruso – Entrada del Parqueadero.

La Figura 3.13 se muestra la imagen almacenada localmente y la Figura 3.14 muestra el correo de alerta generado.

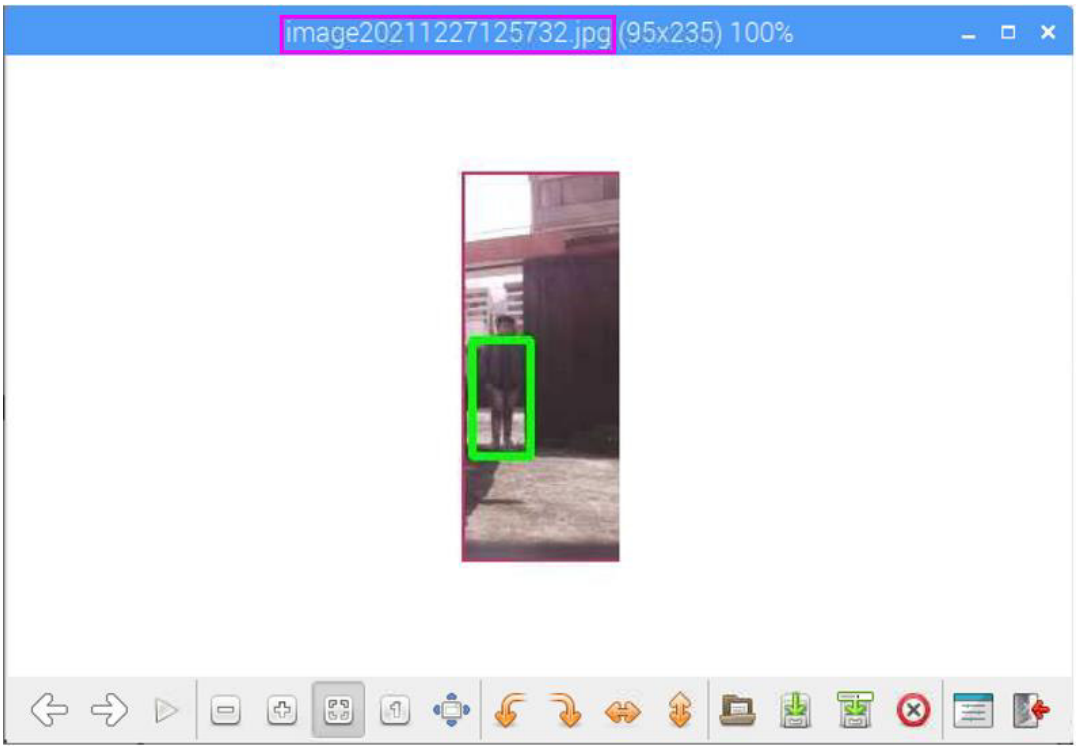

Figura 3.13 Almacenamiento local - Intruso Entrada del Parqueadero.

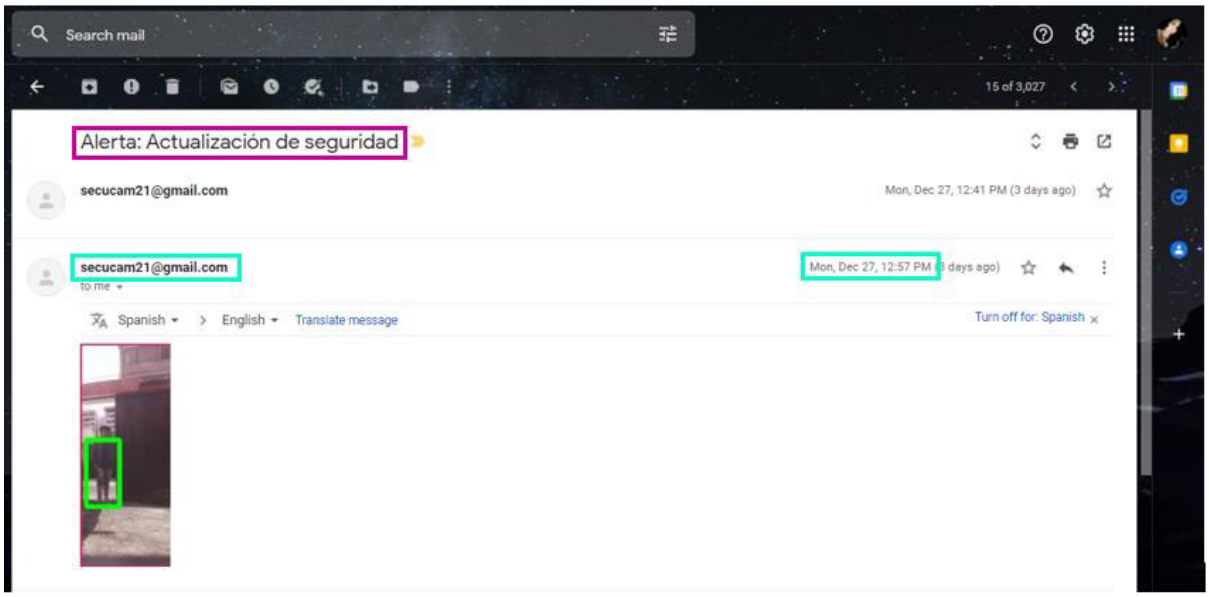

Figura 3.14 Correo de alerta - Entrada del Parqueadero.

Al ser alertas generadas el mismo día y bajo el mismo asunto de correo, la visualización en la bandeja de entrada luce como la Figura 3.14

Se ingresa el link *http://3.21.178.11/detevent.php* para consultar el detalle de los eventos, como se muestra en la Figura 3.15

Figura 3.15 Detalle de los Eventos - Entrada del Parqueadero.

La consulta de la galería se realiza ingresando el link *http://3.21.178.11/galeria.php* colocamos el cursor sobre la imagen para visualizar el nombre de la imagen, como se muestra en la Figura 3.16

Figura 3.16 Galería de imágenes - Entrada del Parqueadero.

3.1.3 ENTRADA TRASERA

Por último, en este escenario se seleccionó como ROI el *"Área de Interés 2"* y se ejecutó el script desde el terminal. La Figura 3.17 muestra el script ejecutándose y en la Figura 3.18 se visualiza el posicionamiento del ROI en la imagen total.

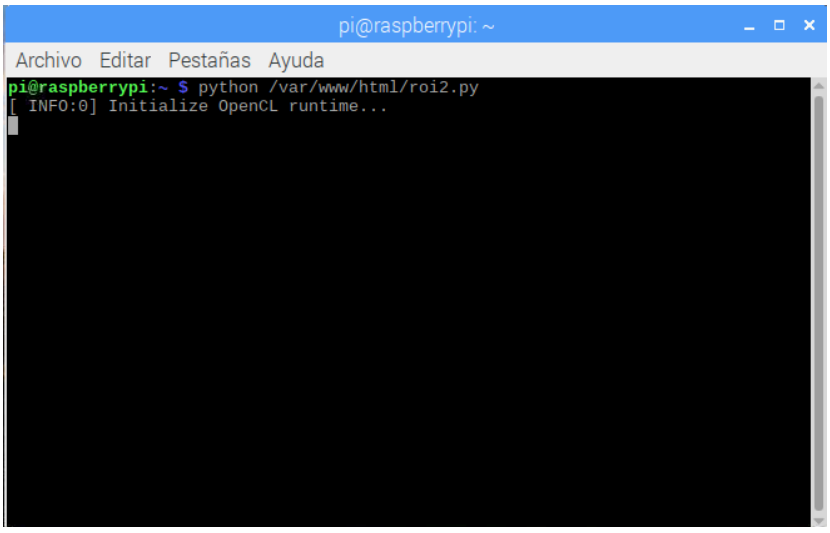

Figura 3.17 Ejecución del Script - Escenario 3.

Figura 3.18 Entrada Trasera.

Al ejecutar el script, el proceso es exactamente igual al expuesto en los dos escenarios anteriores. Inicia con un continuo procesamiento de la imagen capturada por la cámara en tiempo real, al detectar un intruso (*"objeto extraño"*) dentro del área de interés se enmarca este objeto dentro de un rectángulo de color verde y la imagen es almacenada localmente. Dicha imagen se envía al email, al servidor y se inserta la "Fecha" y "Hora" de esta alerta en la base de datos. En la Figura 3.19 se visualiza lo expuesto mediante prints en el terminal de la Rasberry.

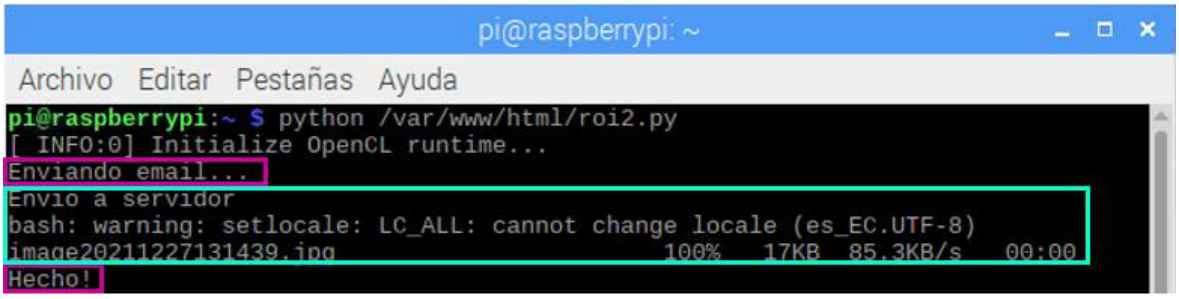

Figura 3.19 Detección de intruso - Entrada Trasera.

En la Figura 3.20 se visualiza la imagen almacenada localmente y la Figura 3.21 muestra el correo de alerta recibido y como se mencionó anteriormente al ser alertas generadas el mismo día con el mismo asunto de correo, esta alerta se suma al hilo de correos existente.

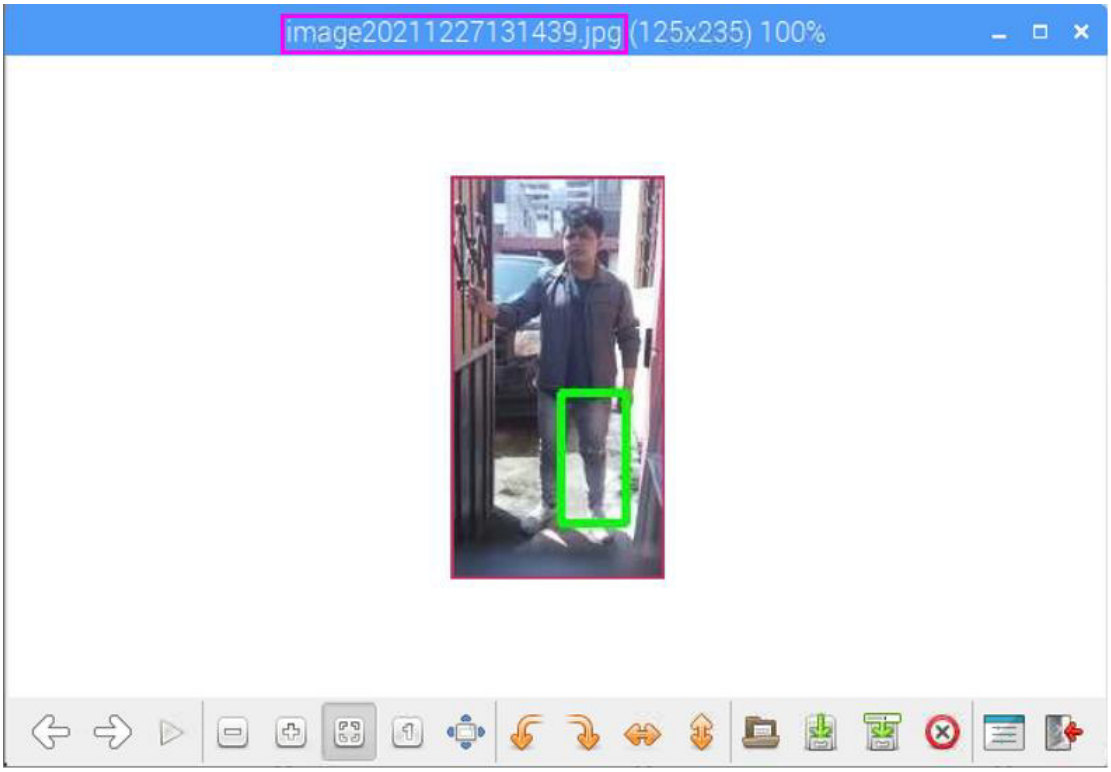

Figura 3.20 Almacenamiento local - Intruso Entrada Trasera.

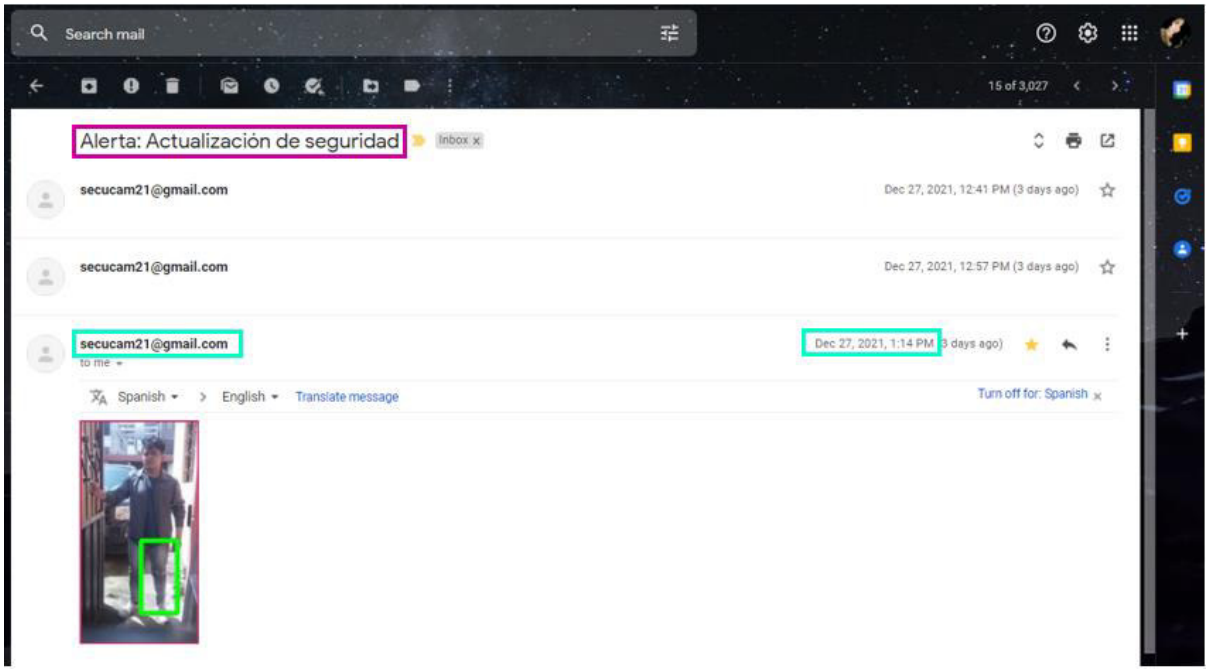

Figura 3.21 Correo de alerta - Entrada Trasera.

Para consultar el detalle de los eventos se ingresa el link *http://3.21.178.11/detevent.php*, como se muestra en la Figura 3.22

Figura 3.22 Detalle de los Eventos - Entrada Trasera.

Para consultar la galería se ingresa en el link *http://3.21.178.11/galeria.php* y colocamos el cursor sobre la imagen para visualizar el nombre de la imagen, como se muestra en la Figura 3.23

Figura 3.23 Galería de imágenes - Entrada Trasera.

3.2 ANÁLISIS DE RESULTADOS

Se verificó mediante los datos insertados en la base de datos y el identificador único que se genera como nombre de la imagen, que el tiempo aproximado entre la detección del intruso y el proceso de almacenamiento local, envío al servidor, envío de email e inserción de datos en la DB; toma entre 4 y 5 segundos. Este tiempo varía debido a la estabilidad de conexión de la red y la distancia que existe entre el nodo y el módem de la red local, lo que resulta en un tiempo de *"delay"* aceptable para que el usuario sea alertado del evento.

Al considerar que el principio de funcionamiento de una cámara es capturar la luz reflejada de los objetos para su posterior análisis o tratamiento, se debe tener en consideración la iluminación con la que se cuente en el área a monitorear. Este es un factor de relevancia ya que permite optimizar el contraste, normalizar cualquier variación del nivel de luz ambiente, debido a que las condiciones nunca son homogéneas [37].

Si bien los escenarios de prueba que fueron definidos contaban con iluminación ambiente, mediante un escenario de parqueadero cerrado se constató la importancia de la iluminación y al no contar con un buen nivel de luz se tuvo que adaptar el posicionamiento de la Raspberry dentro del escenario. La Figura 3.24 y la Figura 3.25 muestra la detección exitosa del intruso a una distancia muy cercana del área a monitorear, lo cual desde un punto de vista técnico no permite optimizar la funcionalidad del proyecto.

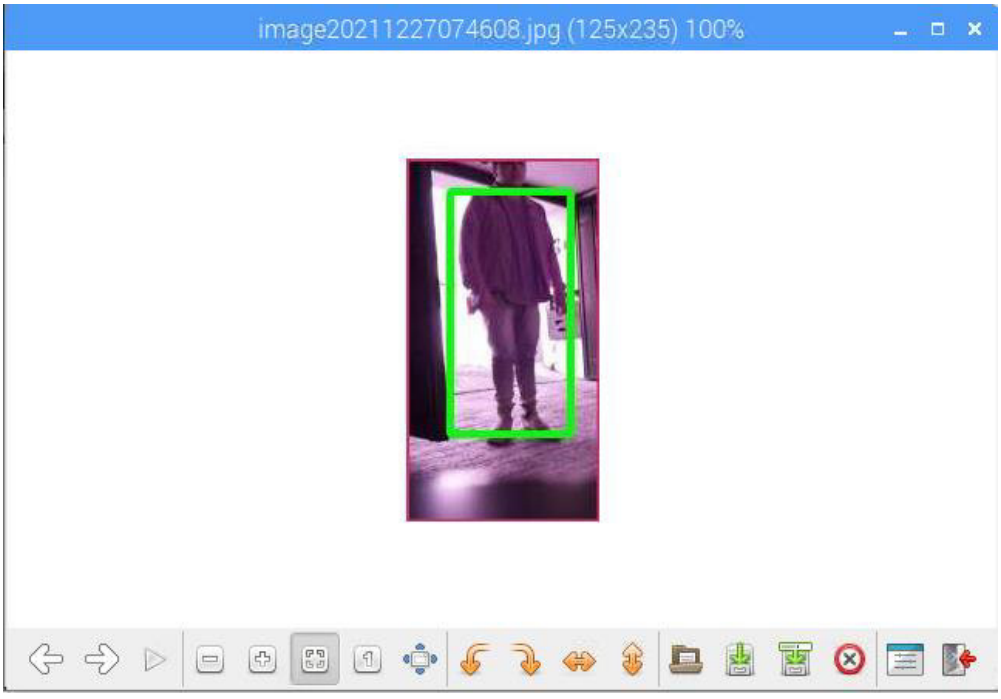

Figura 3.24 Detección de intrusos (1) - Escenario con poca iluminación.

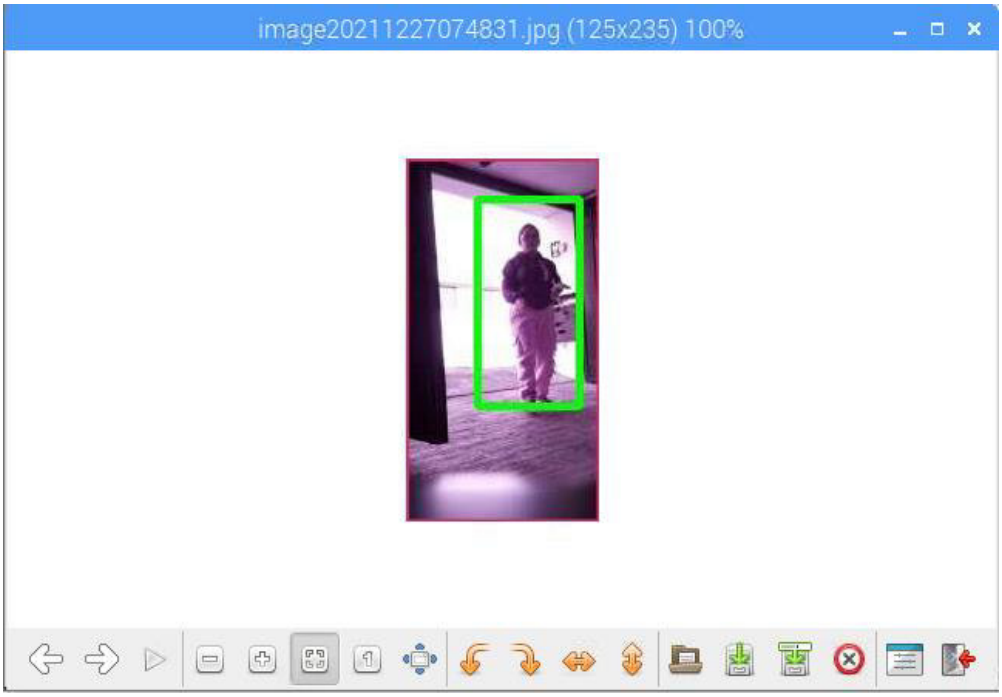

Figura 3.25 Detección de intrusos (2) - Escenario con poca iluminación.

4. CONCLUSIONES Y RECOMENDACIONES

4.1. CONCLUSIONES

El presente proyecto permite fortalecer los mecanismos de seguridad residencial a nivel nacional, ya que el incremento de los niveles de delincuencia en el país, deja cada vez más expuestos y vulnerables a los usuarios residenciales. La inserción de un mecanismo de seguridad que alerte al usuario final dentro de los primeros segundos del evento, contribuye a preservar la integridad personal de los usuarios y la posibilidad de tomar acciones acertadas ante tal vicisitud.

El contar con tecnologías tanto de software y hardware que permiten aprovechar al máximo la inserción de las mismas en el día a día de los usuarios, resulta en aplicaciones de funcionalidad exitosa. Se implementó un nodo que detecta intrusos en residencias, integrando hardware y software open source que permite contribuir en el proceso de alerta temprana.

La Raspberry Pi 3B+ en conjunto con su sistema operativo y la integración del módulo de cámara, dio como resultado un sistema robusto y escalable.

La cámara presentó ciertas limitaciones en el proceso de detección de intrusos en escenarios con bajos niveles de iluminación, por lo que se puede considerar la ocupación de cámaras de mejores características.

Acorde a las pruebas realizadas se concluye que los factores de iluminación son de gran importancia, debido a que influyen en el proceso de detección y un mismo evento bajo condiciones ambientales distintas pueden resultar en detecciones un poco tardías.

Las pruebas de funcionamiento se realizaron en escenarios donde la cobertura de la red WiFi utilizada resultó óptima, lo cual permitió evitar la utilización de cableado adicional al nodo y mantener un enfoque de libre movilización.

Mediante el lenguaje de programación de alto nivel “Python” y el uso de las librerías de Open CV, se pudo garantizar que la Raspberry opere dentro de niveles óptimos de consumo de CPU y cumpla con el objetivo principal de detección y alerta.

El uso de imágenes como mecanismo de constatación visual mediante el aplicativo web y el correo del usuario, permite que se tenga una visualización clara del evento en curso, dentro del área de interés monitoreada.

Al generar alarmas de detección de intrusos vía email, el usuario puede comprobar mediante la aplicación web los detalles del evento en virtud de la fecha y hora de registro. Y a pesar de que las fotografías son de baja resolución permiten efectivamente discriminar la presencia de un intruso en el área de interés.

Se optó por desarrollar el módulo de reportes con un enfoque de acceso público, mientras el módulo de configuración se lo maneja a nivel privado (local), con el fin de que estos parámetros iniciales sean de acceso limitado.

4.2. RECOMENDACIONES

Implementar un sistema de iluminación para poder corregir la variación de nivel de luz según el escenario, considerando a los factores ambientales y de interés del usuario.

Integrar sensores que permitan automatizar el momento en el cual se deba activar la iluminación de apoyo, sensores que controlen el acceso a la residencia (apertura y cierre de puertas) y sensores de movimiento como apoyo para la detección de intrusos.

Optar por la implementación de una cámara de mejores características, la cual mejore el nivel de resolución de las imágenes captadas, con el fin de obtener con mayor detalle las características físicas del intruso.

5. REFERENCIAS BIBLIOGRÁFICAS

[1] Ministerio del Interior, "Indicadores de Seguridad Ciudadana," 2017. [En línea]. Available: http://cifras.ministeriodelinterior.gob.ec/comisioncifras/inicio.php. [Último acceso: 08 2019].

[2] Servicio Integrado de Seguridad ECU 911, "Sistema de video vigilancia," 2017. [En línea]. Available: http://www.ecu911.gob.ec/videovigilancia/. [Último acceso: 2 Agosto 2021].

[3] M. Peter y H. David, Learn Raspberry Pi with Linux, 2015.

[4] Raspberry Pi Foundation, "Raspberry Pi Zero," 2015. [En línea]. Available: https://www.raspberrypi.org/. [Último acceso: 2 Agosto 2019].

[5] OpenCV, 2021. [En línea]. Available: https://opencv.org/about/. [Último acceso: 13 Agosto 2021].

[6] Ministerio del Interior, "Plan Nacional de Seguridad Ciudadana y Convivencia Social," 2019.

[7] Paessler, "CCTV," 2021. [En línea]. [Último acceso: 13 Agosto 2021].

[8] Superinventos, "Sistemas de VideoVigilancia," 9 Agosto 2021. [En línea]. Available: http://www.superinventos.com/sistemas_videovigilancia.htm. [Último acceso: 13 Agosto 2021].

[9] Security Magazine, "Legacy CCTV vs Ip Cameras," 2021. [En línea]. Available: https://www.securitymagazine.com/articles/94565-legacy-cctv-vs-ip-cameras-the-differences-for-security-program-design-and-development. [Último acceso: 13 Agosto 2021].

[10] F. J. García Mata, Introducción al Video IP, Vértice, 2010, pp. 11-14.

[11] S. Martí, "Diseño de un sistema de televigilancia sobre IP para el edificio CRAI de la Escuela Politécnica Superior de Gandía," Universidad Politécnica de Valencia , Gandía-España, 2013.

[12] I. T. Young , J. J. Gerbrands y L. J. Van Vliet, Fundamentals of Image Processing, Delft: Delft University of Technology, 1997.

[13] Raspberry Pi Foundation, "Raspberry Pi 2 Model B," 17 Enero 2017. [En línea]. Available: https://www.raspberrypi.org/products/raspberry-pi-2-model-b/. [Último acceso: 13 Agosto 2021].

[14] Raspberry Pi Foundation, "raspberrypi.org," Noviembre 2018. [En línea]. Available: https://datasheets.raspberrypi.org/rpi3/raspberry-pi-3-a-plus-product-brief.pdf. [Último acceso: 13 Agosto 2021].

[15] Raspberry Pi Foundation, "raspberrypi.org," Marzo 2018. [En línea]. Available: https://datasheets.raspberrypi.org/rpi3/raspberry-pi-3-b-plus-product-brief.pdf. [Último acceso: 13 Agosto 2021].

[16] Raspberry Pi Trading Ltd., "raspberrypi.org," Enero 2021. [En línea]. Available: https://datasheets.raspberrypi.org/rpi4/raspberry-pi-4-product-brief.pdf. [Último acceso: 15 Agosto 2021].

[17] Raspberry Pi Trading Ltd., "raspberrypi.org," Enero 2021. [En línea]. Available: https://datasheets.raspberrypi.org/pico/pico-product-brief.pdf. [Último acceso: 15 Agosto 2021].

[18] Rspberry Pi Foundation, "raspberry.org," 2016. [En línea]. Available: https://www.raspberrypi.org/products/camera-module-v2/. [Último acceso: 15 Agosto 2021].

[19] Raspberry Pi Foundation, "raspberry.org," 07 Mayo 2021. [En línea]. Available: https://www.raspberrypi.org/documentation/computers/os.html. [Último acceso: 15 Agosto 2021].

[20] The Linux Foundation, "linux.com," 2021. [En línea]. Available: https://www.linux.com/what-is-linux/. [Último acceso: 15 Agosto 2021].

[21] The Apache HTTP Server project, "apache.org," 1 Junio 2021. [En línea]. Available: https://httpd.apache.org/. [Último acceso: 15 Agosto 2021].

[22] Oracle, "dev.mysql.com," 2021. [En línea]. Available: https://dev.mysql.com/doc/refman/8.0/en/what-is-mysql.html. [Último acceso: 13 Agosto 2021].

[23] PHP, "php.net," 2001. [En línea]. Available: https://www.php.net/manual/es/intro-whatis.php. [Último acceso: 13 Agosto 2021].

[24] ingenierovirtual, "Conceptos básicos sobre tecnologías de desarrollo web," [En línea]. Available: https://www.ingeniovirtual.com/conceptos-basicos-sobre-tecnologias-de-desarrollo-web/. [Último acceso: 13 Agosto 2021].

[25] A. I. Wasserman, Principles for the Design of Web Applications, Moffett Field: COSI, 2002.

[26] G. Booch, Object-Oriented Analysis and Design, Redwood: Bejamin/Cummings, 1994.

[27] Microsoft, "Azure Database Storage," Microsoft, 2020. [En línea]. Available: https://azure.microsoft.com/es-es/product-categories/databases/. [Último acceso: 14 Agosto 2021].

[28] Raspberry Pi Foundation, "raspberrypi.org," Mayo 2018. [En línea]. Available: https://www.raspberrypi.org/about/. [Último acceso: 22 Septiembre 2021].

[29] Raspberry Pi Foundation, "raspberrypi.org," Agosto 2018. [En línea]. Available: https://www.raspberrypi.org/products/raspberry-pi-3-model-b-plus/. [Último acceso: 22 Septiembre 2021].

[30] OpenCV, "OpenCV: Clasificador en cascada," [En línea]. Available: https://docs.opencv.org/3.4/db/d28/tutorial_cascade_classifier.html. [Último acceso: 22 Septiembre 2021].

[31] Amazon Web Services, "About AWS," [En línea]. Available: https://aws.amazon.com/es/what-is-aws/. [Último acceso: 22 Septiembre 2021].

[32] Modulr, "Tutorials for Developers Laravel/AWS/Linux," [En línea]. Available: https://medium.com/modulr/como-instalar-lamp-stack-linux-apache-mysql-php-en-una-instancia-ec2-de-aws-con-ubuntu-16-04-bcf72b5d9937. [Último acceso: 22 Septiembre 2021].

[33] Oracle Corporation, "MySQL :: MySQL 5.7 Reference Manual :: 6.2.7 Adding Accounts, Assigning Privileges, and Dropping Accounts," [En línea]. Available:

https://dev.mysql.com/doc/refman/5.7/en/creating-accounts.html. [Último acceso: 22 Septiembre 2021].

[34] Oracle Corporation, "MySQL :: MySQL 5.7 Reference Manual :: 6.2.2 Privileges Provided by MySQL," [En línea]. Available: https://dev.mysql.com/doc/refman/5.7/en/privileges-provided.html. [Último acceso: 22 Septiembre 2021].

[35] Amazon Web Services, "Reglas de grupo de seguridad para diferentes casos de uso - Amazon Elastic Compute Cloud," [En línea]. Available: https://docs.aws.amazon.com/es_es/AWSEC2/latest/UserGuide/security-group-rules-reference.html. [Último acceso: 22 Septiembre 2021].

[36] ANYELGUTI, "Bases en la web: Conectar con MySQL," [En línea]. Available: https://aprende-web.net/php/php14_1.php. [Último acceso: 22 Septiembre 2021].

[37] "Aplicación práctica de la visión artificial en el control de procesos industriales. Conocimientos básicos de visión artificial.," Abril 2012. [En línea]. Available: http://www.infoplc.net/files/documentacion/vision_artificial/infoPLC_net_Conocimientos_vISIONARTIFICIAL.pdf. [Último acceso: 22 Septiembre 2021].

ANEXOS

ANEXO A. CÓDIGO FUENTE SCRIPT ROI 1 (Área de interés 1).

ANEXO B. CÓDIGO FUENTE SCRIPT ROI 2 (Área de interés 2).

ANEXO C. CÓDIGO FUENTE SCRIPT ROI 3 (Área de interés 3).

ANEXO D. CREACIÓN DE INSTANCIA AWS - EC2.

ANEXO E. MANUAL DE CONFIGURACIÓN.

ANEXO A

CÓDIGO FUENTE SCRIPT ÁREA DE INTERÉS UNO

```
# Importar libererías

import cv2

import sys

import numpy as np

from matplotlib import pyplot as plt

import MySQLdb

import os

from curses import ascii

import base64

import smtplib

from email.mime.multipart import MIMEMultipart

from email.mime.text import MIMEText

from email.mime.image import MIMEImage

import time

import threading


# Correo electrónico desde el que desea enviar la actualización

fromEmail = 'secucam21@gmail.com'

fromEmailPassword = 'R4sp.mailB'

# Correo electrónico al que desea enviar la actualización

# Lectura archivo registro correo

f = open ('/var/www/html/regcorreo.txt','r')

correodest = f.read()
```

```
toEmail = correodest

#FUNCION PARA ALMACENAR DATOS EN MySQL
def run_query(query=''):
    conectar=MySQLdb.connect("3.21.178.11","admin","RMpa.ad21","Monitoreo")
    cursor=conectar.cursor() #cursor
    cursor.execute(query) #Ejecuta una consulta SQL
    conectar.commit() #finaliza transaccion
    data = None
    cursor.close()
    conectar.close()

# Variables
object_classifier =
cv2.CascadeClassifier("/var/www/html/models/fullbody_recognition_model.xml")
scaling_factor = 0.5
email_update_interval = 30

cam_capture = cv2.VideoCapture(0)
cv2.destroyAllWindows()

# ROI 1, posicion definida
upper_left = (5, 5)
bottom_right = (200, 475)

last_epoch = 0
```

```
# Proceso de detección de intrusos
while True:
    _, image_frame = cam_capture.read()

    # Posicionamieto del ROI en la imagen inicial
    r = cv2.rectangle(image_frame, upper_left, bottom_right, (100, 50, 200), 5)
    rect_img = image_frame[upper_left[1] : bottom_right[1], upper_left[0] : bottom_right[0]]

    # Para detección de movimiento, procesamiento de imagen
    found_objects = False
    rect_img = cv2.resize(rect_img, None, fx=scaling_factor, fy=scaling_factor,
interpolation=cv2.INTER_AREA)
    gray = cv2.cvtColor(rect_img, cv2.COLOR_BGR2GRAY)

    objects = object_classifier.detectMultiScale(
            gray,
            scaleFactor=1.1,
            minNeighbors=5,
            minSize=(30, 30),
            flags=cv2.CASCADE_SCALE_IMAGE
    )

    if len(objects) > 0:
            found_objects = True
```

```
    # Establecimiento de un rectángulo alrededor del intruso detectado.
    for (x, y, w, h) in objects:
        cv2.rectangle(rect_img, (x, y), (x + w, y + h), (0, 255, 0), 3)

        # Identificación y almacenamiento de la imagen procesada (cuando se detectó un intruso)
        tife=time.strftime('%Y%m%d%H%M%S')
        prefijo="image"
        path="/var/www/html/Images"
        n_local=path+"/"+prefijo+tife+".jpg"

    #IMAGEN COMPLETA QUE CAPTURA LA CÁMARA, CON ROI DIBUJADO
    cv2.imshow("CamSecurity ROI 1", image_frame)

    # Definición para envío de alerta
    if found_objects and (time.time() - last_epoch) > email_update_interval:
        last_epoch = time.time()
        print ('Enviando email...')

        # Almacenamiento local de la imagen procesada
        cv2.imwrite(n_local, rect_img)

        # Envio de imagen al servidor AWS-EC2 mediante SCP
        print ('Envio a servidor')
        os.system('sudo scp -r -i /var/www/html/SVR_AWS/servidoraws2021.pem "%s" ubuntu@3.21.178.11:/var/www/html/Imagenes_eventos'%n_local)
```

```
# Parámetros de envío
msgRoot = MIMEMultipart('related')
msgRoot['Subject'] = 'Alerta: Actualización de seguridad'
msgRoot['From'] = fromEmail
msgRoot['To'] = toEmail
msgRoot.preamble = 'Actualización de la cámara de seguridad de la Raspberry Pi'

msgAlternative = MIMEMultipart('alternative')
msgRoot.attach(msgAlternative)
msgText = MIMEText('Posible intruso detectado por cámara de seguridad')
msgAlternative.attach(msgText)

msgText = MIMEText('<img src="cid:image1">', 'html')
msgAlternative.attach(msgText)

image = open(n_local, 'rb').read()
msgImage = MIMEImage(image)
msgImage.add_header('Content-ID', '<image1>')
msgRoot.attach(msgImage)

smtp = smtplib.SMTP('smtp.gmail.com', 587)
smtp.starttls()
smtp.login(fromEmail, fromEmailPassword)
smtp.sendmail(fromEmail, toEmail, msgRoot.as_string())
```

```
        smtp.quit()


        print ('Hecho!')


        # Inserción de Datos en la DB
        Fecha1=time.strftime('%Y-%m-%d')
        Hora1=time.strftime('%H:%M:%S')
        query=("INSERT INTO Eventos" "(Fecha,Hora)" "VALUES ('%s','%s'
)")%(Fecha1,Hora1)
        run_query(query)


    #Presionar ENTER para salir
    if cv2.waitKey(1) == 13:
        break


cam_capture.release()
cv2.destroyAllWindows()
```

ANEXO B

CÓDIGO FUENTE SCRIPT ÁREA DE INTERÉS DOS

```
# Importar libererías

import cv2

import sys

import numpy as np

from matplotlib import pyplot as plt

import MySQLdb

import os

from curses import ascii

import base64

import smtplib

from email.mime.multipart import MIMEMultipart

from email.mime.text import MIMEText

from email.mime.image import MIMEImage

import time

import threading


# Correo electrónico desde el que desea enviar la actualización

fromEmail = 'secucam21@gmail.com'

fromEmailPassword = 'R4sp.mailB'

# Correo electrónico al que desea enviar la actualización

# Lectura archivo registro correo

f = open ('/var/www/html/regcorreo.txt','r')

correodest = f.read()
```

```
toEmail = correodest

#FUNCION PARA ALMACENAR DATOS EN MySQL
def run_query(query=''):
    conectar=MySQLdb.connect("3.21.178.11","admin","RMpa.ad21","Monitoreo")
    cursor=conectar.cursor() #cursor
    cursor.execute(query) #Ejecuta una consulta SQL
    conectar.commit() #finaliza transaccion
    data = None
    cursor.close()
    conectar.close()

# Variables
object_classifier =
cv2.CascadeClassifier("/var/www/html/models/fullbody_recognition_model.xml")
scaling_factor = 0.5
email_update_interval = 30

cam_capture = cv2.VideoCapture(0)
cv2.destroyAllWindows()

# ROI 2, posicion definida
upper_left = (200, 5)
bottom_right = (450, 475)

last_epoch = 0
```

```
# Proceso de detección de intrusos
while True:
    _, image_frame = cam_capture.read()

    # Posicionamieto del ROI en la imagen inicial
    r = cv2.rectangle(image_frame, upper_left, bottom_right, (100, 50, 200), 5)
    rect_img = image_frame[upper_left[1] : bottom_right[1], upper_left[0] : bottom_right[0]]

    # Para detección de movimiento, procesamiento de imagen
    found_objects = False
    rect_img = cv2.resize(rect_img, None, fx=scaling_factor, fy=scaling_factor,
interpolation=cv2.INTER_AREA)
    gray = cv2.cvtColor(rect_img, cv2.COLOR_BGR2GRAY)

    objects = object_classifier.detectMultiScale(
            gray,
            scaleFactor=1.1,
            minNeighbors=5,
            minSize=(30, 30),
            flags=cv2.CASCADE_SCALE_IMAGE
    )

    if len(objects) > 0:
            found_objects = True
```

```
    # Establecimiento de un rectángulo alrededor del intruso detectado.
    for (x, y, w, h) in objects:
        cv2.rectangle(rect_img, (x, y), (x + w, y + h), (0, 255, 0), 3)

        # Identificación y almacenamiento de la imagen procesada (cuando se detectó un
intruso)
        tife=time.strftime('%Y%m%d%H%M%S')
        prefijo="image"
        path="/var/www/html/Images"
        n_local=path+"/"+prefijo+tife+".jpg"

    #IMAGEN COMPLETA QUE CAPTURA LA CÁMARA, CON ROI DIBUJADO
    cv2.imshow("CamSecurity ROI 2", image_frame)

    # Definición para envío de alerta
    if found_objects and (time.time() - last_epoch) > email_update_interval:
        last_epoch = time.time()
        print ('Enviando email...')

        # Almacenamiento local de la imagen procesada
        cv2.imwrite(n_local, rect_img)

        # Envio de imagen al servidor AWS-EC2 mediante SCP
        print ('Envio a servidor')
        os.system('sudo scp -r -i /var/www/html/SVR_AWS/servidoraws2021.pem "%s"
ubuntu@3.21.178.11:/var/www/html/Imagenes_eventos'%n_local)
```

```
# Parámetros de envío
msgRoot = MIMEMultipart('related')
msgRoot['Subject'] = 'Alerta: Actualización de seguridad'
msgRoot['From'] = fromEmail
msgRoot['To'] = toEmail
msgRoot.preamble = 'Actualización de la cámara de seguridad de la Raspberry Pi'

msgAlternative = MIMEMultipart('alternative')
msgRoot.attach(msgAlternative)
msgText = MIMEText('Posible intruso detectado por cámara de seguridad')
msgAlternative.attach(msgText)

msgText = MIMEText('<img src="cid:image1">', 'html')
msgAlternative.attach(msgText)

image = open(n_local, 'rb').read()
msgImage = MIMEImage(image)
msgImage.add_header('Content-ID', '<image1>')
msgRoot.attach(msgImage)

smtp = smtplib.SMTP('smtp.gmail.com', 587)
smtp.starttls()
smtp.login(fromEmail, fromEmailPassword)
smtp.sendmail(fromEmail, toEmail, msgRoot.as_string())
```

```
        smtp.quit()


        print ('Hecho!')


        # Inserción de Datos en la DB
        Fecha1=time.strftime('%Y-%m-%d')
        Hora1=time.strftime('%H:%M:%S')
        query=("INSERT INTO Eventos" "(Fecha,Hora)" "VALUES ('%s','%s'
)")%(Fecha1,Hora1)
        run_query(query)


    #Presionar ENTER para salir
    if cv2.waitKey(1) == 13:
        break


cam_capture.release()
cv2.destroyAllWindows()
```

ANEXO C

CÓDIGO FUENTE SCRIPT ÁREA DE INTERÉS TRES

```
# Importar libererías
import cv2
import sys
import numpy as np
from matplotlib import pyplot as plt
import MySQLdb
import os
from curses import ascii
import base64
import smtplib
from email.mime.multipart import MIMEMultipart
from email.mime.text import MIMEText
from email.mime.image import MIMEImage
import time
import threading

# Correo electrónico desde el que desea enviar la actualización
fromEmail = 'secucam21@gmail.com'
fromEmailPassword = 'R4sp.mailB'
# Correo electrónico al que desea enviar la actualización
# Lectura archivo registro correo
f = open ('/var/www/html/regcorreo.txt','r')
correodest = f.read()
toEmail = correodest

#FUNCION PARA ALMACENAR DATOS EN MySQL
def run_query(query=''):
    conectar=MySQLdb.connect("3.21.178.11","admin","RMpa.ad21","Monitoreo")
```

```
    cursor=conectar.cursor() #cursor
    cursor.execute(query) #Ejecuta una consulta SQL
    conectar.commit() #finaliza transaccion
    data = None
    cursor.close()
    conectar.close()

# Variables
object_classifier =
cv2.CascadeClassifier("/var/www/html/models/fullbody_recognition_model.xml")
scaling_factor = 0.5
email_update_interval = 30

cam_capture = cv2.VideoCapture(0)
cv2.destroyAllWindows()

# ROI 3, posicion definida
upper_left = (450, 5)
bottom_right = (700, 475)

last_epoch = 0

# Proceso de detección de intrusos
while True:
    _, image_frame = cam_capture.read()

    # Posicionamieto del ROI en la imagen inicial
    r = cv2.rectangle(image_frame, upper_left, bottom_right, (100, 50, 200), 5)
    rect_img = image_frame[upper_left[1] : bottom_right[1], upper_left[0] : bottom_right[0]]

    # Para detección de movimiento, procesamiento de imagen
    found_objects = False
```

```
    rect_img = cv2.resize(rect_img, None, fx=scaling_factor, fy=scaling_factor,
interpolation=cv2.INTER_AREA)
    gray = cv2.cvtColor(rect_img, cv2.COLOR_BGR2GRAY)

    objects = object_classifier.detectMultiScale(
            gray,
            scaleFactor=1.1,
            minNeighbors=5,
            minSize=(30, 30),
            flags=cv2.CASCADE_SCALE_IMAGE
    )

    if len(objects) > 0:
            found_objects = True

    # Establecimiento de un rectángulo alrededor del intruso detectado.
    for (x, y, w, h) in objects:
        cv2.rectangle(rect_img, (x, y), (x + w, y + h), (0, 255, 0), 3)

        # Identificación y almacenamiento de la imagen procesada (cuando se detectó un
intruso)
        tife=time.strftime('%Y%m%d%H%M%S')
        prefijo="image"
        path="/var/www/html/Images"
        n_local=path+"/"+prefijo+tife+".jpg"

    #IMAGEN COMPLETA QUE CAPTURA LA CÁMARA, CON ROI DIBUJADO
    cv2.imshow("CamSecurity ROI 3", image_frame)

    # Definición para envío de alerta
    if found_objects and (time.time() - last_epoch) > email_update_interval:
        last_epoch = time.time()
```

```
        print ('Enviando email...')

        # Almacenamiento local de la imagen procesada
        cv2.imwrite(n_local, rect_img)

        # Envio de imagen al servidor AWS-EC2 mediante SCP
        print ('Envio a servidor')
        os.system('sudo scp -r -i /var/www/html/SVR_AWS/servidoraws2021.pem "%s"
ubuntu@3.21.178.11:/var/www/html/Imagenes_eventos'%n_local)

        # Parámetros de envío
        msgRoot = MIMEMultipart('related')
        msgRoot['Subject'] = 'Alerta: Actualización de seguridad'
        msgRoot['From'] = fromEmail
        msgRoot['To'] = toEmail
        msgRoot.preamble = 'Actualización de la cámara de seguridad de la Raspberry Pi'

        msgAlternative = MIMEMultipart('alternative')
        msgRoot.attach(msgAlternative)
        msgText = MIMEText('Posible intruso detectado por cámara de seguridad')
        msgAlternative.attach(msgText)

        msgText = MIMEText('<img src="cid:image1">', 'html')
        msgAlternative.attach(msgText)

        image = open(n_local, 'rb').read()
        msgImage = MIMEImage(image)
        msgImage.add_header('Content-ID', '<image1>')
        msgRoot.attach(msgImage)

        smtp = smtplib.SMTP('smtp.gmail.com', 587)
        smtp.starttls()
```

```
        smtp.login(fromEmail, fromEmailPassword)
        smtp.sendmail(fromEmail, toEmail, msgRoot.as_string())
        smtp.quit()

        print ('Hecho!')

        # Inserción de Datos en la DB
        Fecha1=time.strftime('%Y-%m-%d')
        Hora1=time.strftime('%H:%M:%S')
        query=("INSERT INTO Eventos" "(Fecha,Hora)" "VALUES ('%s','%s'
)")%(Fecha1,Hora1)
        run_query(query)

    #Presionar ENTER para salir
    if cv2.waitKey(1) == 13:
        break

cam_capture.release()
cv2.destroyAllWindows()
```

ANEXO D

CREACIÓN DE INSTANCIA AWS - EC2

Para crear una instancia EC2 de los servicios de la nube de Amazon Web Services (AWS) se debe contar con una cuenta de usuario en AWS, donde se registra la información necesaria en caso de que se generen cobros que estén fuera de la capa gratuita.

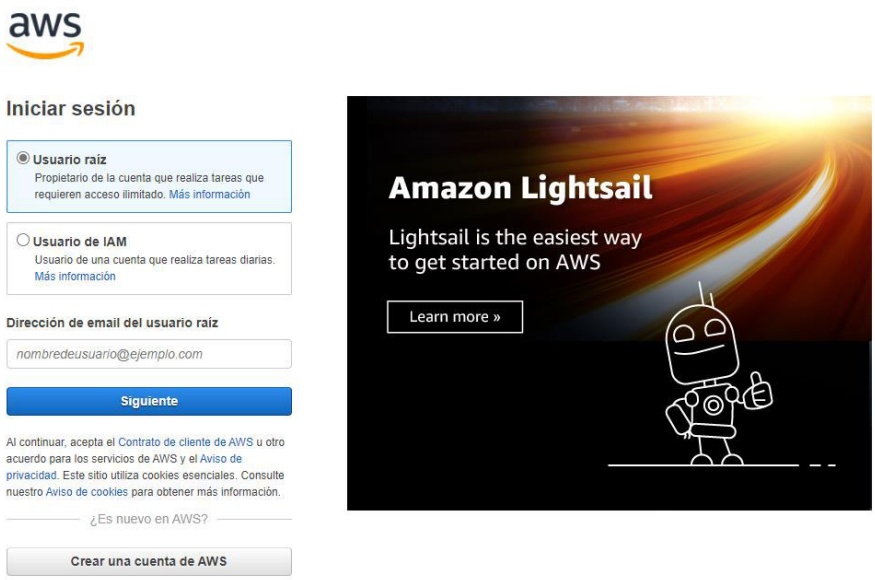

Figura D.1 Página de Login.

Una vez se haya iniciado sesión en la Consola de Administración de AWS, se debe dar clic en la opción *"Launch Services"* para crear y configurar la máquina virtual en donde se albergará el servidor web.

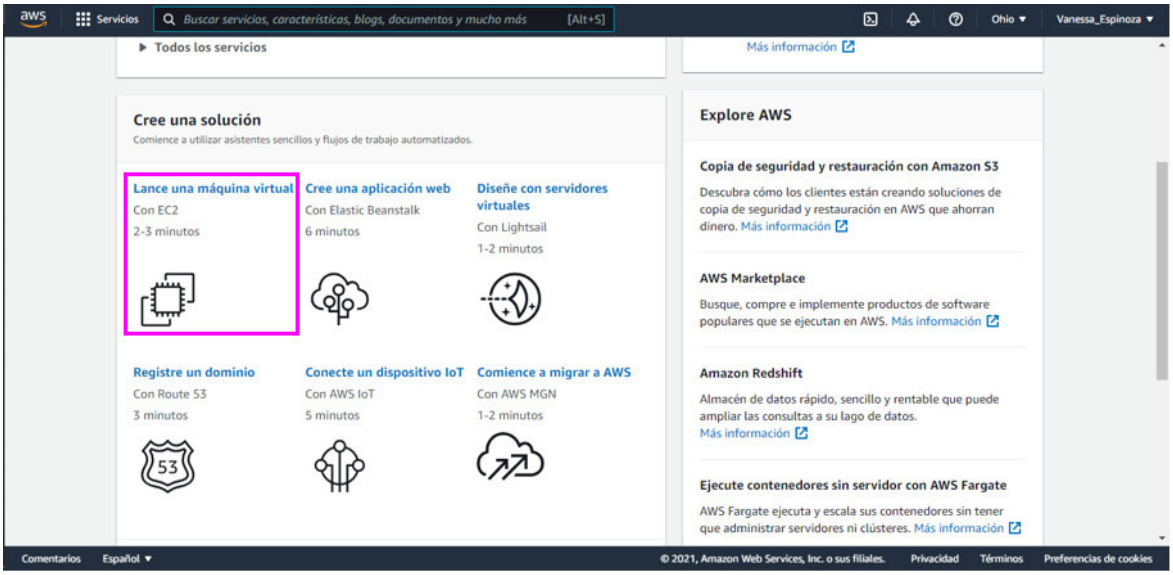

Figura D.2 Lanzamiento de la Instancia.

Seleccionar una imagen de máquinas de Amazon (AMI) que cumpla con las características de software libre, en presente proyecto se opta por la opción Ubuntu Server 16.04 LTS.

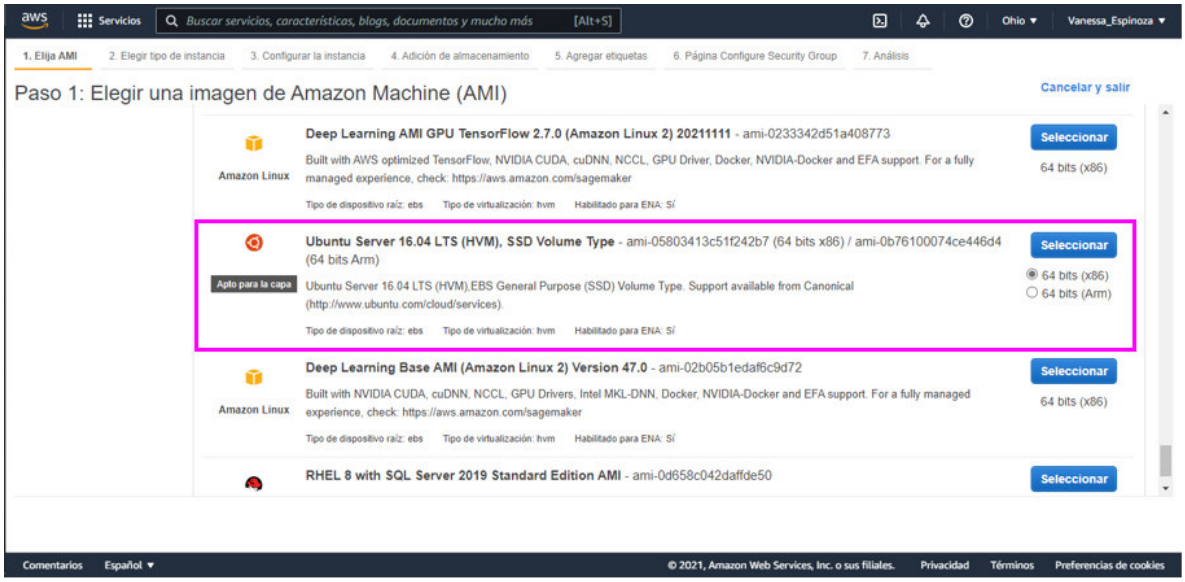

Figura D.3 Selección de imagen de la máquina de Amazon.

Seleccionar la opción de instancia “t2.micro” ya que es la permitida dentro de la capa gratuita.

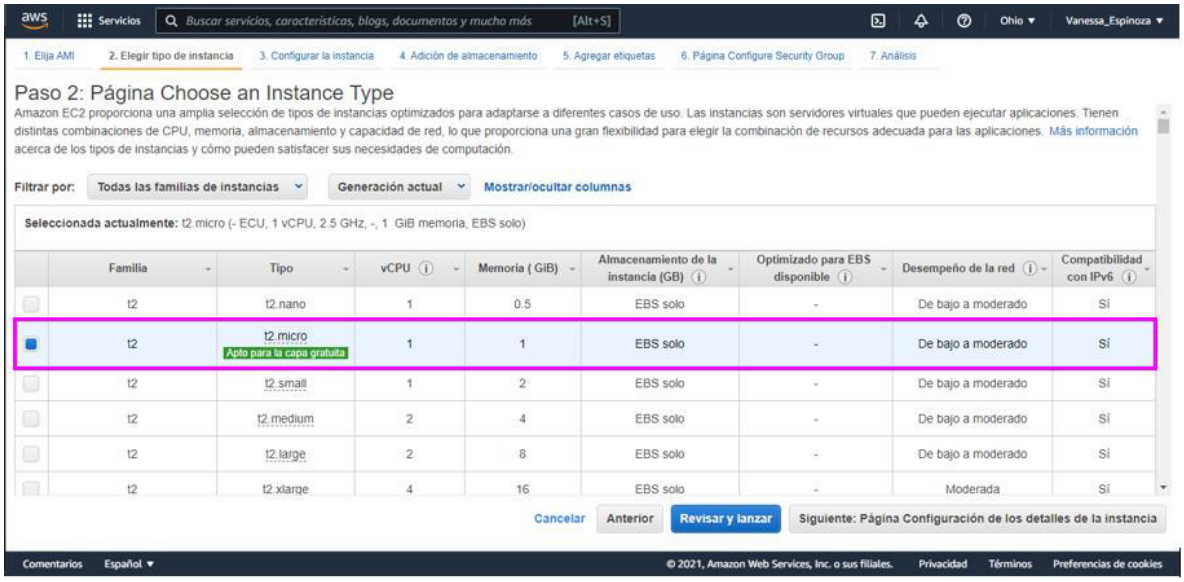

Figura D.4 Selección tipo de instancia.

En “Página Configuración de los detalles de la instancia” no se realiza ninguna configuración.

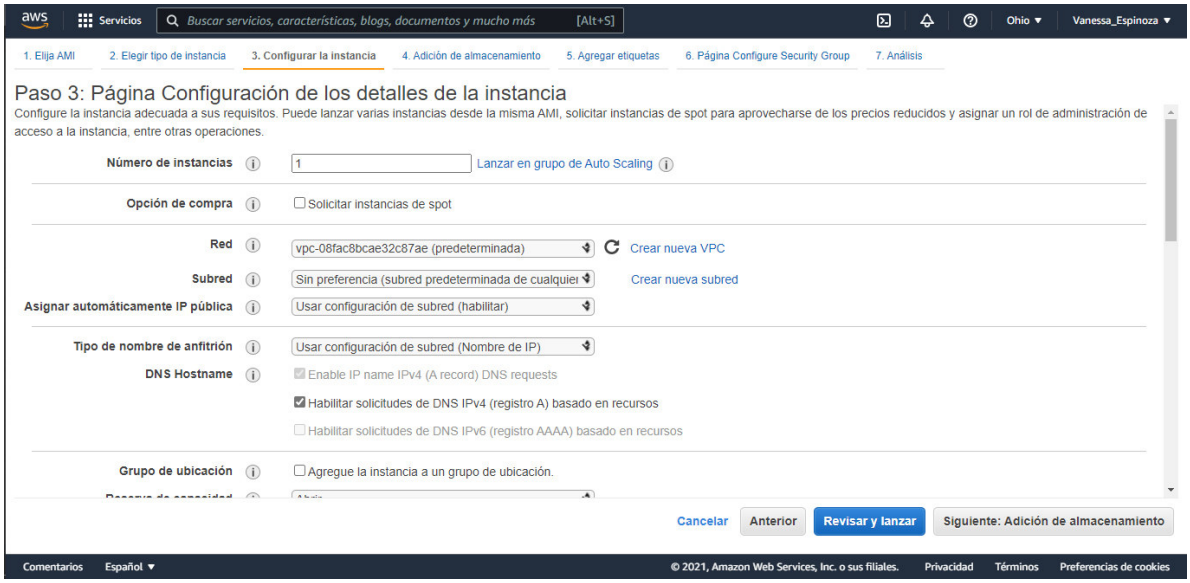

Figura D.5 Detalles de la Instancia.

En la configuración de capacidad de almacenamiento, por defecto se dispone de 8GB los cuales son expandibles hasta 30GB dentro de la capa gratuita.

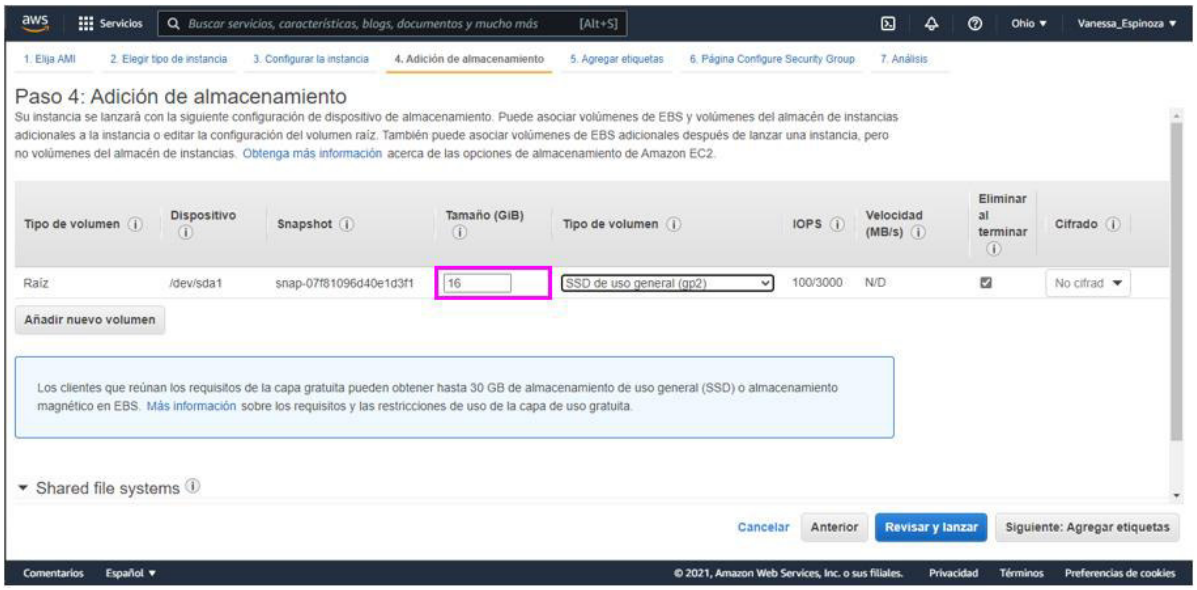

Figura D.6 Configuración de capacidad de almacenamiento.

Asignar un TAG para la instancia creada (opcional).

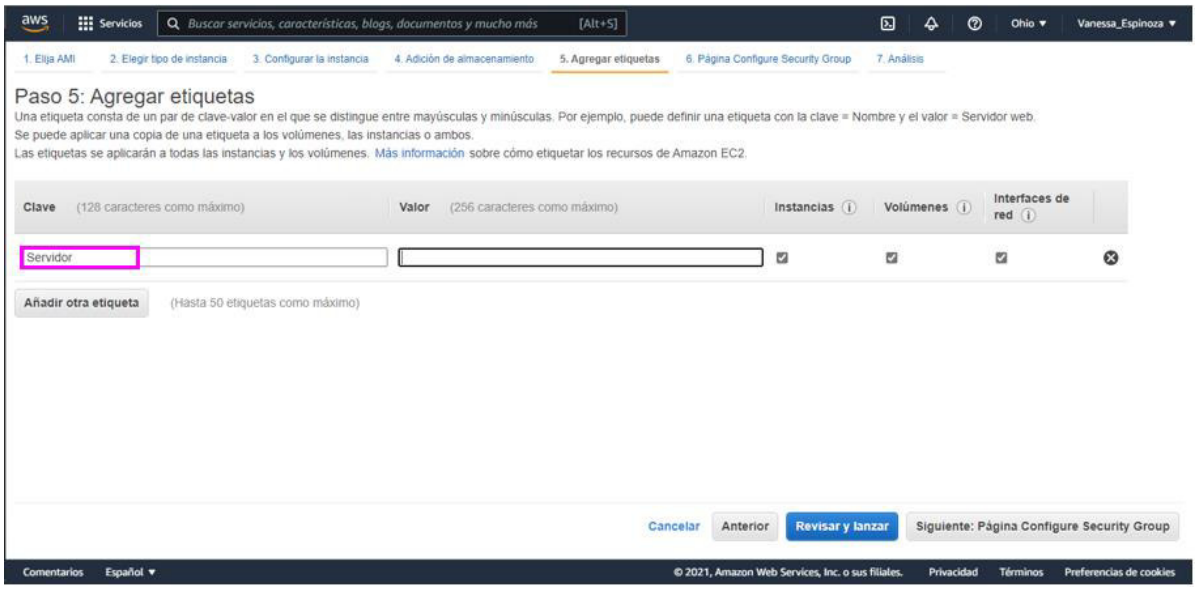

Figura D.7 Asignación TAG a la instancia.

Configurar el firewall virtual *(Security Group)* mediante la habilitación de puertos y direcciones IP que sean requeridos dentro de la instancia.

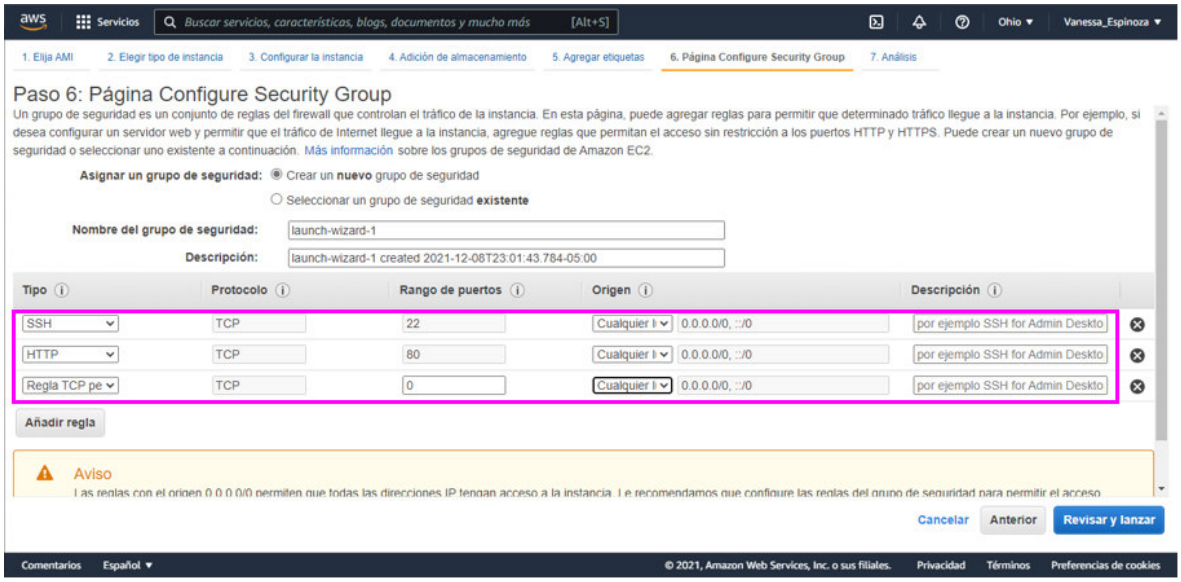

Figura D.8 Configuración Security Group.

Se verifica que estén correctos los parámetros de la instancia para finalmente lanzar la misma.

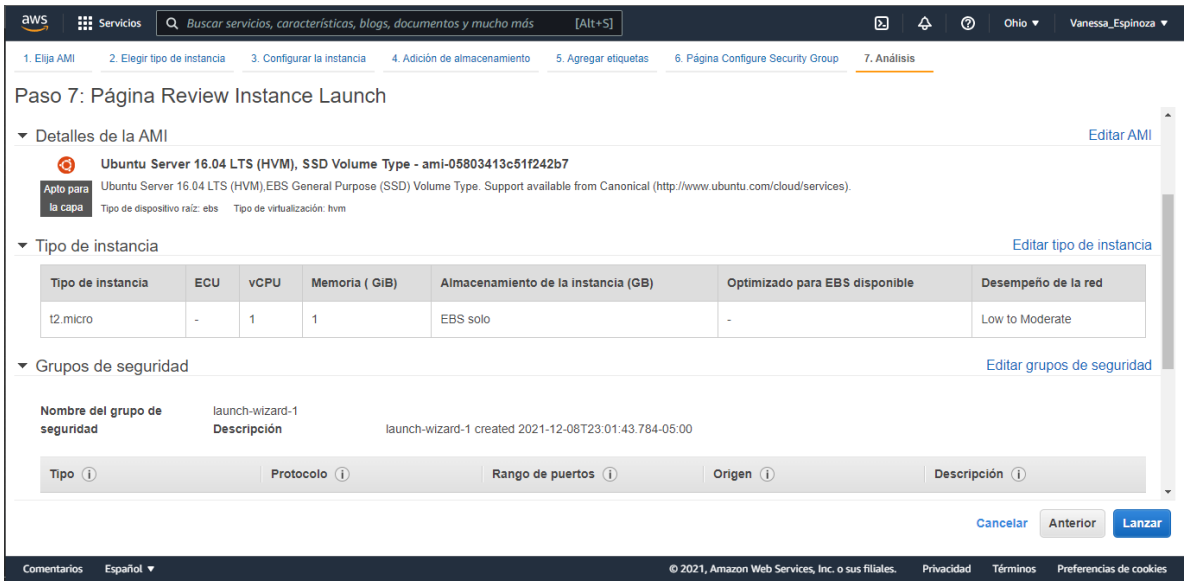

Figura D.9 Lanzamiento de instancia.

Se debe generar una llave digital, la cual permite el acceso al servidor. Esta llave se utilizará posteriormente para llevar a cabo el inicio sesión en la instancia.

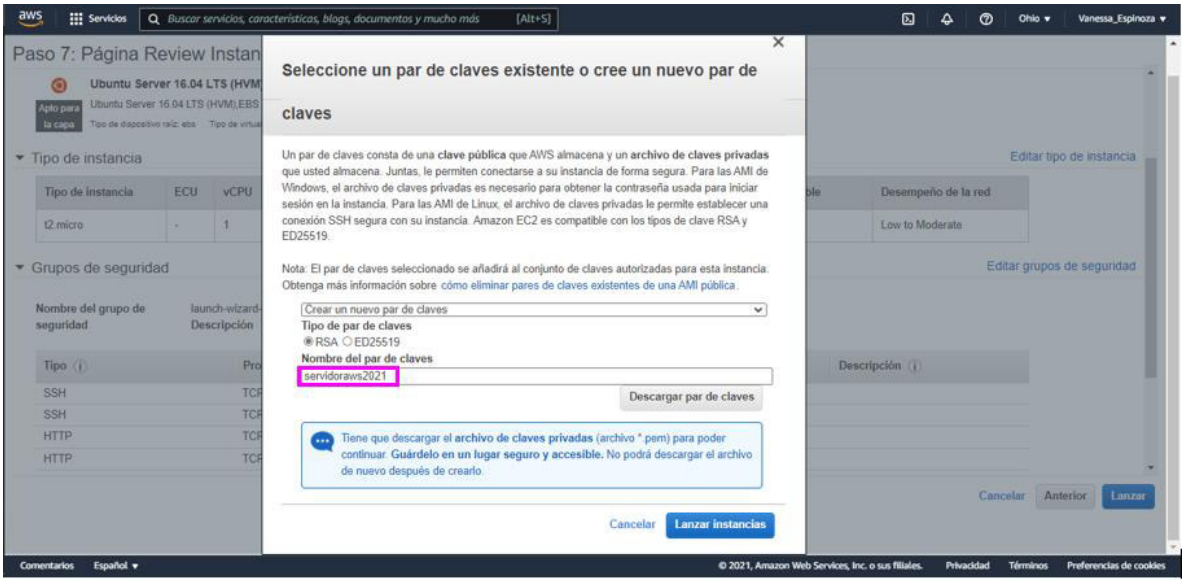

Figura D.10 Generación de llave digital.

Para verificar que la instancia se ha creado correctamente, mediante el panel de la consola se visualiza que el *"Estado de la instancia"* indica *"En ejecución"*.

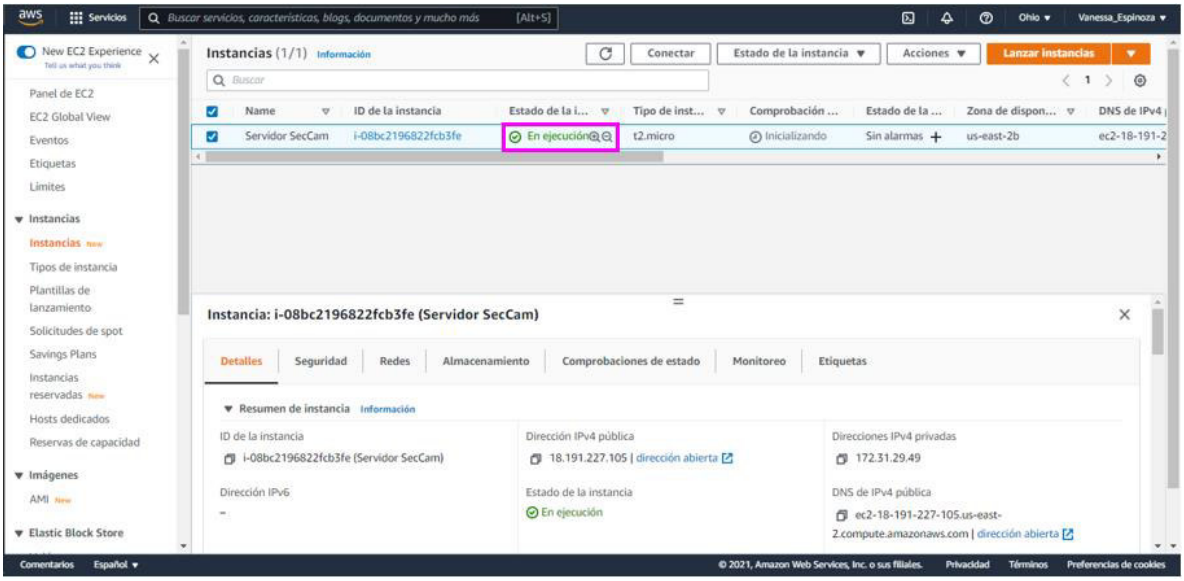

Figura D.11 Estado de la Instancia.

ANEXO E

MANUAL DE CONFIGURACIÓN

Como parte del producto entregado por parte del personal técnico, se ha configurado en su PC, un acceso que permite poner en funcionamiento el proyecto (Figura E.1).

1. Dar clic sobre el acceso directo.

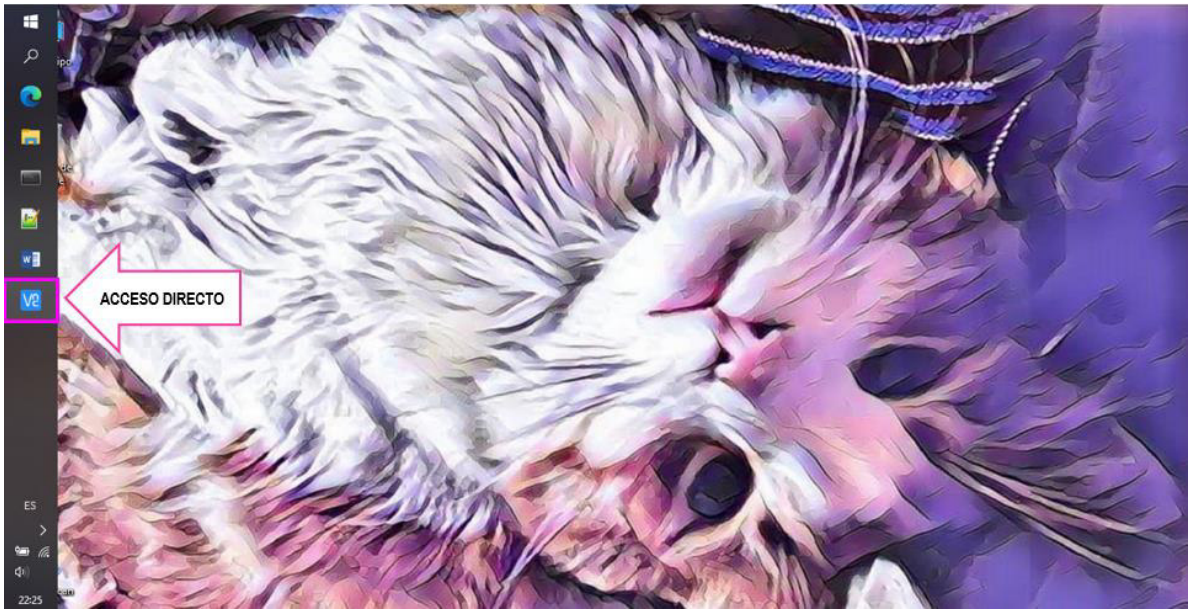

Figura E.1 Acceso al escritorio virtual.

El personal técnico le entregará una dirección IP específica, la cual es igual a la configurada en el acceso al escritorio virtual como muestra la Figura E.2.

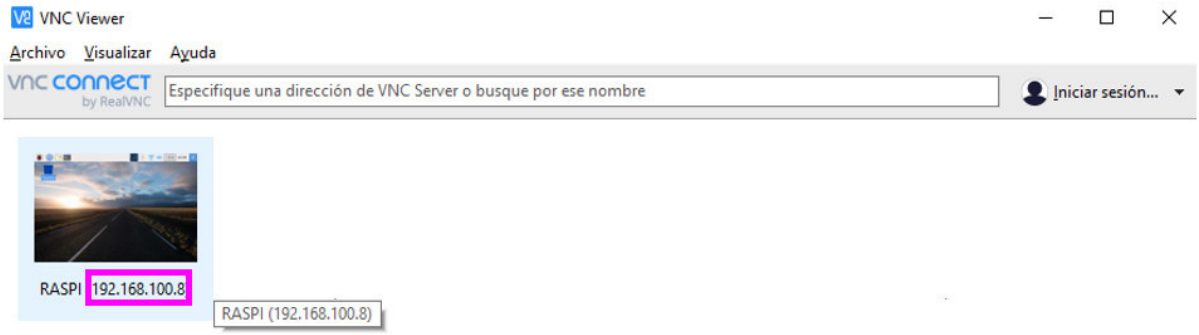

Figura E.2 Dirección IP.

2. Dar doble clic sobre el ícono de la imagen que se muestra en la Figura E.2 para acceder al escritorio de la Raspberry, como se muestra en la Figura E.3

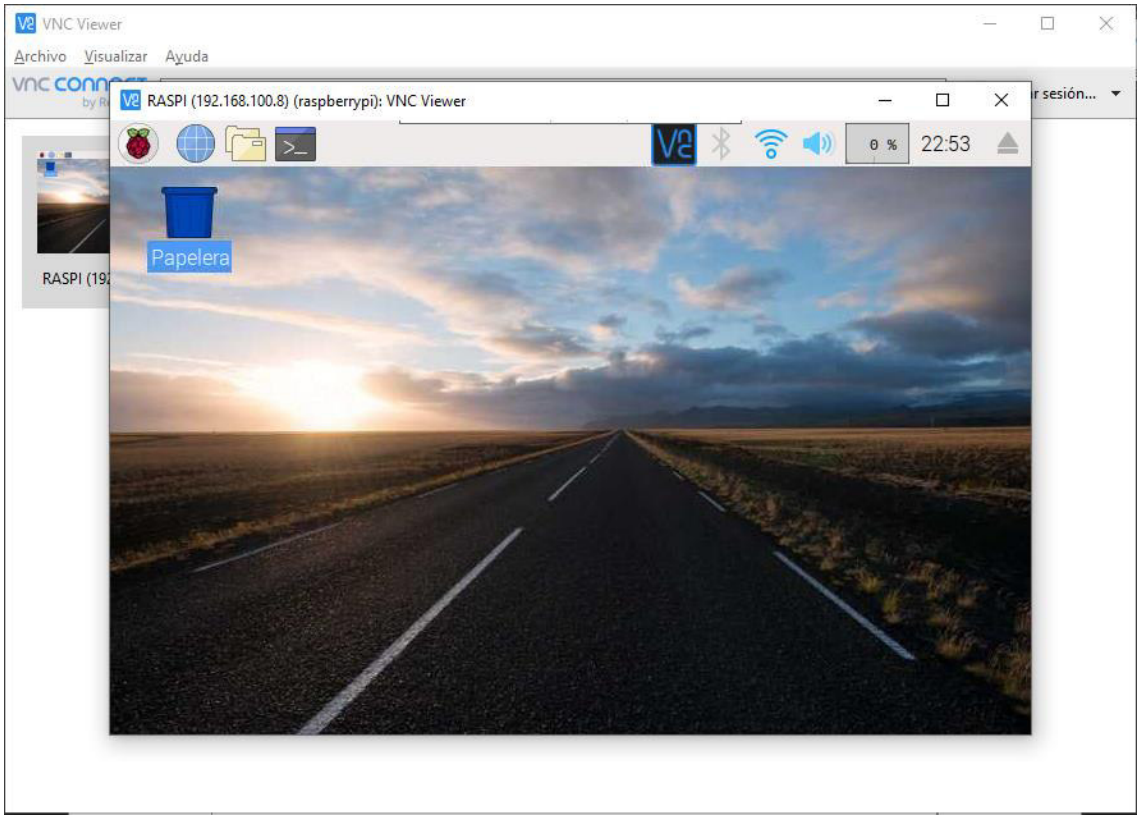

Figura E.3 Interfaz gráfica de escritorio virtual.

3. Abrir el navegador de la PC donde se configuró el acceso directo y en la barra de navegación pegar el siguiente link: **http://192.168.100.87/index.php** y dar *"Enter"*.

   Se debe cambiar los números en rojo, por la dirección IP que le entregó el técnico. Al dar *"Enter"* verá una página igual a la de la Figura E.4.

Figura E.4 Página de Inicio.

4. Dar clic sobre la opción *"Módulo de Configuración"* (Figura E.5)

Figura E.5 Módulos del sistema.

5. Al ingresar en esta opción, debe registrar el correo electrónico en el cual desea recibir los correos de alerta y dar clic en el botón *"Registrar"* (Figura E.6)

Figura E.6 Registro Correo Electrónico.

Aparecerá un mensaje que indica que el registro fue exitoso, como se muestra en la Figura E.7

Figura E.7 Registro Exitoso.

Se tiene 3 opciones del área de interés a monitorear. Se debe seleccionar una de las tres opciones, tomando en consideración el lugar donde se colocó la estructura y la imagen de fondo que va a monitorear.

6. Dar clic en el botón *"copiar"* que se encuentra a lado de cada opción, como se muestra en la Figura E.8.

Figura E.8 Selección del área de interés.

7. En el escritorio de la Raspberry dar clic en el ícono señalado en la Figura E.9.

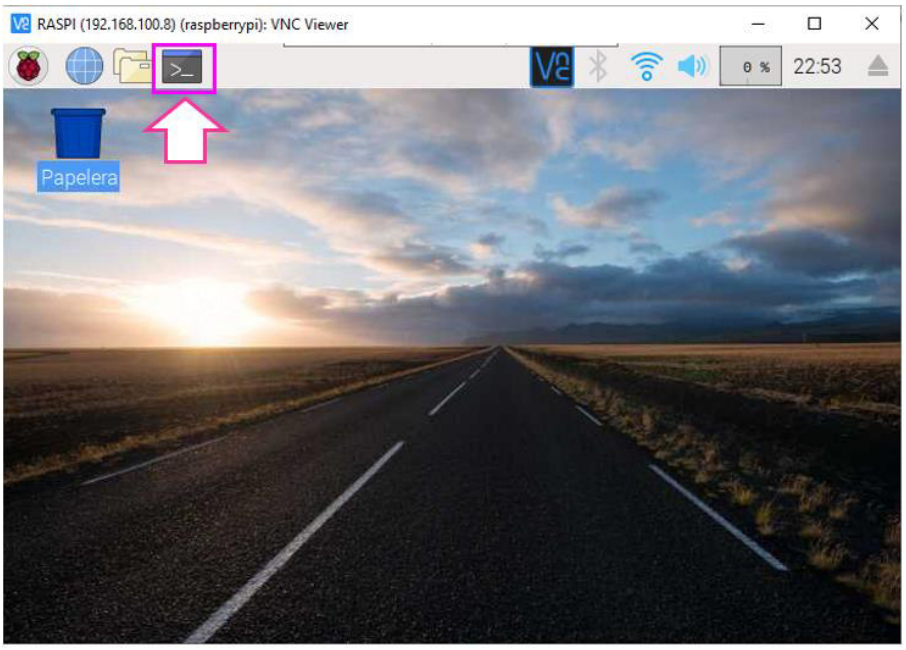

Figura E.9 Apertura del terminal.

8. Se desplegará una pantalla negra en la cual se debe dar clic derecho y seleccionar la opción *"Pegar"*, como se muestra en la Figura E.10.

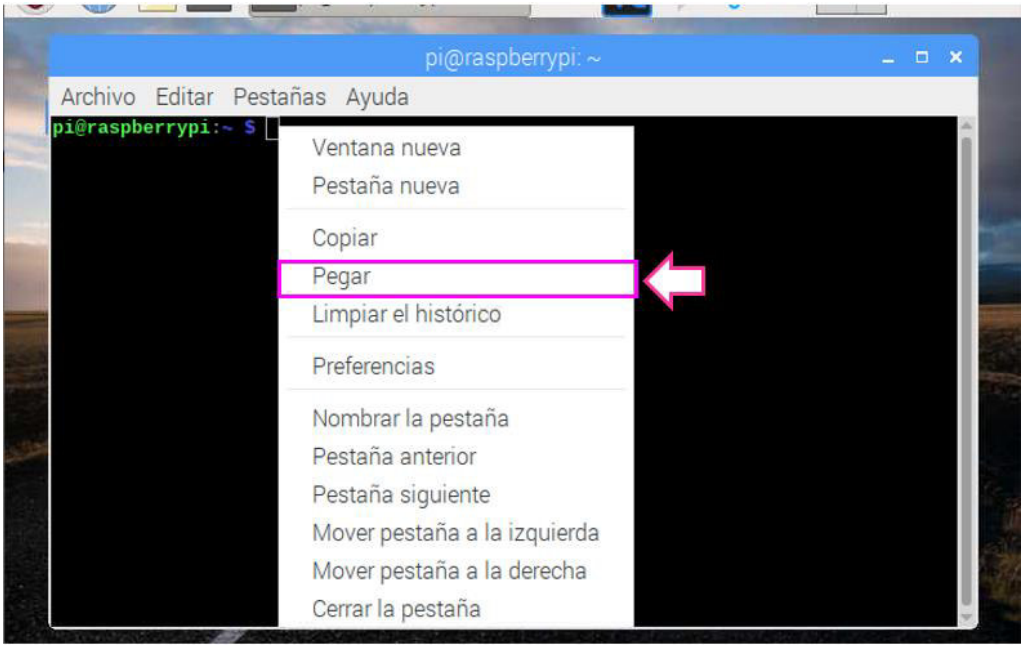

Figura E.10 Pegar texto.

9. Se va a pegar el texto copiado del área de interés seleccionada y por último dar *"Enter"*. (Figura E.11)

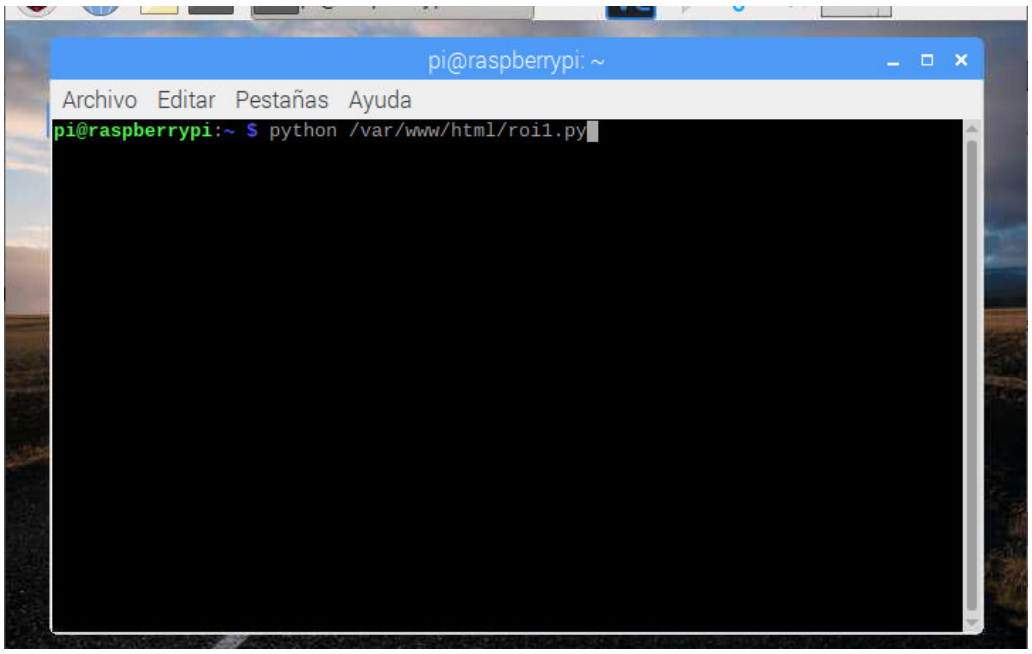

Figura E.11 Ejecución del programa.

10. Se abrirá una ventana en la cual se puede observar el fondo total del escenario, permitiendo mover la estructura de la Raspberry (físicamente) hasta finalmente posicionar de forma más exacta el área de interés a monitorear (recuadro fucsia). Como se muestra en la Figura E.12

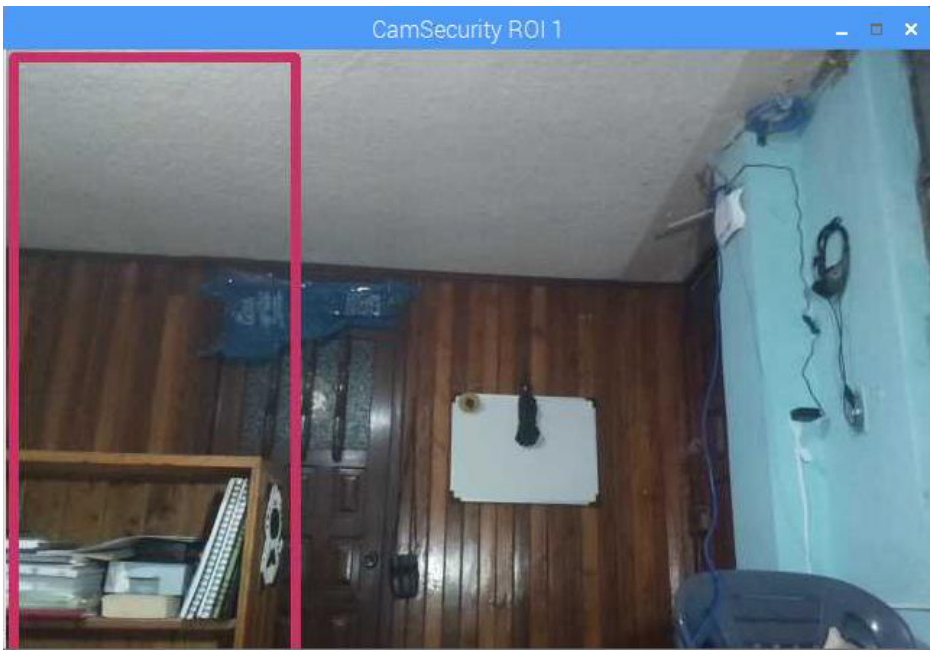

Figura E.12 Ventana de la imagen de fondo.

NOTA: Para detener el programa se debe presionar la tecla *"Enter"* mientras está abierta la ventana de la Figura E.12.

ORDEN DE EMPASTADO