PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ

FACULTAD DE CIENCIAS E INGENIERÍA

DISEÑO DE UN SISTEMA IOT PARA DETERMINAR EL AFORO DE LA ZONA DE ASIENTOS LIBRES DEL COMPLEJO DE INNOVACIÓN ACADÉMICA (CIA)

Tesis para obtener el título profesional de Ingeniero de las Telecomunicaciones

AUTORES:

Carlos Guillermo Minaya Orihuela

Julio Renato Carrión Pardo

ASESORES:

Luis Ángelo Velarde Criado

Cesar Stuardo Lucho Romero

Lima, setiembre, 2024

Informe de Similitud

Yo, Cesar Stuardo Lucho Romero, docente de la Facultad de Ciencias e Ingeniería de la Pontificia Universidad Católica del Perú, asesor de la tesis titulada Diseño de un sistema IoT para determinar el aforo de la zona de asientos libres del Complejo de Innovación Académica (CIA), de los autores Carlos Guillermo Minaya Orihuela y Julio Renato Carrión Pardo, dejo constancia de lo siguiente:

- El mencionado documento tiene un índice de puntuación de similitud de 13%. Así lo consigna el reporte de similitud emitido por el software *Turnitin* el 20/09/2024.
- He revisado con detalle dicho reporte y la Tesis o Trabajo de Suficiencia Profesional, y no se advierte indicios de plagio.
- Las citas a otros autores y sus respectivas referencias cumplen con las pautas académicas.

Lugar y fecha: Lima, Perú, 20/09/2024

Apellidos y nombres del asesor: Lucho Romero, Cesar Stuardo	
DNI: 70326404	Firma
ORCID: 0000-0002-5749-2689	

RESUMEN

En tiempos previos a la pandemia la biblioteca del Complejo de Innovación Académica (CIA), de la PUCP, presentaba un aforo insuficiente para la cantidad de alumnos, ello producía que tanto los espacios de estudios libres y grupales (cubículos) presenten una disponibilidad limitada. Por el lado de los cubículos, estos cuentan con un sistema de reservas que permite una gestión de los mismos; pero por el lado de la zona de estudio de libre disponibilidad no hay presencia de un sistema de monitorización provocando que la búsqueda de asientos libres para un estudiante o grupo de estudiantes tome un tiempo mayor. Ante posible retorno a clases para el 2022, el control de aforos dentro de este establecimiento será necesario y obligatorio, para así, poder evitar posibles contagios.

Esta tesis busca diseñar un sistema IoT para determinar y monitorizar el aforo en tiempo real de los asientos libres de la biblioteca del CIA. Se determinó que el uso de cámaras en combinación con algoritmos de inteligencia artificial corriendo en pequeños equipos de procesamiento (SBCs) como son los Raspberrys, cubren los objetivos propuestos en la tesis. Además, se propone una aplicación web desplegada en la nube, la cual se comunica con el SBC bajo el protocolo de comunicación MQTT. Esta página web permite consultar y administrar determinada información con respecto al aforo dependiendo de los permisos del usuario. El diseño y análisis se centra en un área del primer piso de la biblioteca, pero puede ser aplicada en todos los pisos de este espacio o incluso replicado en otros ambientes que necesiten tal fin.

DEDICATORIA

Redactado por Carlos Minaya

Esta tesis se la dedico primero a Dios por la vida que me dio.

A mi familia quienes por ellos soy lo que soy. Muy especialmente a mis padres, los cuales siempre estuvieron apoyándome en todas las situaciones y me ayudaron a superar diversas adversidades que se presentaron en el camino. Por su cariño, comprensión y dedicación que tiene hacia mi hermano y yo.

También se la dedico a mi abuelo QEPD, el cual siempre estuvo orgulloso de mi.

Redactado por Julio Carrión

A Dios por darme la oportunidad de seguir este camino, a mi familia por darme ese apoyo moral en todo momento y el cariño brindado a lo largo de mi vida, especialmente a mi madre que sin ella no hubiera podido lograr ningún objetivo. Por último, pero no menos importante a mi abuelita QEPD, la cual fue la primera en confiar en mi ciegamente siempre destacándome y dándome las ganas de seguir adelante siempre.

AGRADECIMIENTOS

Redactado por Carlos Minaya

Un agradecimiento a Dios y mis padres por permitirme estudiar en esta universidad la cual me brindo muchos conocimientos y experiencias inolvidables.

También un agradecimiento a mis profesores y amigos de la carrera, con los cuales hemos compartido varios momentos de estudio, diversiones, etc. Un agradecimiento a mis tutores de tesis Stuardo Lucho y Angelo Velarde, los cuales siempre estuvieron disponible para apoyarnos en este proceso. Finalmente, un especial agradecimiento a mi amigo y compañero de tesis Julio, empezamos juntos por allá el 2016 y hoy en día acabamos juntos esta etapa universitaria. Gracias amigo por el constante apoyo en lo académico y en lo fraternal.

Redactado por Julio Carrión

Agradecimiento a Dios por brindarme la posibilidad de llegar a este punto de la vida, a mi madre por permitirme estudiar en esta universidad para mi formación académica y tener experiencias inolvidables. Además, agradecer a mis profesores y amigos de la carrera por lo aprendido y vivido durante estos años, también un agradecimiento especial a nuestros asesores Stuardo Lucho y Ángelo Velarde por acogernos como sus tesistas y brindarnos su disponibilidad de tiempo a lo largo de este año, siempre demostrando un compromiso hacia nosotros para cualquier tipo de actividad y proceso. También agradecer a Arturo Diaz y Antonio Merino por la paciencia y enseñanza brindada a lo largo de todo el año. Por último, pero no menos importante agradecer a mi compañero y amigo Carlos Minaya por permitirme realizar la tesis en conjunto, ya que empezamos la carrera juntos en 2016 y ahora estamos acabando juntos la etapa universitaria. Gracias por tu apoyo académico y fraternal.

ÍNDICE

RESUMEN I
DEDICATORIA II
AGRADECIMIENTOS III
ÍNDICE IV
LISTA DE FIGURAS VII
LISTA DE CUADROS XI
INTRODUCCIÓN 1
Capítulo 1: Presentación de la problemática del aforo en la zona de asientos de libre disponibilidad 2
1.1 Presentación de la Problemática 2
1.2 Estado del Arte 7
1.2.1 Detección mediante sensores 7
1.2.2 Solución mediante redes y radiofrecuencia 11
1.2.3 Captive portal para tecnologías como WiFi 13
1.2.4 Solución mediante una aplicación móvil o Web 14
1.2.5 Cámara de videovigilancia e inteligencia artificial 14
Capítulo 2: Marco teórico sobre las tecnologías implicadas en un sistema de detección de personas basado en procesamiento de imágenes 19
2.1 Marco teórico 19
2.1.1 Edge Computing 19
2.1.2 Inteligencia Artificial 23
2.1.3 Ordenadores de Placa Simple (SBC) 28
2.1.4 Protocolos de Aplicación 32
2.1.5 Frameworks para desarrollo de back end 35
2.1.6 Software de Base de Datos Relacionales 38
2.1.7 Bases de Datos No Relacionales 40
2.1.8 Proveedores de Cloud Computing 43
2.2 Objetivos 47
2.2.1 Objetivo General 47
2.2.2 Objetivos Específicos 48
2.3 Alcance 48

Capítulo 3: Metodología para el diseño de un sistema de detección de personas basado en procesamiento de imágenes aplicada en una zona delimitada de asientos de libre disponibilidad de CIA 49
3.1 Requerimientos del sistema 49
3.2 Elección del protocolo de aplicación 52
3.2.1 Arquitectura 52
3.2.2 Tamaño del mensaje y complejidad 53
3.2.3 Quality of Service (QOS) y funcionalidades especiales 54
3.2.4 Velocidad 54
3.2.5 Seguridad 55
3.3 Elección del equipo de procesamiento de imágenes 56
3.4 Elección del algoritmo de detección de objetos 59
3.4.1 Faster R-CNN 59
3.4.2 YOLO v5 60
3.4.3 TensorFlow Lite 61
3.5 Elección de Framework para backend 62
3.6 Elección de las bases de datos 64
3.6.1 Elección de base de datos para gestión de información del servicio web 66
3.6.2 Elección de base de datos para la información del aforo en tiempo real 68
3.7 Elección de proveedor de servicio Cloud 69
3.8 Arquitectura de la solución 72
Capítulo 4: Configuración del sistema determinado y análisis de pruebas de funcionamiento 75
4.1 Configuración del Mosquitto Broker 76
4.2 Configuración del Raspberry 78
4.3 Configuración de la instancia de MongoDB del MarketPlace 82
4.4 Configuración de la Instancia de PostgreSQL 84
4.5 Configuración de la Plataforma Web (Django) 86
4.6 Pruebas de funcionamiento 101
4.7 Elaboración de costos del servicio 108
4.8 Aportes de la tesis 112
4.8.1 Aportes en el ámbito ambiental 112
4.8.2 Aportes en el ámbito social y legal 112
4.8.3 Aportes en el ámbito económico 112
4.8.4 Aportes en el ámbito cultural 112
4.8.5 Aporte en el ámbito de salud 112

CONCLUSIONES 113
BIBLIOGRAFÍA 115
RECOMENDACIONES 124
TRABAJOS A FUTURO 125
ANEXO 126

LISTA DE FIGURAS

FIGURA 1.1: MAPA DE BIBLIOTECAS EN EL CAMPUS PUCP [3] 4
FIGURA 1.2: FUNCIONAMIENTO DE UN SENSOR BIDIRECCIONAL PARA CONTROL DE AFORO 8
FIGURA 1.3: SISTEMA DE TORNIQUETES DE LA EMPRESA ACCESOR [10] 9
FIGURA 1.4 SISTEMA DE PORTILLO MOTORIZADO DE LA EMPRESA ACCESOR [10] 10
FIGURA 1.5 SISTEMA DE PASILLO DE LA EMPRESA ACCESOR [10] 10
FIGURA 1.6: EJEMPLO DE FUNCIONAMIENTO DE UN SISTEMA DE CONTROL DE AFORO [9] 11
FIGURA 1.7: TRIANGULACION DE UN USUARIO USANDO ACCESS POINT [12] 12
FIGURA 1.8: ETIQUETA NFC CON CODIGO QR IMPRESO EN LA CARATULA [13] 13
FIGURA 1.9: EJEMPLO DE´PORTAL CAUTIVO PARA ENTORNOS UNIVERSITARIOS [15] 14
FIGURA 1.10: SOLUCION IMPLEMENTADA EN UN CENTRO COMERCIAL USANDO LOS ALGORITMOS DE DETECCION Y SEGUIMIENTO [19] 16
FIGURA 2.1: ARQUITECTURA DE EJEMPLO PARA UNA SOLUCIÓN DE EDGE COMPUTING [23] 21
FIGURA 2.2: PROVEEDORES DE NUBE [24] 21
FIGURA 2.3: PRINCIPALES PROVEEDORES DE SERVICIO DE INTERNET EN PERÚ [25] 22
FIGURA 2.4: EQUIPOS DE PROCESAMIENTO [26] 22
FIGURA 2.5: ESTRUCTURA DE UNA RED NEURONAL CONVOLUCIONAL PARA LA CLASIFICACIÓN Y/O DETECCIÓN DE IMÁGENES [27] 23
FIGURA 2.6: ARQUITECTURA DE FASTER R-CNN [28] 24
FIGURA 2.7: GRÁFICO COMPARATIVO DE LA VELOCIDAD DE DETECCION DE DE LOS ALGORITMOS R-CNN, SPP-Net, FAST R-CNN Y FASTER R-CNN [28] 25
FIGURA 2.8: EJEMPLO DE BLOQUES RESIDUALES [29] 26
FIGURA 2.9: PROBABILIDAD DE QUE UN OBJETO SE ENCUENTRE EN EL CUADRO DELIMITADOR [29] 26
FIGURA 2.10: FUNCIONAMIENTO DE IOU [29] 27
FIGURA 2.11: FUNCIONAMIENTO DEL ALGORITMO YOLO [29] 28
FIGURA 2.12: MODELOS DE ARDUINO [32] 30
FIGURA 2.13: RASPBERRY PI 4B [33] 31
FIGURA 2.14: NVIDIA JETSON [34] 31
FIGURA 2.15: ARQUITECTURA DE MQTT [35] 33
FIGURA 2.16: ARQUITECTURA DE HTTP [37] 34
FIGURA 2.17 ARQUITECTURA WEB [38] 35

FIGURA 2.18 FLUJO DE ARQUITECTURA MVC [43] 38
FIGURA 2.19 EJEMPLO DE ESTRUCTURA DE DATOS MONGODB [49] 41
FIGURA 2.20 TIPOS DE INSTANCIAS EN GOOGLE CLOUD - JUNIO 2021[51] 44
FIGURA 2.21 COSTOS POR TIPO DE INSTANCIAS EN GOOGLE CLOUD-JUNIO 2021[51] 44
FIGURA 3.1 ESTRUCTURA DEL PAQUETE MQTT [55] 53
FIGURA 3.2 RESULTADOS DEL TEST [57] 55
FIGURA 3.3 PRUEBA DEL ALGORITMO FASTER R-CNN EN EL RASPBERRY (0.67FPS) [ELABORACIÓN PROPIA] 59
FIGURA 3.4 PRUEBA DEL ALGORITMO YOLO V5 EN EL RASPBERRY [ELABORACIÓN PROPIA] 60
FIGURA 3.5 TIEMPOS DE RESPUESTA EN UN FRAME DE YOLO V5 [ELABORACIÓN PROPIA] 60
FIGURA 3.6 PRUEBA DE TENSORFLOW LITE EN EL RASPBERRY [ELABORACIÓN PROPIA] 62
FIGURA 3.7 COMPARACIÓN DE DJANGO Y SPRING EN DISTINTOS DOMINIOS [63] 64
FIGURA 3.8 COSTO DEL SERVICIO ESTIMADO POR MES [69] 71
FIGURA 3.9 COSTO ESTIMADO DEL SERVICIO POR MES [70] 72
FIGURA 3.10 ARQUITECTURA DE LA SOLUCIÓN EN GOOGLE CLOUD [ELABORACIÓN PROPIA] 73
FIGURA 4.1 ARQUITECTURA DETALLADA DE LA SOLUCIÓN EN GOOGLE CLOUD [ELABORACIÓN PROPIA] 75
FIGURA 4.2 CARACTERÍSTICAS DEL COMPUE ENGINE (E2-MICRO) EN GOOGLE CLOUD [ELABORACIÓN PROPIA] 76
FIGURA 4.3 ESTADO DEL SERVICIO DE MOSQUITTO EN GOOGLE CLOUD [ELABORACIÓN PROPIA] 77
FIGURA 4.4 CONFIGURACIÓN DEL SERVICIO DE MOSQUITTO [ELABORACIÓN PROPIA] 77
FIGURA 4.5 CASE SIN SOPORTE PARA CÁMARA [59] 78
FIGURA 4.6 CASE CON SOPORTE PARA CÁMARA [71] 78
FIGURA 4.7 RASPBERRY PI 4B CON CÁMARA Y CASE [ELABORACIÓN PROPIA] 79
FIGURA 4.8 CONFIGURANDO EL MODELO TFLITE Y EL CÁLCULO DE FRAMES [ELABORACIÓN PROPIA] 80
FIGURA 4.9 PROCESO DE DETECCIÓN DE PERSONAS [ELABORACIÓN PROPIA] 80
FIGURA 4.10 PUBLICACIÓN DE INFORMACIÓN AL TÓPICO ESCOGIDO [ELABORACIÓN PROPIA] 81
FIGURA 4.11 DIAGRAMA DE FLUJOS DEL FUNCIONAMIENTO DEL ALGORITMO DE PROCESAMIENTO DE IMÁGENES [ELABORACIÓN PROPIA] 81

FIGURA 4.12 CARACTERÍSTICAS DEL MARKETPLACE DE MONGODB EN GOOGLE CLOUD [ELABORACIÓN PROPIA] 82
FIGURA 4.13 ESTADO DEL SERVICIO DE MARKETPLACE DE MONGODB EN GOOGLE CLOUD [ELABORACIÓN PROPIA] 83
FIGURA 4.14 RESTRICCIÓN DE IP'S AL PUERTO USADO POR MONGODB [ELABORACIÓN PROPIA] 84
FIGURA 4.15 RESUMEN DE LAS PROPIEDADES DE LA INSTANCIA DE CLOUD SQL [ELABORACIÓN PROPIA] 85
FIGURA 4.16 BASE DE DATOS CREADA EN LA INSTANCIA DE CLOUD SQL [ELABORACIÓN PROPIA] 85
FIGURA 4.17 IP'S PERMITIDAS PARA EL ACCESO A LA BASE DE DATOS EN LA INSTANCIA DE CLOUD SQL [ELABORACIÓN PROPIA] 86
FIGURA 4.18 REQUERIMIENTOS DEL SISTEMA WEB [ELABORACIÓN PROPIA] 86
FIGURA 4.19 CLIENTE MQTT EN BACK END DEL SISTEMA WEB [ELABORACIÓN PROPIA] 87
FIGURA 4.20 CLIENTE PARA MONGODB EN BACK END DEL SISTEMA WEB [ELABORACIÓN PROPIA] 87
FIGURA 4.21 DIAGRAMA DE FLUJOS DEL FUNCIONAMIENTO DE MQTT Y MONGO DB DEL LADO DEL BACKEND DE LA PLATAFORMA WEB [ELABORACIÓN PROPIA] 88
FIGURA 4.22 DIAGRAMA DE FLUJOS DEL FUNCIONAMIENTO DE MQTT Y MONGO DB DEL LADO DEL BACKEND DE LA PLATAFORMA WEB [ELABORACIÓN PROPIA] 89
FIGURA 4.23 VISTA DE INICIO DE SESIÓN DESDE PERSPECTIVA DE COMPUTADORA Y MÓVIL [ELABORACIÓN PROPIA] 90
FIGURA 4.24 DIAGRAMA DE FLUJO DE LA AUTENTICACIÓN DEL USUARIO EN EL SISTEMA WEB [ELABORACIÓN PROPIA] 91
FIGURA 4.25 VISTA DE REGISTRO DESDE PERSPECTIVA DE COMPUTADORA Y MÓVIL [ELABORACIÓN PROPIA] 92
FIGURA 4.26 DIAGRAMA DE FLUJO DEL REGISTRO DE USUARIOS EN EL SISTEMA WEB [ELABORACIÓN PROPIA] 93
FIGURA 4.27 VISTA DE LA LISTA DE USUARIOS DE ROL ESTUDIANTE NO VERIFICADOS [ELABORACIÓN PROPIA] 93
FIGURA 4.28 VISTA DE LA LISTA DE USUARIOS DE ROL ESTUDIANTE VERIFICADOS [ELABORACIÓN PROPIA] 94
FIGURA 4.29 DIAGRAMA DE FLUJO DE LA ADMINISTRACIÓN DE USUARIOS CON ROL DE ESTUDIANTE EN EL SISTEMA WEB [ELABORACIÓN PROPIA] 94
FIGURA 4.30 GRÁFICO DE ESTUDIANTES VERIFICADOS Y NO VERIFICADOS EN EL SISTEMA WEB [ELABORACIÓN PROPIA] 95
FIGURA 4.31 GRÁFICO DE ESTUDIANTES REGISTRADOS POR FACULTAD [ELABORACIÓN PROPIA] 95
FIGURA 4.32 GRÁFICO DE AFORO POR DÍA Y MARCADOR DE AFORO EN TIEMPO REAL [ELABORACIÓN PROPIA] 96

FIGURA 4.33 GRÁFICO DE AFORO DE LA SEMANA TRANSCURRIDA ACTUAL [ELABORACIÓN PROPIA] 96
FIGURA 4.34 VISTA DEL FORMULARIO PARA ACTIVAR O DESACTIVAR NOTIFICACIONES [ELABORACIÓN PROPIA] 97
FIGURA 4.35 DIAGRAMA DE FLUJO DEL FUNCIONAMIENTO DE LAS NOTIFICACIONES EN CASO DE ESTAR ACTIVADAS [ELABORACIÓN PROPIA] 97
FIGURA 4.36 VISTA DEL FORMULARIO PARA LA CREACIÓN DE EVENTOS [ELABORACIÓN PROPIA] 98
FIGURA 4.37 DIAGRAMA DE FLUJO DEL FUNCIONAMIENTO DE CREACIÓN DE EVENTOS [ELABORACIÓN PROPIA] 98
FIGURA 4.38 VISTA DE LA LISTA DE EVENTOS PARA EL ADMINISTRADOR [ELABORACIÓN PROPIA] 99
FIGURA 4.39 VISTA DEL DETALLE DEL EVENTO [ELABORACIÓN PROPIA] 100
FIGURA 4.40 VISTA DE LA EDICIÓN DE UN EVENTO [ELABORACIÓN PROPIA] 100
FIGURA 4.41: PRUEBA CON DOS PERSONAS SENTADAS CONVERSANDO [ELABORACIÓN PROPIA] 101
FIGURA 4.42: PRUEBA 2 CON DOS PERSONAS SENTADAS CONVERSANDIO [ELABORACIÓN PROPIA] 102
FIGURA 4.43: PRUEBA CON PERSONA RECOSTADA [ELABORACIÓN PROPIA] 102
FIGURA 4.44: PRUEBA DE PERSONAS ESTUDIANDO CON MASCARILLA [ELABORACIÓN PROPIA] 103
FIGURA 4.45: PRUEBA DE PERSONAS DANDO SALUDO CON PUÑOS [ELABORACIÓN PROPIA] 103
FIGURA 4.46: PRUEBA CON PERSONAS CAMINANDO [ELABORACIÓN PROPIA] 104
FIGURA 4.47: PRUEBA CON UNA PERSONA EN EL V307 – PUCP [ELABORACIÓN PROPIA] 104
FIGURA 4.48: PRUEBA CON DOS PERSONAS EN EL V307 – PUCP [ELABORACIÓN PROPIA] 105
FIGURA 4.49: GRÁFICO DE AFORO [ELABORACIÓN PROPIA] 105
FIGURA 4.50: PRUEBA CON DETECCION ERRONEA [ELABORACIÓN PROPIA] 106
FIGURA 4.51: PRUEBA DESDE DISTINTO ANGULO [ELABORACIÓN PROPIA] 107
FIGURA 4.52: PRUEBA CON BAJA ILUMINACION [ELABORACIÓN PROPIA] 107
FIGURA 4.53: PRUEBA DESDE DISTINTO ANGULO Y MAYOR ILUMINACION [ELABORACIÓN PROPIA] 108
FUNCIONAMIENTO DEL ALGORITMO R-CNN [28] 126
ARQUITECTURA DE FAST R-CNN [28] 128

LISTA DE CUADROS

CUADRO 2.1: COMPARACIÓN DE MICROCONTROLADOR Y MICROPROCESADOR [31] 29
CUADRO 3.1: LISTA DE REQUERIMIENTOS DE LA PLATAFORMA WEB [ELABORACIÓN PROPIA] 51
CUADRO 3.2 CUADRO COMPARATIVO DE SBC'S [34] [58] 57
CUADRO 3.3 CUADRO COMPARATIVO DE LAS CÁMARAS [59] 58
CUADRO 3.4 CUADRO COMPARATIVO DJANGO Y SPRING BOOT [60] [61] [62] 63
CUADRO 3.5: CUADRO COMPARATIVO ENTRE BASE DE DATOS SQL Y NOSQL [64] 65
CUADRO 3.6 CUADRO COMPARATIVO POSTGRESQL Y MYSQL [65] [66] 67
CUADRO 3.7 SERVICIOS DE AWS, MICROSOFT AZURE Y GOOGLE CLOUD [68] 70
CUADRO 4.1 COSTOS DE INVERSIÓN [ELABORACIÓN PROPIA] 109
CUADRO 4.2 COSTOS DE OPERACIÓN [ELABORACIÓN PROPIA] 111

INTRODUCCIÓN

La Pontificia Universidad Católica del Perú, ha tenido un constante crecimiento en su alumnado. En el 2021-1, se registró 25.994 alumnos de pregrado [1], 2500 más que el año previo, esto afecta directamente en los espacios de estudio, como las bibliotecas, dificultando la búsqueda de asientos libres para estudio, lo que puede generar en los alumnos estrés, preocupación y/o ansiedad. Además, en la actualidad, la pandemia mundial ha generado un recorte de aforos en ambientes cerrados, por lo tanto, tener un control y monitoreo sobre este es indispensable si se quiere evitar aglomeración y posibles contagios. El actual trabajo de investigación se compone del diseño de un sistema IoT para determinar el aforo en tiempo real de la zona de los asientos de libre disponibilidad en CIA.

Primero se realizó una comparación de las tecnologías IoT para el control de aforos, determinando que la mejor opción que cumple con el objetivo principal es el de usar detección de personas mediante cámaras y procesar digitalmente las imágenes usando algoritmos de inteligencia artificial. Luego se analizó las tecnologías involucradas en el desarrollo de la aplicación web, como son los frameworks para el desarrollo del backend y frontend. Además, se realizó una comparación de protocolos de comunicación entre el equipo de procesamiento elegido y la aplicación web. Con el despliegue en la nube, se probó el sistema dando resultados favorables en la detección de personas. Finalmente se determinó que esta solución puede ser extrapolada a otros ambientes que necesiten de un control y monitorización de aforo en tiempo real.

Capítulo 1: Presentación de la problemática del aforo en la zona de asientos de libre disponibilidad

1.1 Presentación de la Problemática

Las telecomunicaciones están teniendo una gran importancia en la actual sociedad peruana porque generan un gran aporte en el crecimiento en todos los aspectos (económico, social, educativo, etc), que se ha demostrado a lo largo de la pandemia provocada por el COVID-19; puesto que la gran mayoría de actividades se han visto forzadas a migrar en un entorno virtual, con lo que se plantean despliegues de nuevas tecnologías, como 5G, y redes ópticas de manera que se pueda soportar la demanda de tráfico en la red por los distintos usuarios. El despliegue de las nuevas tecnologías provoca la posibilidad de nuevas aplicaciones como inteligencia artificial, IoT (*Internet of Things*), entre otras; entonces ello conllevará en un impacto económico a favor del sector de telecomunicaciones y según Sebastián Cabello, CEO de SmC, "traerá consigo un impacto económico estimado de entre US$10.000 y US$15.000 millones hasta el 2030" [2]. Así mismo, la posibilidad de contar con nuevas aplicaciones, ayuda a afrontar de mejor manera problemáticas existentes; por ejemplo, el control de aforos para diferentes espacios ya sean públicos o privados porque están regidos bajo normas y restricciones referidas a la edificación que tienen la finalidad de brindar seguridad a las

personas, ya que disminuir los riesgos de contagios es primordial para así enfrentar la situación actual de salubridad del país. Por lo que, la necesidad de determinación del aforo en distintos establecimientos es imprescindible y si esta información es obtenida en tiempo real sería muy beneficioso.

La Pontificia Universidad Católica del Perú (PUCP) es la universidad con mayor reconocimiento en el país; además en el 2021, 25 991 estudiantes de pregrado y 5 359 postgrado fueron registrados [1]. Esta casa de estudio dispone de 5 bibliotecas, las cuales son:

- Biblioteca Central Luis Jaime Cisneros
- Biblioteca de Ciencias Sociales “Alberto Flores Galindo”
- Biblioteca de Teología
- Biblioteca del Centro de Estudios Orientales
- Biblioteca del Complejo de Innovación Académica (CIA)

La última mencionada posee 2 sótanos y 4 pisos, este lugar va dirigido principalmente a las facultades de arquitectura, Ciencias e Ingeniería y Estudios Generales Ciencias, pero el acceso y uso de espacios es para todos los alumnos de la universidad. La ubicación del complejo se muestra en la Figura 1.1.

FIGURA 1.1: MAPA DE BIBLIOTECAS EN EL CAMPUS PUCP [3]

Este ambiente fue pensado para adaptar una nueva idea, la cual se diferencia de las bibliotecas convencionales; más que ser un repositorio de libros, es un lugar de aprendizaje conjunto. Por el lado de los ambientes de estudio, existen áreas individuales y grupales; los grupales abarcan tanto los cubículos y ciertas salas de estudio, que cuentan con un seguimiento de ocupación por parte de un sistema web y por los empleados de la universidad para así tener un orden por reservas y a la vez no se sobrepase el aforo indicado, mientras que los asientos de libre disponibilidad (individuales) carecen de dicha implementación. Por lo que esta ausencia se considera como un problema de determinación de aforo y ello se explicó previamente.

La razón por la que se ha escogido la biblioteca del Complejo de Innovación académica es la disponibilidad de espacios libres de estudio, ya que es limitada para la cantidad de estudiantes actuales de la universidad. Por lo que el tiempo que conlleva recorrer las instalaciones para encontrar un sitio que no esté ocupado suele tomar un gran rango de tiempo y al ir con un grupo de estudio suele ser peor porque los números de asientos a buscar aumentan. En un contexto

de semi-presencialidad, la interacción con muchas personas es una acción que se tiene que evitar porque puede aumentar el riesgo de contagio, por lo que contar con la información del aforo en tiempo real de la zona de asientos de libre disponibilidad, provocaría un gran aporte a los alumnos como a trabajadores de la biblioteca.

Ante un posible retorno a clases, muchas actividades serán restringidas en la universidad debido a la situación y los reglamentos sanitarios actuales para contrarrestar la propagación del COVID-19, por lo que la posibilidad de brindar la información del aforo en tiempo real y consultarla mediante una aplicación disminuiría el riesgo de contagio. Otra de las razones, es brindarles a las personas encargadas de la administración de la biblioteca del CIA un control ordenado y automatizado para hacer un seguimiento y verificación que ningún asiento este ocupado antes del cierre de la biblioteca y/o distintas acciones que dependan de un seguimiento del aforo del establecimiento como pueden ser estadísticas del días o semanales para saber el horario donde hay más afluencia de alumnos, etc. Además, dependiendo de la tecnología a usarse, esta podría brindar seguridad a los alumnos como al personal de la universidad, por ejemplo: tener un registro grabado por videocámaras, prohibir la salida y/o entrada a la biblioteca en determinados momentos, etc. Finalmente, este proyecto contribuye a una migración a Smart Campus, el cual básicamente es un campus inteligente y sostenible que usa tecnologías en red para facilitar la comunicación, mejorar la seguridad y utilizar los recursos de manera más eficiente mediante el empleo de nuevas aplicaciones tecnológicas [4]. Esto aporta al crecimiento e incentivo de desarrollo de proyectos de esta índole, promoviendo un Smart Campus a futuro en la PUCP. Según Gartner, una empresa consultora y de investigación de las tecnologías de la información, identificó a los Smart Campus como una de las 10 principales tecnologías estratégicas que impactan fuertemente en la educación superior. Esta misma compañía define a Smart campus como un entorno digital o físico en el que los seres

humanos y los sistemas tecnológicos interactúan para crear así, experiencias más inmersivas y automatizadas para los alumnos, profesores y personal de la universidad [5].

Entre las nuevas tecnologías que se implementan en un Smart Campus, está el internet de las cosas (IoT), el cual es un concepto que se basa en la interconexión digital de dispositivos y objetos a través de una red [6]. Los grandes potenciales de esta tecnología son:

- Cobertura: En el caso de la biblioteca CIA se logrará tener un alcance total a todos los espacios de estudio que no presentan actualmente un seguimiento de determinación de aforo.
- Escalabilidad: Una red de Smart Campus debería escalar eficientemente para manejar el tráfico generado por miles de usuarios y dispositivos IoT.
- Diversidad: La conectividad establecida debería tener la capacidad de soportar distintos tipos; desde los más sencillos hasta soluciones que requieran un alto rendimiento y baja latencia.

De acuerdo a lo anteriormente mencionado, la determinación de aforos en tiempo real puede ser enfocado en otros ambientes de la universidad, como son salones, laboratorios, comedores, etc. Sin embargo, para cada escenario habrá condiciones diferentes, debido a que las infraestructuras varían, por ende, la solución debería poder adaptarse a cada una de estas manteniendo la tecnología escogida. Si bien IoT es un concepto muy crítico para la recolección de datos, esta data puede procesarse en los mismos dispositivos o en un dispositivo cercano a la ubicación física del dispositivo recolector. Este tipo de procesamiento define a Edge Computing, la cual tiene como grandes ventajas, ofrecer una menor latencia y una mayor disponibilidad del ancho de banda [7]. Una vez la data sea procesada se convierte en información, la cual llegaría hacia una aplicación y/o servicio web para los usuarios finales,

como lo son principalmente estudiantes y personal administrativo de la universidad, a través de dispositivos de uso diario como lo son laptops, tablets, smartphones, etc.

En conclusión, el conseguir espacios de estudio libres, es un problema que han vivido miles de estudiantes de la PUCP previo a la pandemia. Como estudiante, se tenía que recorrer los 4 pisos y 2 sótanos de la biblioteca del CIA en busca de un asiento disponible para estudiar, lo que significaba un esfuerzo extra al subir y bajar varias escaleras para encontrar dicho lugar. Según, las encuestas realizadas, más del 70% de alumnos se demoraba más de 10 minutos en encontrar sitios de estudio, pudiendo generar así estrés, cansancio, etc; en los estudiantes. Por ende, con la solución que se propondrá en este trabajo de tesis, los alumnos serán los más beneficiados ya que mediante una página web podrán ver el aforo en tiempo real de las zonas de asientos de libre disponibilidad. Además, debido a la pandemia mundial del COVID-19, esta solución evita aglomeraciones y posibles focos de contagios. Por otra parte, el personal administrativo también estaría beneficiado al tener una aplicación para poder realizar un seguimiento y monitorización del aforo en la biblioteca. Finalmente, dependiendo de la tecnología a usar se brindaría una mayor seguridad dentro del establecimiento.

1.2 Estado del Arte

En la actualidad existen diversas soluciones para la medición de aforo las cuales son: detección por medio de sensores, uso de tecnología de redes inalámbricas y por medio de cámaras de videovigilancia junto a algoritmos de procesamiento de imágenes. En el presente capítulo se introducirán lo previamente mencionado.

1.2.1 Detección mediante sensores

Esta es una de las soluciones más usadas en diferentes centros comerciales para la obtención de información del aforo [8]. Se tienen las siguientes soluciones:

- **Sensor Smart Occupancy**: Cuenta con una cámara con la finalidad de tener un escaneo progresivo de manera que se puede examinar la ocupación del lugar mediante la configuración de una línea virtual que delimita la zona interior/exterior. El sensor está ubicado en el techo del ingreso y está apoyado por un algoritmo de reconocimiento; entonces la medición puede ser hecha en tiempo real [8]. Por otro lado, existen modelos bidireccionales y direccionales, donde el primero de ellos hace empleo de dos sensores en un pasillo asignado para la salida y entrada; mientras que el modelo bidireccional hace uso de dos pasillos, con un sensor por pasillo, con lo que se mediría la salida y entrada pasillos independientes. [9]

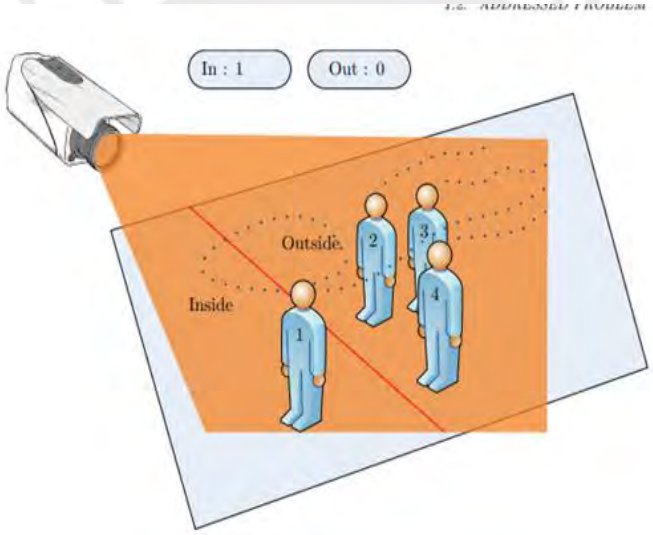


FIGURA 1.2: FUNCIONAMIENTO DE UN SENSOR BIDIRECCIONAL PARA CONTROL DE AFORO [9]

- **Sistema de torniquetes**: Este sistema hace uso de torniquetes, los cuales son dispositivos conformados por 3 o 4 barras metálicas unidas por un extremo y formando un ángulo agudo entre ellos [10]. Su funcionamiento se basa en un cálculo de personas dentro del establecimiento por medio de un conteo de ambas direcciones (personas que entran y salen).

FIGURA 1.3: SISTEMA DE TORNIQUETES DE LA EMPRESA ACCESOR [10]

- **Pasillos y Portillos Motorizados**: Son equipos de control de acceso similar a los torniquetes en cuanto a funcionamiento; sin embargo, en este caso no se cuenta con barras, sino que con una apertura libre. Usualmente en los laterales interiores están alojados los sensores infrarrojos, los cuales aseguran el pase de persona a persona. La diferencia principal entre pasillos y portillos motorizados radica en la seguridad, en el caso de los portillos se tienen barreras o puertas motorizadas, como se aprecia en la figura 1.4, las cuales se despliegan para el pase de cada usuario, por ende, se podría denegar el ingreso y/o salida de personas, mientras que en la figura 1.5 se observan los pasillos en los cuales no existe ningún impedimento físico para el pase [10].

FIGURA 1.4 SISTEMA DE PORTILLO MOTORIZADO DE LA EMPRESA ACCESOR [10]

FIGURA 1.5 SISTEMA DE PASILLO DE LA EMPRESA ACCESOR [10]

En la figura 1.6 se muestra un ejemplo del funcionamiento de un sistema encargado del control de aforo perteneciente a la compañía Magiturno. La solución presentada consta de sensores de determinación de aforo en una sola dirección en la entrada como salida del lugar. Se usa la red Wi-Fi para el envío de información de los sensores hacia el equipo denominado PLAYER, el cual se encarga del procesamiento de información y publicación en tiempo real [9].

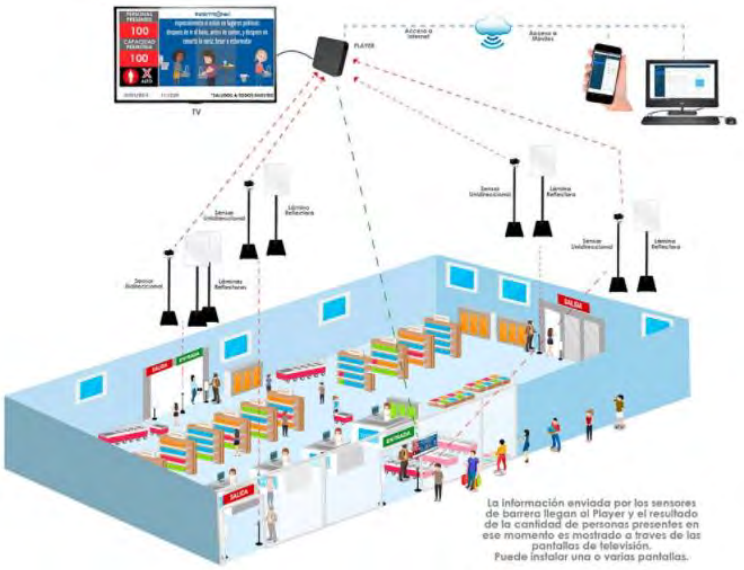


FIGURA 1.6: EJEMPLO DE FUNCIONAMIENTO DE UN SISTEMA DE CONTROL DE AFORO [9]

1.2.2 Solución mediante redes y radiofrecuencia

Por un lado, la compañía Ontrace propone una solución con el uso de sensores PassTracker, el cual detecta equipos móviles con el módulo de Wi-Fi activo en un rango de 70 metros de manera de obtener sus MAC's y nivel de potencia; entonces de esta manera se puede tener una ubicación estimada, pero una desventaja que presenta es que si una persona posee más de un equipo no habría forma de contabilizarlas como una sola persona por lo que para detección de aforo no es conveniente su uso [11]. Esta solución con WiFi, se basa en el uso de los Access Points instalados en el establecimiento, ya que se pueden obtener las MAC's de los equipos que accedan a estos y obtener su ubicación por medio de los niveles de potencia (Figura 1.7).

FIGURA 1.7: TRIANGULACIÓN DE UN USUARIO USANDO ACCESS POINT [12]

Por otro lado, Bluetooth es una tecnología inalámbrica con presencia en todos los equipos móviles de la actualidad; entonces se podrían disponer de receptores distanciados aproximadamente 10 metros entre ellos (rango de funcionamiento de Bluetooth), los cuales contabilizarían el número de personas.

Por último, NFC (*Near Field Communications*) es una tecnología que opera en la frecuencia de 13.56 MHz y se tendrían que colocar etiquetas pasivas NFC (Figura 1.8) en cada asiento con lo que los móviles que tengan esta tecnología se comuniquen con las etiquetas con solo acercarlos. La etiqueta tendría el URL que redirigirá a una página web o app móvil, en la cual se cambiaría automáticamente el estado de disponibilidad del asiento a ocupado. Esta solución podría brindar valores exactos y en tiempo real, pero para ello los celulares de los alumnos deberían poseer NFC, lo que no es común actualmente; por esta razón, para solucionar ese inconveniente, en lugar de etiquetas NFC se podrían usar códigos QR que, al ser escaneados por las cámaras de los celulares, redirijan a la aplicación móvil o web para el conteo de aforo.

FIGURA 1.8: ETIQUETA NFC CON CÓDIGO QR IMPRESO EN LA CARATULA [13]

1.2.3 Captive portal para tecnologías como WiFi

La universidad cuenta con su red Wifi, cuyo SSID es RED PUCP, el cual cubre casi la totalidad del recinto universitario, esta red trabaja en las frecuencias de 2.4 y 5 GHz. El ingreso a esta red es mediante una clave secreta para alumnos de PUCP, sin embargo, se puede hallar "googleando" fácilmente, con lo que cualquier persona que conozca la clave y este cerca de la universidad podría acceder fácilmente ya que no se cuenta con un método de autenticación que verifique que dicha persona está relacionada a la universidad. Por otro lado, un portal cautivo es un programa el cual vigila el tráfico HTTP y obliga a los usuarios a redireccionarse a una página de autenticación, donde el usuario ingresas sus credenciales como su usuario PUCP y contraseña [14]. Implementando esta solución en la universidad, se podrían solucionar problemas como es el conteo de usuarios repetidos al usar la solución de las MACs registradas por los Access Points, debido a que todos los dispositivos de un usuario estarían logueados en la red de la universidad con la misma cuenta PUCP, de tal manera ya no se contaría dispositivos, sino usuarios. Finalmente, un portal cautivo brinda seguridad a la red y responsabiliza a los usuarios de sus acciones que realicen al navegar en internet a través de esta red [14]

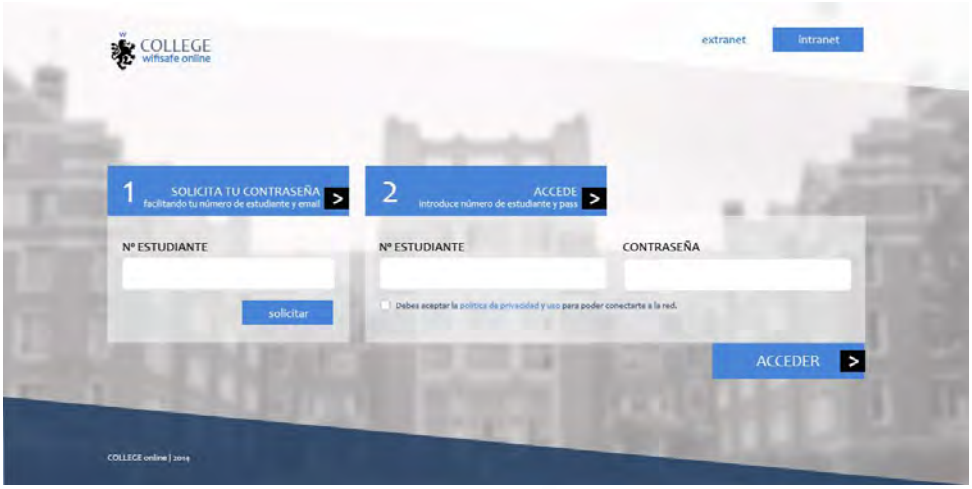


FIGURA 1.9: EJEMPLO DE PORTAL CAUTIVO PARA ENTORNOS UNIVERSITARIOS [15]

1.2.4 Solución mediante una aplicación móvil o Web

En la actualidad existen soluciones como Smart Occupancy, en la cual se pueden gestionar áreas de trabajo, mediante las reservas manuales de asientos o espacios de trabajo mediante su aplicación [16], bastante similar a lo que ofrece la aplicación PUCP móvil, sin embargo, en esta aplicación las reservas son únicamente para ciertas áreas denominadas cubículos, los cuales se ubican en las bibliotecas.

1.2.5 Cámara de videovigilancia e inteligencia artificial

El uso de cámaras de videovigilancia en combinación con algoritmos de inteligencia artificial para realizar el control de aforo, ha sido desarrollada por múltiples compañías, entre ellas se encuentra la marca internacional CANON, la cual desarrolló un software llamado Crowd People Counter acompañado de una cámara con un sensor de alta calidad, permitiendo diferenciar las personas y evitar errores de cálculo en aglomeraciones. En el 2018, durante un partido de rugby, se calculó el aforo entre toda la multitud espectadora con un margen de error del 5%. Según CANON, esta solución es mayormente indicada para espacios abiertos y

aglomeraciones, debido a la complejidad del algoritmo el cual está basada en deep learning e inteligencia artificial [17].

1.2.5.1 Algoritmos

En este campo se cuenta con dos paradigmas, están los algoritmos que detectan objetos y los algoritmos que realizan un seguimiento de objetos. Por un lado, cuando se aplica la detección de objetos, se determina en qué parte de una imagen/frame se encuentra un objeto. Sin embargo, un detector de objetos por lo general suele ser más caro desde un punto de vista computacional y, por lo tanto, más lento que un algoritmo de seguimiento de objetos. Algunos algoritmos de detección de objetos incluyen cascadas de Haar, HOG + SVM lineal y detectores de objetos basados en deep learning, como Faster R-CNN, YOLO y detectores de disparo único (SSD) [18].

Por otro lado, un algoritmo que realiza seguimiento de objeto o también llamado rastreador de objetos, aceptará las coordenadas de entrada (x,y) de la ubicación de un objeto en una imagen y hace lo siguiente:

- Asignar un identificador único a ese objeto en particular. [18]
- Seguir o rastrear al objeto a medida que se mueve alrededor de una secuencia de video, prediciendo la ubicación del nuevo objeto en el siguiente frame usando varios atributos del cuadro como son gradiente, flujo óptico, etc. [18]

Los ejemplos más populares de algoritmos de seguimiento de objetos incluyen MedianFlow, MOSSE, GOTURN, filtros de correlación kernalizados, filtros de correlación discriminatorios, etc. [18]

Si se combinan ambos conceptos de algoritmos de detección de objeto y seguimiento de objetos, se obtiene un rastreador de objetos de alta precisión, el cual consta en su mayoría de dos fases:

- Fase 1 (Detección): en esta fase, se ejecuta el algoritmo de detección de objetos, el cual es computacionalmente más costoso, su trabajo principalmente es detectar si nuevos objetos han ingresado a la vista determinada y ver si se puede encontrar objetos que se "perdieron" o no fueron contados durante la fase de seguimiento. Cada objeto detectado cuenta con sus coordenadas en un cuadro delimitador. Dado que este detector de objetos es más costoso computacionalmente, solo se ejecuta esta fase una vez cada N fotogramas del video. [18]
- Fase 2 (Seguimiento): en esta fase para cada objeto detectado se crea un object tracker, de modo que rastrea al objeto a medida que se mueve alrededor del marco. El object tracker, debería ser más rápido y eficiente que el detector de objetos. El procesamiento de rastreo se realiza hasta que se llega al cuadro N-ésimo o final y luego se vuelve a ejecutar el detector de objetos de la fase 1. Finalmente, el proceso se repite una y otra vez. [18]

Los beneficios de una solución híbrida es que se pueden aplicar métodos de detección y seguimientos de objetos de alta precisión sin tanta carga computacional, obteniendo así, un sistema más eficiente. La implementación de esta solución se puede apoyar en el lenguaje de programación python y su librería OpenCV [18].

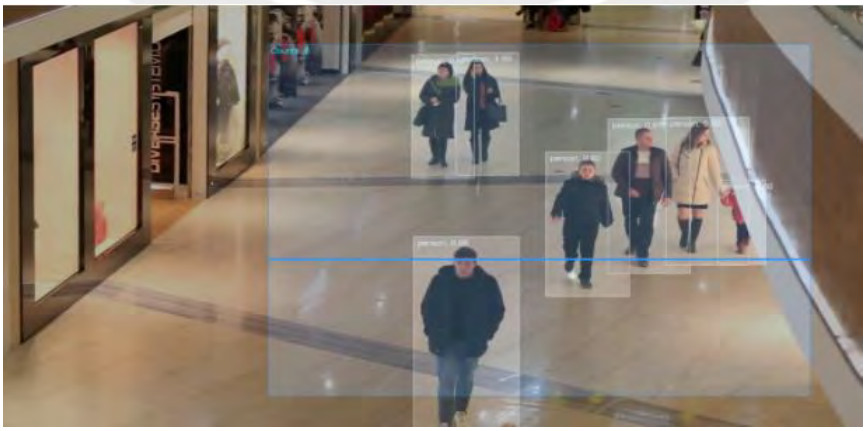


FIGURA 1.10: SOLUCIÓN IMPLEMENTADA EN UN CENTRO COMERCIAL USANDO LOS ALGORITMOS DE DETECCIÓN Y SEGUIMIENTO [19]

1.2.5.2 Algoritmo de detección facial

En ciertas aplicaciones, el cuerpo de la persona es parcialmente visible y los rostros suelen estar completamente a la vista. Además, por su lado, la detección de rostros es un algoritmo que se ha estudiado durante más tiempo y se ha generalizado en la última década, lo que hace que la tecnología sea más confiable. Por lo tanto, la solución para contabilizar a las personas en un video es mediante el conteo de rostros en un frame. Sin embargo, se tienen ciertas desventajas, ya que en ciertos entornos puede haber una superposición de un objeto o de una persona sobre el rostro de otra. Además, la detección facial es un procesamiento mucho más costoso en términos de computación, por lo tanto, no es preferible aplicarla a cada frame del video. Por esta razón, se combina con un algoritmo de seguimiento de objeto (object tracker) el cual básicamente rastrean objetos únicos de un cuadro al siguiente, de esta manera se reduce la carga computacional [20]. La estructura de funcionamiento descrita por StreamLogic [20] es la siguiente:

1) Se inicializa la cuenta y el algoritmo rastreador de objetos.
2) Para cada frame de video.
    a. Actualizar el rastreador de objetos con nuevos fotogramas de vídeo.
    b. Para cada N fotograma de video:
        i. Empieza la detección de rostro.
        ii. Se hace match de rostros con ubicaciones de objetos que ya se están rastreando.
        iii. Incrementa la cuenta por cada nuevo rostro reconocido.
        iv. Se agrega cada nuevo rostro al object tracker.
        v. Se remueve los objetos del tracker que no hayan hecho match.

Sin embargo, existe un problema que resolver, en el caso que una persona sea reconocida por el algoritmo, luego se desplaza fuera del rango de visualización por unos segundos y finalmente

vuelva a este rango, el algoritmo contará dos veces a la misma persona y existirán duplicados en la cuenta. Por esta razón, se debe hacer uso de otro algoritmo que determine si dos rostros son los mismos. La Carnegie Mellon University (CMU) ha desarrollado un algoritmo llamado OpenFace, el cual utiliza una imagen de la cara como entrada y genera una firma numérica para dicho rostro, similar a las firmas manuscritas. La firma producida por OpenFace para dos imágenes del rostro de la misma persona no es exactamente igual, sin embargo, se puede comparar por similitudes [20].

Capítulo 2: Marco teórico sobre las tecnologías implicadas en un sistema de detección de personas basado en procesamiento de imágenes

2.1 Marco teórico

2.1.1 Edge Computing

Edge computing es una infraestructura informática distribuida que se desarrolla localmente en la ubicación física del usuario o de la fuente de datos. Esta proximidad permite que los servicios sean consumidos de manera más rápida y confiable, debido a que los tiempos de respuesta son menores y se tiene una mayor disponibilidad del ancho de banda.

El crecimiento explosivo y la creciente potencia informática de los dispositivos IoT están generando grandes cantidades de volúmenes de datos. Estos volúmenes de datos seguirán creciendo a medida que se vaya desplegando las redes 5G y aumenten el número de dispositivos móviles conectados [21].

Años anteriores el compromiso que ofrecía las redes cloud y la IA era automatizar y acelerar la innovación extrayendo conocimientos prácticos de los datos. Sin embargo, estos dispositivos conectados creaban datos a una gran escala y complejidad que se han superado en muchos casos las prestaciones de infraestructura y red [21]. El envió de todos los datos generados por

los dispositivos a un centro de datos centralizado o al cloud genera problemas de ancho de banda y latencia. Edge computing resuelve esta problemática, ofreciendo una alternativa más eficiente ya que los datos son procesados y analizados localmente, como los datos no tienen que atravesar una red hasta un cloud o centro de datos para recién ser procesados y analizados, el tiempo de respuesta y latencia se reduce significativamente, generando así una mejor experiencia para el cliente final. [21]

Muchos casos en la vida real de Edge computing surgen del paradigma de procesar datos de manera local e inmediata, porque al usar el sistema tradicional de enviar hacia un centro para procesar la data, se pueden generar niveles de latencia inaceptables. Un ejemplo claro es en las plantas de fabricación moderna, los sensores IoT generan flujos constantes de datos, que puede llegar hasta 2200 terabytes de datos al mes, considerando 2000 equipos. Por esta razón, es más rentable y rápido procesar esos datos, en equipos que se encuentren cerca de los sensores IoT. Otro ejemplo se da en los servicios populares de internet, las redes de distribución de contenido implementan servidores de datos cerca de los usuarios finales, permitiendo así, que los sitios web carguen rápidamente y se agilicen los servicios de transmisión de video [22]. Además, el Edge computing reduce el riesgo de permitir el acceso a los datos confidenciales, ya que mantiene toda esa potencia informática en un lugar cercano al de obtención de los datos.

Por otro lado, un sistema de Edge computing tiende a ser más complejo y menos adaptable que una arquitectura de nube centralizada, por lo tanto, se debe detectar si realmente es la solución adecuada y necesaria, ya que una implementación prematura, sin la adecuada planeación, puede ser contraproducente. Incorporar varios servidores de Edge computing en varias áreas pequeñas puede ser más complejo y costoso que solo aumentar la capacidad equivalente en un solo centro de datos principal [22].

En la figura 2.1, se observa una arquitectura de Edge Computing, en la cual se pueden ver en la base a los dispositivos que sensan y/o recopilan la información. En el siguiente nivel, se

encuentra el Edge, que es donde están los servidores y/o equipos encargados de procesar la data recolectada. Finalmente se sube a la red de internet esta data ya procesada, la cual es de menor volumen que la data inicial.

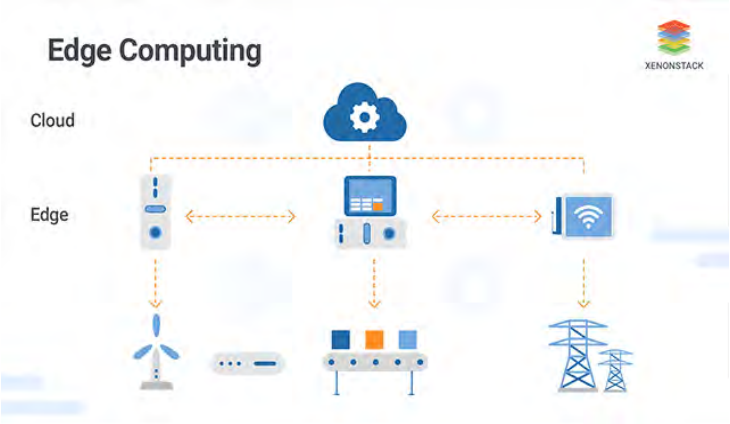


FIGURA 2.1: ARQUITECTURA DE EJEMPLO PARA UNA SOLUCIÓN DE EDGE COMPUTING [23]

2.1.1.1 Partes de una red de Edge Computing

Según RedHat [22], las partes que conforman una red Edge computing son:

- **Sistema principal de la empresa o del proveedor**: son los sistemas tradicionales que no se encuentran en el "extremo de la red", son aquellos que pertenecen a los proveedores de nube pública, proveedores de telecomunicaciones, etc [22].

FIGURA 2.2: PROVEEDORES DE NUBE [24]

- **Extremo del proveedor de servicio**: son aquellos que se encuentran entre los centros de datos y el último tramo de la red. Generalmente pertenecen a empresas de telecomunicaciones o proveedores de servicio de internet, desde aquí es donde estos proveedores prestan sus servicios a varios clientes [22].

FIGURA 2.3: PRINCIPALES PROVEEDORES DE SERVICIO DE INTERNET EN PERÚ [25]

- **Extremo de las instalaciones del usuario final**: este tramo es el Edge computing de la empresa. Es el equipo de procesamiento que recolecta la data y la convierte en información [22].

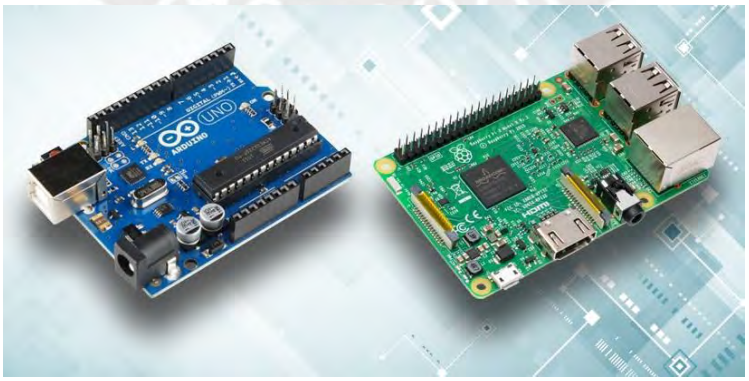


FIGURA 2.4: EQUIPOS DE PROCESAMIENTO [26]

- **Extremo del dispositivo**: Sistemas independientes que conectan directamente los sensores a través de protocolos que no son de internet. Es el extremo más alejado de la red [22].

2.1.2 Inteligencia Artificial

La red neuronal convolucional (CNNs) es una técnica de inteligencia artificial para el procesamiento de imágenes, pero en especial es dedicada al uso en métodos de reconocimiento y clasificación. A partir de un dato de entrada se tiene una clasificación por medio de un modelo entrenado; entonces esta idea puede ser llevada a entradas pertenecientes a una imagen de modo que se pueda ser clasificada o detectar un objeto y/o persona; para ello las redes neurales convolucionales usan un procesamiento pequeño con respecto a las demás redes neuronales convencionales. La CNN son entrenadas por imágenes usando píxeles y etiquetas como entrada, de modo que usan filtros para aplicarlos a las imágenes que se desean procesar; ya que sobre la imagen a detectar se aplican filtros para así obtener patrones y conocer donde el filtro coincide con el contenido proporcionado por la foto [27]. En la figura 2.5 se muestra la estructura de una red neuronal convolucional para la clasificación y/o detección de imágenes.

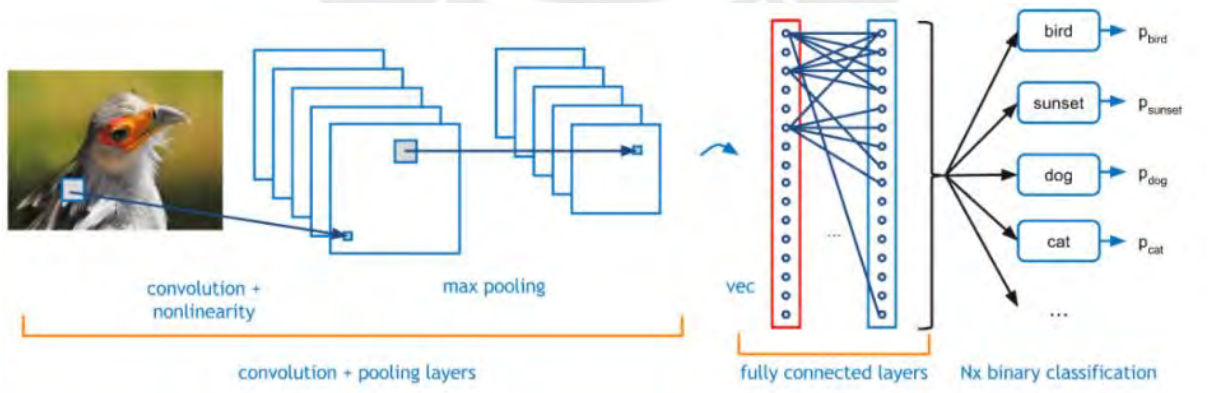


FIGURA 2.5: ESTRUCTURA DE UNA RED NEURONAL CONVOLUCIONAL PARA LA CLASIFICACIÓN Y/O DETECCIÓN DE IMÁGENES [27]

Estos tipos de redes presentan algoritmos para la clasificación y detección de objetos, los cuales se explicarán a continuación.

2.1.2.1 Faster R-CNN

En el caso de CNN se requerían de la búsqueda selectiva de regiones, lo cual lo convertía en un proceso lento que afectaba directamente en el rendimiento de la red neuronal, es por ello que no debería usarse una búsqueda selectiva de regiones, sino que en su lugar la misma red aprenda de las regiones. En Faster R-CNN la imagen se proporciona como entrada a una red convolucional que proporciona un mapa de características convolucional. En lugar de usar un algoritmo de búsqueda selectiva en el mapa de características para identificar las propuestas de región, se usa una red separada para predecir las propuestas de región; estas luego se reforman usando una capa de agrupación de "Rol" para la clasificación y predicción de valores de desplazamiento para los cuadros delimitadores. En la Figura 2.6 se muestra la arquitectura de Faster R-CNN [28].

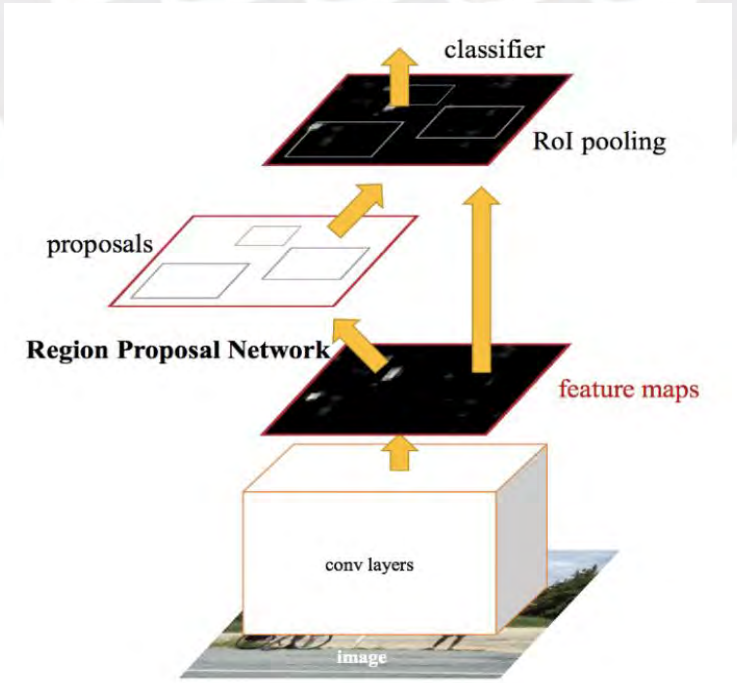


FIGURA 2.6: ARQUITECTURA DE FASTER R-CNN [28]

En la figura 2.7 un gráfico comparativo de la velocidad de detección de los algoritmos en segundos.

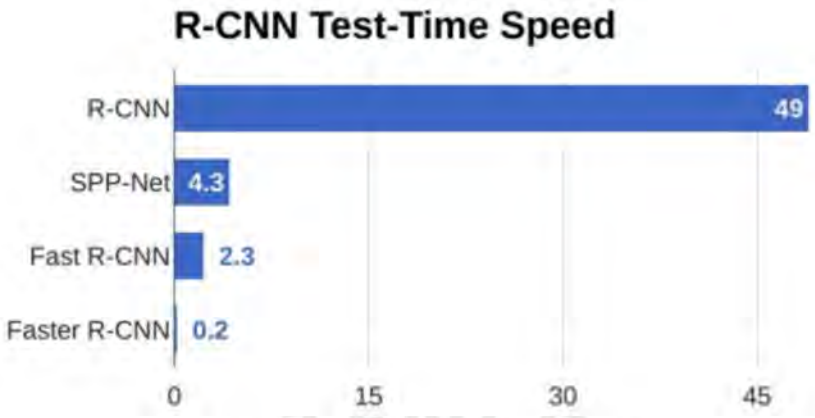


FIGURA 2.7: GRÁFICO COMPARATIVO DE LA VELOCIDAD DE DETECCIÓN DE LOS ALGORITMOS R-CNN, SPP-NET, FAST R-CNN Y FASTER R-CNN [28]

Como se observa en la figura 2.7, Faster R-CNN es más rápido que sus antecesores, por lo que permite que pueda ser usado para aplicaciones en tiempo real.

2.1.2.2 You Only Look Once (YOLO)

Este algoritmo es distinto a los previamente explicados, debido a que no se basa en el uso de regiones para la detección en imágenes, si no predice cuadros delimitadores y probabilidades de clases para ellos; además este es usado en aplicaciones de detección en tiempo real por su velocidad y precisión. El funcionamiento de tres técnicas donde la primera de ellas es el uso de bloque residuales y ella consiste en el fraccionamiento de la imagen en cuadrículas con una dimensión de SxS detectando los objetos en cada una de ellas [29]. La figura 2.8 se muestra un ejemplo de bloques residuales.

FIGURA 2.8: EJEMPLO DE BLOQUES RESIDUALES [29]

En segundo lugar, la regresión del cuadro delimitador es la técnica encargada de resaltar el objeto detectado en la imagen a partir de un valor probabilístico, designado a las cuadrículas (técnica previa) [29]. En la figura 2.9 se muestra la probabilidad de que un objeto se pueda encontrar en el cuadro delimitador.

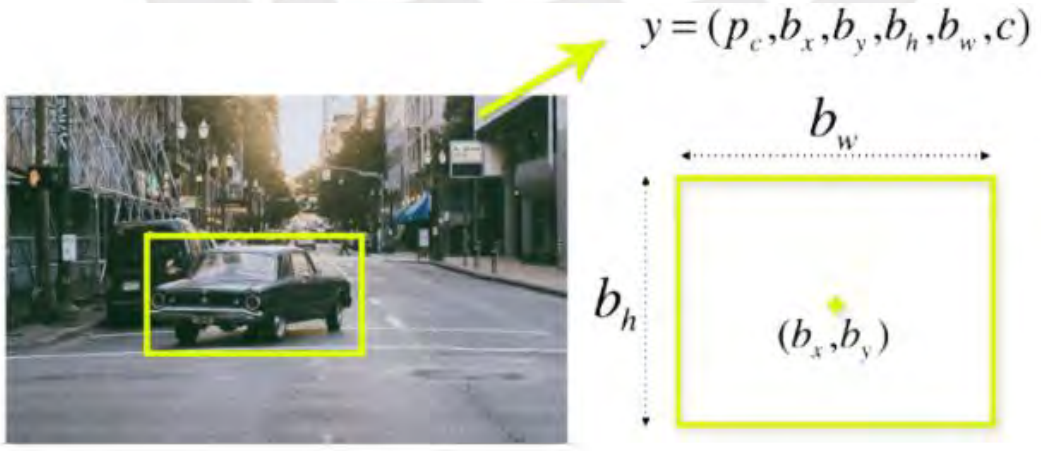


FIGURA 2.9: PROBABILIDAD DE QUE UN OBJETO SE ENCUENTRE EN EL CUADRO DELIMITADOR [29]

Por último, se tiene la intersección sobre unión (IOU) que es un fenómeno común en detección de objetos descrito como la superposición de cuadros; entonces YOLO usa IOU para mostrar a la salida el cuadro que englobe al objeto detectado partiendo de que, si el valor del IOU es uno, el cuadro delimitador es el mismo que el cuadro real por lo que así se eliminan los cuadros delimitadores que no se asemejen al objeto detectado [29]. En la figura 2.10 se muestra el funcionamiento de IOU, puesto que el cuadro verde representa el cuadro delimitador real, mientras que el azul es el que se predijo de manera que YOLO se encarga de que estos sean los mismos.

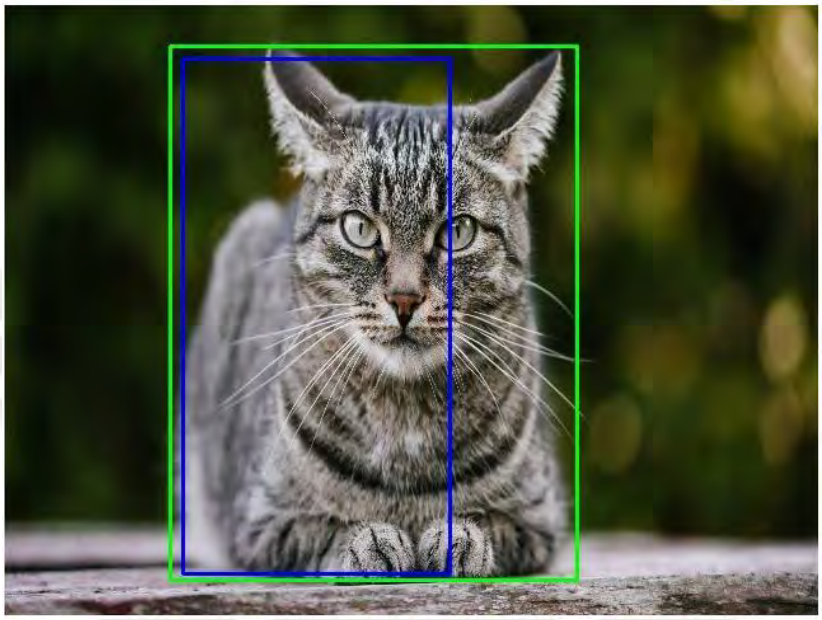


FIGURA 2.10: FUNCIONAMIENTO DE IOU [29]

Con lo ya mencionado, YOLO hace uso de las técnicas para su funcionamiento en aplicaciones de detección de objetos y en la figura 2.11 se muestra la aplicación y funcionamiento en conjunto.

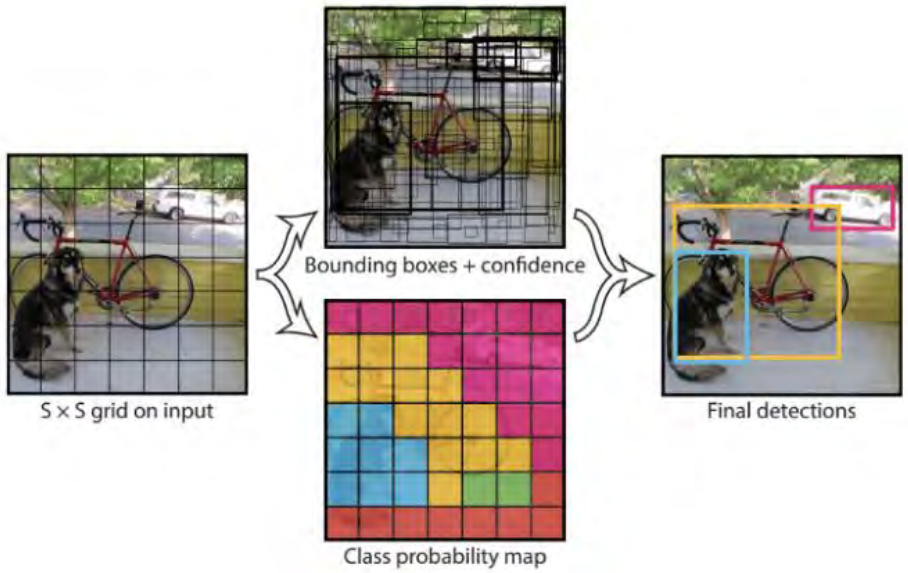


FIGURA 2.11: FUNCIONAMIENTO DEL ALGORITMO YOLO [29]

Por otro lado, uno de los grandes inconvenientes que posee YOLO es la detección de objetos muy pequeños como es el caso de un accesorio de una mascota; entonces limitaría su uso para determinadas aplicaciones [29].

2.1.3 Ordenadores de Placa Simple (SBC)

Una computadora de una sola placa o single-board computer (SBC) en inglés, es una computadora completa construida en un solo circuito con microprocesadores, memoria, periféricos de entradas/salidas y demás características requeridas de tamaño reducido que tiene una computadora funcional [30].

Por otro lado, se tienen los microcontroladores, el cual es un circuito integrado programable, capaz de ejecutar ordenes grabadas en su memoria. Al igual que el SBC incluye en su interior las 3 principales unidades funcionales: unidad central de procesamiento, memoria y periféricos de entrada y salida. Se usa para aplicaciones en específico, en comparación de una computadora

de una sola placa que está conformada por microprocesadores, son más lentos ya que realizan menos instrucciones por segundo [31]. A continuación, en el cuadro 2.1 se muestran las diferencias entre ambos paradigmas.

CUADRO 2.1: COMPARACIÓN DE MICROCONTROLADOR Y MICROPROCESADOR [31]

Microcontrolador	**Microprocesadores como parte de una SBC**
El microcontrolador actúa como el corazón del sistema integrado	El/Los microprocesador(es) actúa(n) como el corazón del sistema informático.
La memoria y los componentes de entrada y salida están presentes internamente. Por lo tanto, el circuito es menos complejo	La memoria y los periféricos de entrada y salida están conectados externamente. Por lo tanto, el circuito es más complejo
Se pueden utilizar en sistemas bien compactos	Es ineficaz en sistemas compactos debido a su mayor tamaño
El microcontrolador tiene más registros. Por lo tanto, un programa es más fácil de escribir.	El microprocesador tiene menos registros, por lo tanto, la mayoría de las operaciones se basan en la memoria.
Lenta comparado con el microprocesador, la cual tiene velocidades de operación en el orden de los KHz a MHz	Una mayor frecuencia de reloj, llegando al orden de los GHz
El costo de un microcontrolador es económico comparado a un microprocesador o SBC	Un microprocesador es más costoso.
Como ejemplo se tiene a Arduino, una placa basada en un microcontrolador ATMEL	Como ejemplo se tiene a Raspberry, el cual es un Single-board computer compuesto de varios microprocesadores.

2.1.3.1 Arduino

Es una placa de desarrollo libre de hardware que incorpora un microcontrolador reprogramable y una serie de pines hembra que permiten establecer conexiones entre el microcontrolador y los diferentes sensores y actuadores de una manera realmente sencilla. Se puede construir dispositivos digitales y dispositivos interactivos que puedan detectar y controlar objetos del mundo real [32]. Además, Arduino como empresa también ofrece software libre mediante la plataforma Arduino IDE, el cual es un entorno de programación para crear aplicaciones para las placas Arduino.

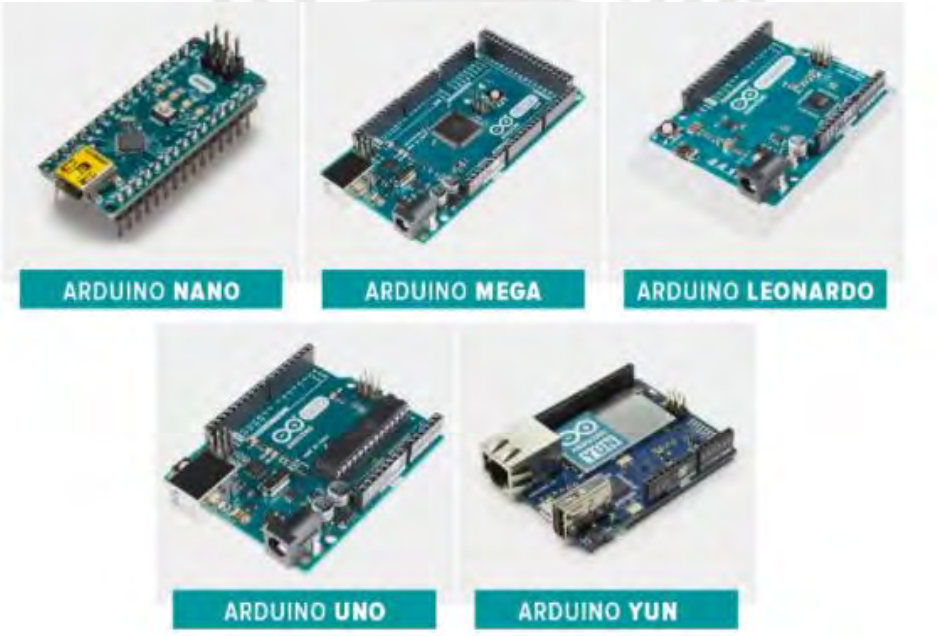


FIGURA 2.12: MODELOS DE ARDUINO [32]

2.1.3.2 Raspberry PI

Es una serie de pequeños single board computer, la cual fue desarrollada en Reino unido para promover la informática y la creación digital en las escuelas. Sin embargo, hoy en día, es usado en más ámbitos como es la robótica [33]. No incluye periféricos de entrada y salida. En la

actualidad ya se encuentra en la versión 4 modelo B, lanzada en junio de 2019, con versiones de 2 GB,4GB y 8GB de RAM.

FIGURA 2.13: RASPBERRY PI 4B [33]

2.1.3.3 NVIDIA Jetson

La marca americana diseñadora de unidades de procesamiento de gráficos GPU, también desarrolla placas informáticas integradas. Se tienen los modelos Jetson TK1, TX1 y TX2 los cuales integran un procesador tegra que contiene una CPU de arquitectura ARM. Está diseñada para aplicaciones embebidas, IoT y aplicaciones de inteligencia artificial, debido a su pequeño tamaño, potente rendimiento y una buena eficiencia energética [34].

FIGURA 2.14: NVIDIA JETSON [34]

2.1.4 Protocolos de Aplicación

El ordenar de placa simple (SBC) será el encargado de procesamiento de la información, pero para que ella pueda ser enviada a un web server existen dos protocolos de aplicación de manera que la información procesada puede ser vista en tiempo real, estos son:

2.1.4.1 MQTT

Este es un protocolo de mensajería estándar de OASIS para aplicaciones de Internet en las cosas (IoT). MQTT está diseñado como transporte de mensajería publicación/suscripción muy liviana que funciona idealmente para la conexión de equipos remotos que tengan una huella de código pequeña; además de funcionar con un ancho de banda de red limitado, es por ello que su uso es frecuente en áreas industriales, automotriz, fabricación, telecomunicaciones, etc [35]. Los beneficios proporcionados por este protocolo son:

- **Ligero y Eficiente**: Los clientes usados en MQTT son pequeños, por lo que requieren de recursos mínimos como el caso de microprocesadores. Las cabeceras de este protocolo son pequeñas para así optimizar el ancho de banda de la red [35].
- **Entrega de mensajes confiables**: Como la seguridad es importante en muchos tipos de aplicaciones de IoT, MQTT presenta 3 niveles de Calidad de servicio (QoS) los cuales son: 0 - como máximo una vez (se puede no recibir mensajes), 1 – al menos una vez (se asegura la recepción de mensajes, pero puede haber duplicados) y 2 – exactamente una vez (El mensaje es recibido sin duplicados) [35].
- **Soporte para redes no confiables**: Puede darse el caso de que la conexión de equipos esté bajo una red celular no confiable, por ello el soporte de este protocolo para sesiones persistentes reduce el tiempo para la reconexión del cliente con el broker [35].

- **Seguridad**: Presenta cifrado de mensajes por medio de TSL y autenticación entre clientes por medio de Oauth [35].
- **Escala a millones de cosas**: Escala para conectarse con millones de dispositivos IoT [35].

La arquitectura de publicación/suscripción de MQTT a diferencia del modelo tradicional de cliente y servidor, los clientes MQTT son divididos en dos grupos donde uno es el remitente (el publicador) y el consumidor de datos (el suscriptor), donde estos dos desconocen información uno del otro porque no están conectados directamente; ya que el tercer componente es el MQTT broker encargado de dirigir la data del publicador a los puntos finales (suscriptores) [35]. En la figura 2.15 se muestra la arquitectura de MQTT.

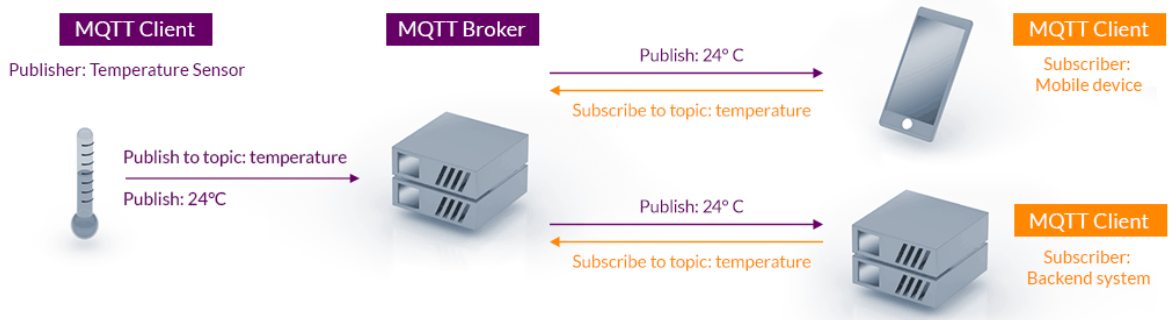


FIGURA 2.15: ARQUITECTURA DE MQTT [35]

2.1.4.2 HTTP

Este protocolo lo que permite es intercambio de datos hacia un web server bajo el modelo de cliente servidor, lo cual implica que comúnmente las solicitudes sean iniciadas por el navegador web; además se envía por medio de TCP en una conexión cifrada con TLS [36]. Los aspectos presentados por HTTP son:

- **Extensible**: La cabecera de este protocolo hace que este sea sencillo de extender y experimentar, ya que se pueden implementar nuevas funcionalidades por medio de un acuerdo entre el cliente y servidor sobre un nuevo encabezado [36].
- **Sin estado, pero con sí con sesión**: No tiene estados, puesto que no existe algún vínculo entre dos solicitudes que se produzcan sucesivamente sobre una misma conexión; mientras que las cookies HTTP permiten que exista el uso de sesiones con estado [36].
- **Entrega de mensajes**: HTTP no requiere un protocolo de transporte subyacente que esté basado en conexiones, solo requiere que sea confiable es decir que no pierda mensajes (como mínimo presentando un error los casos) [36].

La arquitectura de HTTP esta dado bajo el modelo de cliente y servidor, donde por un lado el cliente encargado de realizar consultas tipo GET o POST de forma de publicar o recibir información de un web server; además este siempre es el ente que inicia la solicitud. Por otro lado, se tiene el servidor encargado de resolver la consulta establecida con el cliente [36]. En la figura 2.16 se muestra la arquitectura de HTTP.

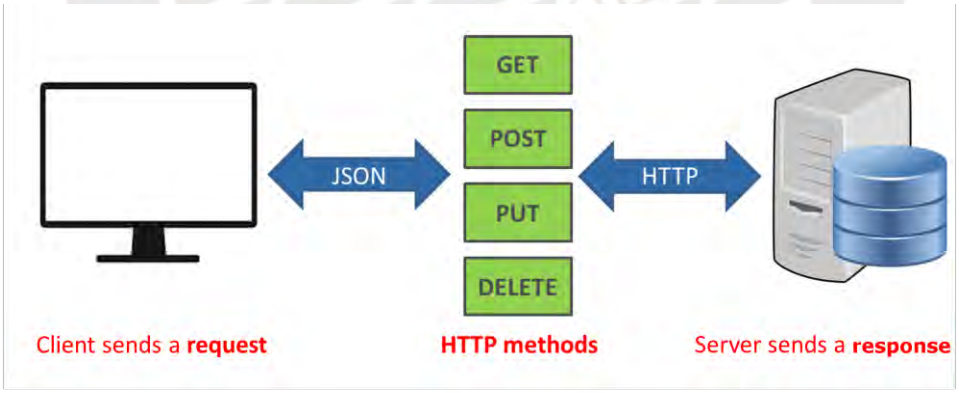


FIGURA 2.16: ARQUITECTURA DE HTTP [37]

2.1.5 Frameworks para desarrollo de back end

Back end representa el grupo de recursos del servicio web por lado del servidor dentro de un modelo de cliente-servidor; por ejemplo, la conexión a base de datos, obtención de información mediante API's y guardado en base de datos son ejemplos de funcionalidades que se tienen comúnmente del lado del servidor. En la figura 2.17 se muestra la arquitectura web tradicional de forma que se tenga una diferencia clara entre back end y front end.

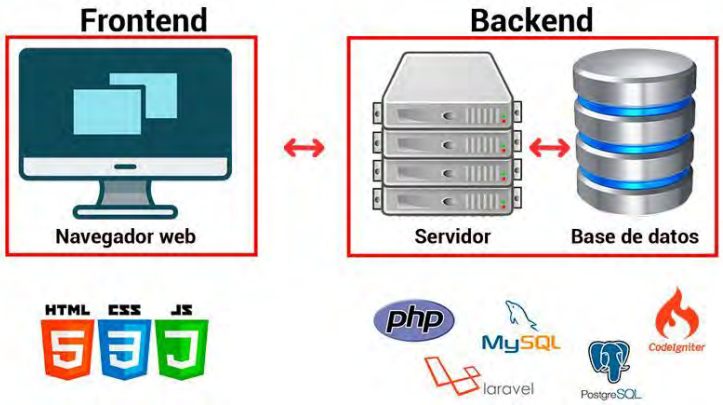


FIGURA 2.17 ARQUITECTURA WEB [38]

Para cada lenguaje de programación existen distintos tipos de frameworks para desarrollo de back end, usados de acuerdo al nivel de proyecto que se desea llegar; por ello estos deben conseguir abarcas con la gran mayoría de funcionamientos requeridos por un proyecto, de manera que los requerimientos determinados se realicen y también sean funcionales a medida de los avances tecnológicos que tengan las librerías y APIS usadas; de igual manera que front end, un diseño escalable y buenas prácticas de programación son características importantes en la etapa de realización. A partir de lo ya mencionado, la investigación se va a centrar en los frameworks para desarrollo web de los lenguajes de programación de Java y Python, ya que según PYPL (Popularity of programming language) [39], estos dos lenguajes ocupan los 2 primeros puestos en su top 10 de lenguajes de programación más populares. Por otro parte, se describirá Django y Spring Boot que son los 2 frameworks que ocupan los mejores puestos con

respecto al lenguaje en el que están basados; además de estar dentro del market share del 2021 brindado por la página web Space technologies, asimismo en el benchmark del 2020 publicado en el sitio web Black4app [40].

2.1.5.1 Django

Django es un framework para desarrollo web capaz de realizar aplicaciones de todo tipo de complejidad; asimismo los lanzamientos de estas se realizan de manera eficaz y rápida, ya que este se encarga de gran parte de las funciones requeridas. Django está dado para el lenguaje de programación Python y cuenta con el uso de librerías (funcionalidades propias de Python) que refuerzan el proyecto para la integración de funciones disponibles como envíos de correos, videoconferencias, chats, pasarela de pagos, entre otros. Por otro lado, al ser de código abierto (open source) lo que involucra que sea gratis y presente una documentación para su instalación y uso, dentro de la información brindada se encuentran módulos con la capacidad de trabajar con un mayor orden y sobre todo se encarga de tareas de manera automatizada, de manera que se tiene un gran ahorro de tiempo. Por ejemplo, dentro de los módulos están DjangoForms y Models, donde el primero de ellos facilita la creación y validación de campos dentro de formularios solo configurando parámetros de entrada en funciones propias al módulo; mientras que el último de ellos permite el modelamiento de base de datos desde la misma aplicación especificando el tipo de datos y características de cada uno, asimismo cuenta con un control de historial de migraciones (cambios en base de datos), lo cual es conveniente para un buen manejo de los cambios realizados cuando se trabaja en conjunto. Con lo previamente mencionado, Django es una herramienta completa para el desarrollo de un servicio web, aunque presenta algunos contras como no ser bueno para aplicaciones a pequeña escala, no se pueden manejar múltiples solicitudes de manera simultánea, entre otras [41].

2.1.5.2 Spring Boot

Este framework enfocado en el lenguaje de Java y siendo basado en Spring (framework para desarrollo web), por lo que las configuraciones necesarias no son tan complejas; además hace uso de las librerías de terceros para facilitar el trabajo y hace uso de ellas como propias por medio de dependencias añadidas al proyecto, las cuales son las librerías starter. Por otra parte, se tienen características importantes como es el manejo de seguridad de acceso a los módulos correspondientes a roles determinados, por lo que se garantiza que los usuarios no accedan a vistas no permitidas; asimismo cuenta con librerías como Spring Session y Spring Data JPA, donde el primero de ellos es útil para el uso de información volátil durante una sesión activa y un gran conjuntos de drivers como JDBC (base de datos), Redis (base de datos en memoria), entre otras; mientras que el segundo permite mapear y hacer consultas de una tabla completa en base de datos con el sistema de Entidad, Repositorios y Controladores [42]. A partir de lo mencionado previamente, definiremos lo siguiente:

- Entidad: Son las clases y objetos de java que representan una tabla de base de datos, que son identificadas por medio de la etiqueta "@Entity" y a la vez son referenciadas a la tabla por "@Table" indicando el nombre de la misma.[42]
- Repositorio: Son las interfaces que permiten la gestión de acceso a la información de base de datos y se hace uso de uno por cada entidad; entonces se pueden hacer CRUD's por medio de consultas usando métodos existentes o con el tag "@Query" que permite hacer una consulta personalizada. [42]
- Controlador: Controlador es la clase de java encargada del funcionamiento de todos los servicios brindados en el sistema y está denotado por "@Controller" y de igual manera lo común es el uso de uno de estos por entidad o roles determinados. [42]

Los módulos explicados forman parte de la arquitectura MVC (Modelo-Vista-Controlador) usada por Spring para poder seguir el patrón de Front Crontoller, sin embargo, se presentan

ciertos aspectos negativos; por ejemplos que la integración de algunas aplicaciones de terceros llega a complicarse en cierto instante, también es un framework que requiere de experiencia de uso por parte del desarrollador a un mayor nivel, etc. [43] El flujo de la arquitectura MVC se presenta en la Figura 2.18.

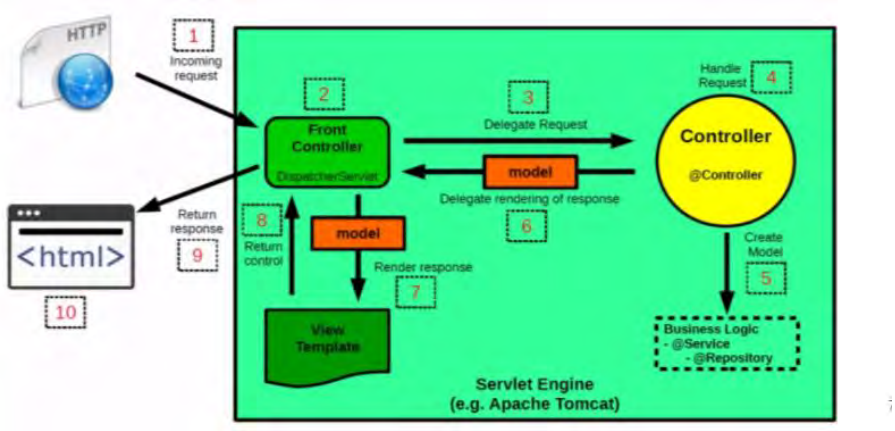


FIGURA 2.18 FLUJO DE ARQUITECTURA MVC [43]

2.1.6 Software de Base de Datos Relacionales

Estos softwares nos permitirán es el almacenamiento de data en bases de datos con la finalidad de que la información pueda ser usada para las diferentes posibles modelos; así como Datos del usuario, términos y condiciones, sesión, roles, entre otros; son llamados relacionales porque se emplea el uso de tablas y estas pueden tener una relación por medio de una llave foránea, la cual es el parámetro identificador de una tabla. Según Datanyz [44], las dos tecnologías open source (código abierto) en su market share realizado en el 2021 para bases de datos relacionales son MySQL y PostgresSQL, por lo que se explicarán en este subcapítulo.

2.1.6.1 PostgreSQL

PostgreSQL pretende cumplir con el estándar SQL, de manera que cumpla con características generales y no guíe al usuario a optar por decisiones erróneas para su arquitectura. A partir del lanzamiento de su versión 13 en septiembre del 2020 cumple con 170 de los 179 aspectos

obligatorios por SQL Core 2016, que hasta en esa fecha era el único que cumplía plenamente con el estándar. Cuenta con los siguientes aspectos: [45]

- Tipos de datos: Cuenta con distintos tipos de datos al igual que otros, pero su diferencia entra en que se pueden customizar y crear tipos de datos configurables. [45]
- Rendimiento: Cuenta con indexaciones comunes y avanzadas de maneras que la búsqueda de información es mucho más rápida, asimismo cuenta con un planificador de consultas sofisticado y optimizado, transacciones y transacciones anidadas. Por otro lado, cuenta con un control de versiones de manera que hay un control de cambios en las tablas existentes. [45]
- Fiabilidad y recuperación frente a desastres: Hace uso del protocolo de registro de escritura anticipada (WAL) por lo que se tiene un seguro con respecto al almacenamiento de datos; por otro parte, cuenta con replicación síncrona, asíncrona y lógica para la recuperación en un momento determinado (PITR) en caso de fallos. [45]
- Seguridad: Presenta autenticaciones como GSSAPI, SSPI, LDAP, SCRAM-SHA-256, Certificado y más; del mismo modo, Seguridad a nivel de columna y fila, un sistema de control del acceso a la información y Autenticación multifactor con certificados. [45]

Estas son las razones por las que PostgreSQL es escogida por múltiples personas y organizaciones para almacenar información sensible.

2.1.6.2 MySQL

MySQL es la base de datos open source más usado en el mundo, puesto que brinda de manera rentable el envío de transmisión confiable, asimismo es de alto rendimiento y escalable, procesamiento de transacciones integradas y aplicaciones integradas de base de datos. MySQL ofrece facilidades de uso para el usuario; además de un conjunto completo de controladores de

bases de datos e interfaces visuales (Workbench) para administración y gestión. Se proporcionan las siguientes características: [46]

- Presenta un esquema de rendimiento para su supervisión y el conjunto de recursos a nivel de usuario como de aplicación. [46]
- Posee independencia de la plataforma, debido a que ofrece flexibilidad para la implementación de múltiples sistemas operativos. [46]
- Gestión de tablas relacionadas y archivos JSON sin la necesidad de un esquema en una única base de datos. [46]
- En seguridad cuenta con autenticación para limitar el acceso a los usuarios; además protege la información por medio de cifrado de data transparente (TDE). [46]
- La posibilidad de generación de backups para el respaldo y pueden ser completas, incrementales y parciales para los casos de fallos. [46]
- Posee un firewall en caso de posibles ataques por medio de inyección SQL, por lo que protege la información sensible. [46]

Por lo ya mencionado, este software nos ofrece un funcionamiento completo y con un gran rendimiento para el alojamiento de información requerida.

2.1.7 Bases de Datos No Relacionales

Las bases de datos no relacionales (NoSQL) están diseñadas para modelos de bases de datos característicos y con flexibles esquemas para su uso en aplicaciones y enfocado al uso flexible de información (gran volumen) y de alto rendimiento [47]. Según Techgig [48], las 2 bases de datos no relacionales basadas en documentos (se hacen consultas hacia las mismas en un formato de documento como JSON) son MongoDb y Amazon DynamoDB, por lo que se explicarán continuación.

2.1.7.1 MongoDB

Según su plataforma oficial [49], MongoDB es la base de datos NoSQL opensource basada en documentos más popular en el mercado; ya que envía entre a 3 y 5 veces más rápido la data. El modelo de documentos de este hace que sea sencillo de usar y brinda a los desarrolladores variedades de funcionalidades para los requerimientos más complicados y a diferentes escalas, también proporcionan drivers para más de diez lenguajes; por lo que su popularidad es muy grande y creciente [49]. A continuación, se presentarán sus principales beneficios:

- La estructura de puede cambiar todo el tiempo debido a que se almacenan datos en documentos flexibles parecidos al formato JSON. En la figura 2.19 se muestra un ejemplo de la estructura de datos [49].

FIGURA 2.19 EJEMPLO DE ESTRUCTURA DE DATOS MONGODB [49]

- Para hacer más sencillo el trabajo de datos, el modelo de documentos es asignado a los objetos en el código desarrollado de la aplicación [49].
- La alta disponibilidad, escalabilidad horizontal y distribución son integradas y simples de usar; ya que MongoDB es una base de datos distribuida en su núcleo [49].

Por lo ya mencionado, MongoDB es usada en diferentes casos como Internet en las Cosas (IoT), videojuegos, Pagos, entre otros; por ello entre sus clientes figuran Google, eBay, Adobe, Sega, EA, Verizon, etc.

2.1.7.2 Amazon DynamoDB

Amazon DynamoDB es una base de datos no relacional que acepta los modelos de datos de documentos y clave valor. Este servicio es usado para aplicaciones modernas y sin servidores (serverless) que pueden iniciar desde una escala pequeña y luego llegar a una escala mundial de modo que se permitan hasta petabytes de data y decenas de millones de solicitudes tanto de escritura como lectura [50]. A continuación, se presentarán sus principales aspectos:

- **Rendimiento a Escala**: DynamoDB admite tables de casi cualquier tamaño con un escalado horizontal, lo que permite que escale a más de 10 billones de solicitudes de escritura y lectura por día; además de soportar hasta un pico de 20 millones de solicitudes por segundo [50].
- **Sin servidor**: No presenta servidores que administrar, por lo que no hay softwares que instalar, mantener o usar; además esta base de datos permite reducir o aumentar de forma automática las tablas de forma que ajustan las capacidades y mantienen el rendimiento sin necesidad de ser administrados [50].
- **Listo para uso empresarial**: Está diseñado para soportar cargas de trabajo de nivel crítico, también incluye las transacciones ACID para múltiples conjuntos de aplicaciones que necesiten de una lógica de negocio compleja [50].

Por lo mencionado previamente, este servicio de base de datos NoSQL es una posible opción para almacenar un gran volumen de información.

2.1.8 Proveedores de Cloud Computing

Un servicio web desplegado en un servidor cloud puede ser accedido desde cualquier parte del mundo y en cualquier momento por los usuarios objetivos (alumnos y/o personal de la biblioteca), debido a que estaría situado en la red global. Por lo tanto, se deben tener en cuenta las ventajas y desventajas de los proveedores de tal manera que en balance se pueda aprovechar los recursos brindados a un costo moderado; por lo que se explicarán 3 proveedores de Cloud Computing considerados como los más usados a partir del reporte publicado por Canalys [50] en el 2021, sin embargo, solo serán mencionados los módulos necesarios para el alojamiento web.

2.1.8.1 Google Cloud

Este proveedor nos brinda directamente el servicio de alojamiento por medio de Google App Engine, la cual es la encargada de tener activa (compilado) la arquitectura de software definida por medio de configuraciones necesarias en la consola brindada por Google. Las características claves son:

- Compatibilidad con lenguajes de programación: Es capaz de compilar distintos grupos de lenguajes, pero lo importantes para esta investigación es que tanto Python como Java pertenecen al grupo. [51]
- Abierto y flexible: Gracias al uso de suministro de un contenedor Docker, posee la capacidad de usar cualquier librería y framework por medio de entornos de ejecución customizado. [51]
- Control de versiones de aplicaciones: Puede manejar tanto una versión de desarrollo, etapas de pruebas y una versión de producción de la misma aplicación sin complicación alguna. [51]

- Seguridad para aplicaciones: Al presentar un firewall propio de App Engine, se definen reglas para el acceso; asimismo aprovecha los certificados SSL/TLS disponibles de forma predeterminada del dominio propio de la página sin un cargo extra. [51]
- Balanceo de Carga: En caso de una gran demanda de personas, App Engine cuenta con un propio balanceador de carga para soportar muchos usuarios simultáneamente. [51]

Google brinda instancias (máquinas virtuales) dentro de App Engine para la compilación del sistema. A continuación, presentaremos sus tipos y sus costos respectivos:

Clase de instancia	Límite de memoria	Límite de CPU	Tipos de escalamiento admitidos
F1 (predeterminada)	256 MB	600 MHz	automático
F2	512 MB	1.2 GHz	automático
F4	1,024 MB	2.4 GHz	automático
F4_1G	2,048 MB	2.4 GHz	automático
B1	256 MB	600 MHz	manual, básico
B2 (predeterminada)	512 MB	1.2 GHz	manual, básico
B4	1,024 MB	2.4 GHz	manual, básico
B4_1G	2,048 MB	2.4 GHz	manual, básico
B8	2,048 MB	4.8 GHz	manual, básico

FIGURA 2.20 TIPOS DE INSTANCIAS EN GOOGLE CLOUD - JUNIO 2021 [51]

Clase de instancia	Costo por hora, por instancia
B1	$0.05
B2	$0.10
B4	$0.20
B4_1G	$0.30
B8	$0.40
F1	$0.05
F2	$0.10
F4	$0.20
F4_1G	$0.30

FIGURA 2.21 COSTOS POR TIPO DE INSTANCIAS EN GOOGLE CLOUD-JUNIO 2021 [51]

Si bien los costos mostrados pertenecen a la región más cercana al Perú, dependen de la ubicación de la instancia. Por otro lado, si bien exponen que uno de sus beneficios es que escala de forma automáticamente a medida del flujo de usuarios, la infraestructura es desconocida; entonces sería una "caja negra" por lo que no se podrá observar ni modificar la topología de red usada, puesto que Google lo maneja en su totalidad.

2.1.8.2 AWS Educate

Por el lado de Amazon, debemos tener en cuenta el uso de máquinas virtuales (instancias EC2) en conjunto con su servicio de base de datos relacionales (RDS), ambos conforman la arquitectura deseada para el alojamiento. Por el lado de las instancias, cuentan con muchos tipos que van acorde a las características de hardware necesarias para el rendimiento necesario y el precio será mayor al tener mejores aspectos; sin embargo, si bien es un beneficio poder tener una gran distribución de máquinas virtuales, la arquitectura no llega a ser escalable porque dependerá de lo máximo que puedas usar de tu máquina virtual. Por otra parte, la topología dependerá del usuario de AWS, ya que tanto la seguridad como el acceso a la red es configurable; además las máquinas pueden ser apagadas y prendidas en cualquier momento para una posible reducción de costos en una posible etapa de desarrollo, también cuenta con la posibilidad de usar IP's elásticas (costo adicional) para que se mantenga siempre al iniciar el EC2 y acceder a la página web desde cualquier navegador. Por lo mencionado previamente, notamos que dependiendo del flujo de personas simultáneas en la página hará que sea necesario el uso de balanceo de carga, pero AWS no cuenta con uno por lo que se necesitaría usar uno externo para solucionar su ausencia [52]. En caso del RDS, se tienen los siguientes beneficios:

- Sencilla administración: Brinda la facilidad de distintos medios de uso, los cuales son la consola de administración de AWS, la interfaz de líneas de comandos de AWS RDS, o llamadas a la API, también al levantar instancias de tipo RDS presentan

configuraciones predeterminadas adecuadas al motor escogido, Amazon se encarga de las actualizaciones necesarias del software seleccionada, brinda sugerencias sobre buenas prácticas, etc. [52]

- Rendimiento: El almacenamiento usado es de tipo SSD (Unidad de estado sólido) con base de 2 IOPS, de manera que las entradas y salidas son muy rápidas, predecibles y uniformes. [52]
- Seguridad: Mediante AWS Key Management Service (KMS) se cifran las bases de datos, tanto cuando está en reposo (activo, pero sin solicitudes) como cuando hay transporte de información; además con AWS Identity and Access Management (IAM) se pueden manejar la seguridad de los usuarios que tienen acceso al RDS. [52]
- Precio: la particularidad que tiene es que presenta una cuota al mes por el uso por cada instancia RDS, entonces traería problemas en caso de que la instancia no sea usada de manera constante. Sus costos dependen del tipo de software usado; por ejemplo, el precio de PostgresSQL es distinto al de MySQL. [52]

Para tener la arquitectura completa para el alojamiento se deberá ingresar las credenciales de la base de datos en el proyecto (funcionando en el EC2) para generar la conexión con la base de datos.

2.1.8.3 Microsoft Azure

Este proveedor ofrece servicio App Service que tiene la finalidad de poder desplegar toda la arquitectura de software en su nube con los diferentes módulos que ofrece como son máquinas virtuales para compilar el código, Azure SQL para Database el almacenamiento de información, Web app firewall para proveer de seguridad a nuestra aplicación. App Service tiene las siguientes características:

- Es un servicio con la posibilidad de administración completa, por lo que en caso de cambios en cualquier aspecto puede ser realizado, asimismo cuenta con mantenimiento de infraestructura integrada, parches de seguridad y escalabilidad. [53]
- Implementación de control de versiones con la mayoría de sistemas de control de versiones, por los que no se tendría que generar los cambios manualmente en el servidor. [53]
- Integración directa con redes sociales y capacidad de ser generado en un entorno de App Service ya sea aislado o dedicado. [53]
- Posee una autenticación propia que puede ser usada para las aplicaciones por medio de Azure Active Directory, de tal manera que se restringe el acceso. [53]
- Tiene monitorización en tiempo real e interactivo de rendimiento y estado de las aplicaciones desplegadas. [53]

A partir de lo mencionado se puede notar que el servicio brindado es completo; sin embargo, presentan muchos módulos que no necesariamente serían utilizados y tienen que ser pagados porque pertenecen al precio del paquete.

2.2 Objetivos

2.2.1 Objetivo General

La presente tesis tiene como finalidad principal el diseño e implementación de un sistema basado en procesamiento de imágenes y detección de personas para la determinación de aforo en las zonas de asientos de libre disponibilidad del complejo de innovación académica (CIA), con el objetivo de mostrar en tiempo real el valor del aforo evitando aglomeraciones y brindar un sistema automatizado para controlar y monitorizar las actividades por parte del personal administrativo de la biblioteca.

2.2.2 Objetivos Específicos

- Determinar el dispositivo IoT que recolecte información del aforo en la biblioteca del CIA.
- Determinar el/los algoritmos(s) de seguimiento y detección de objetos a través de imágenes.
- Determinar el dispositivo que se encargará de procesar la data recolectada por el dispositivo IoT.
- Diseño de la arquitectura en nube para el envío, almacenamiento de datos y alojamiento de un servicio web.
- Diseño de una interfaz web adaptable a distintos tipos de dispositivos.

2.3 Alcance

Para la presente tesis se ha determinado por la evaluación del aforo de una zona específica para estudio ubicado en el primer piso del complejo de innovación académica, puesto que este piso posee solo un sector de sitios de libre disponibilidad; mientras que en otros pisos existen 2 áreas. Sin embargo, La coyuntura actual del país provocada por el virus COVID-19 ha generado que el acceso hacia estas zonas sea restringido para el público, por lo que se usarán los espacios de estudios de cada hogar de los autores de la tesis para la evaluación del diseño. Por otro lado, no se tomarán en cuenta gestiones administrativas que pueden afectar el funcionamiento típico de las zonas de estudio individuales; por ejemplo, cierre de zonas para determinados eventos. El sistema se desarrollará sobre asientos de libre disponibilidad para todo usuario.

Capítulo 3: Metodología para el diseño de un sistema de detección de personas basado en procesamiento de imágenes aplicada en una zona delimitada de asientos de libre disponibilidad de CIA

3.1 Requerimientos del sistema

Una vez recopilada la información mencionada anteriormente de cada una de las tecnologías involucradas para la solución del proyecto, se mencionará a continuación los requerimientos del diseño:

- **Equipo de procesamiento**: Se requiere de un dispositivo con una unidad de procesamiento gráfica para el procesado de imágenes, recursos computacionales de memoria y una unidad de procesamiento que soporte la ejecución de algoritmos de reconocimiento como CNN o YOLO. Además, se necesita de una fácil conexión de periféricos de entradas y salidas, como la cámara que se usará para la recolección de datos.
- **Cámara**: Se requiere de un módulo de cámara para la captura de imágenes compatible con el equipo de procesamiento. Este módulo debe tener un buen sensor fotográfico con la capacidad de grabar en alta definición.

- **Sistema de comunicación**: La información debe transmitirse mediante un protocolo de comunicación que sea ligero para transportar mensajes entre dispositivos constantemente. Además, debe ser eficiente en el uso de ancho de banda y tener una máxima fiabilidad posible.
- **Framework para desarrollo de back end**: Se requiere un framework capaz del desarrollo de una plataforma web donde tanto los alumnos como administradores del sistema puedan tener acceso al aforo en tiempo real y una gráfica sobre el aforo a lo largo del día, donde este último es exclusivo del administrador.
- **Software de base de datos**: Se tienen dos enfoques porque por el lado de la información procesada del aforo de la zona cubierta por la cámara se necesita de un gran volumen de data; mientras que por el lado de la plataforma web es necesario una base de datos enfocada a la administración de información de manera segura, consistente y basada en reglas.
- **Proveedor de servicio en la nube**: Se requiere que el lanzamiento de la plataforma web desarrollada sea sencillo; además de una gestión sencilla. Por otro lado, el almacenamiento de datos enviados por el equipo de procesamiento es un necesario.

Para poder tener una arquitectura en la nube se deben tener en cuenta los requerimientos del sistema web tendrá, por ello en el cuadro 3.1 se presenta la lista de requerimientos que debe cumplir la plataforma desarrollada con las funcionalidades que se deben tener.

CUADRO 3.1: LISTA DE REQUERIMIENTOS DE LA PLATAFORMA WEB

[ELABORACION PROPIA]

ID	REQUISITO
Rol: Estudiante y Administrador	
1	Inicio de Sesión con credenciales (Correo y contraseña). Solo podrán iniciar sesión estudiantes verificados y administradores.
Rol: Administrador	
2	Dashboard para presentar contador del aforo en tiempo real, gráfico aforo de por día, gráfico de aforo semanal (semana actual transcurrida), gráfico de estudiantes verificados/no verificados y estudiantes registrados por facultad. Se usan gráficos de barras, líneas y/o circulares.
3	Verificación de alumnos en el Sistema (Aprobación Manual): En esta sección el administrador verifica a los estudiantes luego de su registro.
4	Lista de estudiantes verificados y no verificados en el sistema.
5	Publicación de anuncios (CRUD) en el sistema para promover eventos indicando lo siguiente: - Título del anuncio - Foto del anuncio y video (este último siendo opcional) - Descripción del anuncio. - Fechas de inicio y fin del evento anunciado.
Rol: Estudiante	
6	Vista de visualización de anuncios publicados en el sistema, pero no se mostrarán anuncios de eventos ya culminados.
7	Perfil del estudiante, donde este puede cambiar su celular y Facultad

8	Módulo de Recuperación de Contraseña. El usuario da clic en la opción de "¿Olvidaste tu contraseña?" e ingresara su correo electrónico registrado para recibir un correo con un link de recuperación.
9	La visualización de aforo en tiempo real de la sección de prueba de la biblioteca del CIA.
10	Registro de estudiantes para el sistema indicando los siguientes datos: - Nombre y Apellidos. - Correo electrónico. (No se pueden repetir) - DNI (No se puede Repetir) - Código PUCP (No se puede repetir) - Celular (No se puede repetir) - Contraseña - Facultad (Combobox)

A partir de lo ya mencionado, el sistema debe ser escalable para la cantidad de alumnos a los que se está enfocando; además de poder gestionar correctamente a los estudiantes, recibir y mostrar información en tiempo real, almacenar información y gestionar eventos a modo de publicidad en beneficio de la universidad.

3.2 Elección del protocolo de aplicación

En esta sección se analizará la tecnología con la que se va transmitir los datos. Compararemos dos tecnologías MQTT y HTTP.

3.2.1 Arquitectura

- MQTT es un protocolo más enfocado en la transmisión de datos a nivel de bytes, mientras que HTTP se centra en la transmisión de documento.[55]

- MQTT es un protocolo de publicación/suscripción a un tema, mediante un agente que gestiona este tipo de peticiones, llamado bróker. Por otro lado, HTTP es un modelo clásico de cliente/servidor. [55]
- HTTP crea una conexión cada vez que se quiera enviar una petición y mantiene abierta esta conexión durante un breve período mientras un dispositivo envía y recibe datos. Mientras que MQTT abre una conexión constante en el cual se comparten "heartbeats" para mantenerla abierta. [55]
- HTTP establece una comunicación TCP half-duplex, aunque en su versión 3 ya admite comunicación UDP. Mientras que MQTT establece una comunicación TCP full-duplex. [55]

3.2.2 Tamaño del mensaje y complejidad

MQTT tiene una especificación más liviana y menos compleja que HTTP. En MQTT los tipos de paquetes principales para desarrolladores son CONNECT, PUBLISH, SUBSCRIBE, UNSUBSCRIBE y DISCONNECT. Mientras las especificaciones de HTTP son mucho más largas, permitiendo así un mejor control de aplicaciones. [55]

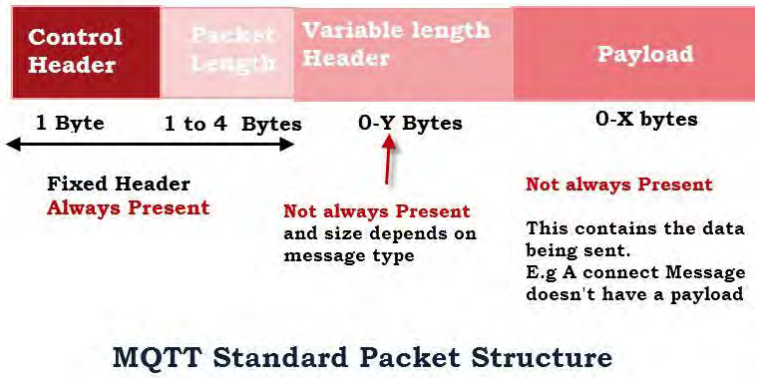


FIGURA 3.1 ESTRUCTURA DEL PAQUETE MQTT [55]

En la figura 3.1 se puede ver la estructura estándar de un paquete MQTT, en donde la cabecera más pequeña es Fixed Header, la cual puede tener un tamaño de 2-5 Bytes, mientras que en

HTTP la cabecera más pequeña es de 26 Bytes [55]. Esto genera una comunicación más ligera sin sobrecargar tanto el canal de comunicación. En cuanto a la carga, MQTT es capaz de transmitir datos en binario directamente, a comparación de HTTP que requiere de una codificación de tipo ASCII. [55]

3.2.3 Quality of Service (QOS) y funcionalidades especiales

HTTP de forma nativa no implementa por sí mismo calidad de servicio (QOS). En cambio, MQTT, estable tres diferentes niveles de QOS, las cuales fueron vistas anteriormente.

Además, MQTT ofrece 2 funcionalidades extras de las cuales carece HTTP:

- Last Will & testament (LWT): cuya funcionalidad indica que, en caso de una desconexión repentina de un suscriber, todos los demás clientes suscritos a ese topic recibirán una notificación por parte del broker [55].
- Retención de mensajes: Permite que un cliente recién suscrito a un tema conseguirá una actualización de estado inmediata sobre el resto de clientes. [55]

3.2.4 Velocidad

Debido a que MQTT nació como un protocolo de comunicación M2M (Machine to Machine), posee características específicas para este modelo. La velocidad es una de ellas, según RMI (Royal Meteorological Institute of Belgium) [56], un instituto meteorológico que usa redes de sensores, indico que en redes 3G, el uso del protocolo MQTT es 93 veces más rápido, usa 8 veces menos tráfico y gasta 170 veces menos energía en los receptores que el protocolo HTTP [56].

Por otro lado, Flespi una empresa con más de 16 años de experiencia en la práctica de ingeniería de software en el área de rastreo GPS, realizó un test de consumo de energía en un Raspberry PI 2B. Para el test se usó una batería de 5V y 2000 mAh y se realizó un bucle infinito de sesiones MQTT y HTTP por separado, para medir así [57]:

- El número de sesiones hasta que se agote la batería
- La vida útil del dispositivo con esta batería
- Uso del CPU promedio

Raspberry power consumption	3G modem Internet connection		Ethernet internet connection	
	REST API via curl	**Mosquitto MQTT client**	**REST API via curl**	**Mosquitto MQTT client**
Number of sessions	9243	11728	20964	22864
Test lifetime	2h 39min	2h 50min	5h 53min	6h 21min
mAh per session	0.2164	0.17	0.0954	0.089
Messages per second	0.97	1.15	0.99	0.98
CPU usage, %	64	60	69	64

FIGURA 3.2 RESULTADOS DEL TEST [57]

Como se puede observar en la figura 3.2 MQTT es hasta un 22% más eficiente energéticamente y un 15% más rápido. Y no depende de si el tipo de conexión es 3G o Ethernet [57].

3.2.5 Seguridad

- MQTT y HTTP pueden operar sobre TLS/ SSL y autorización mediante JWT (JSON Web Token). Por lo tanto, en términos de seguridad son similares.[55]
- El broker MQTT tiene como una de sus funciones controlar quien se suscribe y quien publica en determinado topic [55]. Por lo tanto, ofrece una mejor gestión y administración de estos.
- Según Flespi, MQTT sobre SSL consume menos energía que HTTP tanto en conexiones cableadas como inalámbricas. Por lo tanto, recomiendan usar MQTT tanto en dispositivos independientes como en dispositivos portátiles.[57]

Después de haber analizado la arquitectura, tamaño del mensaje y complejidad, Quality of Service (QOS), velocidad y seguridad entre MQTT y HTTP. Se concluye que MQTT está

mucho más optimizado en cuanto a tiempos de respuesta, consumo de batería, ancho de banda y throughput. Si a esto se añade la posibilidad de implementar QoS y otras funcionalidades específicas de MQTT como Last will & Testament (LWT) y Retención de mensajes, lo posiciona en la mejor elección para aplicaciones relacionadas a Internet de las cosas (IoT) debido a que el tipo de información que se envía son pequeños paquetes constantes. Si se necesitara enviar grandes bloques de datos, HTTP sería el protocolo elegido.

3.3 Elección del equipo de procesamiento de imágenes

Entre los Single Board Computers (SBC) que se van a analizar y comparar son el Raspberry Pi 4 y el NVIDIA Jetson Nano. El dispositivo de Arduino no entrara en la comparación debido a sus limitaciones de Hardware y Software. Por su parte, este dispositivo es un microcontrolador en el cual se puede programar pequeñas aplicaciones para usos concretos, es decir, tareas sencillas de realizar, como prender un foco led. Este no cuenta con un sistema operativo, por lo cual no se le pueden cargar programas complejos, librerías, etc.

CUADRO 3.2 CUADRO COMPARATIVO DE SBC'S [34] [58]

	NVIDIA JETSON NANO	**RASPBERRY PI 4**
Procesador	Quad-core ARM A57 @ 1.43 GHz	Broadcom BCM2711, Quad core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz
Memoria RAM	4 GB 64-bit LPDDR4	2GB, 4GB o 8GB LPDDR4-3200 SDRAM (depende del modelo)
GPU	128-core Maxwell	VideoCore VI @ 500 MHz
Conectividad	Gigabit Ethernet, M.2 Key E	2.4 GHz and 5.0 GHz IEEE 802.11ac wireless, Bluetooth 5.0, BLE Gigabit Ethernet
Puertos	4x USB 3.0, USB 2.0 Micro-B	2 USB 3.0 ports 2 USB 2.0 ports 2 × micro-HDMI ports 2-lane MIPI DSI display port 2-lane MIPI CSI camera port
Potencia de entrada	10 W	5V DC vía USB-C conector (mínimo 3A*)
Dimensiones	69 mm x 45 mm	88 x 58 x 19.5 mm
Precio	Aprox. 150 USD	Aprox. 55 USD

De acuerdo al cuadro 3.2, se llegó a la conclusión de que el equipo Raspberry Pi 4, era el más adecuado para la solución teniendo en cuenta su potencia y costo del equipo, el cual es prácticamente 1/3 del dispositivo de NVIDIA.

Para el equipo seleccionado Raspberry Pi 4, se tienen varias cámaras disponibles, desde versiones oficiales lanzadas por la misma marca hasta versiones OEM. En el cuadro 3.3 se realizará una comparación de 4 cámaras.

CUADRO 3.3 CUADRO COMPARATIVO DE LAS CÁMARAS [59]

	Cámara V2 Raspberry Oficial	**Cámara Raspberry Oficial V2 NoIR**	**Cámara V1 OEM**	**Cámara Raspberry Pi HQ**
Megapíxeles	8 MPX	8 MPX	5 MPX	12.3 MPX
Sensor	IMX219PQ SONY	IMX219 SONY sin filtro IR, para no bloquear el espectro infrarrojo	OmniVision OV5647 5Mpx	Sony IMX477 de 1/3"
Calidad de Video	1080p30, 720p60 y 640x480p90	1080p30, 720p60 y 640x480p90	1080p30, 720p60 y 640x480p90	1080p30, 720p60 y 640x480p90
Dimensiones	25mm x 23mm x 9mm	25mm x 23mm x 9mm	20mm x 25mm x 9mm	38mm x 38mm x 17.23 mm
Precio	165 soles	140 soles	40 soles	289 soles

De acuerdo al cuadro 3.3, se puede comparar las distintas cámaras presentes en el mercado. La solución necesita de una cámara para detección y conteo de personas mediante el procesamiento de video, entonces se descarta la cámara V1 OEM, ya que la calidad podría ser baja y repercutir en el correcto funcionamiento del algoritmo. Por otro lado, la cámara Raspberry Pi HQ, posee un buen sensor, acompañado de una buena apertura focal para captar mayor iluminación, sin embargo, el precio es considerablemente más alto. Por lo tanto, la

cámara V2 de 8MPX sería la opción ideal por su calidad precio, se cuenta con un mejor sensor y una mayor cantidad de megapíxeles, comparado con la V1 OEM, para así obtener una detección clara y precisa.

3.4 Elección del algoritmo de detección de objetos

En base a lo explicado en el Capítulo 2 y la elección del Raspberry Pi 4 como equipo de procesamiento para la detección de personas, se probará el rendimiento de Faster RCNN y YOLO v5 de modo que se determine qué algoritmo es el que obtiene mejores resultados en el Raspberry.

3.4.1 Faster R-CNN

En este caso, se obtuvieron resultados de detección positivas en el aspecto de determinación de personas; sin embargo, su gran problemática está situada en el consumo de recursos del equipo; ya que los Fotogramas Por Segundo (FPS) obtenidos son de entre 0.45-0.75, lo que generaría un gran retraso, entre 8 a 10 segundos, frente a lo que está pasando en tiempo real con respecto a la respuesta. En la figura 3.3 se muestra una de las pruebas hechas con este algoritmo.

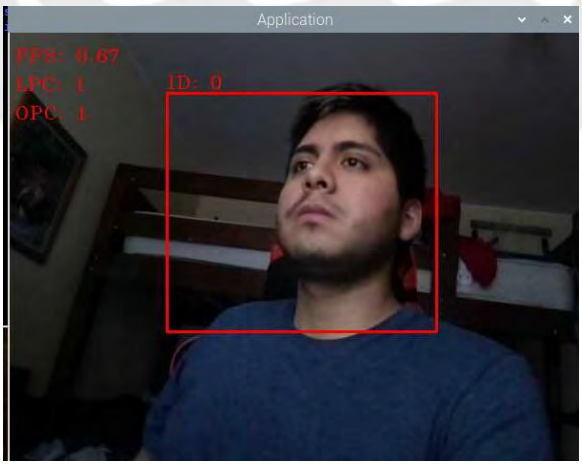


FIGURA 3.3 PRUEBA DEL ALGORITMO FASTER R-CNN EN EL RASPBERRY (0.67FPS) [ELABORACIÓN PROPIA]

3.4.2 YOLO v5

YOLO es un algoritmo más ligero con respecto a Faster R-CNN, sin embargo, sus tiempos de respuesta siguen siendo considerablemente lentos porque si bien en este caso no se muestran los FPS, tenemos los tiempos de respuesta una vez obtenido el frame del video con valores de entre 3.45 a 8.15 segundos, por lo que sus FPS tendrían un rendimiento solo un poco por encima con respecto al algoritmo previamente expuesto. En la figura 3.4 se tiene una prueba del algoritmo YOLO v5 en el Raspberri Pi 4, así mismo en la figura 3.5 se observan los tiempos obtenidos por frame y qué objetos fueron detectados en dicho frame.

FIGURA 3.4 PRUEBA DEL ALGORITMO YOLO V5 EN EL RASPBERRY

[ELABORACIÓN PROPIA]

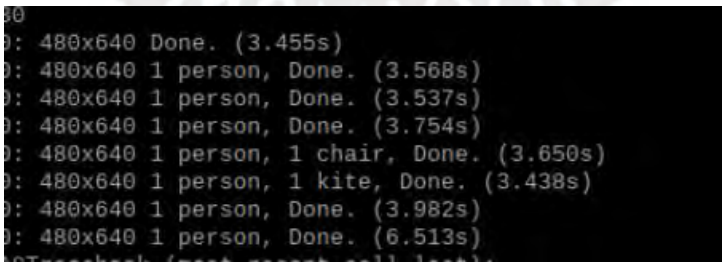


FIGURA 3.5 TIEMPOS DE RESPUESTA EN UN FRAME DE YOLO V5

[ELABORACIÓN PROPIA]

Luego de analizar los resultados mostrados de los algoritmos se observó que, si bien cumplen la función de detección de personas, no presentan un rendimiento eficiente para la solución, debido a que la carga computacional es muy elevada dando así unos valores de FPS muy bajos (menor a 1 FPS). Es por ello que se encontró una alternativa llamada TensorFlow Lite con un rendimiento mejor con respecto a las anteriores pruebas.

3.4.3 TensorFlow Lite

Este programa desarrollado por Google es una biblioteca open source de extremo a extremo encargada del aprendizaje automático por medio de un rango de tareas; además satisface las necesidades de los sistemas de redes neuronales para la detección de objetos y descifrar patrones, lo que genera una respuesta de forma más rápida. Para esta prueba se usó su versión Lite que optimiza el aprendizaje automático integrado en el dispositivo y aceleración de hardware de manera que presenta un rendimiento mayor. En la Figura 3.6 se obtiene el resultado de la prueba de Tensor Flow Lite en el Raspberry con un mínimo de 4.25 FPS y un máximo de 5 FPS, por lo que se obtiene un resultado diferenciador con respecto a los 2 algoritmos previamente analizados, también su tiempo de respuesta es menor cumpliendo con requerimientos propuestos y siendo posible su uso para la aplicación en tiempo real.

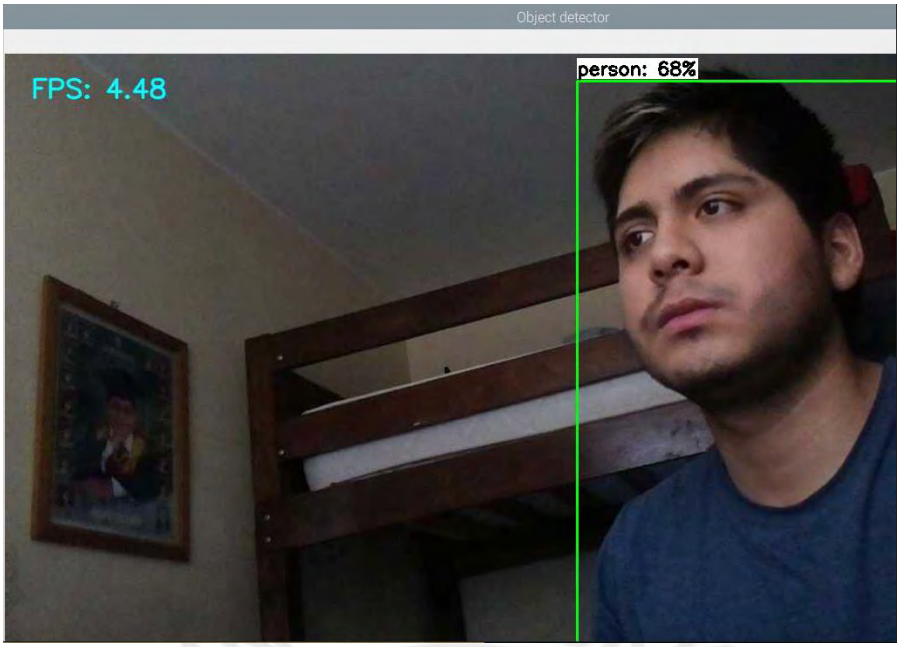


FIGURA 3.6 PRUEBA DE TENSORFLOW LITE EN EL RASPBERRY [ELABORACIÓN PROPIA]

A partir de lo mencionado, TensorFlow Lite es la alternativa escogida para la detección de personas en el Raspberry.

3.5 Elección de Framework para backend

Se tienen los frameworks Django y Spring Boot desarrollados para los lenguajes de programación Python y Java respectivamente; sobre los cuales se realiza una comparación en el cuadro 3.4.

CUADRO 3.4 CUADRO COMPARATIVO DJANGO Y SPRING BOOT [60] [61] [62]

	Django	**Spring Boot**
¿Por qué los desarrolladores lo usan?	Características de desarrollo rápido, aprendizaje sencillo, código abierto, presenta una gran comunidad.	Características como código abierto, muy potente, configuración sencilla, mayor desarrollo empresarial.
Herramientas usadas	Python, Laravel, PyCharm, Sentry, Django REST framework, Stream, etc.	Java, Spring Boot, Bugsnag, Auth0, Hazelcast, Java EE, QueryDSL, etc.
Interfaz de administración	Si cuenta con ello.	No cuenta con ello.
Tipos de aplicaciones más beneficiadas	Monolíticas: aplicaciones que necesitan de la interfaz de usuario (frontend) y de la capa de acceso (backend).	Aplicaciones de micro servicios como chats, videoconferencias, entre otras.
Ventajas	Cobertura de usuarios en distintos sitios web, búsqueda completa de textos, poder implementar etiquetas personalizadas para su uso en las vistas, entre otras.	Facilidad de configuración, su conexión a base de datos es sencilla, integración con aplicaciones de Spring, el uso de Maven y Gradle facilita el empleo de distintas herramientas, entre otras.
Desventajas	No se pueden gestionar múltiples solicitudes, no se aprovecha el rendimiento del framework para aplicaciones muy pequeñas (demos).	Presenta una mayor complicación para su aprendizaje y uso con respecto a Django, no presenta una búsqueda de texto completo, No cuenta con una solución de supervisión de seguridad y gestión.

Según SimilarTech [63], Django tiene una mejor cobertura de uso en más categorías de sitios web, tal como se observa en la figura 3.7.

FIGURA 3.7 COMPARACIÓN DE DJANGO Y SPRING EN DISTINTOS DOMINIOS [63]

En función a la información expuesta, Django y Spring poseen la capacidad para el desarrollo del proyecto integrando distintas librerías para la realización de una aplicación sostenible y automatizada, pero Django presenta ventajas diferenciales como un sistema de control y manejo de administración de la página web; además de un aprendizaje más sencillo generando así el resultado en la imagen anterior.

3.6 Elección de las bases de datos

Se tienen 2 enfoques para el uso de bases de datos, donde el primero consiste en la gestión de información ingresada en el servicio web desarrollado; mientras que el segundo está relacionado a la información del aforo determinada por el equipo de procesamientos de imágenes, por ello se realizará un balance entre los dos tipos de bases de datos explicados en el capítulo 2. A partir de lo mencionado, el cuadro comparativo 3.5 mostrará información correspondiente a los tipos de base de datos relacionales (SQL) y no relacionales (NoSQL).

CUADRO 3.5: CUADRO COMPARATIVO ENTRE BASE DE DATOS SQL Y NOSQL [64]

	SQL	**NoSQL**
Naturaleza	Una base de datos relacional en la naturaleza.	Una base de datos no relacional por naturaleza.
Diseño	Modelado en base al concepto de "tablas".	Modelado en base al concepto de "documento".
Escalable	Ser de naturaleza relacional puede ser una tarea difícil para escalar big data.	Big Data fácilmente escalable en comparación con relacional.
Modelo	El modelo de base de datos detallado debe estar en su lugar antes de la creación.	No es necesario desarrollar un modelo de base de datos detallado.
Estandarización	SQL es un lenguaje estándar.	Falta de un lenguaje de consulta estándar.
Esquema	El esquema es rígido.	El esquema dinámico es un beneficio clave de NoSQL.
Flexibilidad	No tan flexible en cuanto al diseño, la inserción de nuevas columnas o campos afecta un diseño.	Se pueden insertar nuevas columnas o campos sin un diseño existente.

Con lo ya mencionado, se puede decir que las bases de datos relacionales son convenientes para el manejo y control de data para el servicio web, puesto que está regido bajo un modelo de tablas y no es flexible en cuanto diseño porque presenta un esquema fijo y estructura establecida; además su uso es conveniente para sistemas relacional como es el caso de los flujos webs; mientras que para la información en tiempo real del aforo es conveniente el uso de una base de datos no relacional, ya que son fácilmente escalable para un gran volumen de información, flexibles puesto a que su estructura estaría definida por el desarrollador y es por ello que uno de sus casos de uso principal es el análisis de información en tiempo real porque cada vez que se actualiza la información en la base de datos puede ser reflejado en la vista de la página web por lo que se pueden mostrar gráficos estadísticos con información en tiempo real.

3.6.1 Elección de base de datos para gestión de información del servicio web

Para base de datos relacionales se contemplaron los softwares PostgreSQL y MySQL siendo el más avanzado y el más usado respectivamente; por ello se realizará un cuadro comparativo con el fin de hacer un balance para escoger una alternativa para un posible diseño.

CUADRO 3.6 CUADRO COMPARATIVO POSTGRESQL Y MYSQL [65] [66]

	PostgreSQL	**MySQL**
Conocido como	Ser la base de datos más avanzada.	Ser la base de datos más popular.
Desarrollo	Es de código abierto.	Es de código abierto.
Sistema de gestión de base de datos	Basado en objetos.	Basado en relaciones.
Herramienta GUI	PgAdmin.	MySQL Workbench.
ACID (Atomicidad, Consistencia, Aislamiento y Durabilidad)	De manera completa.	Solo en caso de usar motores de clúster InnoDB y NDB.
Lenguaje de implementación	C	C/C++
Funciones de DROP de tablas	Soporta un borrado en cascada.	No soporta borrado en cascada.
Incremento de Columnas	Serial.	Auto incremental.
Tipos de Datos	Soporta los tipos de datos estándar y avanzados, como lo son matices, hstore o tipos establecidos por el usuario.	Solo admite tipo de datos estándar.
Función de control de concurrencia de múltiples versiones (MVCC)	Soporte completo.	Lo proporciona, pero de manera limitada.

En base a la comparativa hecha y lo redactado previamente, las características brindadas por PostgreSQL son más flexibles con respecto a MySQL; además de contar con un soporte completo de control de versiones para proyectos que tengan distintas etapas como la de desarrollo y producción; sin embargo, los dos presentan un rendimiento muy similar para el almacenamiento de información. Con lo ya mencionado PostgreSQL sería el software de base de datos más completo.

3.6.2 Elección de base de datos para la información del aforo en tiempo real

Para este caso se tendrá una comparativa entre MongoDB y DynamoDB que, si bien ofrecen funciones y características parecidas, presentan diferencias las cuales son:

- **Entorno y estrategia de implementación**

Por el lado de MongoDB se tiene una plataforma independiente que puede ser configurado para su ejecución en cualquier proveedor de servicio de cloud; además los usuarios tienen un mayor nivel de control porque administran su infraestructura y configuraciones. Mientras que DynamoDB solo puede ser usado en la plataforma de AWS, por lo que no es flexible su uso; asimismo esta base de datos es completamente administrada lo que si bien permite su uso inmediato hace que su control sea limitado en ciertos aspectos de configuración [67].

- **Modelo y esquema de datos**

DynamoDB presenta un máximo tamaño de 400kB por documento y MongoDB un tamaño máximo por documento de 16 MB. Por otro lado, los dos servicios de base de datos no relacionales no presentan un esquema, pero MongoDB hace posible que los usuarios apliquen un esquema pro medio de su esquema de validación incorporada en caso sea necesario. Estos pueden validar estructura del documento, tipos de datos, los campos, entre otros [67].

- **Consulta de datos e índices**

MongoDB admite la creación de índices para todo campo en un documento, también por el lado de la consulta de datos, este brinda mayor flexibilidad, puesto que los usuarios pueden agregar y consultar datos de distintas formas nativas como llaves individuales, rangos, recorridos gráficos, uniones, etc. En cambio, DynamoDB admite consultas nativas solo en caso de valor clave y para el caso de indexación presenta 2 tipos de índices secundarios, los cuales son Índice secundario global (GSI) y Índice secundario local (LSI) [67].

- **Seguridad de la base de datos**

Por el lado de seguridad DynamoDB cuenta con las prácticas de seguridad usadas por AWS por lo que la autorización es mediante Identity and Access Management (IAM) y en caso de MongoDB dependerá de la configuración del usuario [67].

En base a todos a todo lo explicado, MongoDB es una opción más flexible con capacidades limitadas por el usuario en los aspectos expuesto; por lo que sería una mejor opción frente a DynamoDB.

3.7 Elección de proveedor de servicio Cloud

En este caso se cuenta con 3 servicios de cloud, que son AWS, Google Cloud (App Engine) y Microsoft Azure; entonces primero se presentarán los nombres de servicios usados por cada proveedor, tanto para el alojamiento de archivos, compilación del proyecto y base de datos.

CUADRO 3.7 SERVICIOS DE AWS, MICROSOFT AZURE Y GOOGLE CLOUD [68]

Servicio	AWS	Microsoft Azure	Google Cloud
Computación	Elastic Cloud Compute (EC2)	Virtual Machines	Compute Engine
Alojamiento de la aplicación	Elastic Beanstalk	Cloud Services	App Engine
Almacenamiento de archivos	S3 Storage Service	Azure Storage	Cloud Storage
Base de datos	Relational Database Service (RDS)	Azure SQL Database	Cloud SQL
Balanceador de carga	Elastic Load Balancing	Load Balancer	Cloud Load Balancing

Teniendo en cuenta el cuadro 3.7, se tendrá que evaluar costos con respecto al rendimiento brindado para el alojamiento; sin embargo, como se especificó en el capítulo anterior Azure presenta la gran desventaja del pago por el conjunto de módulos que no serían usados y ello generaría un sobrecosto. Con lo ya mencionado, AWS y Google Cloud serían las 2 posibles opciones; entonces se adjuntan imágenes de costos estimados, determinados a partir de las calculadoras oficiales de cada empresa, para el levantamiento del servicio considerando que el alcance de la aplicación es para una zona focalizada.

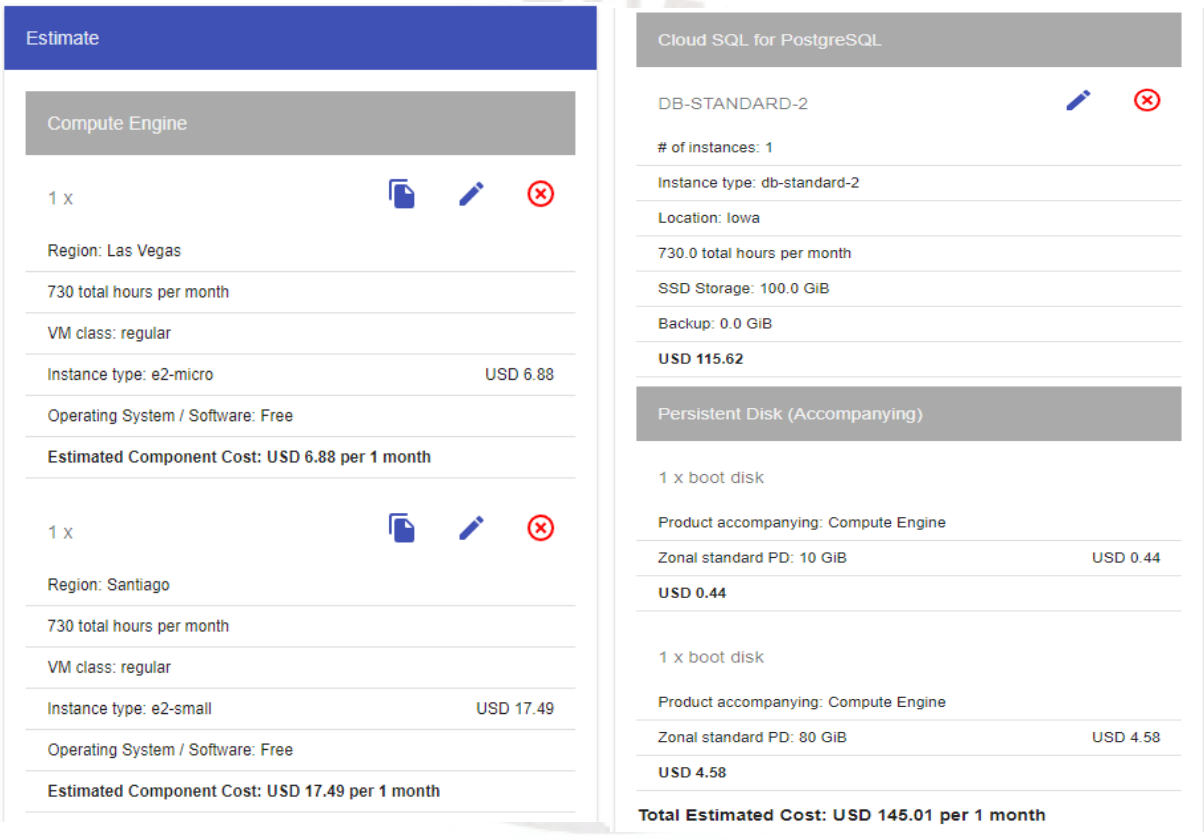


FIGURA 3.8 COSTO DEL SERVICIO ESTIMADO POR MES [69]

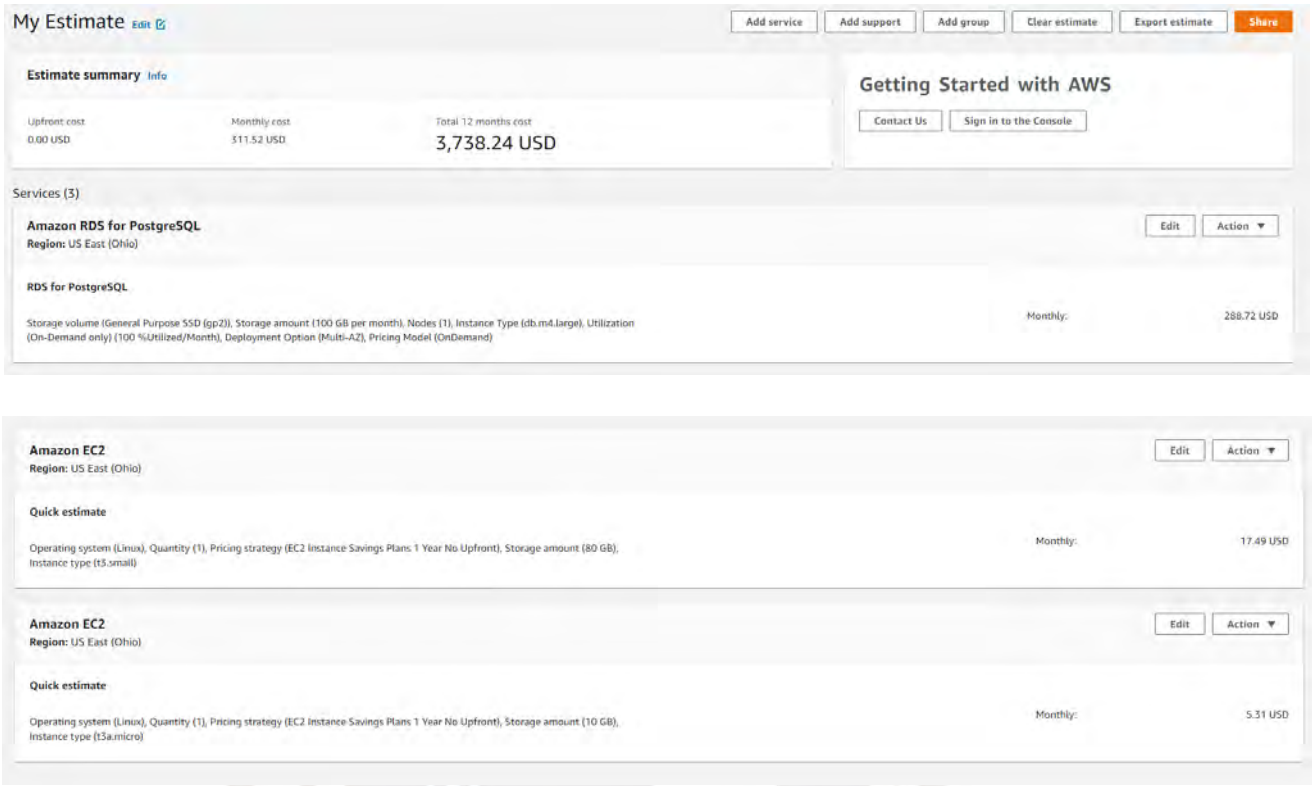


FIGURA 3.9 COSTO DEL SERVICIO ESTIMADO POR MES [70]

A partir de las imágenes brindadas, Google Cloud sería el elegido frente a AWS y esto se debe a que el precio es más barato con respecto al otro proveedor; además de también presentar más facilidades con respecto a configuración porque si bien se está estimando una máquina virtual de 2 vCPU, este pertenece al servicio de MongoDB que por el lado de Google viene ya configurado y corriendo para su uso instantáneo; mientras que en AWS se tiene que instalar desde 0. Por todo lo ya mencionado, el proveedor de servicios Cloud será Google.

3.8 Arquitectura de la solución

La arquitectura determinada está basada en las elecciones hechas durante el capítulo 3, por lo que se tiene un sistema desplegado en la nube del proveedor Google Cloud encargada de la gestión de información enviada por el Raspberry; además de contar con un servicio web para la visualización en tiempo real del aforo de la sección abarcada. Los servicios usados en la nube serán Cloud Balancing, App Engine, CloudSQL y Cloud DataStore.

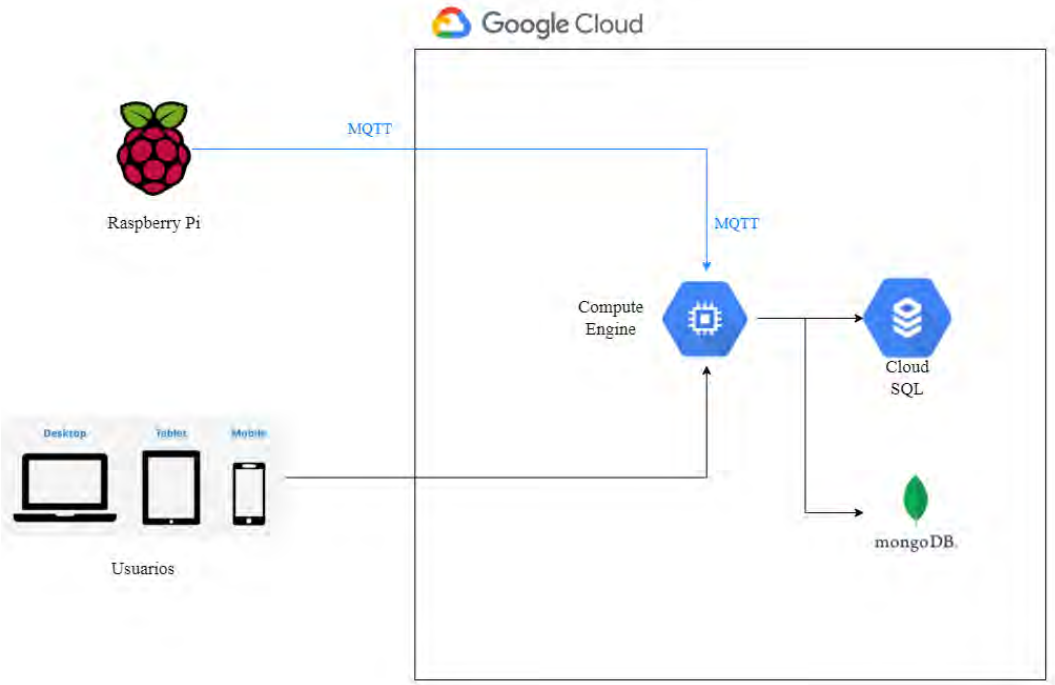


FIGURA 3.10 ARQUITECTURA DE LA SOLUCIÓN EN GOOGLE CLOUD

[ELABORACIÓN PROPIA]

A continuación, se listarán las tecnologías determinadas y se describirán los servicios en la nube elegidos para el diseño realizado:

- **Equipo de procesamiento:** Raspberry Pi 4.
- **Cámara del equipo:** Cámara V2 Raspberry Oficial.
- **Tecnología para procesamiento de imágenes:** TensorFlow.
- **Protocolo de Aplicación:** MQTT.
- **Framework para Backend:** Django.
- **Base de Datos Relacional:** PostgreSQL.
- **Base de Datos no Relacional:** MongoDB.
- **Proveedor de Servicios Cloud:** Google Cloud.

- **Compute Engine**

  Este es el servicio de Máquinas virtuales que presenta la plataforma de Google cloud de modo que servirá para el despliegue del sistema web y como broker usando mosquitto.

- **CloudSQL**

  Este es el servicio de base de datos relacionadas administrado para PostgreSQL y con una sencilla administración.

- **MongoDB MarketPlace**

  Este es un servicio brindado por Google Cloud, donde el proveedor lanza una instancia (Compute Engine) ya configurada con la plataforma de MongoDB y con almacenamiento decidido por el usuario corriendo como servicio; además de contar con una IP elástica para su uso en conexiones.

Capítulo 4: Configuración del sistema determinado y análisis de pruebas de funcionamiento

En base a la información brindada en el capítulo 3, se realizó la figura 4.1.

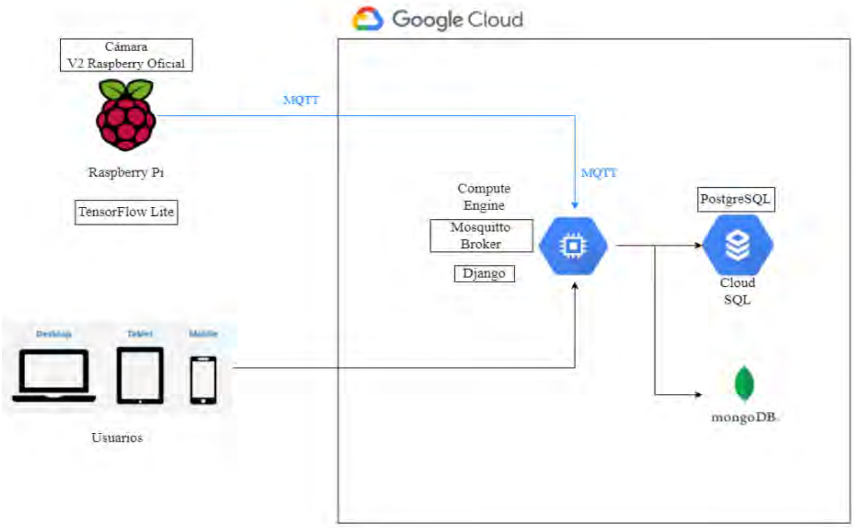
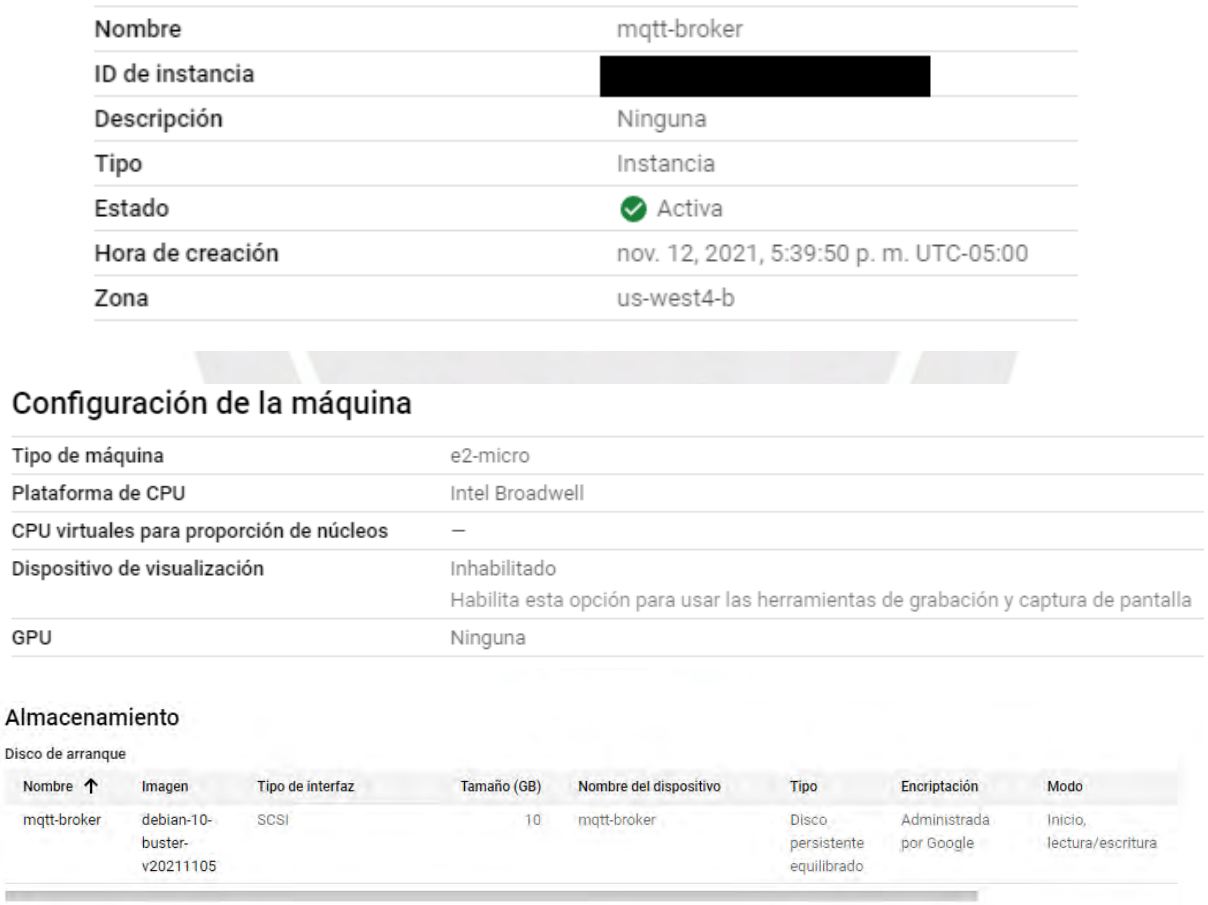


FIGURA 4.1 ARQUITECTURA DETALLADA DE LA SOLUCIÓN EN GOOGLE CLOUD [ELABORACIÓN PROPIA]

4.1 Configuración del Mosquitto Broker

Es el componente central en la comunicación, el bróker usado es MOSQUITTO debido a su ligereza y su capacidad de uso en diversidad de dispositivos, desde computadoras de placa única (SBC) hasta servidores completos.

En este caso, se usó una instancia "e2-micro" con procesador Intel Broadwell del Google Cloud Engine, la cual corre una distribución de Linux, lo cual se comprueba en la figura 4.2 donde se tiene un resumen de las propiedades de la instancia que será usado para el broker y para el alojamiento web.



Nombre	mqtt-broker
ID de instancia	[redacted]
Descripción	Ninguna
Tipo	Instancia
Estado	Activa
Hora de creación	nov. 12, 2021, 5:39:50 p. m. UTC-05:00
Zona	us-west4-b

Configuración de la máquina

Tipo de máquina	e2-micro
Plataforma de CPU	Intel Broadwell
CPU virtuales para proporción de núcleos	–
Dispositivo de visualización	Inhabilitado Habilita esta opción para usar las herramientas de grabación y captura de pantalla
GPU	Ninguna

Almacenamiento

Disco de arranque

Nombre	Imagen	Tipo de interfaz	Tamaño (GB)	Nombre del dispositivo	Tipo	Encriptación	Modo
mqtt-broker	debian-10-buster-v20211105	SCSI	10	mqtt-broker	Disco persistente equilibrado	Administrada por Google	Inicio, lectura/escritura

FIGURA 4.2 CARACTERÍSTICAS DEL COMPUTE ENGINE (E2-MICRO) EN GOOGLE CLOUD [ELABORACIÓN PROPIA]

Por otro lado, para la instalación de MOSQUITTO se efectúa por medio del siguiente comando en modo root:

```
sudo apt-get install mosquitto
```

En la figura 4.3 se puede observar que el servidor MOSQUITTO está corriendo en dicha máquina virtual.

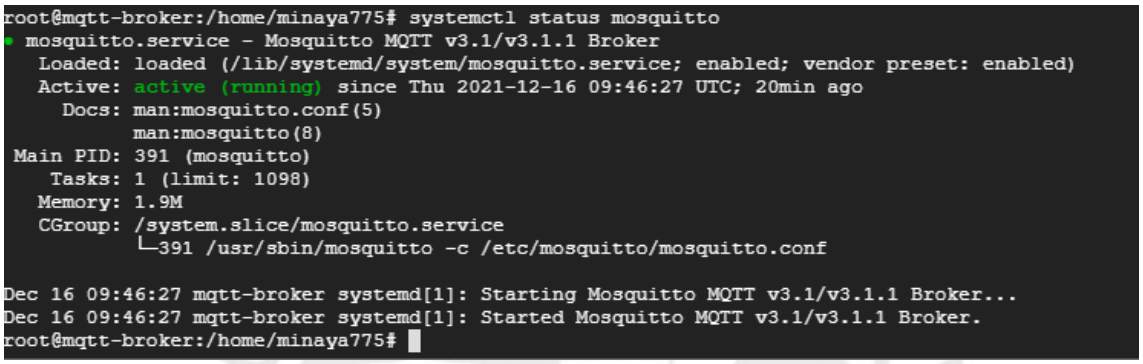


FIGURA 4.3 ESTADO DEL SERVICIO DE MOSQUITTO EN GOOGLE CLOUD [ELABORACIÓN PROPIA]

Por otro lado, al tener un cliente MQTT en java script se debió agregar la siguiente configuración en mosquitto ello se comprueba en la figura 4.4.

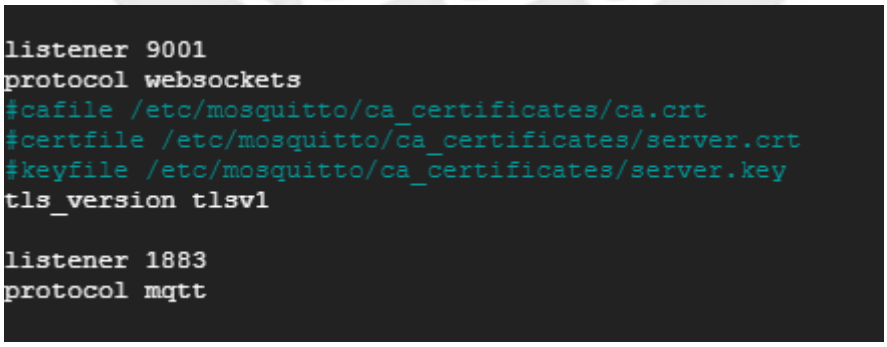


FIGURA 4.4 CONFIGURACIÓN DEL SERVICIO DE MOSQUITTO [ELABORACIÓN PROPIA]

4.2 Configuración del Raspberry

El kit de Raspberry Pi 4 adquirido, incluyó consigo un case, el cual se puede apreciar en la figura 4.5.

FIGURA 4.5 CASE SIN SOPORTE PARA CÁMARA [59]

La cámara se conecta al Raspberry mediante un cable FCC, sin embargo, el case al no contar con un soporte para la cámara, esta no puede suspenderse directamente en una superficie, lo cual nos limita en la comodidad de instalar el equipo en determinado espacio. Existen otros cases para el Raspberry con soporte de cámara, sin embargo, el precio es ligeramente mayor.

FIGURA 4.6 CASE CON SOPORTE PARA CÁMARA [71]

Finalmente, el conjunto de equipos quedaría de la siguiente manera como se puede apreciar en la figura 4.7.

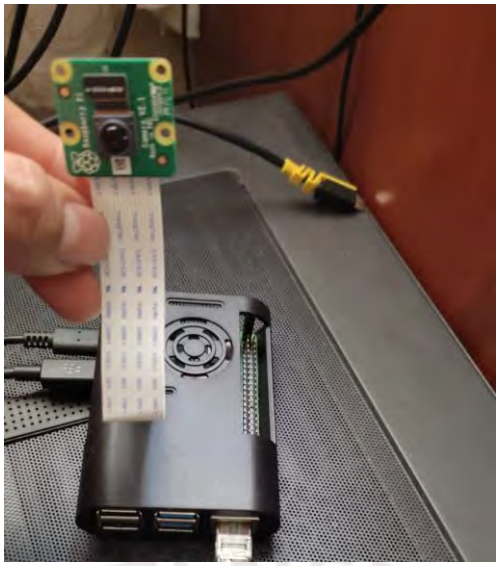


FIGURA 4.7 RASPBERRY PI 4B CON CÁMARA Y CASE

En primer lugar, se tienen librerías requeridas para el funcionamiento de este algoritmo (Se está usando python 3.7), las cuales son:

```
sudo apt-get -y install libjpeg-dev libtiff5-dev
libjasper-dev libpng12-dev
sudo apt-get -y install libavcodec-dev libavformat-dev
libswscale-dev libv4l-dev
sudo apt-get -y install libxvidcore-dev libx264-dev
sudo apt-get -y install qt4-dev-tools libatlas-base-dev
pip3 install opencv-python==3.4.6.27
pip3 install paho-mqtt
```

En este Raspberry se tiene la lógica para la detección y envío de información; entonces para la etapa de detección se tiene un algoritmo que usa modelos pre entrenados de Tensor Flow Lite, las cuales utilizan como entrada los frames por segundo capturados por la cámara del Raspberry. En la figura 4.8 se muestra las líneas de código para cargar y obtener los modelos de TensorFlow Lite.

```
Load the Tensorflow Lite model.
If using Edge TPU, use special load_delegate argument
 use_TPU:
  interpreter = Interpreter(model_path=PATH_TO_CKPT,
                            experimental_delegates=[load_delegate('libedgetpu.so.1.0')])
  print(PATH_TO_CKPT)
se:
  interpreter = Interpreter(model_path=PATH_TO_CKPT)

terpreter.allocate_tensors()

Get model details
put_details = interpreter.get_input_details()
tput_details = interpreter.get_output_details()
ight = input_details[0]['shape'][1]
dth = input_details[0]['shape'][2]

oating_model = (input_details[0]['dtype'] == np.float32)

put_mean = 127.5
put_std = 127.5

Initialize frame rate calculation
ame_rate_calc = 1
eq = cv2.getTickFrequency()
```

FIGURA 4.8 CONFIGURANDO EL MODELO TFLITE Y EL CÁLCULO DE FRAMES [ELABORACIÓN PROPIA]

Luego se hace un bucle sobre todas las detecciones y se dibuja un cuadro de detección alrededor del objeto detectado solo si la fiabilidad se encuentra por encima del umbral mínimo, en este caso es del 50%, lo cual se muestra en la figura 4.9. Posteriormente, se realiza un contador en tiempo real de las personas detectas en cada frame, para que finalmente, mediante un cliente PAHO, se publican los temas con el mensaje de aforo y fecha actual como se puede apreciar en la figura 4.10.

```
for i in range(len(scores)):
    if ((scores[i] > min_conf_threshold) and (scores[i] <= 1.0)):
        object_name = labels[int(classes[i])]
        if object_name == "person":
            # Get bounding box coordinates and draw box
            # Interpreter can return coordinates that are outside of image dimensions, nee
            ymin = int(max(1,(boxes[i][0] * imH)))
            xmin = int(max(1,(boxes[i][1] * imW)))
            ymax = int(min(imH,(boxes[i][2] * imH)))
            xmax = int(min(imW,(boxes[i][3] * imW)))

            cv2.rectangle(frame, (xmin,ymin), (xmax,ymax), (10, 255, 0), 2)

            # Draw label
            # Look up object name from "labels" array using class index
            label = '%s: %d%%' % (object_name, int(scores[i]*100)) # Example: 'person: 72%
            labelSize, baseLine = cv2.getTextSize(label, cv2.FONT_HERSHEY_SIMPLEX, 0.7, 2)
            label_ymin = max(ymin, labelSize[1] + 10) # Make sure not to draw], label_ymin

            lpc_count += 1
            cv2.putText(frame, label, (xmin, label_ymin-7), cv2.FONT_HERSHEY_SIMPLEX, 0.7,
            #print(lpc_count)
```

FIGURA 4.9 PROCESO DE DETECCIÓN DE PERSONAS [ELABORACIÓN PROPIA]

```
if lpc_count_old != lpc_count:
    lpc_count_old = lpc_count
    msg = str(lpc_count_old) +","+ datetime.today().strftime('%Y-%m-%d %H:%M:%S')
    client.publish("tesis/aforo",msg )
    print(lpc_count_old)
```

FIGURA 4.10 PUBLICACIÓN DE INFORMACIÓN AL TÓPICO ESCOGIDO [ELABORACIÓN PROPIA]

En la figura 4.11 se tiene el diagrama de flujo de modo que el proceso se especifique a mayor nivel.

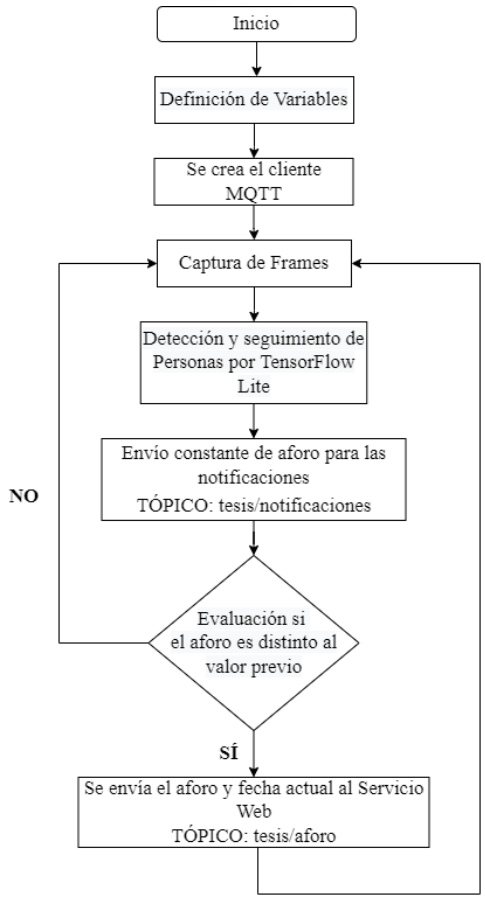


FIGURA 4.11 DIAGRAMA DE FLUJOS DEL FUNCIONAMIENTO DEL ALGORITMO DE PROCESAMIENTO DE IMÁGENES [ELABORACIÓN PROPIA]

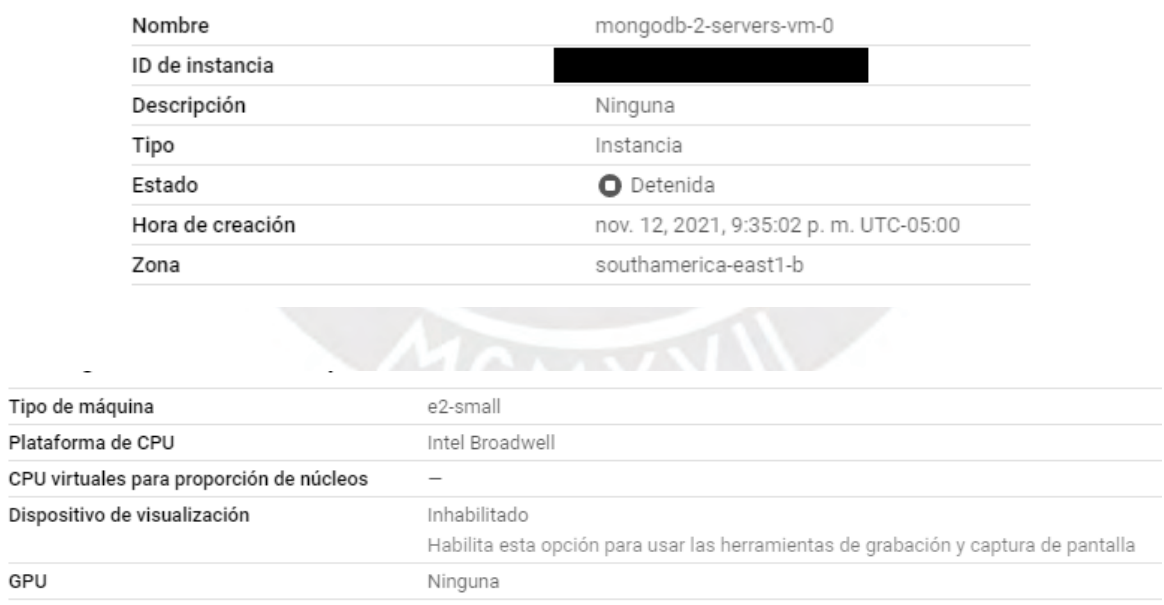
Por último, para la ejecución del algoritmo se usa el siguiente comando:

```
python3 TFLite_detection_webcam.py --
modeldir=Sample_TFLite_model
```

Se usa este algoritmo basado en TensorFlow Lite debido a la limitación de hardware que se tiene en el Raspberry, reduciendo así el costo computacional de las redes neuronales. Por otro, lado TensorFlow en su versión completa es mucho más potente ya que puede construir y entrenar modelos de Machine Learning, sin embargo, el costo de cómputo es mucho más elevado.

4.3 Configuración de la instancia de MongoDB del MarketPlace

Para este componente se realizó un despliegue automático por el lado de Google cloud donde como usuario se tiene que especificar la zona donde será albergada y al ser un Compute Engine preparado para el uso instantáneo de MongoDB, se debe indicar las especificaciones de la máquina virtual (número de CPU y número de RAM). En la figura 4.12 se mostrará las especificaciones seleccionadas para el diseño propuesto.



Nombre	mongodb-2-servers-vm-0
ID de instancia	[redacted]
Descripción	Ninguna
Tipo	Instancia
Estado	Detenida
Hora de creación	nov. 12, 2021, 9:35:02 p. m. UTC-05:00
Zona	southamerica-east1-b

Tipo de máquina	e2-small
Plataforma de CPU	Intel Broadwell
CPU virtuales para proporción de núcleos	–
Dispositivo de visualización	Inhabilitado Habilita esta opción para usar las herramientas de grabación y captura de pantalla
GPU	Ninguna

FIGURA 4.12 CARACTERÍSTICAS DEL MARKETPLACE DE MONGODB EN GOOGLE CLOUD [ELABORACIÓN PROPIA]

Luego ello, se comprobó que el servicio se encuentre con un estatus de activo para así asegurar el funcionamiento correcto de MongoDB, es por ello que en la figura 4.13 se tiene la corroboración del servicio con un estado activo.

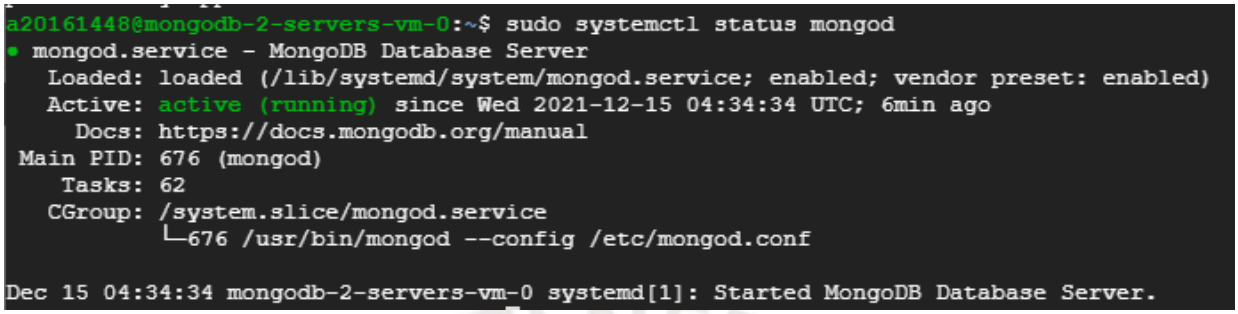


FIGURA 4.13 ESTADO DEL SERVICIO DE MARKETPLACE DE MONGODB EN GOOGLE CLOUD [ELABORACIÓN PROPIA]

Para garantizar que el acceso hacia esta base de datos se por únicamente los usuarios que necesitan del ingreso para el desarrollo la plataforma y por el servicio web se detalló una regla de seguridad en el Firewall de VPC (Red virtual en la nube) a la que pertenece la instancia lanzada de modo que se logrará que solo las IP detalladas puedan acceder al puerto donde el servicio está corriendo; es por ello que en la figura 4.14 se tiene dicha regla ingresada donde la última dirección de red pertenece a la máquina virtual encargada del lanzamiento de la plataforma web y las dos restantes son para pruebas locales.

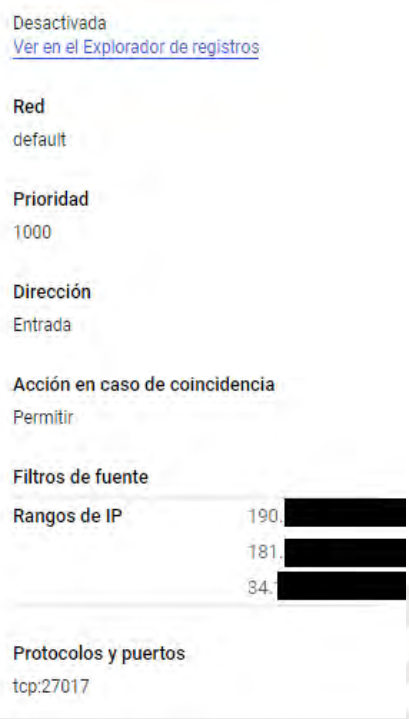


FIGURA 4.14 RESTRICCIÓN DE IP'S AL PUERTO USADO POR MONGODB

[ELABORACIÓN PROPIA]

4.4 Configuración de la Instancia de PostgreSQL

Para poder configurar este servicio en la nube se tuvo que crear una instancia de Cloud SQL con la versión de Base de datos de PostgreSQL 13; además para desplegar esta instancia lo que se tuvo que realizar es especificar la Región donde estará y características suficientes para soportar los requerimientos de almacenamiento (Data de gestión de usuarios y eventos). En la figura 4.15 se muestra un resumen de las características que presenta la instancia desplegada.

Resumen

Región	us-central1 (Iowa)
Versión de la base de datos	PostgreSQL 13
CPU virtuales	4 CPU virtual(es)
Memoria	26 GB
Almacenamiento	100 GB
Capacidad de procesamiento de la red (MB/s)	1,000 de 2,000
Capacidad de procesamiento del disco (MB/s)	Lectura: 48.0 de 240.0 Escritura: 48.0 de 240.0
IOPS	Lectura: 3,000 de 15,000 Escritura: 3,000 de 15,000
Conexiones	IP pública
Copia de seguridad	Automatizada
Disponibilidad	Varias zonas (con alta disponibilidad)
Recuperación de un momento determinado	Habilitada

FIGURA 4.15 RESUMEN DE LAS PROPIEDADES DE LA INSTANCIA DE CLOUD SQL [ELABORACIÓN PROPIA]

Por otro lado, si bien se creó la instancia, se deben crear la base de datos a usar para el sistema web a desarrollar por lo que en la figura 4.16 se muestra la base de datos creada.

FIGURA 4.16 BASE DE DATOS CREADA EN LA INSTANCIA DE CLOUD SQL [ELABORACIÓN PROPIA]

Como el servicio de Cloud SQL es la que se encarga de todo el aspecto en seguridad, faltaría indicar las direcciones de red que pueden acceder a la instancia para hacer uso de los recursos, por lo que en la figura 4.17 se tienen las IP's permitidas.

FIGURA 4.17 IP'S PERMITIDAS PARA EL ACCESO A LA BASE DE DATOS EN LA INSTANCIA DE CLOUD SQL [ELABORACIÓN PROPIA]

4.5 Configuración de la Plataforma Web (Django)

Para la aplicación web desarrollada, se consumen los servicios de MQTT, PostgreSQL y MongoDB. Entonces para ello en la figura 4.18 se tienen los requerimientos (librerías de python) usadas para el desarrollo del proyecto.

```
django==3.2.3
django-storages
google-cloud-storage
gunicorn
Pillow==8.2.0
psycopg2-binary
django-sendgrid-v5==0.9.0
paho-mqtt
mqtt
pymongo
```

FIGURA 4.18 REQUERIMIENTOS DEL SISTEMA WEB [ELABORACIÓN PROPIA]

El primer flujo presentando consta en el guardado de información proveniente del Raspberry, por lo que en la figura 4.19 se muestra creación del cliente MQTT usado en el lado del servidor (back end) encargado de guardar la información del aforo en la base de datos no relacional en tiempo real.

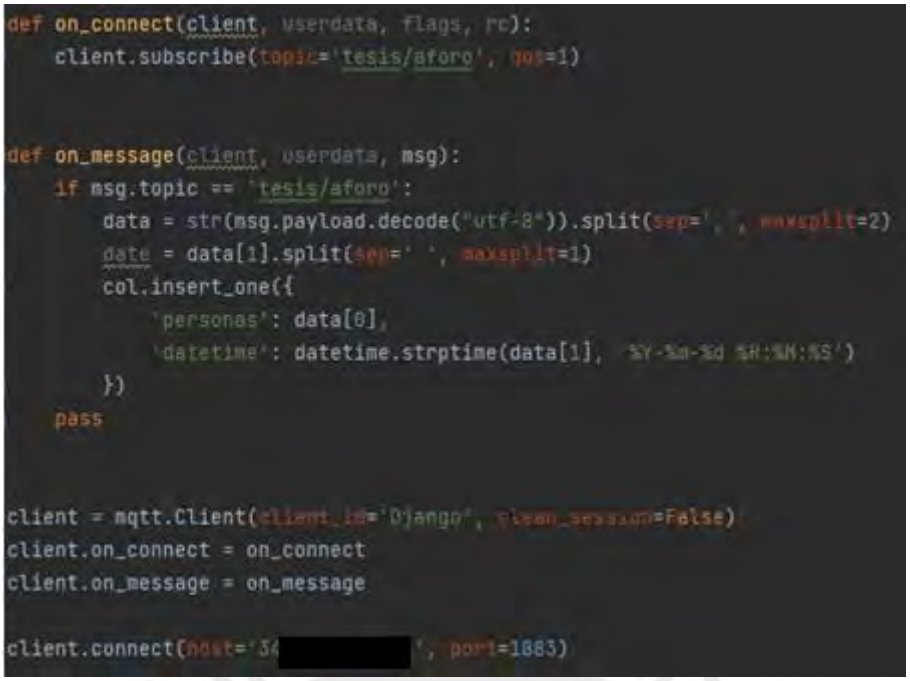


FIGURA 4.19 CLIENTE MQTT EN BACK END DEL SISTEMA WEB [ELABORACIÓN PROPIA]

Para el guardado en MongoDB también se necesita generar una conexión para efectuar el guardado de información del aforo en tiempo real, por lo que en la figura 4.20 se efectúa.

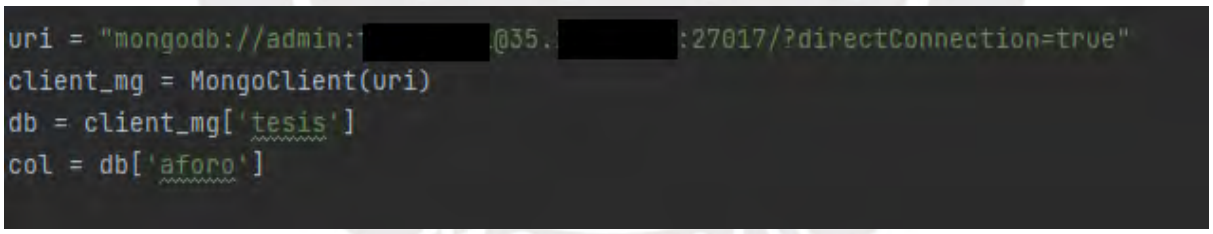


FIGURA 4.20 CLIENTE PARA MONGODB EN BACK END DEL SISTEMA WEB [ELABORACIÓN PROPIA]

Para el funcionamiento de estos componentes mencionados se tiene el siguiente diagrama de flujos en la Figura 4.21, el cual explica que una vez los clientes hayan sido creados se evaluará que el mensaje recibido pertenezca al tópico “tesis/aforo” para así guardar la data del aforo y el tiempo en el que fue capturado en la base de datos no relacional.

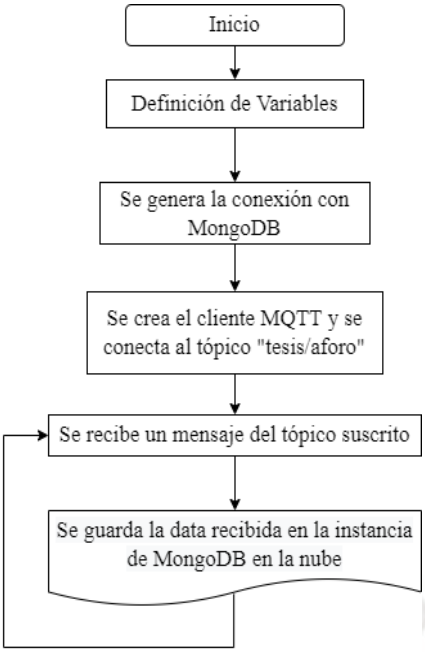


FIGURA 4.21 DIAGRAMA DE FLUJOS DEL FUNCIONAMIENTO DE MQTT Y MONGODB DEL LADO DEL BACKEND DE LA PLATAFORMA WEB

[ELABORACIÓN PROPIA]

Para el desarrollo de la página web se tiene el consumo de la base de datos relacional PostgreSQL creada previamente, es por ello que en la figura 4.22 se produce la conexión con la base de datos remota para así crear las tablas correspondientes por medio del comando “python manage.py makemigrations” y “python manage.py migrate” que son parte de Django, las cuales permiten modelar la base de datos desde python y reflejarlos en la base de datos.

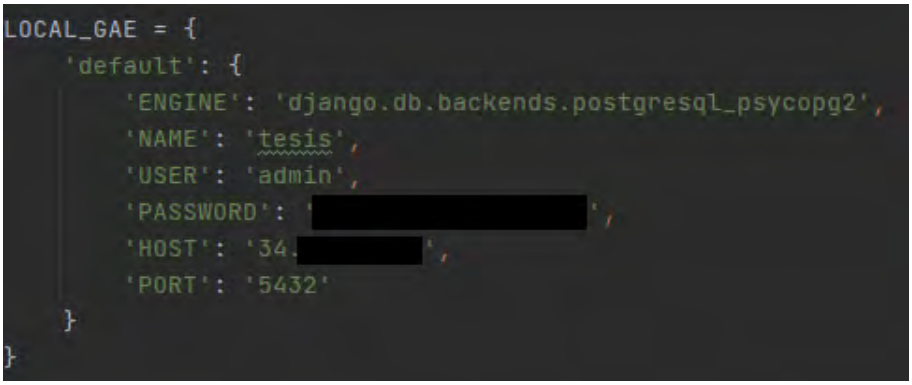


FIGURA 4.22 DIAGRAMA DE FLUJOS DEL FUNCIONAMIENTO DE MQTT Y MONGODB DEL LADO DEL BACKEND DE LA PLATAFORMA WEB

[ELABORACIÓN PROPIA]

Partiendo de esta base se desarrollarán los flujos que interactúan directamente con el usuario a través de las vistas de la página web y cumpliendo con los requerimientos expuestos en el cuadro 3.1; entonces en primer lugar para el inicio de sesión se da por medio del ingreso de las credenciales de correo electrónico (para el administrador esto puede ser cambiado por medio de soporte) y contraseña y para que sea exitoso las credenciales deben estar registradas en la base de datos y con un estado de usuario de activo. En la figura 4.23 se muestra la vista de inicio de sesión desde la perspectiva de una computadora y un móvil

FIGURA 4.23 VISTA DE INICIO DE SESIÓN DESDE PERSPECTIVA DE COMPUTADORA Y MÓVIL [ELABORACIÓN PROPIA]

En la figura 4.24 se tiene el diagrama de flujo de la autenticación del usuario en la plataforma del sistema de modo que se detalle a mayor nivel este proceso.

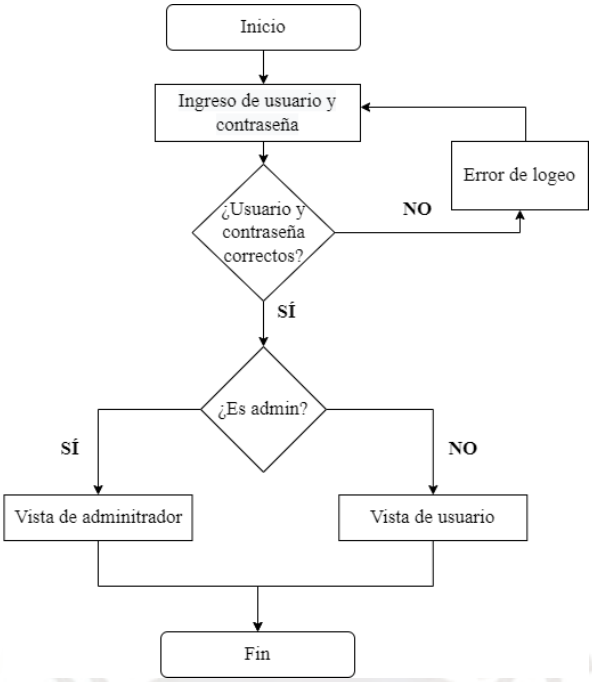


FIGURA 4.24 DIAGRAMA DE FLUJO DE LA AUTENTICACIÓN DEL USUARIO EN EL SISTEMA WEB [ELABORACIÓN PROPIA]

Por el lado del registro en la plataforma solo permitirá crear un usuario con rol de estudiante y con las validaciones indicadas en el cuadro 3.1, de forma que si los datos ingresados cumplen las reglas estipuladas la creación será hecha en base de datos, pero con un estado de inactivo para el sistema, lo que significa que no podrá iniciar sesión en la página hasta que sea aprobado por el administrador. A partir de lo mencionado, en la figura 4.25 se muestra la vista del registro en versión web y móvil; mientras que en la figura 4.26 se detalla el proceso por medio de un diagrama de flujo.

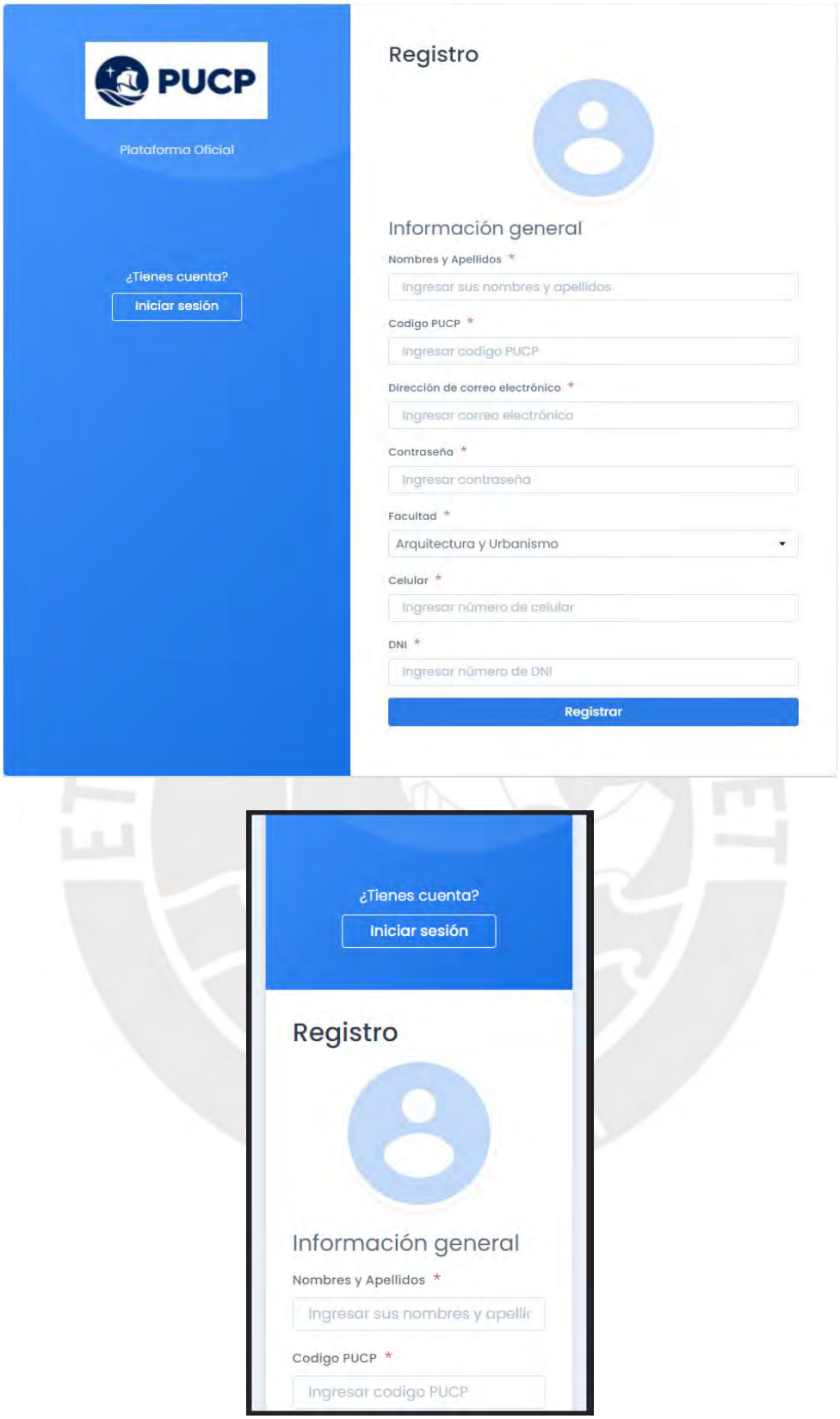


FIGURA 4.25 VISTA DE REGISTRO DESDE PERSPECTIVA DE COMPUTADORA Y MÓVIL [ELABORACIÓN PROPIA]

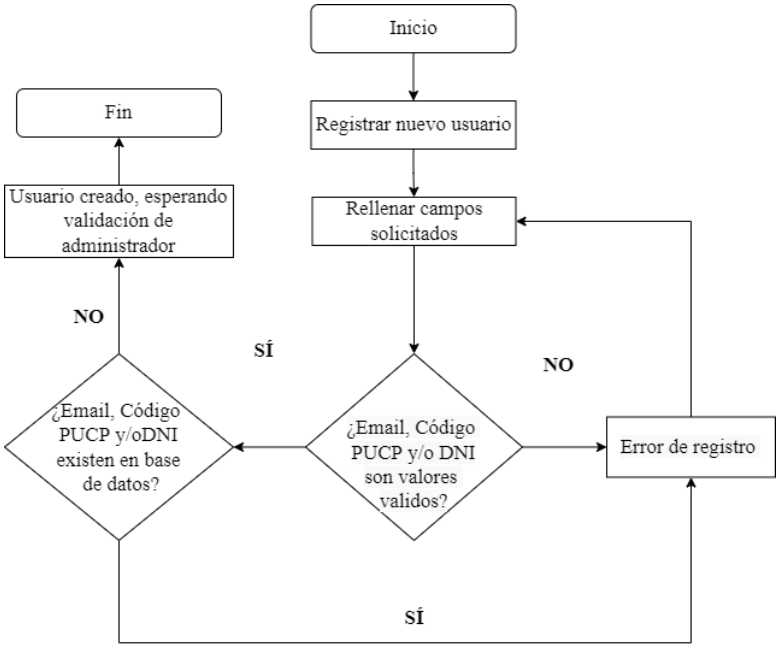


FIGURA 4.26 DIAGRAMA DE FLUJO DEL REGISTRO DE USUARIOS EN EL SISTEMA WEB [ELABORACIÓN PROPIA]

Luego el sistema presenta un sistema de gestión de usuarios con rol estudiante para el administrador, de forma que solo se acepten personas validando su información ingresada y para ello se tiene una lista de estudiantes no verificados (Figura 4.27)

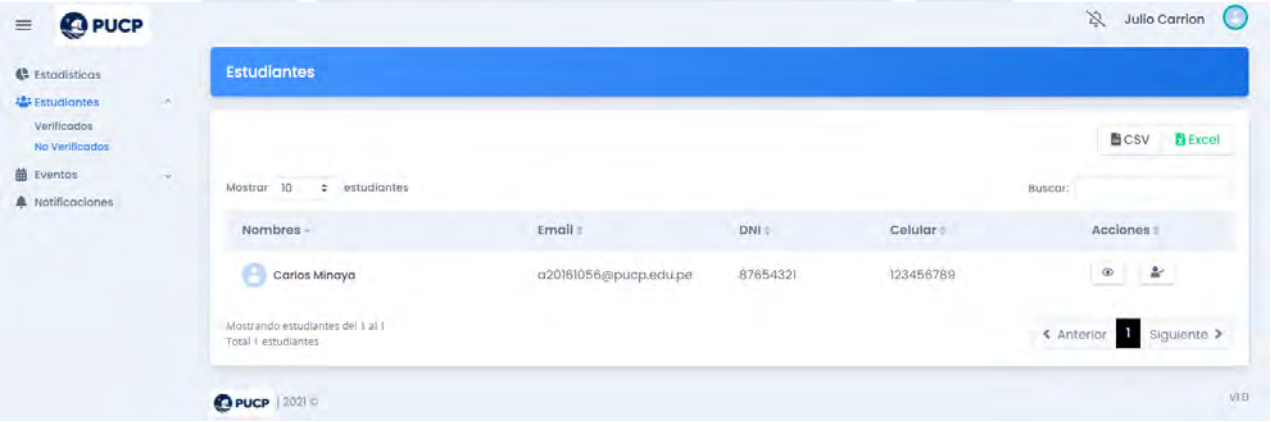


FIGURA 4.27 VISTA DE LA LISTA DE USUARIOS DE ROL ESTUDIANTE NO VERIFICADOS [ELABORACIÓN PROPIA]

Luego de verificar al estudiante, se actualiza su estado a activo de manera que pasa a la lista de usuarios verificados y ello se puede mostrar en la figura 4.28.

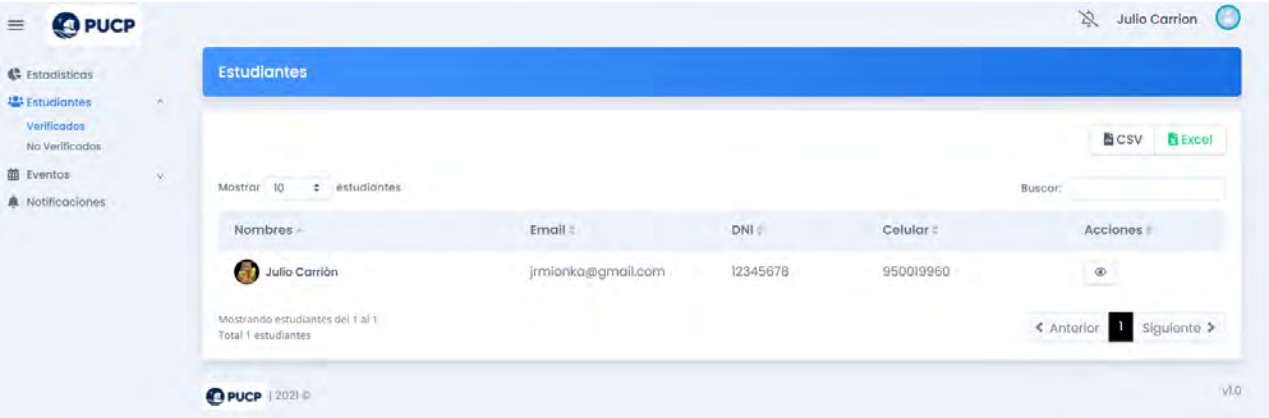


FIGURA 4.28 VISTA DE LA LISTA DE USUARIOS DE ROL ESTUDIANTE VERIFICADOS [ELABORACIÓN PROPIA]

Para brindar un mayor detalle de este proceso se tiene la figura 4.29 que es un diagrama de flujo de la administración de usuarios del lado del usuario de rol administrador.

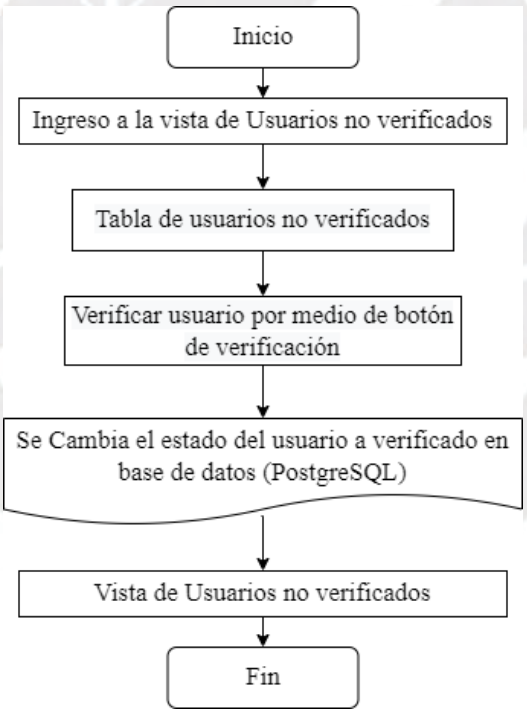


FIGURA 4.29 DIAGRAMA DE FLUJO DE LA ADMINISTRACIÓN DE USUARIOS CON ROL DE ESTUDIANTE EN EL SISTEMA WEB [ELABORACIÓN PROPIA]

Luego de ello se tiene la vista de Estadísticas del sistema, donde se muestra el aforo en tiempo real de la zona de detección, gráfico estadístico por día del aforo, gráfico de la semana

transcurrida del aforo, gráfico de cantidad por estudiantes registrados y gráfico de cantidad de estudiantes verificados y no verificados. Todo ello es mostrado en las imágenes 4.30, 4.31, 4.32 y 4.33.

FIGURA 4.30 GRÁFICO DE ESTUDIANTES VERIFICADOS Y NO VERIFICADOS EN EL SISTEMA WEB [ELABORACIÓN PROPIA]

FIGURA 4.31 GRÁFICO DE ESTUDIANTES REGISTRADOS POR FACULTAD [ELABORACIÓN PROPIA]

FIGURA 4.32 GRÁFICO DE AFORO POR DÍA Y MARCADOR DE AFORO EN TIEMPO REAL [ELABORACIÓN PROPIA]

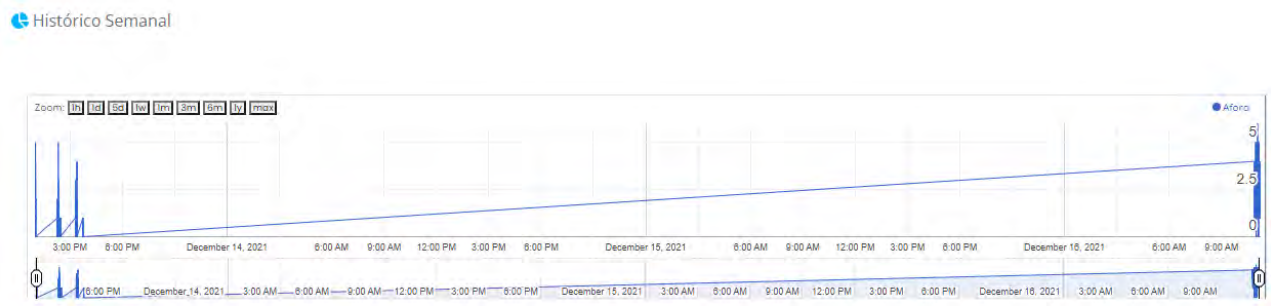


FIGURA 4.33 GRÁFICO DE AFORO DE LA SEMANA TRANSCURRIDA ACTUAL [ELABORACIÓN PROPIA]

Luego para el flujo de notificaciones se tiene el siguiente formulario de modo que será decisión del administrador si quiere tener una alerta que al exceder el tiempo indicado el aforo, se muestre un modal como aviso. En la figura 4.34 se muestra el formulario para activar las notificaciones y la figura 4.35 se tiene el diagrama de flujo de manera más precisada.

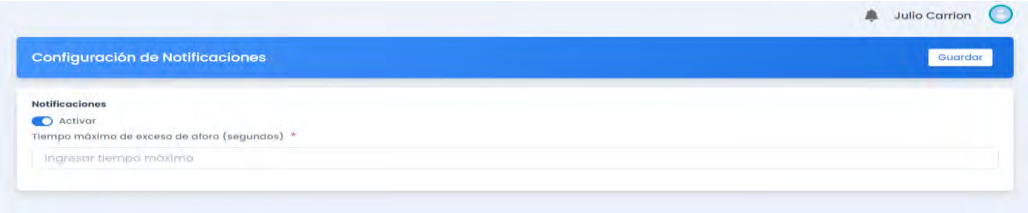


FIGURA 4.34 VISTA DEL FORMULARIO PARA ACTIVAR O DESACTIVAR NOTIFICACIONES [ELABORACIÓN PROPIA]

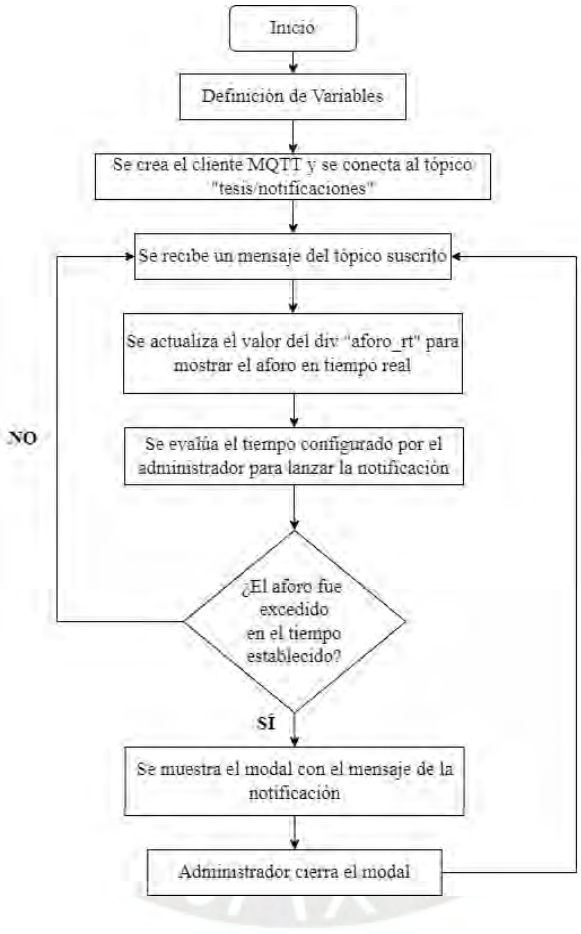


FIGURA 4.35 DIAGRAMA DE FLUJO DEL FUNCIONAMIENTO DE LAS NOTIFICACIONES EN CASO DE ESTAR ACTIVADAS [ELABORACIÓN PROPIA]

Finalmente, para que la plataforma tenga un valor agregado se implementó un sistema de gestión de eventos a forma de brindar cierta publicidad, por lo que se tiene primero la etapa de creación de eventos, que al igual que los flujos previos cumplen con los requerimientos expuestos en el capítulo 3 para las validaciones; entonces para efectuar la creación se debe

llenar un formulario con información del evento. Con lo mencionado, en la figura 4.36 se tiene el formulario para la creación de eventos y la figura 4.37 representa el diagrama de flujos con la función de especificar el proceso.

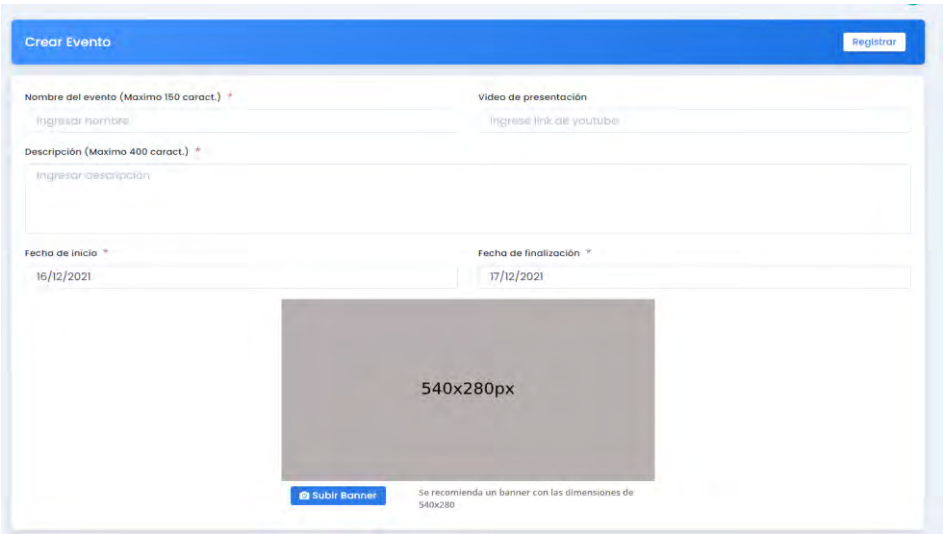


FIGURA 4.36 VISTA DEL FORMULARIO PARA LA CREACIÓN DE EVENTOS [ELABORACIÓN PROPIA]

FIGURA 4.37 DIAGRAMA DE FLUJO DEL FUNCIONAMIENTO DE CREACIÓN DE EVENTOS [ELABORACIÓN PROPIA]

Una vez creado el evento, este puede ser editado y eliminado solo si la fecha y hora de inicio del evento ha sido transcurrido; además este deja de mostrarse en el sistema si es que la fecha y hora fin ha pasado; por ello en la figura 4.38 se muestra la lista de eventos del sistema (En caso de ser administrador aparecen botones de edición y eliminación),

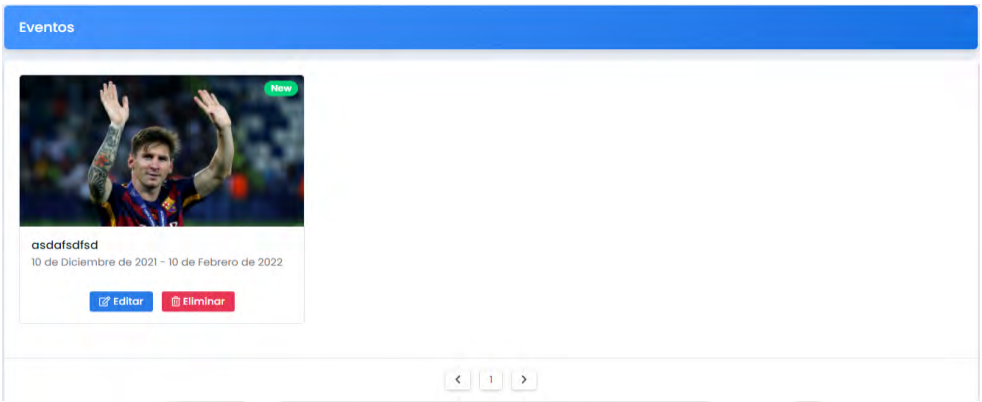


FIGURA 4.38 VISTA DE LA LISTA DE EVENTOS PARA EL ADMINISTRADOR

[ELABORACIÓN PROPIA]

Las dos últimas vistas del sistema serían el detalle del evento que contiene toda la información ingresada en el formulario y la modificación del evento, por lo que en la figura 4.39 y 4.40 se muestran respectivamente.

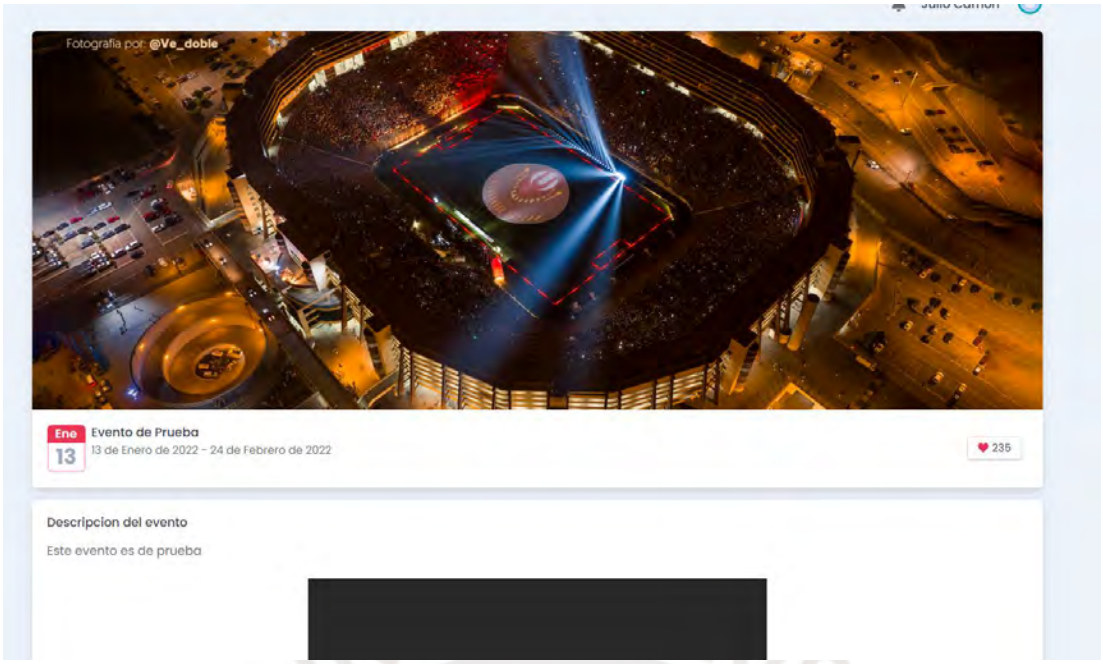


FIGURA 4.39 VISTA DEL DETALLE DEL EVENTO [ELABORACIÓN PROPIA]

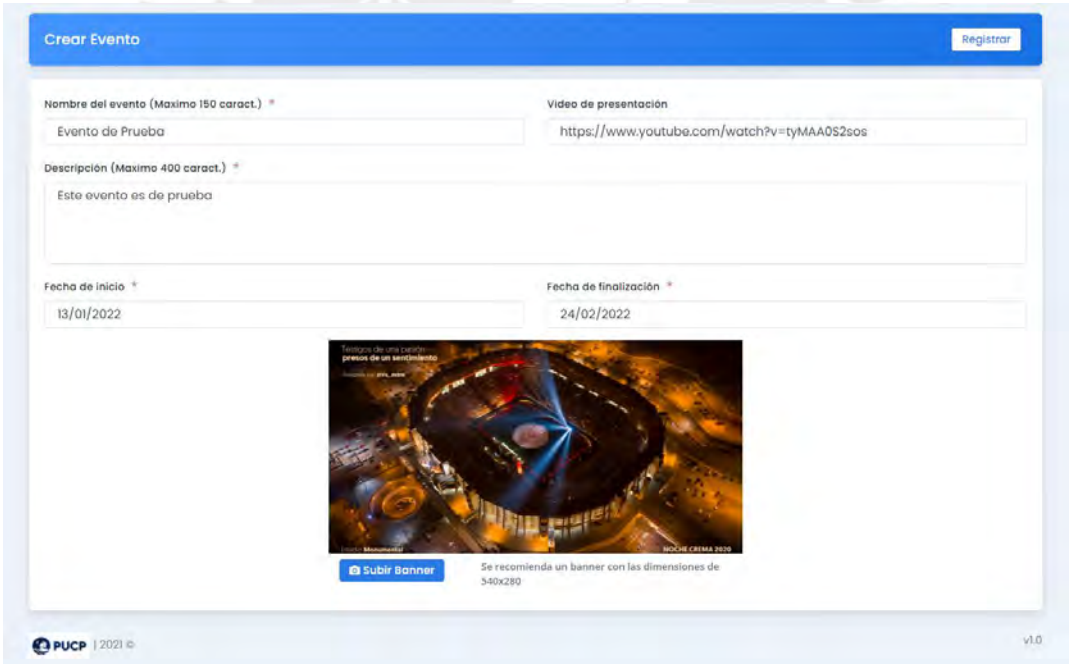


FIGURA 4.40 VISTA DE LA EDICIÓN DE UN EVENTO [ELABORACIÓN PROPIA]

4.6 Pruebas de funcionamiento

Se realizaron las pruebas de funcionamiento en el domicilio de Julio Carrion, simulando un área de estudio en la sala. Esto se debe a que no se tuvo acceso a las instalaciones de la universidad para poder probar el sistema en clases piloto con una mayor cantidad de aforo. La cámara con el Raspberry fue situadas perpendicularmente a la mesa. En las figuras 4.41 y 4.42 se puede observar mediciones correctas, obteniendo unos FPS de 4.46 en promedio. Además, el sistema detecta dos personas con una precisión aproximada del 74.5%.

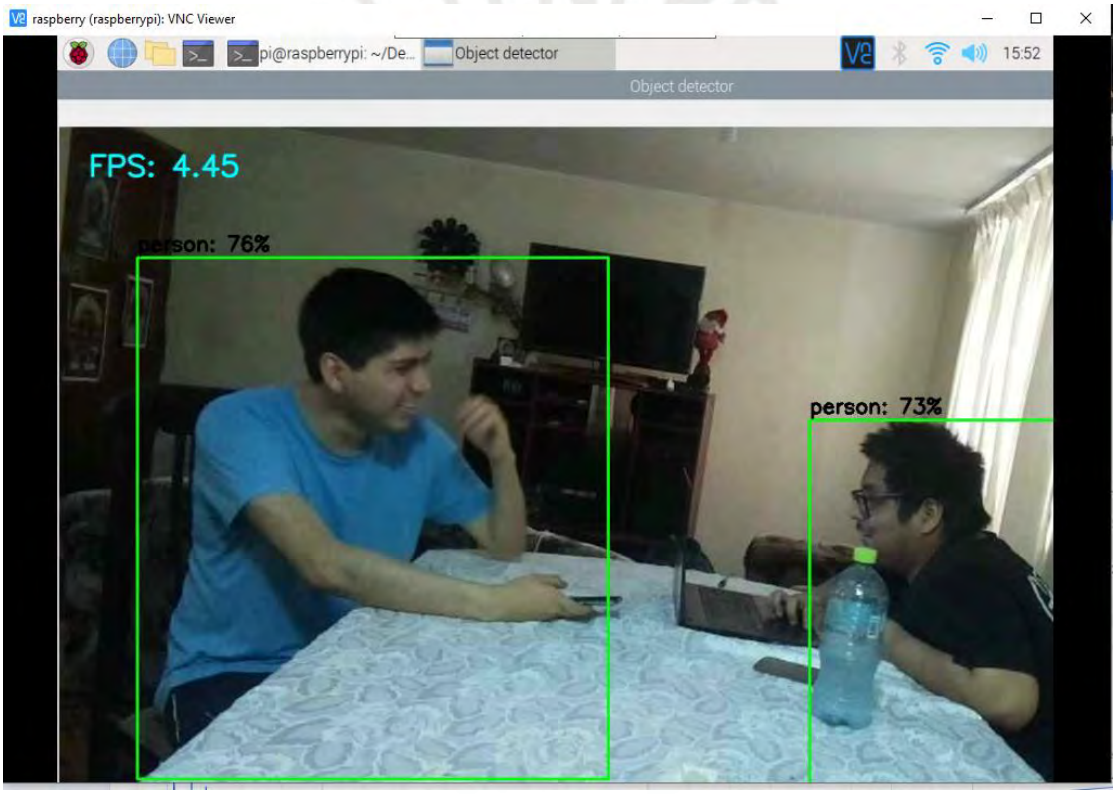


FIGURA 4.41: PRUEBA CON DOS PERSONAS SENTADAS CONVERSANDO

[ELABORACIÓN PROPIA]

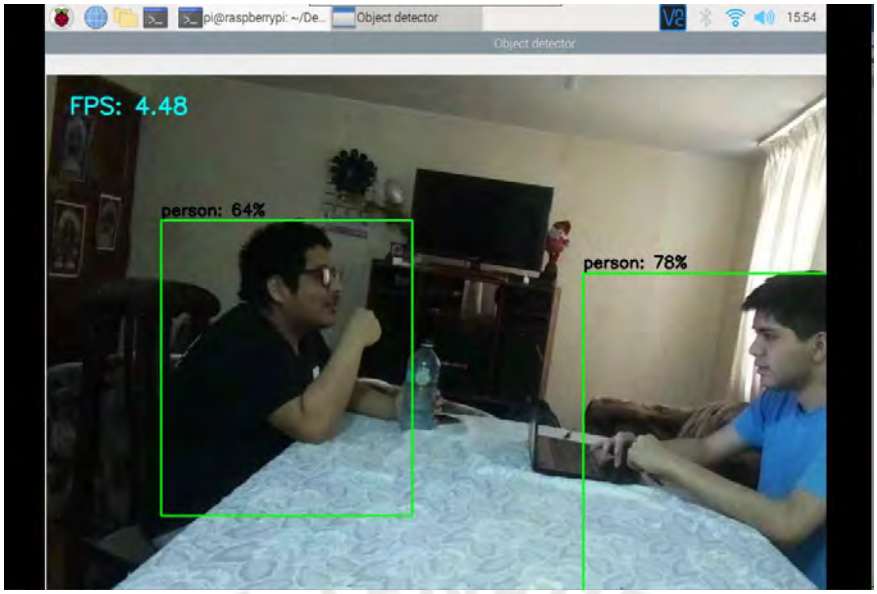


FIGURA 4.42: PRUEBA 2 CON DOS PERSONAS SENTADAS CONVERSANDO [ELABORACIÓN PROPIA]

En la figura 4.43, se puede observar una persona recostada en la mesa y el algoritmo de detección puede reconocer que es una persona con una probabilidad del 64%. Mientras que la otra persona de al frente, mantiene la misma probabilidad de antes. En promedio los FPS se mantienen en valores similares.

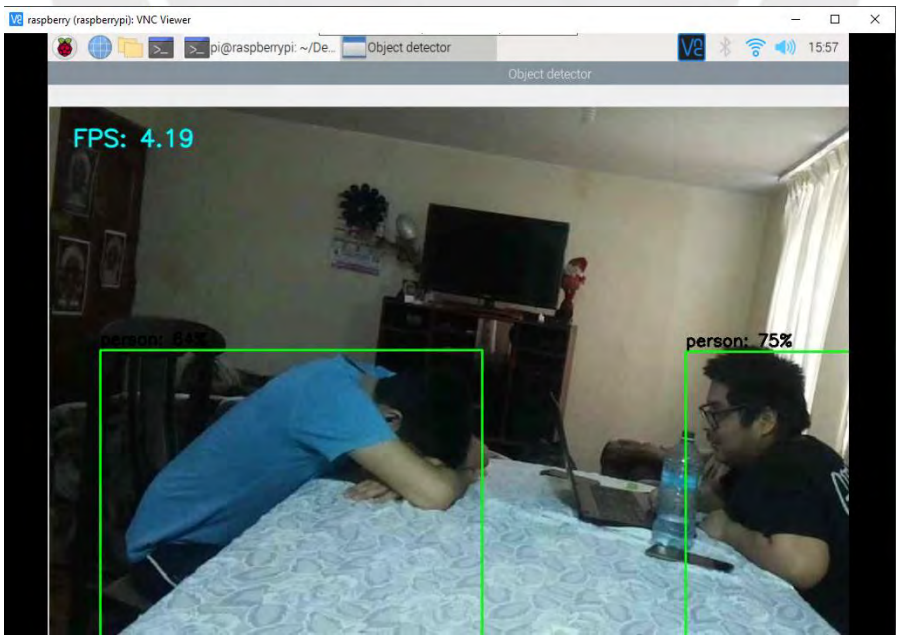


FIGURA 4.43: PRUEBA CON PERSONA RECOSTADA [ELABORACIÓN PROPIA]

En las figuras 4.44, 4.45 y 4.46, se puede observar dos personas interactuando con su respectiva mascarilla, lo cual simula una situación actual en cualquier espacio cerrado público. Por su parte, el algoritmo de detección no sufre de problemas al detectar a las personas, obteniendo una probabilidad de confianza en promedio del 65%.

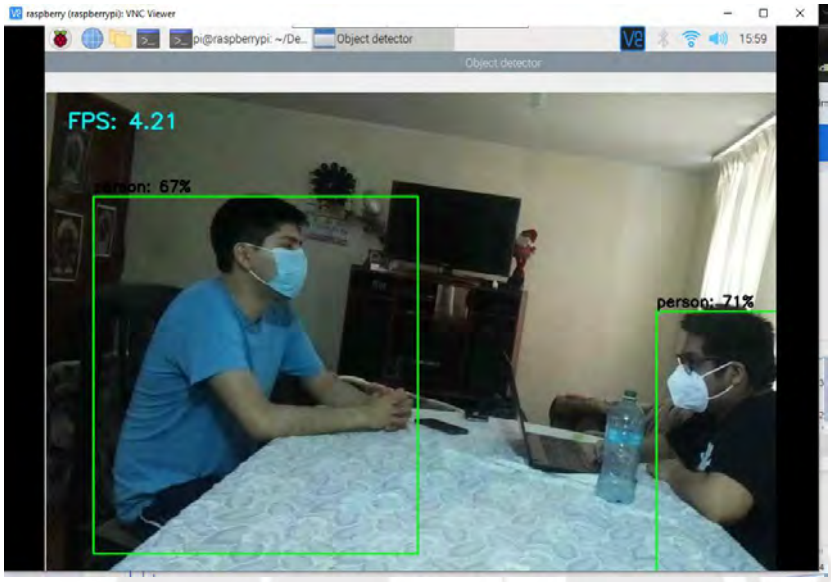


FIGURA 4.44: PRUEBA DE PERSONAS ESTUDIANDO CON MASCARILLA

[ELABORACIÓN PROPIA]

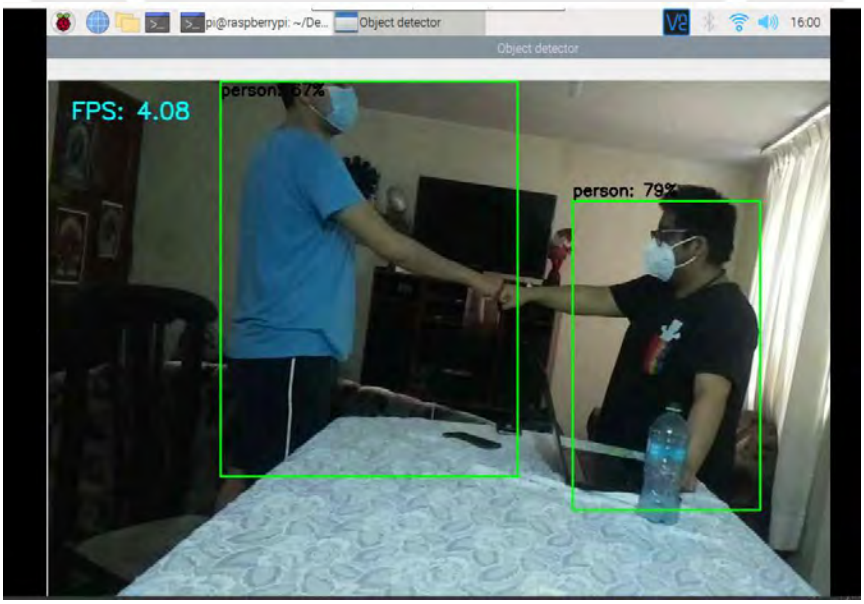


FIGURA 4.45: PRUEBA DE PERSONAS DANDO SALUDO CON PUÑOS

[ELABORACIÓN PROPIA]

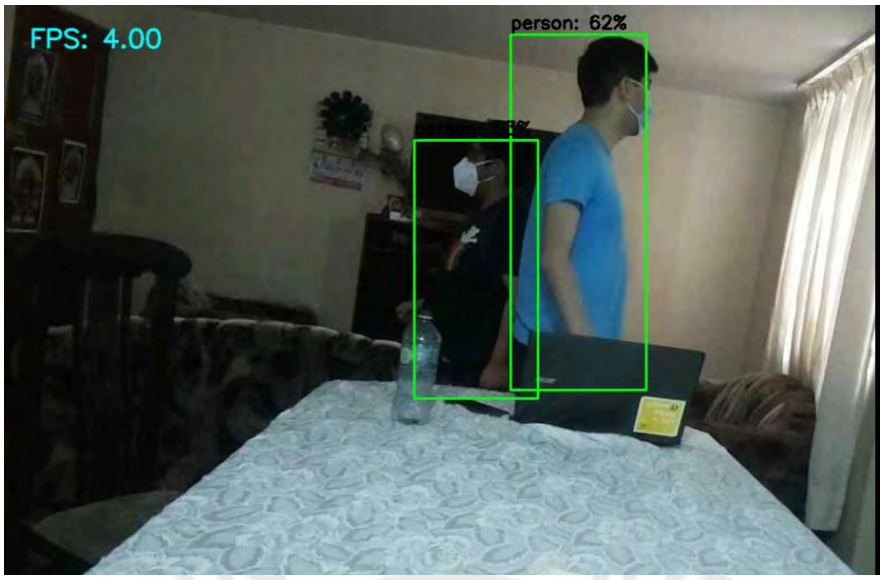


FIGURA 4.46: PRUEBA CON PERSONAS CAMINANDO [ELABORACIÓN PROPIA]

Con el permiso de la sección de ingeniería, se pudo ingresar al salón 307 del pabellón V en la Pontificia Universidad Católica del Perú (PUCP). Se instalo la solución presentada en la esquina superior del aula y como se observa en la figura 4.47 y 4.48, la detección de personas es correcta y acertada con una probabilidad en promedio del 69%

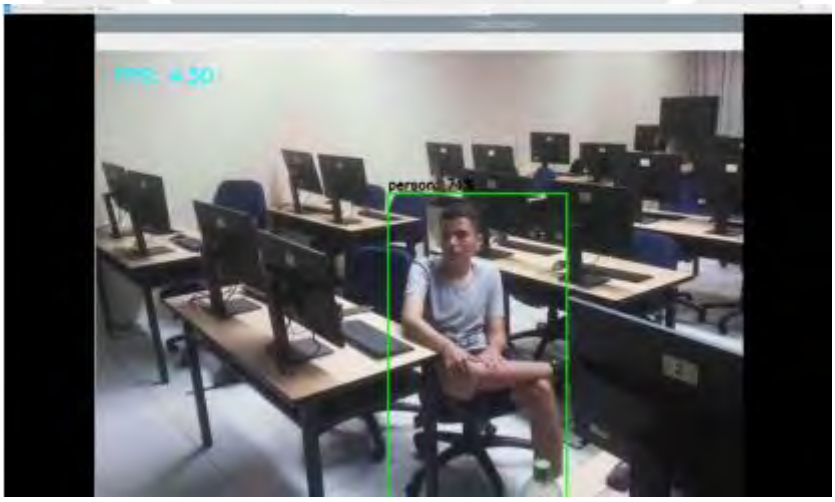


FIGURA 4.47: PRUEBA CON UNA PERSONA EN EL V307 – PUCP [ELABORACIÓN PROPIA]

FIGURA 4.48: PRUEBA CON DOS PERSONAS EN EL V307 – PUCP [ELABORACIÓN PROPIA]

En la página web, el usuario y administrador pueden observar el aforo en tiempo real. Adicionalmente, el administrador puede ver graficas relacionadas al aforo de una fecha en específico o de un resumen semanal.

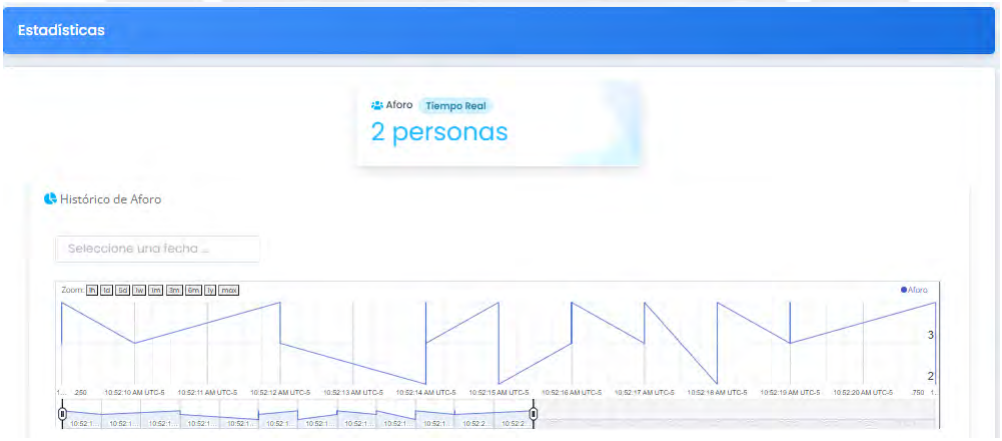


FIGURA 4.49: GRÁFICO DE AFORO [ELABORACIÓN PROPIA]

Debido a que se está limitado en términos de hardware por el Raspberry Pi 4B, el algoritmo usado es uno más sencillo y ligero que el Tensor Flow en su versión completa. Por ende, en

ciertos momentos la detección presenta errores en el conteo de personas. Esto incrementa al tener personas que interactúan de manera muy cercana como se puede ver en la figura 4.50, en la que se tienen dos personas cerca que son contabilizadas como tres. Se debe tener en cuenta que el valor del umbral para detectar una persona está configurado en 50%, esto ayuda a detectar a las personas en situaciones poco favorables, como por ejemplo cuando no muestran todo su cuerpo, la iluminación no es la adecuada, etc.

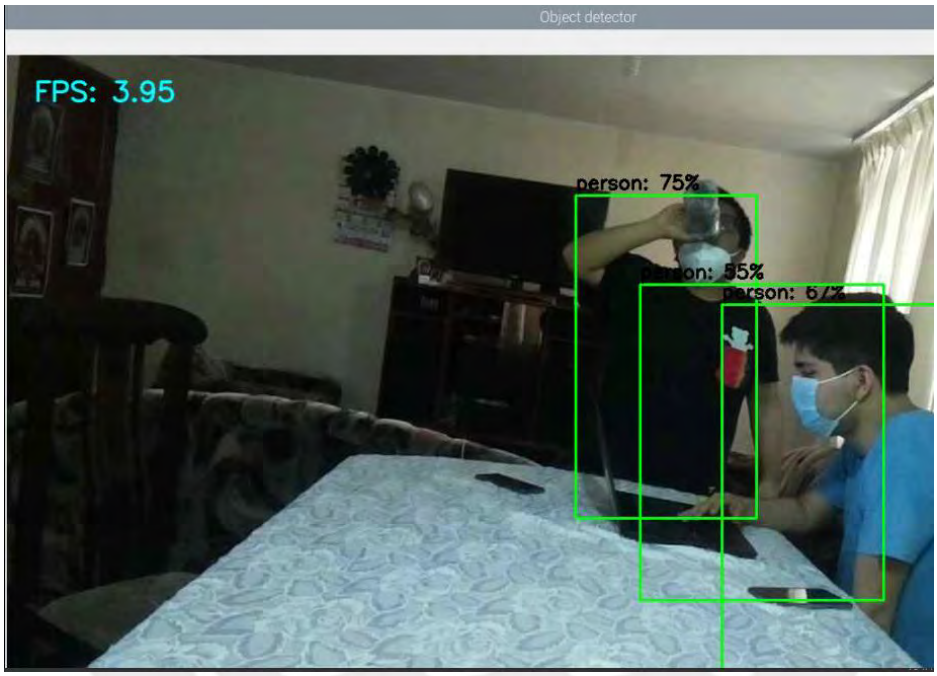


FIGURA 4.50: PRUEBA CON DETECCIÓN ERRÓNEA [ELABORACIÓN PROPIA]

Por otro lado, también se realizaron pruebas con el equipo ubicado en la pared superior de la casa con cierto ángulo de inclinación hacia abajo. Se puede observar en las figuras 4.51 y 4.52, como la detección sigue funcionando de manera correcta, incluso con mascarillas puestas.

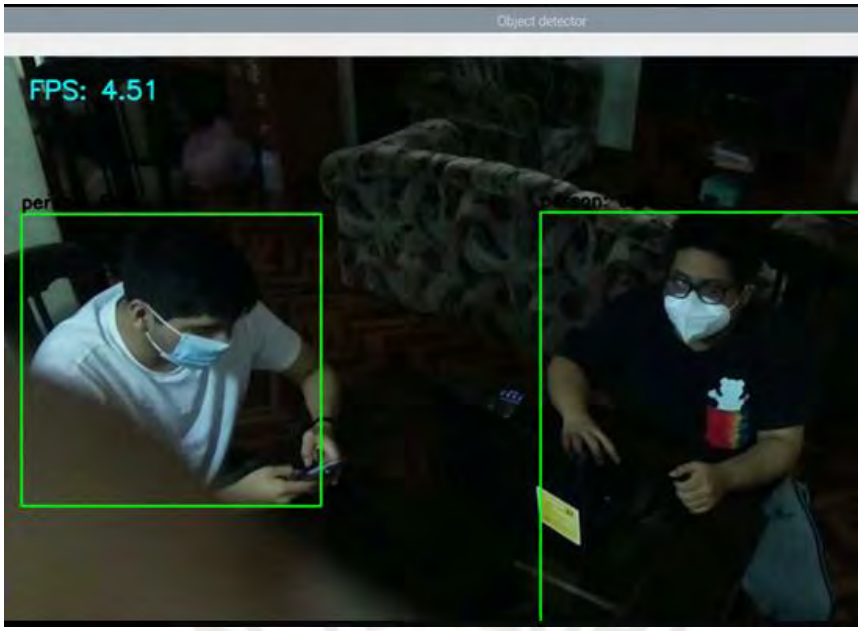


FIGURA 4.51: PRUEBA DESDE DISTINTO ÁNGULO [ELABORACIÓN PROPIA]

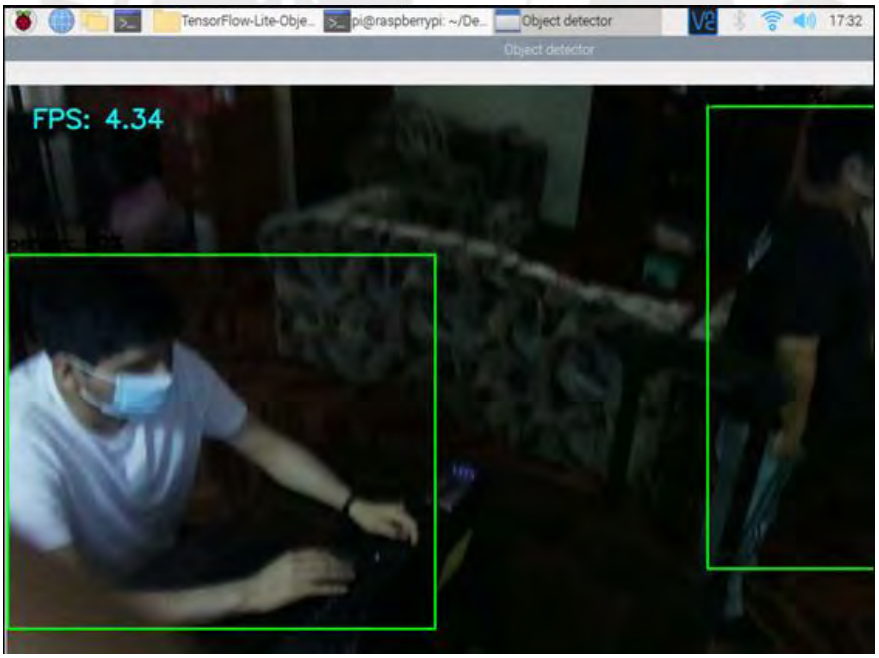


FIGURA 4.52: PRUEBA CON BAJA ILUMINACIÓN [ELABORACIÓN PROPIA]

En la figura 4.53, se tiene el ambiente con una mayor iluminación, la detección sigue siendo clara y precisa.

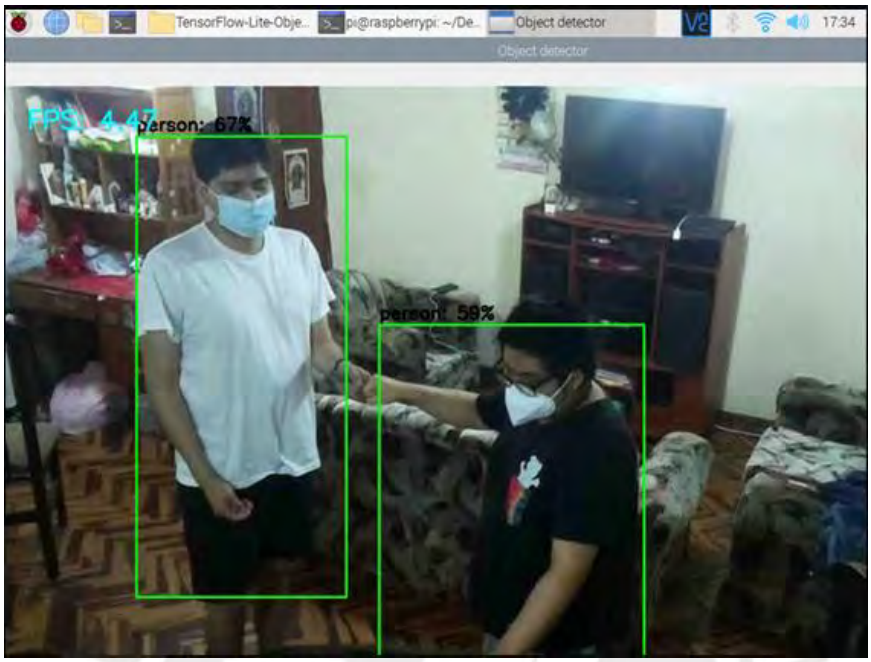


FIGURA 4.53: PRUEBA DESDE DISTINTO ÁNGULO Y MAYOR ILUMINACIÓN

[ELABORACIÓN PROPIA]

4.7 Elaboración de costos del servicio

Después de haber analizado y seleccionado todas las tecnologías y componentes implementados, se mencionarán los costos de CAPEX y OPEX de la solución propuesta. (Se considera un tipo de cambio de 1$ = S/.4)

CUADRO 4.1 COSTOS DE INVERSIÓN [ELABORACIÓN PROPIA]

	Descripción	Cantidad	Costo Total (Soles)	Costo Total (dólares)
Sector Edge				
Equipo	Kit Raspberry Pi 4 de 4GB RAM, incluye microSD de 32 GB clase 10, case, cooler, fuente compatible 5.0V 3A, 1 Pack de 3 disipadores y 1 cable micro HDMI a HDMI de 1.5 m.	1	S/.400	100$
Módulo de Cámara	Cámara Raspberry Oficial v2 de 8Mpx.	1	S/.140	35$
Sector Eléctrico				
Material	Adaptador de fuente con las siguientes características: alimentación de 10W, 2A, CA 110V-220V a CC 5V	1	S/.16	4$
Material	Cable LH 2.5 mm Azul x 25 m	1	S/.47.92	$11.98

Material	Cable LH 2.5 mm Verde/Amarillo x 25 m	1	S/.59.76	$14.94
Material	Cable LH 2.5 mm Rojo x 25 m	1	S/.50.92	$12.73
Material	Toma Doble Universal Blanco	1	S/.11.60	$2.90
Material	Caja Universal Toma eléctrica	1	S/.6	$1.50
Implementación				
Servicio de implementación	Mano de obra para la instalación, conexión y configuración de dispositivos SBC.	1	S/.1368	342$
Total			S/.2100.2	525.05$

CUADRO 4.2 COSTOS DE OPERACIÓN [ELABORACIÓN PROPIA]

	Descripción	Cantidad	Duración	Costo Total (Soles)	Costo Total (dólares)
Sector Cloud					
Infraestructura	Servicio de Compute Engine (e2-micro) con 10GB de almacenamiento y una IP pública para el mosquitto broker y el servicio web.	1	Mensual	S/.48.96	12.24$
Infraestructura	Servicio de Compute Engine (e2-small) con 80GB de almacenamiento para MongoDB	1	Mensual	S/.99.92	24.98$
Infraestructura	Servicio de Cloud SQL (db-estándar-1) para PostgreSQL con 100 SSD Gbi y una IP pública	1	Mensual	S/.371.36	92.84$
Mantenimiento					
Servicio de implementación	Mantenimiento físico y lógico de los dispositivos SBC.	1	Mensual	S/.160	40$
Total				S/.680.24	170.06$

4.8 Aportes de la tesis

4.8.1 Aportes en el ámbito ambiental

Esta solución usa equipos de procesamiento de bajo consumo energético, el Raspberry Pi 4 en modo de estrés puede llegar a gastar 6.25 W [72]. Entonces, teniendo en cuenta que su funcionamiento es de lunes a sábado de 8am hasta 10pm y considerando el precio de Kwh de 0.780 soles. Se tiene un consumo aproximado de 20 soles anuales, lo cual representa un gasto mínimo de electricidad, generando un impacto energético mínimo al medio ambiente.

4.8.2 Aportes en el ámbito social y legal

En el ámbito social y legal, esta solución no afecta en la confidencialidad de las personas, ya que las grabaciones no son guardadas. Además, no se detectan los rostros, sino el cuerpo de la persona.

4.8.3 Aportes en el ámbito económico

El consumo energético es mínimo, de aproximadamente 20 soles al año. Lo cual no generaría un impacto económico apreciable.

4.8.4 Aportes en el ámbito cultural

Esta tesis no tiene aporte en el ámbito cultura.

4.8.5 Aporte en el ámbito de salud

Como se mencionó en el capítulo 1, la demora en la búsqueda de espacios de estudio puede generar estrés en los estudiantes, afectando directamente a su salud mental. Con la solución propuesta en la tesis, se brinda el aforo en tiempo real mediante una aplicación web, que puede ser accedida fácilmente por el alumno desde su dispositivo móvil.

CONCLUSIONES

- Se logró cumplir con el objetivo general al implementar un diseño IoT basado en procesamiento de imágenes y detección de personas, con la finalidad de la determinación de aforo en las zonas de los asientos de libre disponibilidad gracias al uso de algoritmos de seguimiento y detección de objetos usando imágenes desarrollados en el Raspberry Pi 4. Además de una plataforma web con la finalidad de mostrar esta información recolectada tanto en tiempo real como en gráficas de históricos por día y por semana actual transcurrida siendo capaz de ser visualizada en distintos tipos de dispositivos.
- Para el envío de información recolectada se usó el protocolo de aplicación MQTT, debido a que este puede trabajar con anchos de banda limitados y de ese modo que la red de la universidad no sea saturada. Por otro lado, se tienen 2 bases de datos, donde el primero es un SQL relacionado a la información de gestión del servicio web, mientras que la base de datos NoSQL está enfocado al guardado de información recolectada por el Raspberry.
- Se realizó pruebas de funcionamiento en el la arquitectura desplegada en la nube con instancias de MongoDB del marketplace de Google Cloud, que es una máquina virtual llamado Compute Engine con 2 vCPU, 2 GB de memoria RAM y almacenamiento de 80 GB; mientras que para el broker de mosquitto y el despliegue de la página web es un Compute Engine con 2 vCPU, 1GB de RAM y almacenamiento de 10GB; asimismo para la instancia de la base de datos PostgreSQL se usa una instancia de Cloud SQL. Con lo mencionado se obtuvo que tanto el algoritmo como el sistema web trabajan en conjunto de manera esperada para los requerimientos detallados en la presente tesis como son las actualizaciones en tiempo real de valores de aforo.

- Se demuestra que el sistema desarrollado es capaz de obtener, guardar y mostrar el aforo en tiempo real para las condiciones de la zona de pruebas; además de presentar pruebas desde varios ángulos de la cámara, demostrando así los límites del algoritmo usado y su desempeño. Por otro lado, las gráficas estadísticas del aforo permiten el seguimiento continuo a lo largo del día o semana, facilitando así la gestión administrativa de la biblioteca, como son la apertura y clausura del establecimiento.

BIBLIOGRAFÍA

[1] PUCP. (2019, 23 julio). *Datos académicos*. PUCP | Pontificia Universidad Católica del Perú. Recuperado 2021, de https://www.pucp.edu.pe/la-universidad/nuestra-universidad/pucp-cifras/datos-academicos/?seccion=comunidad-universitaria&area=pregrado

[2] Cabello, S. (2020, 21 octubre). Telecomunicaciones en el Perú: ¿Cómo generar un impacto que alcance los US$15 mil millones? *El Comercio Perú*. https://elcomercio.pe/economia/dia-1/telecomunicaciones-telecomunicaciones-en-el-peru-como-generar-un-impacto-que-alcance-los-us15-mil-millones-movistar-claro-entel-bitel-noticia/

[3] Google Maps. *Mapa de bibliotecas en la PUCP* [Mapa]. Recuperado 15 de Mayo de 2021 de https://www.google.com/maps/search/bibliotecas+en+pucp/@-12.0696014,-77.0811353,16z

[4] ProcessMaker. (2020, marzo). *Cómo convertir su universidad en un campus inteligente*. https://www.processmaker.com/es/blog/how-to-turn-your-university-into-a-smart-campus/

[5] Jones, M. (2021, 9 noviembre). *What is a Smart Campus and the Benefits to College Students and Faculty*. Cox Business. Recuperado 20 de Abril de 2021, de https://www.coxblue.com/what-is-a-smart-campus-and-the-benefits-to-college-students-and-faculty/

[6] Garcia, M. (2019, 8 enero). *IoT - Internet Of Things*. Deloitte Spain. Recuperado 15 de noviembre de 2021, de https://www2.deloitte.com/es/es/pages/technology/articles/IoT-internet-of-things.html

[7] RedHat. (s. f.). *El concepto del edge computing*. Recuperado 18 de octubre de 2021, de https://www.redhat.com/es/topics/edge-computing

[8] ACCESOR. (2021, 3 junio). *Control de aforo*. Recuperado 19 de noviembre de 2021, de https://www.accesor.com/control-de-aforo/

[9] Magiturno. (s. f.). *CONTADORES DE PERSONAS ELECTRÓNICOS PARA CONTROL DE AFORO CON PROTOCOLOS DE BIOSEGURIDAD"*. Recuperado 18 de julio de 2021, de http://magiturno.com/producto/sistemas-de-conteo-de-personas-en-para-control-de-aforos/

[10] ACCESOR. (s. f.). *Torniquetes, Portillos y Tornos de acceso | Accesor*. Recuperado 18 de junio de 2021, de https://www.accesor.com/-/torniquetes-portillos-tornos/

[11] Grupo Omega. (2020, mayo). *SOLUCIÓN PARA EL CONTROL DE AFOROS*. https://grupo-omega.es/wp-content/uploads/2020/05/SOLUCIO%CC%81N-PARA-CONTROL-DE-AFOROS-COVID-19.pdf

[12] Cortez, A. (2018, octubre). *Planificación en redes de área local inalámbricas en escenarios internos: Elementos, herramientas y cuestiones prácticas*. https://doi.org/10.33412/pri.v9.1.2062

[13] Nfctagfactory. (s. f.). *NFC tag qr code*. Recuperado 20 de diciembre de 2021, de https://www.nfctagfactory.com/products/non-standard-shape-NFC-tag-qr-code.html#.YcBFPWjMLIW

[14] Cahuana, J. L. (2021, 28 enero). *¿POR QUÉ ES IMPORTANTE TENER UN PORTAL CAUTIVO?* Nettix Perú. Recuperado 20 de diciembre de 2021, de https://www.nettix.com.pe/blog/web-blog/por-que-es-importante-tener-un-portal-cautivo/

[15] WifiSafe. (s. f.). *Distribuidor WiFi - Ejemplos de portal cautivo para diferentes sectores con HotSpot | WifiSafe - WifiSafe*. Recuperado 20 de diciembre de 2021, de https://www.wifisafe.com/blog/ejemplos-portales-cautivos-diferentes-sectores/

[16] Plain Concepts. (2021, 14 septiembre). *Aplicación Smart Occupancy*. Recuperado 20 de diciembre de 2021, de https://www.plainconcepts.com/es/smart-occupancy-app/

[17] Peco, R. (2020, 5 junio). Aumentan las tecnologías que controlan el aforo en toda clase de espacios. *La Vanguardia*. https://www.lavanguardia.com/tecnologia/20200605/481599787542/distanciamiento-social-distancia-seguridad-covid-videocamaras-videovigilancia-canon-vivotek-hikvision-contar-personas-multitudes.html#foto-1

[18] Rosebrock, A. (2021, 13 julio). *OpenCV People Counter*. PyImageSearch. Recuperado 20 de diciembre de 2021, de https://www.pyimagesearch.com/2018/08/13/opencv-people-counter/

[19] Meel, V. (2021, 20 agosto). *People Counting System: How To Make Your Own in Less Than 10 Minutes*. Viso.Ai. Recuperado 20 de diciembre de 2021, de https://viso.ai/applications/people-counting-system/

[20] StreamLogic. (s. f.). *StreamLogic - Counting People*. Recuperado 20 de diciembre de 2021, de https://streamlogic.io/counting-people-in-motion.html

[21] IBM. (s. f.). *¿Qué es edge computing?* España | IBM. Recuperado 20 de diciembre de 2021, de https://www.ibm.com/es-es/cloud/what-is-edge-computing

[22] RedHat. (2021, 31 marzo). *¿Qué es edge computing?* Recuperado 20 de diciembre de 2021, de https://www.redhat.com/es/topics/edge-computing/what-is-edge-computing

[23] Kaur, J. (2021, 2 septiembre). *Overview of Edge Computing, The Impact of Edge Computing on IoT*. XenonStack. Recuperado 20 de diciembre de 2021, de https://www.xenonstack.com/blog/edge-computing

[24] Stalcup, K. (2019, 12 septiembre). *How Much Do the Differences Between Cloud Providers Actually Matter?* ParkMyCloud. Recuperado 20 de diciembre de 2021, de https://www.parkmycloud.com/blog/cloud-providers/

[25] Riofrío, M. M. (2020, 26 febrero). ¿Está perjudicialmente concentrado el mercado de Internet fijo en el Perú? *El Comercio Perú*. https://elcomercio.pe/economia/dia-1/internet-fijo-en-el-peru-esta-perjudicialmente-concentrado-el-mercado-osiptel-operadores-movistar-claro-entel-noticia/

[26] Gudino, M. (2020, 6 marzo). *Raspberry Pi 3 vs. Arduino Uno Rev3*. Arrow.com. https://www.arrow.com/es-mx/research-and-events/articles/comparing-arduino-uno-and-raspberry-pi-3

[27] Torre, A. (2020, 4 noviembre). *Deep Learning: clasificando imágenes con redes neuronales*. LIS Data Solutions. Recuperado 20 de diciembre de 2021, de https://www.lisdatasolutions.com/blog/deep-learning-clasificando-imagenes-con-redes-neuronales/

[28] Gandhi, R. (2018, 3 diciembre). *R-CNN, Fast R-CNN, Faster R-CNN, YOLO — Object Detection Algorithms*. Medium. Recuperado 20 de diciembre de 2021, de https://towardsdatascience.com/r-cnn-fast-r-cnn-faster-r-cnn-yolo-object-detection-algorithms-36d53571365e

[29] *Introduction to YOLO Algorithm for Object Detection*. (2021, 15 abril). Engineering Education (EngEd) Program | Section. Recuperado 20 de diciembre de 2021, de https://www.section.io/engineering-education/introduction-to-yolo-algorithm-for-object-detection/

[30] BaeSystems. (s. f.). *What are single-board computers?* BAE Systems | United States. Recuperado 20 de diciembre de 2021, de https://www.baesystems.com/en-us/definition/what-are-single-board-computers

[31] Shunlongwei. (2021, 20 mayo). *Microprocesador vs Microcontrolador: ¿Cuál es la diferencia? - Co. Ltd de Shunlongwei*. Recuperado 20 de diciembre de 2021, de https://www.shunlongwei.com/es/microprocessor-vs-microcontroller-what-is-the-difference/

[32] ARDUINO. (2021, 12 julio). *¿Qué es Arduino?* Arduino.cl. Recuperado 20 de diciembre de 2021, de https://arduino.cl/que-es-arduino/

[33] Opensource. (s. f.). *What is a Raspberry Pi?* Opensource.Com. Recuperado 20 de diciembre de 2021, de https://opensource.com/resources/raspberry-pi

[34] NVIDIA. (2021, 30 marzo). *Jetson Nano*. NVIDIA Developer. Recuperado 20 de diciembre de 2021, de https://developer.nvidia.com/embedded/jetson-nano

[35] MQTT. (s. f.). *MQTT - The Standard for IoT Messaging*. Recuperado 20 de diciembre de 2021, de https://mqtt.org/

[36] MDN contributors. (2021, 18 diciembre). *Generalidades del protocolo HTTP - HTTP | MDN*. Mozilla. Recuperado 20 de diciembre de 2021, de https://developer.mozilla.org/es/docs/Web/HTTP/Overview

[37] *API REST*. (2019, 27 octubre). Aprendiendo Arduino. Recuperado 20 de diciembre de 2021, de https://aprendiendoarduino.wordpress.com/tag/arquitectura-api/

[38] Suratica. (2021, 16 septiembre). *Qué es el Frontend*. SURATICA SOFTWARE. Recuperado 20 de diciembre de 2021, de https://www.suratica.es/que-es-el-frontend/

[39] PYPL. (s. f.). *PYPL PopularitY of Programming Language index*. Recuperado 20 de diciembre de 2021, de https://pypl.github.io/PYPL.html

[40] Clark, J. (2021, 13 septiembre). *Los 10 mejores marcos de trabajo de backend*. Back4App Blog. Recuperado 20 de diciembre de 2021, de https://blog.back4app.com/es/los-10-mejores-marcos-de-trabajo-de-backend/

[41] Django. (s. f.). *Django overview | Django*. Recuperado 20 de diciembre de 2021, de https://www.djangoproject.com/start/overview/

[42] Spring. (s. f.). *Spring Framework*. Recuperado 20 de diciembre de 2021, de https://spring.io/projects/spring-framework

[43] Universidad de Alicante. (2014, 26 junio). *Introducción a MVC en Spring* [Diapositivas]. Jtech. http://www.jtech.ua.es/j2ee/publico/spring-2012-13/sesion03-apuntes.html

[44] Datanyze. (s. f.). *Databases Market Share Report | Competitor Analysis | Microsoft SQL Server, MySQL, Microsoft Access*. Recuperado 20 de diciembre de 2021, de https://www.datanyze.com/market-share/databases--272

[45] PostgreSQL. (s. f.). *PostgreSQL: About*. The PostgreSQL Global Development Group. Recuperado 20 de diciembre de 2021, de https://www.postgresql.org/about/

[46] MySql. (s. f.). *MySQL :: MySQL 8.0: Up to 2x Faster*. Recuperado 20 de diciembre de 2021, de https://www.mysql.com/products/enterprise/database/

[47] Amazon. (s. f.). *Bases de datos en AWS: la herramienta correcta para el trabajo adecuado*. Amazon Web Services, Inc. Recuperado 20 de diciembre de 2021, de https://aws.amazon.com/es/nosql/

[48] Correspondent, T. (2020, 26 septiembre). *Top 5 NoSQL databases for Data Scientists in 2020*. TechGig. Recuperado 20 de diciembre de 2021, de https://content.techgig.com/top-5-nosql-databases-for-data-scientists-in-2020/articleshow/78330888.cms

[49] MongoDB. (s. f.). *¿Qué es MongoDB?* Recuperado 20 de diciembre de 2021, de https://www.mongodb.com/es/what-is-mongodb

[50] Amazon. (s. f.-a). *AWS | Servicio de base de datos gestionada NoSQL (DynamoDB)*. Amazon Web Services, Inc. Recuperado 20 de diciembre de 2021, de https://aws.amazon.com/es/dynamodb/

[51] Canalys. (2020, 29 abril). *Global cloud services market Q1 2021*. Recuperado 20 de diciembre de 2021, de https://www.canalys.com/newsroom/global-cloud-market-Q121

[52] Google. (s. f.). *App Engine Application Platform* |. Google Cloud. Recuperado 20 de diciembre de 2021, de https://cloud.google.com/appengine

[53] Amazon. (s. f.). *AWS | Cloud Computing - Servicios de informática en la nube*. Amazon Web Services, Inc. Recuperado 20 de diciembre de 2021, de https://aws.amazon.com/es/?nc1=h_ls

[54] Microsoft. (s. f.). *Conozca Azure*. Microsoft Azure. Recuperado 20 de diciembre de 2021, de https://azure.microsoft.com/es-es/overview/

[55] Albarracín, Á. H. (2020, 18 abril). *MQTT vs HTTP: ¿qué protocolo es mejor para IoT?* BorrowBits. Recuperado 20 de diciembre de 2021, de https://borrowbits.com/2020/04/mqtt-vs-http-que-protocolo-es-mejor-para-iot/#:%7E:text=Diferencias%20en%20la%20arquitectura&text=En%20MQTT%20se%20mantiene%20la,en%20MQTT%20es%20full%2Dduplex.

[56] MagneticValley. (2016). *MQTT* [Diapositivas]. Intermagnet. https://www.intermagnet.org/publications/dinant2016/mqtt.pdf

[57] Bartnitsky, J. (2018, 23 enero). *HTTP vs MQTT performance tests*. Flespi. Recuperado 20 de diciembre de 2021, de https://flespi.com/blog/http-vs-mqtt-performance-tests

[58] *4 Model B specifications –*. (s. f.). Raspberry Pi. Recuperado 20 de diciembre de 2021, de https://www.raspberrypi.com/products/raspberry-pi-4-model-b/specifications/

[59] *CREATIVIDAD AHORA – Tienda de productos electrónicos y cursos online*. (s. f.). CREATIVIDADAHORA. Recuperado 20 de diciembre de 2021, de https://creatividadahora.com/

[60] Bakshi, N., Kumar, P., Rastog, A., & Surya, D. (2020, junio). *Spring Framework vs Django Framework: A Comparative Study*. IRJET. https://www.irjet.net/archives/V7/i6/IRJET-V7I61162.pdf

[61] Olkhovsky, P. (2021, 20 abril). *Web frameworks: Django vs Spring*. Edgica. Recuperado 20 de diciembre de 2021, de https://www.edgica.com/web-frameworks-django-vs-spring/

[62] *Django VS Spring: Detailed comparison* -. (2020, 5 agosto). NIMAP INFOTECH. Recuperado 20 de diciembre de 2021, de https://nimapinfotech.com/blog/django-vs-spring/

[63] Similartech. (s. f.). *Django VS Spring - Framework Technologies Market Share Comparison*. Recuperado 20 de diciembre de 2021, de https://www.similartech.com/compare/django-vs-spring

[64] Pedamkar, P. (2021, 1 marzo). *MySQL vs NoSQL*. EDUCBA. Recuperado 20 de diciembre de 2021, de https://www.educba.com/mysql-vs-nosql/

[65] *StackPath*. (s. f.). Postgresqltutorial. Recuperado 20 de diciembre de 2021, de https://www.postgresqltutorial.com/postgresql-vs-mysql/

[66] GeeksforGeeks. (2021, 29 octubre). *Difference between MySQL and PostgreSQL*. Recuperado 20 de diciembre de 2021, de https://www.geeksforgeeks.org/difference-between-mysql-and-postgresql/

[67] GeeksforGeeks. (2020, 15 septiembre). *SQL vs NoSQL: Which one is better to use?* Recuperado 20 de diciembre de 2021, de https://www.geeksforgeeks.org/sql-vs-nosql-which-one-is-better-to-use/?ref=rp

[68] ParkMyCloud. (2018, 20 abril). *Cloud Services Comparison*. Recuperado 20 de diciembre de 2021, de https://www.parkmycloud.com/cloud-services-comparison/

[69] Google. (s. f.). *Google Cloud Pricing Calculator*. Google Cloud. Recuperado 20 de diciembre de 2021, de https://cloud.google.com/products/calculator

[70] Amazon. (s. f.). *AWS Pricing Calculator*. AWS. Recuperado 20 de diciembre de 2021, de https://calculator.aws/#/estimate

[71] The Pi Hut. (s. f.). *Raspberry Pi 4 Camera Case*. Recuperado 20 de diciembre de 2021, de https://thepihut.com/products/raspberry-pi-4-3-camera-case

[72] ¿Cuál es el consumo de una Raspberry Pi 3/4? (2021, 26 marzo). Bugeados. Recuperado 20 de diciembre de 2021, de https://bugeados.com/raspberry/cual-es-el-consumo-de-una-raspberry-pi-3-4/

RECOMENDACIONES

A continuación, se presentan las siguientes recomendaciones:

- El uso de un case de Raspberry con soporte para cámara ayudaría en la comodidad para un mejor manejo del equipo. Además, brindaría una mayor facilidad al instalar el equipo en la biblioteca del CIA.
- Como pudimos observar en las pruebas de funcionamiento, este algoritmo puede presentar error en ciertas situaciones. Por ende, implementar TensorFlow en su versión completa mejoraría los resultados finales. Esto sería posible añadiendo al Raspberry un dispositivo llamado Coral, el cual es un USB que acelera el procesamiento, mejorando el rendimiento y los FPS.
- Se recomienda usar un Cloud Storage de Google para el almacenamiento de archivos estáticos, fotos de los eventos y fotos de perfil de los usuarios.
- Se recomienda usar una conexión HTTP segura, de esta manera, se podría usar el Google App Engine (GAE), para levantar la página web. Este GAE, ofrece una monitorización y automatización en el despliegue en la nube.

TRABAJOS A FUTURO

A partir de la solución propuesta:

- Se puede implementar la validación de distanciamiento entre personas a modo que se cumpla con el distanciamiento social obligatorio generado por el COVID 19.
- Se puede utilizar las cámaras a modo de seguridad de videovigilancia, por lo que puede ser un sistema de apoyo para las cámaras actuales que se tienen en la zona de la biblioteca.
- Se puede entrenar un modelo customizado para detección de modo de que esta esté enfocada a las personas, lo cual no está dado por parte del modelo escogido para TensorFlow Lite.

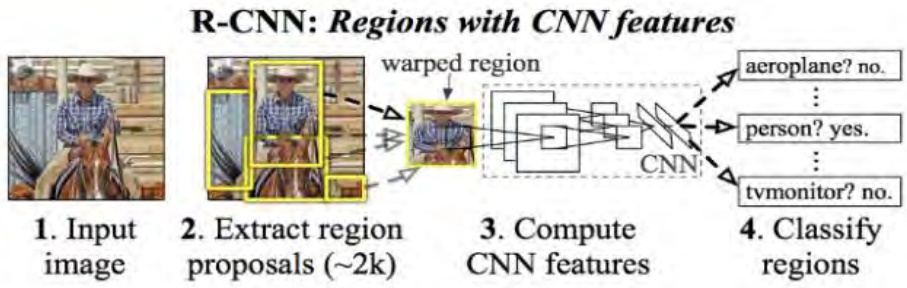
ANEXO

RCNN

Como se sabe para la detección y calificación se extraían todas las regiones de la imagen a procesar sin embargo estas podrían ser muchas de modo que el resultado tomaría un tiempo prolongado; por lo que en R-CNN se hace una búsqueda selectiva bajo la premisa de solo 2000 regiones de la imagen llamadas "Regiones Propuestas" y estas son generadas bajo el siguiente criterio de búsqueda selectiva:

1. Generación de subsegmentación inicial, donde se tienen regiones candidatas.
2. Uso de un algoritmo para combinar de forma recursiva regiones que sean similares en regiones mucho más grandes.
3. Usar las regiones grandes como "Regiones propuestas" candidatas.

Una vez que se obtienen las 2000 "Regiones propuestas" candidatas, estas son deformadas en un recuadro y son alimentadas de la red neuronal convolucional produciéndose un vector de 4096 dimensiones en la salida. Todas las características extraídas de la imagen a la salida se introducen bajo una Máquina de Vectores de Soporte con la función de que se puedan clasificar los objetos detectados en cada región candidata; además se predice la existencia de un objeto bajo 4 valores importantes que aumentan la precisión de la misma. En la figura mostrada se muestra el procedimiento del algoritmo R-CNN.



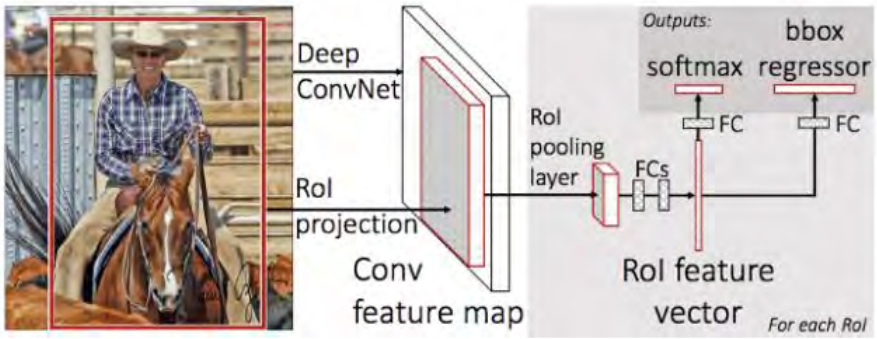
FUNCIONAMIENTO DEL ALGORITMO R-CNN [28]

Si bien este algoritmo cumple con la función de detección y clasificación de elementos, presenta ciertos problemas, los cuales son:

- El tiempo necesario para capacitar a la red neuronal sigue siendo alto, debido a que se requieren de 2000 regiones.
- Este algoritmo no puede ser usado para una aplicación en tiempo real por el tiempo que requiere por imagen.
- Este algoritmo puede tener "Regiones Propuestas" erróneas de forma que su búsqueda selectiva es fija.

Fast RCNN

Este algoritmo parte del R-CNN resolviendo ciertos inconvenientes, por lo que al ser una versión más rápida se le denominó Fast R-CNN. Si bien el concepto es similar, en lugar de obtener "Regiones Propuestas", se envía la imagen al CNN de forma que se genera un mapa de característica convolucional; por lo que, partiendo del mapa se identifican cuadros a través de una capa de agrupación denominada "Rol" luego se pasa por una capa Softmax para así predecir la clase de la región y los valores de compensación para el cuadro delimitador. La razón por la que este algoritmo es más rápido con respecto a R-CNN es que, en lugar de enviar 2000 regiones a la red neuronal cada vez, solo se realiza una vez por imagen y así se genera una capa característica a partir de ella. En la figura mostrada se muestra la arquitectura de Faster R-CNN.



ARQUITECTURA DE FAST R-CNN [28]

Por otro lado, el mayor problema que tiene este algoritmo es que la inclusión de las regiones ya mencionadas genera un cuello de botella afectando así su rendimiento.