UNIVERSIDAD NACIONAL DE CHIMBORAZO
FACULTAD DE INGENIERÍA
CARRERA DE INGENIERÍA EN TECNOLOGÍAS DE LA INFORMACIÓN

Desarrollo de prototipo móvil-web para gestionar disponibilidad y alertar intrusiones mediante autenticación facial en parqueaderos Cooperativa "Señor del Buen Suceso"

Trabajo de Titulación para optar al título de
Ingeniero en Tecnologías de la Información

Autor:
Guaman Pineda John Jairo

Tutor:
Ing. Miryan Estela Narváez Vilema, PhD

Riobamba, Ecuador. 2026

DECLARATORIA DE AUTORÍA

Yo, John Jairo Guaman Pineda, con cédula de ciudadanía 0605034933, autor del trabajo de investigación titulado: Desarrollo de prototipo móvil-web para gestionar disponibilidad y alertar intrusiones mediante autenticación facial en parqueaderos Cooperativa "Señor del Buen Suceso", certifico que la producción, ideas, opiniones, criterios, contenidos y conclusiones expuestas son de mí exclusiva responsabilidad.

Asimismo, cedo a la Universidad Nacional de Chimborazo, en forma no exclusiva, los derechos para su uso, comunicación pública, distribución, divulgación y/o reproducción total o parcial, por medio físico o digital; en esta cesión se entiende que el cesionario no podrá obtener beneficios económicos. La posible reclamación de terceros respecto de los derechos de autor de la obra referida será de mi entera responsabilidad; librando a la Universidad Nacional de Chimborazo de posibles obligaciones.

En Riobamba, 20 de abril de 2026.

John Jairo Guaman Pineda

C.I: 0605034933

DICTAMEN FAVORABLE DEL PROFESOR TUTOR

Quien suscribe, Miryan Estela Narváez Vilema catedrático adscrito a la Facultad de Ingeniería, por medio del presente documento certifico haber asesorado y revisado el desarrollo del trabajo de investigación titulado: Desarrollo de prototipo móvil-web para gestionar disponibilidad y alertar intrusiones mediante autenticación facial en parqueaderos Cooperativa "Señor del Buen Suceso", bajo la autoría de John Jairo Guaman Pineda; por lo que se autoriza ejecutar los trámites legales para su sustentación.

Es todo cuanto informar en honor a la verdad; en Riobamba, a los 27 días del mes de abril de 2026.

Validar únicamente en FirmaEC.
Firmado electrónicamente por:
MIRYAN ESTELA
NARVAEZ VILEMA

Miryan Estela Narváez Vilema

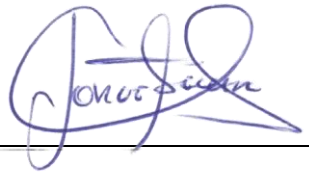
C.I: 0603576778

CERTIFICADO DE LOS MIEMBROS DEL TRIBUNAL

Quienes suscribimos, catedráticos designados Miembros del Tribunal de Grado para la evaluación del trabajo de investigación Desarrollo de prototipo móvil-web para gestionar disponibilidad y alertar intrusiones mediante autenticación facial en parqueaderos Cooperativa "Señor del Buen Suceso", presentado por John Jairo Guaman Pineda, con cédula de identidad número 0605034933, bajo la tutoría de PhD. Miryan Estela Narváez Vilema; certificamos que recomendamos la APROBACIÓN de este con fines de titulación. Previamente se ha evaluado el trabajo de investigación y escuchada la sustentación por parte de su autor; no teniendo más nada que observar.

De conformidad a la normativa aplicable firmamos, en Riobamba el 9 de junio del 2026.

Mgs. Jorge Delgado
PRESIDENTE DEL TRIBUNAL DE GRADO



Mgs. Lady Espinoza
MIEMBRO DEL TRIBUNAL DE GRADO

Mgs. Danny Velasco
MIEMBRO DEL TRIBUNAL DE GRADO

C E R T I F I C A C I Ó N

Que, **JOHN JAIRO GUAMAN PINEDA** con CC: **0605034933**, estudiante de la Carrera **INGENIERÍA EN TECNOLOGÍAS DE LA INFORMACIÓN**, Facultad de **INGENIERIA**; ha trabajado bajo mi tutoría el trabajo de investigación titulado "**DESARROLLO DE PROTOTIPO MÓVIL-WEB PARA GESTIONAR DISPONIBILIDAD Y ALERTAR INTRUSIONES MEDIANTE AUTENTICACIÓN FACIAL EN PARQUEADEROS COOPERATIVA SEÑOR DEL BUEN SUCESO**", cumple con el **9 %,** de acuerdo al reporte del sistema Anti plagio **COMPILATIO**, porcentaje aceptado de acuerdo a la reglamentación institucional, por consiguiente autorizo continuar con el proceso.

Riobamba, 27 de abril de 2026

Validar únicamente en FirmaEC.
Firmado electrónicamente por:
MIRYAN ESTELA
NARVAEZ VILEMA

Miryan Narváez
TUTORA

DEDICATORIA

A mis padres Galo Elías Guaman Ramírez y Mirian Magali Pineda Villarroel, quienes con su amor incondicional y sacrificio silencioso sembraron en mí la semilla del esfuerzo y la perseverancia desde el día que nací. Cada paso de este camino lleva su huella. A mi abuela, cuya sabiduría y ternura han sido refugio en los momentos más difíciles. Este logro es también el fruto de sus oraciones y de su fe inquebrantable en mí. A mi familia, por ser el pilar que sostiene mis sueños y el hogar al que siempre regreso.

John Guaman

AGRADECIMIENTO

El presente trabajo no habría sido posible sin el apoyo de quienes me acompañaron a lo largo de este proceso.

A mis padres, por su entrega sin condiciones, por cada sacrificio que hicieron para que yo pudiera llegar hasta aquí y por enseñarme que la constancia es la forma más noble de amor.

A mis abuelos, por sus palabras de aliento en los momentos de duda y por recordarme siempre de lo que soy capaz.

A mi familia, por su comprensión, su paciencia y su presencia en cada etapa de este camino.

A mis docentes y asesores, por compartir su conocimiento y orientar este trabajo con profesionalismo y dedicación, especialmente a mi tutora de tesis Miryan Estela Narváez Vilema que me supo entender y apoyar en todo este camino y no solo de la tesis sino todo este ciclo académico.

A mis amigos, en especial a Wilson Tierra, por los momentos compartidos, por el ánimo brindado en los días más exigentes y por demostrarme que la amistad verdadera también es un sostén en los grandes retos de la vida.

A todos quienes, de una u otra manera, contribuyeron a hacer posible este logro: mi más sincero agradecimiento.

John Guaman

ÍNDICE GENERAL

DECLARATORIA DE AUTORÍA

DICTAMEN FAVORABLE DEL PROFESOR TUTOR

CERTIFICADO DE LOS MIEMBROS DEL TRIBUNAL

CERTIFICADO ANTIPLAGIO

DEDICATORIA

AGRADECIMIENTO

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

RESUMEN

ABSTRACT

CAPÍTULO I. INTRODUCCIÓN 14

1.1 Planteamiento del problema 14

1.2 Justificación 15

1.3 Formulación del problema 15

1.4 Objetivos 16

Objetivo General 16

Objetivos Específicos 16

CAPÍTULO II. MARCO TEÓRICO 17

2.1 Sistemas inteligentes de gestión de parqueaderos 17

2.1.1 Evolución y fundamentos de los sistemas de estacionamiento inteligente (SEI) .. 17

2.1.2 Tecnologías utilizadas en la gestión de parqueaderos 17

2.1.3 Beneficios y limitaciones de los SEI actuales 17

2.2 Visión por computadora en la detección vehicular 19

2.2.1 Principios de la visión por computadora aplicada a entornos vehiculares 19

2.2.2 Procesamiento de imágenes y detección de objetos con OpenCV y YOLOv8 .. 19

2.2.3 Reconocimiento facial como método de autenticación biométrica 20

2.3 Framework Flutter para el desarrollo móvil-web 20

2.3.1 Arquitectura y ventajas de Flutter frente a otros frameworks 20

2.3.2 Integración de Flutter con servicios backend y APIs para monitoreo en tiempo real 21
2.4 Fiabilidad, evaluación e impacto del Prototipo 21
2.4.1 Evaluación del software mediante la norma ISO/IEC 25010:2023 21
2.4.2 Pruebas de ausencia a fallos, disponibilidad y tolerancia a fallos 22
2.5 Metodología de desarrollo kanban 23
CAPÍTULO III. METODOLOGÍA 25
3.1 Diseño de la investigación 25
3.2 Tipo de investigación 25
3.3 Población de estudio y tamaño muestra 25
3.4 Técnicas de recolección de datos 26
3.5 Identificación de variables 26
3.5.1 Variable dependiente 26
3.5.2 Variable independiente 26
3.6 Operacionalización de variables 26
3.7 Metodología de desarrollo 28
3.7.1 Fase 1: Análisis 28
3.7.2 Fase 2: Diseño 31
3.7.3 Fase 3: Implementación 38
3.7.4 Fase 4: Pruebas y evaluación del sistema 45
CAPÍTULO IV. RESULTADOS Y DISCUSIÓN 48
4.1 Resultados 48
4.2 Discusión 54
CAPÍTULO V. CONCLUSIONES Y RECOMENDACIONES 56
5.1 Conclusiones 56
5.2 Recomendaciones 56
BIBLIOGRÁFIA 58
ANEXOS 60

ÍNDICE DE TABLAS

Tabla 1. Comparativa de tecnologías de los Sistemas de Estacionamiento 18
Tabla 2. Ventajas y limitaciones de la metodología Kanban 23
Tabla 3: Operacionalización de Variables 27
Tabla 4: Requerimientos funcionales 29
Tabla 5: Requerimientos no funcionales 29
Tabla 6: Historias de usuario 30
Tabla 7: Tabla Authorized Plates 34
Tabla 8: Tabla Notification Settings 34
Tabla 9: Tabla Parking Slots 34
Tabla 10: Tabla Users 34
Tabla 11: Tabla Users Alerts 35
Tabla 12: Tabla Users Faces 35
Tabla 13: Users Tokens 35
Tabla 14: Comprobación de completitud funcional 48
Tabla 15: Prueba de madurez 50
Tabla 16: Prueba de tolerancia a fallos 51
Tabla 17: Prueba de disponibilidad 53

ÍNDICE DE FIGURAS

Figura 1: Revisión de algoritmos de detección y seguimiento de objetos 20
Figura 2: Modelo ISO/IEC 25010:2023 22
Figura 3: Tablero Kanban 28
Figura 4: Tablero Kanban Fase 1 31
Figura 5: Diagrama de casos de uso 31
Figura 6: Diagrama de actividades 32
Figura 7: Diagrama de despliegue o arquitectura 32
Figura 8: Diagrama de base de datos 33
Figura 9: Script de la base de datos 33
Figura 10: Diagrama de interfaces del sistema 37
Figura 11: Tablero Kanban Fase 2 38
Figura 12: Patrón de diseño 39
Figura 13: Conexión con la cámara IP 41
Figura 14: Modulo de reconocimiento facial con OpenCV 42
Figura 15: Módulo de detección de placas vehiculares 42
Figura 16: Módulo de navegación y visualización web 43
Figura 17: Módulo de aplicación móvil 43
Figura 18: App móvil 44
Figura 19: Página web - detección de placas 44
Figura 20: Página web - detección de rostros 45
Figura 21: Tablero Kanban Fase 3 45
Figura 22: Prueba de Madurez 49
Figura 23: Prueba de disponibilidad 50
Figura 24: Prueba de tolerancia a fallos 51
Figura 25: Recursos utilizados 52
Figura 26: Tablero Kanban fase 4 52
Figura 27: Grafica de madurez en ausencia a fallos 53
Figura 28: Grafica de tolerancia a fallos 54

RESUMEN

Se desarrolló una aplicación móvil-web para gestionar la disponibilidad de parqueaderos en tiempo real y alertar intrusiones mediante autenticación facial en los parqueaderos de la Cooperativa de Taxis “Señor del Buen Suceso” de la ciudad de Riobamba. Para ello, se investigaron modelos de reconocimiento facial, tecnologías móviles-web y una metodología de gestión adecuada para el desarrollo del prototipo, estructurando el trabajo por fases bajo un enfoque Kanban.

Se implementó un prototipo que integró el registro y control de ocupación de espacios, el proceso de autenticación facial para verificar la identidad del conductor asociado al vehículo y un mecanismo de alertas automáticas ante accesos no autorizados. La solución permitió visualizar el estado de disponibilidad y generar notificaciones cuando la verificación facial no coincidió con el conductor registrado, aportando a un control más seguro del retiro del vehículo y a una mejor organización operativa del parqueadero.

Finalmente, se evaluó la fiabilidad del prototipo mediante criterios basados en la norma ISO/IEC 25010:2023, considerando características relacionadas con disponibilidad y tolerancia a fallos. En conjunto, se concluyó que la aplicación propuesta aportó a la reducción de vulnerabilidades por procesos manuales, fortaleció la seguridad frente a intrusiones y mejoró la eficiencia en la gestión del parqueo dentro de la cooperativa, demostrando su factibilidad como solución tecnológica replicable en entornos similares.

Palabras claves: autenticación facial, disponibilidad en tiempo real, intrusión, parqueaderos, seguridad.

ABSTRACT

A mobile-web application was developed to manage parking availability in real time and alert users to intrusions using facial authentication at the parking lots of the "Señor del Buen Suceso" Taxi Cooperative in the city of Riobamba. This involved researching facial recognition models, mobile-web technologies, and a suitable management methodology for prototype development, structuring the work in phases using a Kanban approach.

A prototype was implemented that integrated the registration and occupancy control of spaces, the facial authentication process to verify the identity of the driver associated with the vehicle, and an automatic alert mechanism for unauthorized access. The solution allowed users to view the availability status and generate notifications when the facial verification did not match the registered driver, contributing to safer vehicle retrieval and improved operational organization of the parking lot.

Finally, the reliability of the prototype was evaluated using criteria based on the ISO/IEC 25010:2023 standard, considering characteristics related to availability and fault tolerance. Overall, it was concluded that the proposed application contributed to reducing vulnerabilities in manual processes, strengthened security against intrusions, and improved the efficiency of parking management within the cooperative, demonstrating its feasibility as a replicable technological solution in similar environments.

Keywords: facial authentication, real-time availability, intrusion, parking, security.

Firmado electrónicamente por:
JESSICA MARIA GUARANGA LEMA

Validar únicamente con FirmaEC

Reviewed by:
Mgs. Jessica María Guaranga Lema
ENGLISH PROFESSOR
C.C. 0606012607

CAPÍTULO I. INTRODUCCIÓN

El crecimiento acelerado de vehículos en la ciudad de Riobamba ha evidenciado deficiencias importantes en la gestión del estacionamiento vehicular, especialmente en las instalaciones de las cooperativas de transporte, generando problemas de congestión, ocupación indebida y riesgos relacionados con la seguridad vehicular. En particular, la Cooperativa de Taxis "Señor Buen Suceso", ubicada en un sector céntrico y de alta circulación, enfrenta diariamente dificultades para organizar sus espacios de parqueo y garantizar la seguridad de los vehículos estacionados, siendo frecuentes casos de intrusión o intentos de robo.

Actualmente, los procesos en esta cooperativa dependen completamente de procedimientos manuales o de observación humana, incrementando el margen de error y dejando vulnerabilidades significativas en la seguridad. Al mismo tiempo, el contexto tecnológico actual presenta una oportunidad clara para la implementación de sistemas inteligentes basados en reconocimiento facial y plataformas digitales integradas a dispositivos móviles. El país posee una alta penetración de teléfonos inteligentes e infraestructura adecuada para desplegar soluciones en la nube, permite proponer un sistema automático, escalable y confiable.

Ante esta realidad, surge la necesidad de implementar un prototipo móvil-web que gestione automáticamente la disponibilidad de parqueaderos en tiempo real y utilice autenticación facial para verificar que la persona que retire el vehículo sea la misma que lo estacionó originalmente, generando alertas inmediatas ante posibles intrusiones. Esta propuesta, además de aportar a la seguridad y eficiencia operativa, contribuirá a mejorar la gestión urbana, facilitando datos estadísticos útiles para futuras decisiones estratégicas de la cooperativa y las autoridades locales.

1.1 Planteamiento del problema

Según los datos registrados en la Dirección de Gestión de Movilidad Tránsito y Transporte del GAD Municipal de Riobamba, se evidencia un crecimiento en la cantidad de vehículos hasta rozar los 37342 vehículos registrados [1], cifra que casi se duplicó en la última década. La oferta formal de estacionamiento intenta compensar ese aumento con el SEROT, Sistema de Estacionamiento Rotativo Ordenado Tarifado que hoy delimita 171 calles azules en el centro urbano, aun así, la demanda diaria desborda la capacidad del sistema y obliga a los conductores a deambular o detenerse en doble fila, prácticas que agravan la congestión y generan sanciones.

Para los socios de la Cooperativa de Taxis "Señor del Buen Suceso" el problema se traduce en minutos improductivos, gasto extra de combustible y un servicio menos competitivo frente a plataformas de viaje que ya muestran niveles de eficiencia superiores. Esta brecha define una demanda insatisfecha de estacionamiento gestionado en tiempo real dentro de un entorno digitalmente preparado. ¿El desarrollo de un prototipo móvil-web con

reconocimiento facial mejorará el control de acceso y la seguridad en el parqueadero de la Cooperativa de Taxis "Señor del Buen Suceso"?

Responder a esta interrogante permitirá intervenir dos procesos clave en la gestión de parqueaderos: la optimización del tiempo de búsqueda de plazas mediante monitoreo automático en tiempo real, y la reducción efectiva de incidentes de seguridad al verificar facialmente la identidad del conductor que retira cada vehículo, emitiendo alertas inmediatas en caso de detección de intrusiones.

1.2 Justificación

Este trabajo de investigación surge de la necesidad de mejorar la gestión de parqueaderos y aumentar la seguridad vehicular en la Cooperativa de Taxis “Señor del Buen Suceso” en Riobamba. El crecimiento constante del número de vehículos como se menciona en el planteamiento del problema ha superado la capacidad de control y supervisión manual de los espacios disponibles. Actualmente, los métodos de estacionamiento y vigilancia en la cooperativa dependen en gran medida de la observación humana, incrementando el margen de error y deja expuestos a robos, intrusiones o uso no autorizado de los espacios.

En este contexto, desarrollar un prototipo móvil-web que integre tecnologías de reconocimiento facial y modelos entrenados mediante YoloV8 permitió automatizar el proceso de autenticación de conductores y el control de disponibilidad de espacios en tiempo real. Este tipo de soluciones ha demostrado ser efectivo en diversos entornos, ayudando a reducir incidentes de seguridad y optimizar recursos logísticos al combinar visión computacional y monitoreo continuo.

Además, la aplicación propuesta se alinea con los objetivos de modernización tecnológica y transformación digital en la gestión del transporte urbano, donde se busca que las cooperativas adopten sistemas inteligentes que mejoren su eficiencia operativa y competitividad. Su implementación traerá beneficios concretos: mejor administración del parqueadero, reducción del tiempo perdido buscando espacios, mayor confianza de los socios y generación de alertas automáticas ante accesos no autorizados.

1.3 Formulación del problema

¿De qué manera el desarrollo de un prototipo móvil-web que gestione la disponibilidad de parqueaderos en tiempo real y emita alertas ante intrusiones mediante autenticación facial, mejorará la seguridad y la eficiencia operativa en la Cooperativa de Taxis “Señor del Buen Suceso” de la ciudad de Riobamba?

1.4 Objetivos

Objetivo General

Desarrollar un prototipo móvil-web para gestionar disponibilidad y alertar intrusiones mediante autenticación facial en parqueaderos Cooperativa "Señor del Buen Suceso".

Objetivos Específicos

- Investigar tecnologías móviles-web y modelos de reconocimiento facial para disponibilidad y alertas de intrusiones.
- Implementar un prototipo móvil-web que integre disponibilidad en tiempo real y reconocimiento facial para autenticación del conductor con sistema automático de alertas frente a accesos no autorizados.
- Evaluar la fiabilidad del prototipo móvil-web, utilizando la norma ISO/IEC 25010:2023.

CAPÍTULO II. MARCO TEÓRICO

2.1 Sistemas inteligentes de gestión de parqueaderos

Los sistemas fueron creados como soluciones tecnológicas para optimizar la administración de los espacios vehiculares dentro de entornos públicos y privados, a través del uso de sensores, cámaras, redes y plataformas digitales. Su función principal fue mejorar la eficiencia operativa, fortalecer la seguridad en parqueaderos y reducir el tiempo de búsqueda de un lugar de estacionamiento.

Estos sistemas fueron parte de las ciudades inteligentes, al integrar IoT, además de análisis de datos y herramientas de automatización que ayudaron a gestionar de manera dinámica la disponibilidad y el acceso vehicular. De esta forma, el estacionamiento dejo de realizarse como un proceso manual para transformarse en un sistema interactivo.

2.1.1 Evolución y fundamentos de los sistemas de estacionamiento inteligente (SEI)

En los últimos años los sistemas de estacionamiento inteligente evolucionaron desde métodos convencionales de control manual hacia plataformas digitales que permitieron gestionar la disponibilidad de plazas en tiempo real. Recientemente, se documentó que estos sistemas ofrecieron información al conductor, optimizaron el uso de espacios y facilitaron el pago automático, representando un avance frente a los sistemas tradicionales [2].

Los sistemas de estacionamiento inteligente surgieron con la necesidad de mejorar la movilidad y reducir el tiempo en encontrar estacionamiento, lo cual también ayudo a reducir la congestión vehicular. También fue impulsado debido a la tendencia de las ciudades inteligentes. Estos sistemas acogieron un enfoque integral que abarcó la gestión del espacio físico de aparcamiento como la interacción digital con el conductor.

2.1.2 Tecnologías utilizadas en la gestión de parqueaderos

Durante el desarrollo reciente de los sistemas de estacionamiento inteligente se emplearon diversas tecnologías para detección de ocupación, comunicación y análisis. Respecto a la disponibilidad de parqueaderos, de utilizaron sensores, infrarrojos, cámaras con Computer Visión y redes de sensores.

Por parte de la comunicación y procesamiento, se utilizó la arquitectura del IoT, redes inalámbricas y plataformas en la nube para procesar datos en tiempo real. La capa de interfaz implemento aplicación móvil y web, sistema de pago digital, reservación de espacios y en algunos casos algoritmos de predicción basados en IA para estimar la demanda en la disponibilidad de la misma.

2.1.3 Beneficios y limitaciones de los SEI actuales

Los SEI trajeron múltiples beneficios como la reducción del tiempo en que un conductor busca un estacionamiento disponible, optimizaron la gestión de espacios, mejoraron la experiencia del conductor y contribuyeron datos útiles para la toma de decisiones. A demás permitieron monetizar de una mejor forma los espacios de aparcamiento

y ayudaron a reducir emisiones vehiculares asociadas a la búsqueda de un lugar de estacionamiento.

Sin embargo, los sistemas afrontaron ciertas limitaciones en el transcurso como por ejemplo la instalación de cámaras y sensores debido al costo elevado, algunas tecnologías presentaron problemas de escalabilidad en el momento de producción, y la dependencia de conectividad e infraestructura represento un reto, especialmente en espacios que no fueron diseñados para este tipo de sistema. Por otro lado, temas como la seguridad de datos, la privacidad del usuario y la interoperabilidad entre sistemas se reflejaron como barreras para una adopción mucho más amplia y efectiva en casos reales.

Mediante el análisis de los beneficios y limitaciones descritos, resulto acertado sintetizar las principales tecnologías aplicadas en los sistemas de estacionamiento inteligente, junto a sus ventajas operativas y restricciones técnicas. Esta recopilación permitió identificar los componentes más relevantes para el desarrollo de un prototipo móvil-web adaptada a la gestión de parqueaderos con autenticación facial y disponibilidad de plazas, como se aprecia en la Tabla 1, donde se presenta una comparativa entre las tecnologías más utilizadas, sus aportes y desafíos dentro de este tipo de sistemas.

Tabla 1. Comparativa de tecnologías de los Sistemas de Estacionamiento

Tecnología principal	Aplicación en SEI	Beneficios	Limitaciones
Sensores ultrasónicos	Detección individual de vehículos en cada plaza de estacionamiento.	Alta precisión en detección de ocupación; fácil integración con microcontroladores.	Alto costo de instalación y mantenimiento; sensibilidad a condiciones ambientales.
Visión por Computadora (Cámaras + IA)	Reconocimiento de vehículos, matrículas y detección de intrusiones.	Permite análisis avanzado, autenticación facial y seguridad visual en tiempo real.	Requiere procesamiento elevado y conectividad estable; riesgo de invasión de privacidad.
Redes IoT	Transmisión de datos entre sensores y servidores de gestión.	Comunicación inalámbrica eficiente y adaptable; bajo consumo energético.	Requiere procesamiento elevado y conectividad estable; riesgo de invasión de privacidad.
Plataformas en la nube	Procesamiento y almacenamiento de información de disponibilidad y alertas.	Escalabilidad, disponibilidad remota y capacidad analítica avanzada.	Dependencia de conectividad; costos de servicio y posibles vulnerabilidades.

Aplicaciones móviles y web	Interfaz para usuarios y administradores de parqueaderos.	Facilitan interacción, reservas y pagos; permiten alertas en tiempo real.	Requieren diseño adaptable y ciberseguridad robusta; riesgo de saturación del sistema.
Algoritmos de IA	Predicción de demanda, patrones de ocupación y anomalías.	Aumentan eficiencia operativa y precisión en el control de flujos.	Necesitan grandes volúmenes de datos para entrenamiento; posibles sesgos algorítmicos.

2.2 Visión por computadora en la detección vehicular

Computer Vision se transformó en un componente importante y necesario para los sistemas inteligentes de gestión de parqueaderos al permitir que las cámaras, sensores y algoritmos puedan procesar imágenes en tiempo real para detectar, contar y clasificar vehículos, placas y demás datos necesarios dependiendo el contexto. Esta disciplina combinó técnicas de procesamiento de imágenes, aprendizaje autónomo y análisis de video para automatizar tareas que antes era necesario operadores humanos, contribuyendo así a una mejor eficiencia y seguridad operativa. Se implementó mediante librerías como OpenCv y modelos como YOLOv8, además de la aplicación del reconocimiento facial como método biométrico de autenticación son opciones open source y fáciles de entrenar para este tipo de casos.

2.2.1 Principios de la visión por computadora aplicada a entornos vehiculares

Computer Vision o visión por computadora aplicada al entorno vehicular se basó en el análisis de secuencias de vídeo o imágenes para obtener información relevante sobre los vehículos, como su propia presencia, tipo, dirección o posición. Esta disciplina usó técnicas como la segmentación de objetos, el reconocimiento de patrones y el seguimiento de trayectorias para generar datos útiles para la gestión de parqueaderos. Los algoritmos desarrollados permitieron identificar cuándo una plaza se desocupa o se ocupaba únicamente usando cámaras, y alertar sobre uso indebido o accesos no autorizados, transformando la operación manual en un proceso automatizado mediante modelos pre-entrenados.

2.2.2 Procesamiento de imágenes y detección de objetos con OpenCV y YOLOv8

Se implementaron librerías como OpenCV para la manipulación y pre-procesamiento de imágenes (ajuste de contraste, detección de bordes, extracción de contornos), mientras que modelos de detección de objetos más avanzados como YOLOv8 permitieron localizar y clasificar vehículos en tiempo real con alta precisión además de reconocer los caracteres de las placas vehiculares. Estudios recientes demostraron que YOLOv8, combinado con OpenCV, alcanzó resultados adecuados para entornos urbanos, incluso en condiciones dinámicas de iluminación y tráfico [3]. Esta integración ayudó a la identificación de vehículos, la estimación de velocidad o conteo y la reducción de la intervención manual, lo cual es directamente aplicable al caso de la cooperativa que gestiona plazas de estacionamiento.

A fin de entender de forma visual el proceso interno de una red convolucional empleada en la detección de objetos, se ilustra el flujo general que sigue una imagen desde su entrada hasta la etapa de clasificación. En este esquema se observa cómo los filtros aplican operaciones de convolución y submuestreo para extraer características relevantes que posteriormente son procesadas por capas de clasificación. En la Figura 1, se muestra la estructura básica de una red neuronal convolucional utilizada en los procesos de reconocimiento y detección de imágenes.

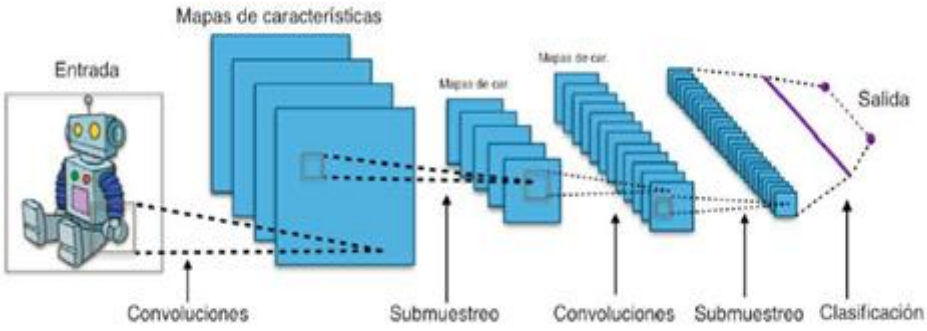


Figura 1: Revisión de algoritmos de detección y seguimiento de objetos
Fuente: [4]

2.2.3 Reconocimiento facial como método de autenticación biométrica

El reconocimiento facial constituyó un método biométrico de autenticación que sirve para verificar la identidad del individuo que solicita el acceso o retiro de un vehículo, con el objetivo de prevenir accesos no autorizados. Gracias al uso de redes neuronales profundas y técnicas de aprendizaje automático, los sistemas de reconocimiento facial modernizados alcanzaron mejoras significativas en cuanto a su exactitud, aunque persisten retos importantes en materia de sesgos, privacidad y robustez frente a ataques y sin olvidar el tema del margen de error [5]. Esta tecnología permitió establecer que sólo el conductor autenticado previamente podría liberar el vehículo sin generar alertas, integrando así gestión de disponibilidad con seguridad biométrica.

2.3 Framework Flutter para el desarrollo móvil-web

El framework se presentó como una solución moderna que permitió desarrollar aplicaciones móviles y web desde una única base de código, facilitando la convergencia de plataformas y redujo considerablemente el esfuerzo de desarrollo [6]. Con su arquitectura orientada a widgets y un motor de renderizado propio, Flutter brinda una experiencia de usuario fluida, con alto rendimiento, diseño consistente en múltiples dispositivos y además compatible con multiplataformas siendo excelente para poder escalar en proyectos a futuro.

2.3.1 Arquitectura y ventajas de Flutter frente a otros frameworks

La arquitectura de Flutter se compuso de varios módulos que proveyeron los componentes necesarios para el desarrollo de una aplicación. Esta arquitectura se distinguió por elementos clave como el motor gráfico 2D Skia [7], optimizado para ofrecer un alto rendimiento, y un núcleo de framework que gestionaba los ciclos de vida de la aplicación y permitía un desarrollo basado en estados.

Hablando de sus ventajas, se destacan varias importantes entre ellas: el desarrollo ágil y rápido, gracias a la funcionalidad de hot-reload, que permitía a los programadores observar los cambios en el código al instante. Se mencionó que con Flutter se lograba una calidad en el producto final comparable a la de las aplicaciones nativas, maximizando el rendimiento del dispositivo. Además, se reportaron tiempos de carga inferiores a un segundo tanto en iOS como en Android.

En una comparativa con otros frameworks como React Native e Ionic, el desempeño de Flutter fue calificado como "Excelente", superando al de React Native ("Bueno") y al de Ionic ("Medio") [7].

2.3.2 Integración de Flutter con servicios backend y APIs para monitoreo en tiempo real

La integración de Flutter con servicios backend y APIs facilitó la creación de módulos que gestionaron datos en tiempo real, suscripciones a cambios y actualización dinámica de interfaces mediante mecanismos como WebSockets, Streams o bases de datos en tiempo real [8]. Flutter puede servir para interactuar con un servidor REST y un sistema de mensajería en tiempo real que notifica al usuario sobre cambios en la disponibilidad de plazas o alertas de intrusión, garantizando una notificación inmediata sin necesidad de refrescar manualmente la interfaz. Además, la arquitectura permite separar la lógica de interfaz, la lógica de negocio y la comunicación con el backend, mejorando la mantenibilidad y escalabilidad del sistema.

2.4 Fiabilidad, evaluación e impacto del Prototipo

La fiabilidad del prototipo se abordó como uno de los factores esenciales para garantizar que la aplicación móvil-web gestionara disponibilidad de plazas y autenticación facial sin fallos críticos. Se implementó un enfoque estructurado de evaluación del software, donde la norma ISO/IEC 25010:2023 sirvió como marco de referencia para definir características de calidad tales como fiabilidad, usabilidad y seguridad [9]. La norma ISO/IEC 25010:2023 definió el modelo internacional de calidad para productos y sistemas de software, estableciendo un conjunto de características y sub características que permitieron evaluar su fiabilidad, usabilidad, seguridad, mantenibilidad y eficiencia en el contexto de su funcionamiento y uso previsto [10].

2.4.1 Evaluación del software mediante la norma ISO/IEC 25010:2023

La evaluación del software se basó en el modelo de calidad que propone la norma ISO/IEC 25010:2023. Esta norma no solo actualiza versiones anteriores, sino que también añade nuevas dimensiones, como la seguridad y la protección de datos, adaptándose a los entornos de software modernos [9].

La aplicación de este marco metodológico permitió analizar la calidad del sistema, considerando tanto los atributos internos del software (como la estabilidad del código, la eficiencia algorítmica y modularidad) como los externos (tales como la experiencia del usuario, la accesibilidad y la seguridad).

En la Figura 2, se presenta un esquema simplificado que representa la estructura general del modelo ISO/IEC 25010:2023, compuesto por sus nueve características principales y sus relaciones jerárquicas.

CALIDAD DEL PRODUCTO SOFTWARE								
ADECUACIÓN FUNCIONAL	EFICIENCIA DE DESEMPEÑO	COMPATIBILIDAD	CAPACIDAD DE INTERACCIÓN	FIABILIDAD	SEGURIDAD	MANTENIBILIDAD	FLEXIBILIDAD	PROTECCIÓN
COMPLETITUD FUNCIONAL CORRECCIÓN FUNCIONAL PERTINENCIA FUNCIONAL	COMPORTAMIENTO TEMPORAL UTILIZACIÓN DE RECURSOS CAPACIDAD	COEXISTENCIA INTEROPERABILIDAD	RECONOCIBILIDAD DE ADECUACIÓN APRENDIZABILIDAD OPERABILIDAD PROTECCIÓN FRENTE A ERRORES DE USUARIO INVOLUCRACIÓN DEL USUARIO INCLUSIVIDAD ASISTENCIA AL USUARIO AUTO-DESCRIPTIVIDAD	AUSENCIA DE FALLOS DISPONIBILIDAD TOLERANCIA A FALLOS RECUPERABILIDAD	CONFIDENCIALIDAD INTEGRIDAD NO-REPUDIO RESPONSABILIDAD AUTENTICIDAD RESISTENCIA	MODULARIDAD REUSABILIDAD ANALIZABILIDAD CAPACIDAD DE SER MODIFICADO CAPACIDAD DE SER PROBADO	ADAPTABILIDAD ESCALABILIDAD INSTALABILIDAD REEMPLAZABILIDAD	RESTRICCIÓN OPERATIVA IDENTIFICACIÓN DE RIESGOS PROTECCIÓN ANTE FALLOS ADVERTENCIA DE PELIGRO INTEGRACIÓN SEGURA

Figura 2: Modelo ISO/IEC 25010:2023
Fuente: [11]

En síntesis, la aplicación del modelo de calidad propuesto por la ISO/IEC 25010:2023 permitió establecer una metodología estructurada de evaluación que abarcó todo el ciclo de vida del software.

2.4.2 Pruebas de ausencia a fallos, disponibilidad y tolerancia a fallos

Las pruebas de ausencia de fallos intentan asegurarse de que el sistema funcione sin errores durante la ejecución normal de sus procesos y que responda de manera adecuada ante condiciones de operación continuas. Este tipo de verificación es fundamental para validar la calidad interna del software, ya que un sistema confiable debe garantizar que sus módulos operen correctamente sin interrupciones inesperadas ni pérdida de datos como lo dicta la norma. Esta aplicación incluyó la revisión estructural del código, la ejecución de casos de prueba unitarios y la simulación de escenarios críticos con el fin de detectar comportamientos anómalos y asegurar la integridad de los resultados [12].

Paralelamente, las pruebas de disponibilidad miden la proporción de tiempo en que el sistema permanece operativo frente al total del periodo de observación. La disponibilidad operativa (A) se expresó mediante la relación entre el Tiempo Medio Entre Fallos (MTBF) y la suma de este con el Tiempo Medio para la Recuperación (MTTR) por el 100% del tiempo, de acuerdo con el modelo clásico de confiabilidad [13]:

$$DISPONIBILIDAD = \frac{MTBF}{MTBF + MTTR} \times 100$$

Este indicador permite identificar las ventanas de mantenimiento, tiempos de recuperación y periodos de inactividad, dando una visión cuantitativa de la estabilidad del sistema en escenarios reales de funcionamiento.

La tolerancia a fallos, por su parte, se evalúa a través de la capacidad del sistema para mantener sus funciones críticas operativas cuando ocurrían errores o interrupciones en uno o más componentes [14]. Durante la fase de pruebas, se simulan fallos en módulos de conexión, sensores de video y procesos de autenticación para observar si el sistema es capaz de recuperarse automáticamente, generar alertas y continuar con la ejecución del resto de sus servicios sin intervención manual. Los resultados demostraron que la arquitectura distribuida y el manejo de excepciones en tiempo real fueron determinantes para lograr la continuidad del servicio.

2.5 Metodología de desarrollo kanban

Es una de las herramientas más utilizadas dentro de la gestión ágil de proyectos de software, debido a su enfoque visual y adaptable para controlar el flujo de trabajo y mejorar la eficiencia del equipo. Fundamentada en principios Lean, Kanban se realiza mediante tableros visuales que permite comprobar el progreso del trabajo en etapas definidas como "Backlog", "En progreso" y "Listo" [15].

Esta metodología destacó por su simplicidad y adaptabilidad, porque permitió gestionar proyectos de diferente escala sin imponer roles estrictos o calendarios cerrados. En vez de basarse en ciclos temporales como el modelo Scrum, Kanban priorizó el flujo continuo de valor, teniendo así un equilibrio entre la demanda de trabajo y la capacidad del equipo.

A diferencia de otras metodologías ágiles más estructuradas, Kanban no exigió una transformación organizacional completa, sino que se integró progresivamente sobre los procesos existentes. Su enfoque visual y su énfasis en la colaboración lo convirtieron en una herramienta ampliamente utilizada en equipos de desarrollo de software, diseño y mantenimiento de sistemas informáticos.

En la Tabla 2 se resume las principales ventajas y limitaciones que caracterizaron al enfoque Kanban en el ámbito de la ingeniería de software:

Tabla 2. Ventajas y limitaciones de la metodología Kanban

Aspecto	Descripción
Ventaja 1	Permite una visualización clara del flujo de trabajo y las prioridades.
Ventaja 2	Mejora la eficiencia al limitar el trabajo en curso (WIP) y reducir interrupciones.
Ventaja 3	Facilita la adaptación continua sin planificación rígida ni sprints definidos.
Ventaja 4	Promueve la comunicación y la colaboración entre los miembros del equipo.
Ventaja 5	Ofrece métricas precisas (Lead Time, Cycle Time) para decisiones basadas en datos.

Limitación 1	Requiere disciplina del equipo para mantener actualizado el tablero.
Limitación 2	No establece roles ni tiempos fijos, lo que puede generar ambigüedad en equipos nuevos.
Limitación 3	Puede volverse menos efectivo en proyectos grandes sin un marco complementario.

CAPÍTULO III. METODOLOGÍA

La metodología es lo más importante del proceso de investigación, porque ayuda a definir el tipo de estudio, el diseño metodológico, las técnicas para recolectar información y los métodos que se utilizó para analizar y procesar los datos. La metodología se enfocó en la construcción, implementación y evaluación de un prototipo móvil-web con autenticación facial y gestión automatizada de parqueaderos. El objetivo fue mejorar la seguridad y optimizar el uso de los espacios en la Cooperativa de Taxis “Señor del Buen Suceso”.

3.1 Diseño de la investigación

El diseño de investigación se centró en el desarrollo de un prototipo funcional. Se empleado un enfoque de investigación mixto, combinando elementos cuantitativos y cualitativos. Este enfoque es fundamental porque la problemática abarca tanto aspectos medibles de rendimiento tecnológico como percepciones y necesidades humanas. La parte cuantitativa se aplicó en la evaluación de la fiabilidad del sistema y el rendimiento técnico de los algoritmos de visión por computadora (YOLOv8) y reconocimiento facial (OpenCV) a través de métricas numéricas, así como la fiabilidad general del prototipo según la norma ISO/IEC 25010:2023.

Por otro lado, la parte cualitativa se enfocó en comprender la dinámica actual del parqueadero y las necesidades de la cooperativa mediante observación directa y entrevistas semiestructuradas, y en recabar feedback de los usuarios sobre la usabilidad y experiencia del prototipo durante las pruebas piloto.

3.2 Tipo de investigación

La investigación fue de tipo aplicada. Este enfoque busca la resolución de un problema práctico y específico en un contexto real, que en este caso es la optimización de la gestión de parqueaderos y el incremento de la seguridad vehicular en la Cooperativa de Taxis "Señor del Buen Suceso" en Riobamba. Se orienta a la creación e implementación de una solución tecnológica funcional (el prototipo móvil-web con autenticación facial y gestión de disponibilidad) que pueda ser utilizada de inmediato para abordar las deficiencias actuales. A través de este desarrollo, se buscó aplicar conocimientos teóricos y tecnológicos existentes (visión por computadora, reconocimiento facial, desarrollo móvil) para generar resultados concretos y medibles que beneficien directamente a la cooperativa y sus usuarios.

3.3 Población de estudio y tamaño muestra

La población y muestra de estudio son infinitas. Esto se debe a que el enfoque principal de este proyecto es el desarrollo y la evaluación técnica de un prototipo de software. Al tratarse de un sistema automatizado de gestión y seguridad de parqueaderos, las "observaciones" o "eventos" (como la detección de vehículos, el reconocimiento facial o los intentos de intrusión) son continuos y potencialmente ilimitados. La fiabilidad del prototipo se midió en función de su capacidad para operar bajo diversas condiciones y ante un flujo constante de datos, lo que trasciende la necesidad de definir un número fijo de sujetos o eventos.

3.4 Técnicas de recolección de datos

Para la recolección de datos, se utilizó las siguientes técnicas:

- Observación directa: Observación del funcionamiento actual del parqueadero para identificar problemas de congestión y seguridad.
- Entrevistas semiestructuradas: Entrevistas con el personal clave de la cooperativa (administradores, socios y conductores) para definir los requisitos funcionales y no funcionales.
- Recolección de datos visuales: Captura de videos e imágenes de los parqueaderos a través de cámaras para entrenar y probar los modelos de visión por computadora (YOLOv8 y reconocimiento facial).
- Herramienta de pruebas: Se utilizo JMeter para evaluar la ausencia de fallos, disponibilidad y tolerancia a fallos del sistema. A través de pruebas de carga y estrés.

3.5 Identificación de variables

3.5.1 Variable dependiente

Fiabilidad del prototipo móvil-web.

3.5.2 Variable independiente

Desarrollo de un prototipo móvil-web con reconocimiento facial y monitoreo de disponibilidad.

3.6 Operacionalización de variables

La Tabla 3, presenta la operacionalización de variables.

Tabla 3: Operacionalización de Variables

PROBLEMA	TEMA	OBJETIVOS	VARIABLES	CONCEPTUALZACION	DIMENSION	INDICADORES
¿La implementación de un prototipo móvil-web utilizando técnicas de reconocimiento facial mejorará la fiabilidad de disponibilidad y alertas de intrusiones en parqueaderos de la Cooperativa de Taxis "Señor del Buen Suceso"?	Desarrollo de prototipo móvil-web para gestionar disponibilidad y alertar intrusiones mediante autenticación facial en parqueaderos Cooperativa "Señor del Buen Suceso"	**GENERAL** Desarrollar un prototipo móvil-web para gestionar disponibilidad y alertar intrusiones mediante autenticación facial en parqueaderos Cooperativa "Señor del Buen Suceso".	**DEPENDIENTE** Fiabilidad del prototipo móvil-web.	La fiabilidad del prototipo se define como la capacidad del software de mantener un nivel de rendimiento especificado bajo condiciones dadas, durante un período de tiempo determinado (ISO/IEC 25010:2023). Es una medida crítica de la calidad que evalúa la ausencia de fallos, la disponibilidad, la tolerancia a fallos y la capacidad de recuperación.	Fiabilidad de los prototipos.	• Ausencia a fallos • Disponibilidad • Tolerancia a fallos
		ESPECÍFICOS • Investigar tecnologías móviles-web y modelos de reconocimiento facial para disponibilidad y alertas de intrusiones. • Implementar un prototipo móvil-web que integre disponibilidad en tiempo real y reconocimiento facial para autenticación del conductor con sistema automático de alertas frente a accesos no autorizados. • Evaluar la fiabilidad del prototipo móvil-web, utilizando la norma ISO/IEC 25010:2023.	**INDEPENDIENTE** Desarrollo de un prototipo móvil-web con reconocimiento facial y monitoreo de disponibilidad.	Se refiere al proceso de diseño, construcción e implementación del prototipo informático accesible vía web y móvil. Este desarrollo integra componentes de visión por computadora (YOLOv8, OpenCV) para la detección de vehículos y monitoreo de plazas, así como módulos de reconocimiento facial para autenticación y alerta de intrusiones. El proceso culmina en un sistema funcional que gestiona automáticamente los parqueaderos	Prototipo móvil-web.	• Funcionalidades desarrolladas. • Nivel de integración de módulos. • Uso de tecnologías específicas.

3.7 Metodología de desarrollo

El desarrollo del prototipo móvil-web se realizó utilizando la metodología ágil Kanban. Esta metodología permitió gestionar el flujo de trabajo de una manera visual, flexible y continua, asegurando que siempre existiera un progreso constante desde la idea inicial del prototipo hasta su implementación final. Fue la opción más acertada gracias a su capacidad de adaptarse a cambios en los requisitos funcionales, su enfoque en la eficiencia del proceso y su compromiso con la mejora continua en cada iteración.

Kanban se centra en la visualización de las tareas, la limitación del trabajo en curso y la entrega continua de valor [16]. Permite priorizar actividades relacionadas con la integración de módulos tecnológicos como OpenCV, YOLOv8, Flutter y la base de datos, manteniendo un flujo equilibrado entre diseño, codificación, pruebas y ajustes.

En la planificación general se definieron cuatro fases: Análisis, Diseño, Implementación y Pruebas. En la Figura 3 se observa la fase preliminar del tablero de Kanban con sus primeras actividades.

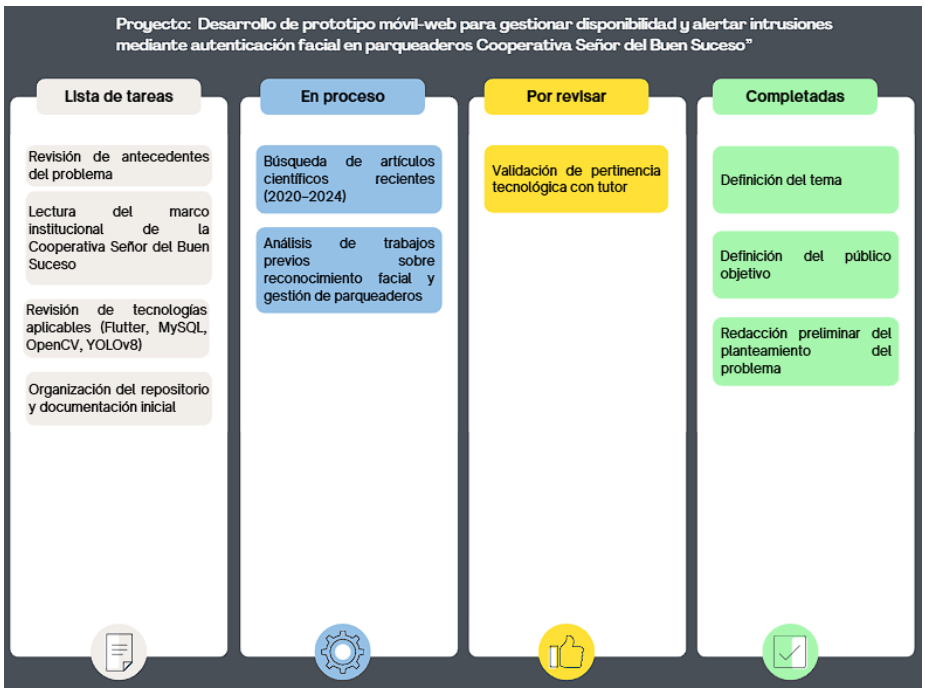


Figura 3: Tablero Kanban

3.7.1 Fase 1: Análisis

La fase de análisis se orientó a la identificación de los requerimientos funcionales y no funcionales del sistema, así como la comprensión de los procesos operativos que se desarrollaban en el parqueadero de la cooperativa. Esto resultó fundamental para garantizar que el prototipo respondiera de manera efectiva a las necesidades de los usuarios finales y a las condiciones tecnológicas.

Durante esta etapa, se aplicaron técnicas de levantamiento de información cualitativa y cuantitativa, entre las que destacaron:

- Entrevistas estructuradas al personal administrativo y a los conductores afiliados, con el objetivo de conocer las principales limitaciones del sistema actual de control manual y los requerimientos deseables para una solución automatizada.
- Observación directa del flujo de vehículos dentro del parqueadero, identificando tiempos de espera, procesos repetitivos y riesgos de accesos no autorizados.
- Análisis documental de registros internos de ingreso y salida de vehículos, reportes de incidentes y métodos internos de seguridad vehicular.

En las Tablas 4 y 5, se observan los requerimientos funcionales y requerimientos no funcionales del sistema.

Tabla 4: Requerimientos funcionales

ID	Descripción
RF01	El sistema debía registrar en tiempo real la disponibilidad de cada plaza de estacionamiento.
RF02	Se debía integrar un módulo de autenticación facial mediante librerías como OpenCV y el modelo YOLOv8, capaz de reconocer el rostro del conductor que realiza el retiro del vehículo.
RF03	El sistema debía generar **alertas automáticas** ante la detección de rostros no autorizados o intentos de intrusión.
RF04	Los registros de ocupación, accesos y alertas debían almacenarse en una base de datos centralizada y accesible desde el entorno web.
RF05	La aplicación debía permitir la consulta de disponibilidad desde la interfaz móvil y el control administrativo desde la interfaz web.
RF06	El sistema debía permitir la visualización del estado de vehículos (autorizado o no autorizado) en una interfaz.
RF07	El sistema debía poder permitir llenar con la información pre eliminar escogida de los socios de la cooperativa

Tabla 5: Requerimientos no funcionales

ID	Descripción
RNF01	El código fuente debe seguir principios de diseño simple y contener pruebas automatizadas para asegurar calidad continua.
RNF02	La interfaz del sistema debe ser intuitiva y permitir al usuario recibir alertas sin distracción.
RNF03	El sistema debía ofrecer seguridad y confidencialidad en el manejo de datos biométricos y credenciales de usuario.
RNF04	La arquitectura debía garantizar escalabilidad y mantenimiento, permitiendo la futura integración con cámaras IP adicionales.
RNF05	La comunicación entre módulos debía realizarse de forma eficiente mediante servicios web y una API REST segura.
RNF06	La interfaz debía ser intuitiva, adaptable y responsiva, desarrollada bajo el framework Flutter para garantizar compatibilidad entre plataformas.

Además, se definieron las historias de usuario necesarias para el desarrollo del prototipo, las cuales describen los requerimientos específicos y guían la implementación de las funcionalidades principales. Cada historia se formuló considerando los criterios de aceptación que garantizan su cumplimiento dentro del sistema.

La Tabla 6, presenta las historias de usuario establecidas para el proyecto:

Tabla 6: Historias de usuario

ID	Descripción	Criterios de aceptación
HU01	El sistema debía capturar en tiempo real las imágenes provenientes de las cámaras instaladas en los puntos del parqueadero.	La cámara debía activarse automáticamente al iniciar el sistema y sus módulos. Las transmisiones de video debían mantenerse estables con una tasa mínima de 15 fps en adelante. Si alguna cámara se desconectaba o no enviaba señal, el sistema debía registrar el evento y notificar al administrador.
HU02	El sistema debía detectar y localizar automáticamente el rostro del conductor dentro del área de monitoreo, utilizando modelos de detección y segmentación basados en YOLOv8 y OpenCV.	La detección debía ejecutarse en tiempo real con una precisión mínima del 75 %. El sistema debía trazar un contorno facial visible en la interfaz de supervisión. En caso de no identificar ningún rostro, debía mostrarse un mensaje de rostro no detectado.
HU03	El sistema debía comparar el rostro capturado con los registros existentes en la base de datos de la cooperativa para verificar la identidad del conductor.	La coincidencia debía superar un umbral del 85 % de similitud facial. El proceso de autenticación debía completarse en menos de diez segundos. Cada autenticación debía registrarse con fecha, hora y resultado (autorizado / no autorizado).
HU04	El sistema debía actualizar el estado de las plazas de estacionamiento de forma automática según la detección de ingreso o salida de vehículos.	El cambio de estado debía reflejarse en tiempo real en el mapa digital del parqueadero. Los espacios disponibles debían visualizarse en color verde y los ocupados en color rojo. Cada actualización debía registrarse en la base de datos con su marca de tiempo.
HU05	El sistema debía garantizar la sincronización entre la plataforma web y el módulo móvil, permitiendo la supervisión en tiempo real de la disponibilidad y las alertas generadas.	Las actualizaciones del sistema debían reflejarse instantáneamente en ambos entornos. Las interrupciones de conexión debían generar avisos automáticos y reintentos de sincronización.
HU06	El sistema debía asegurar la integridad y confidencialidad de los datos personales y biométricos registrados durante el proceso de autenticación.	Solo el personal autorizado debía acceder al módulo de gestión de usuarios y rostros. Se debía garantizar el cumplimiento de las políticas de protección de datos personales.

En la Figura 4, se muestra el tablero Kanban en su fase 1.

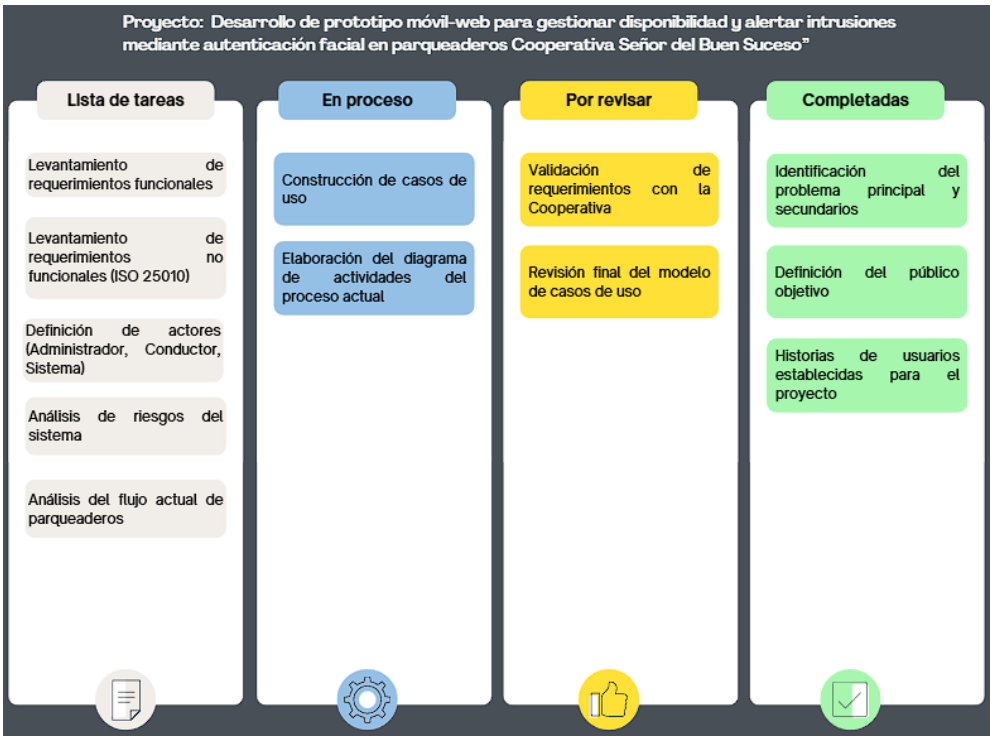


Figura 4: Tablero Kanban fase 1

3.7.2 Fase 2: Diseño

En esta etapa se muestran los diferentes gráficos de ingeniería de software necesarios para organizar y plantear la solución conforme a los requerimientos obtenidos.

Casos de uso

Los casos de uso se detallan en la Figura 5.

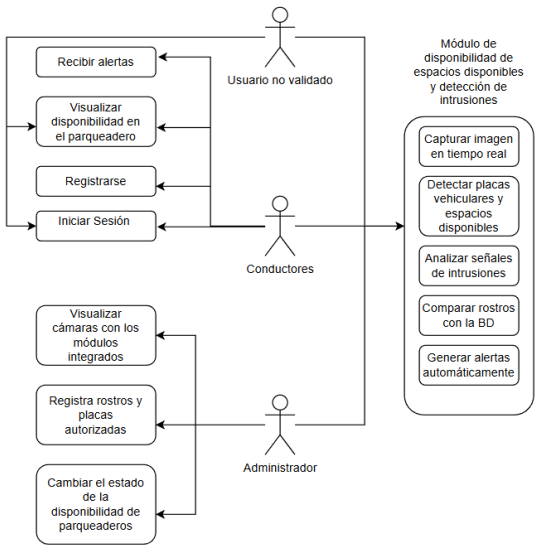


Figura 5: Diagrama de casos de uso

Diagrama de actividades

El diagrama de actividades se observa en la Figura 6.

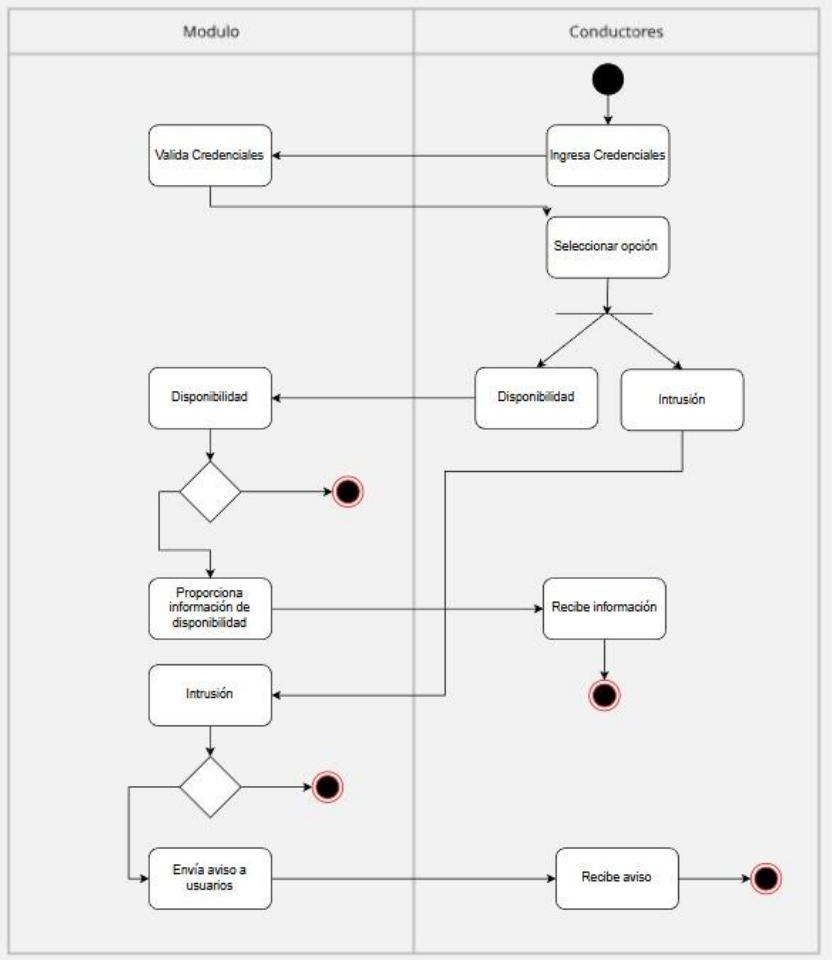


Figura 6: Diagrama de actividades

Diagrama de Despliegue o Arquitectura

El diagrama de despliegue o la arquitectura se detalla en la Figura 7.

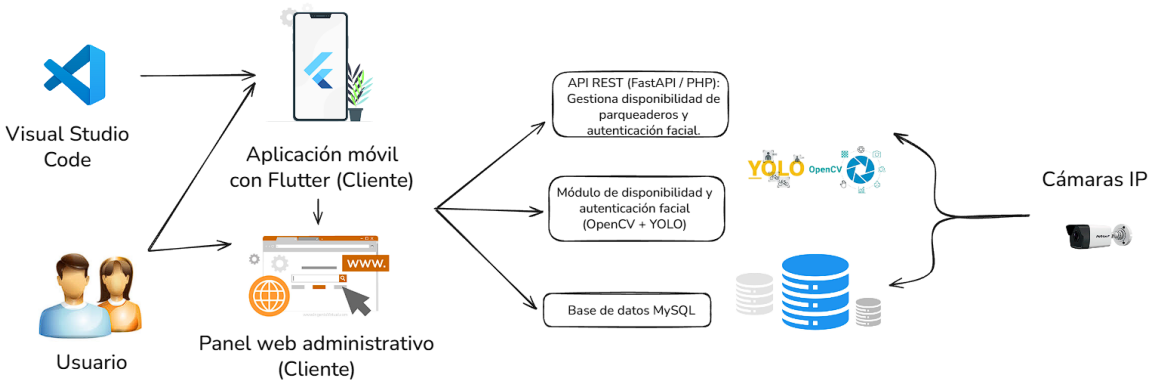


Figura 7: Diagrama de despliegue o arquitectura

Diagrama físico de la base de datos (y script)

El diagrama físico de base de datos se ha elaborado con la herramienta Diseñador de MySQL y se observa en la Figura 8.

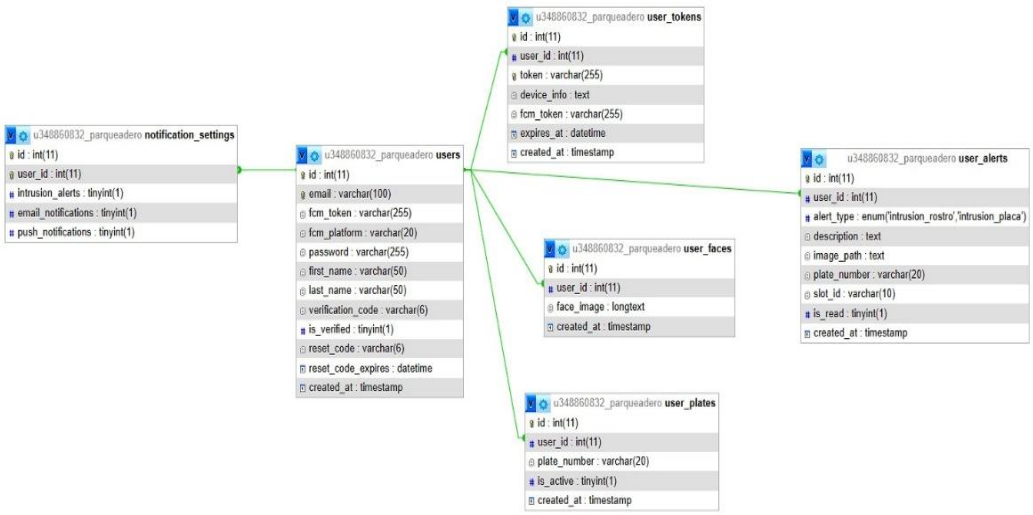


Figura 8: Diagrama de base de datos

Además, se observa el script de dicha base de datos en la Figura 9.

taxi_app.sql - not connected*

```
--
Base de datos: `taxi_app`
-- --------------------------------------------------------
-- Estructura de tabla para la tabla `usuarios`
CREATE TABLE `usuarios` (
  `id` int(11) NOT NULL,
  `email` varchar(100) NOT NULL,
  `password` varchar(255) NOT NULL,
  `nombre` varchar(50) NOT NULL,
  `apellido` varchar(50) NOT NULL,
  `verified_at` datetime DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
-- --------------------------------------------------------
-- Estructura de tabla para la tabla `verificaciones`
CREATE TABLE `verificaciones` (
  `id` int(11) NOT NULL,
  `email` varchar(100) NOT NULL,
  `password` varchar(255) NOT NULL,
  `nombre` varchar(50) NOT NULL,
  `apellido` varchar(50) NOT NULL,
  `codigo` varchar(6) NOT NULL,
  `verificado` tinyint(1) DEFAULT 0,
  `intentos` int(11) NOT NULL DEFAULT 0,
  `expires_at` datetime NOT NULL,
  `fecha` timestamp NOT NULL DEFAULT current_timestamp()
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
-- Indices de la tabla `usuarios`
ALTER TABLE `usuarios`
  ADD PRIMARY KEY (`id`),
  ADD UNIQUE KEY `email` (`email`),
  ADD UNIQUE KEY `unique_email` (`email`);
-- Indices de la tabla `verificaciones`
ALTER TABLE `verificaciones`
  ADD PRIMARY KEY (`id`),
  ADD UNIQUE KEY `unique_email_pending` (`email`);
-- AUTO_INCREMENT de la tabla `usuarios`
ALTER TABLE `usuarios`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=14;
-- AUTO_INCREMENT de la tabla `verificaciones`
ALTER TABLE `verificaciones`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=15;
COMMIT;
```

Figura 9: Script de la base de datos

Diccionario de datos de la base de datos

Tabla de placas autorizadas

La tabla para el authorized_plates_mobile se detalla en la Tabla 7.

Tabla 7: Tabla Authorized Plates

Campo	Tipo	Descripción
id	INT(11)	Identificador único de la tabla
plate_number	VARCHAR(20)	Numero de placa
owner_email	VARCHAR(100)	Correo del dueño de la placa
owner_name	VARCHAR(100)	Nombre del dueño de la placa
is_active	TINYINT(1)	Placa activa o no
create_at	TIMESTAMP	Fecha en el que se creó el registro
update_at	TIMESTAMP	Fecha en el que se actualizo el registro

Tabla de Notificaciones

La tabla para notification_settings se detalla en la Tabla 8.

Tabla 8: Tabla Notification Settings

Campo	Tipo	Descripción
id	INT(11)	Identificador único de la notificación
user_id	INT(11)	Identificador único del usuario
intrusion_alerts	TINYINT(1)	Estado de la intrusión
email_notifications	TINYINT(1)	Estado de la notificación
push_notifications	TINYINT(1)	Se muestra la notificación

Tabla de Parqueadero

La tabla para parking_slots se detalla en la Tabla 9.

Tabla 9: Tabla Parking Slots

Campo	Tipo	Descripción
slot_id	VARCHAR(10)	Identificador único del espacio
zone	VARCHAR(1)	Numero de zona
is_occupied	TINYINT(1)	Ocupado o no
plate_number	VARCHAR(20)	Número de placa
occupied_since	DATETIME	Hora desde que está ocupado
last_update	TIMESTAMP	Ultima hora de actualización

Tabla de Users

La tabla para users se detalla en la Tabla 10.

Tabla 10: Tabla Users

Campo	Tipo	Descripción
id	INT(11)	Identificador único del usuario
email	VARCHAR(20)	Email del usuario

fcm_token	VARCHAR(100)	Token de registro único
fcm_platform	VARCHAR(100)	Vincula con el teléfono
password	VARCHAR(1)	Contraseña del usuario
firts_name	VARCHAR(1)	Nombre
last_name	VARCHAR(1)	Apellido
verification_code	VARCHAR(1)	Código de verificación
is_verified	TINYINT(1)	Estado del código
reset_code	VARCHAR(1)	Cambio de contraseña
reset_code_expires	DATETIME	Expira el código para cambio de contraseña
created_at	TIMESTAMP	Fecha de creación del usuario

Tabla de usuarios con alertas

La tabla para users_alerts se especifica en la Tabla 11.

Tabla 11: Tabla Users Alerts

Campo	Tipo	Descripción
id	INT(11)	Identificador único de la alerta
user_id	INT(11)	Identificador único del usuario
alert_type	ENUM	Tipo de alerta
description	TEXT	Descripción de la alerta
image_path	TEXT	Ruta de la imagen
plate_number	VARCHAR(20)	Número de placa
slot_id	VARCHAR(10)	Id del slot
is_read	TINYINT(1)	Estado de la alerta
created_at	TIMESTAMP	Fecha de creación de la alerta

Tabla de rostros de usuarios

La tabla para users_face se especifica en la Tabla 12.

Tabla 12: Tabla Users Faces

Campo	Tipo	Descripción
id	INT(11)	Identificador único del rostro
user_id	INT(11)	Identificador único del usuario
face_image	LONGTEXT	Imagen del rostro
created_at	TIMESTAMP	Fecha de creación del rostro

Tabla de tokens de usuarios

La tabla para users_token se especifica en la Tabla 13.

Tabla 13: Users Tokens

Campo	Tipo	Descripción
id	INT(11)	Identificador único del token
user_id	INT(11)	Identificador único del usuario
token	VARCHAR(255)	Token de registro único de firebase

device_info	TEXT	Información del dispositivo
fcm_token	VARCHAR(255)	Token de registro único
expires_at	DATETIME	Fecha en el que se expira
create_at	TIMESTAMP	Fecha en el que se creó el token de usuario

Modelado de interfaz

En la herramienta CANVA se realizó el diseño de las interfaces del sistema de disponibilidad de intrusiones, como se observa en la Figura 10, donde se diseñó el Login, creación de usuarios, monitoreo y dashboard.

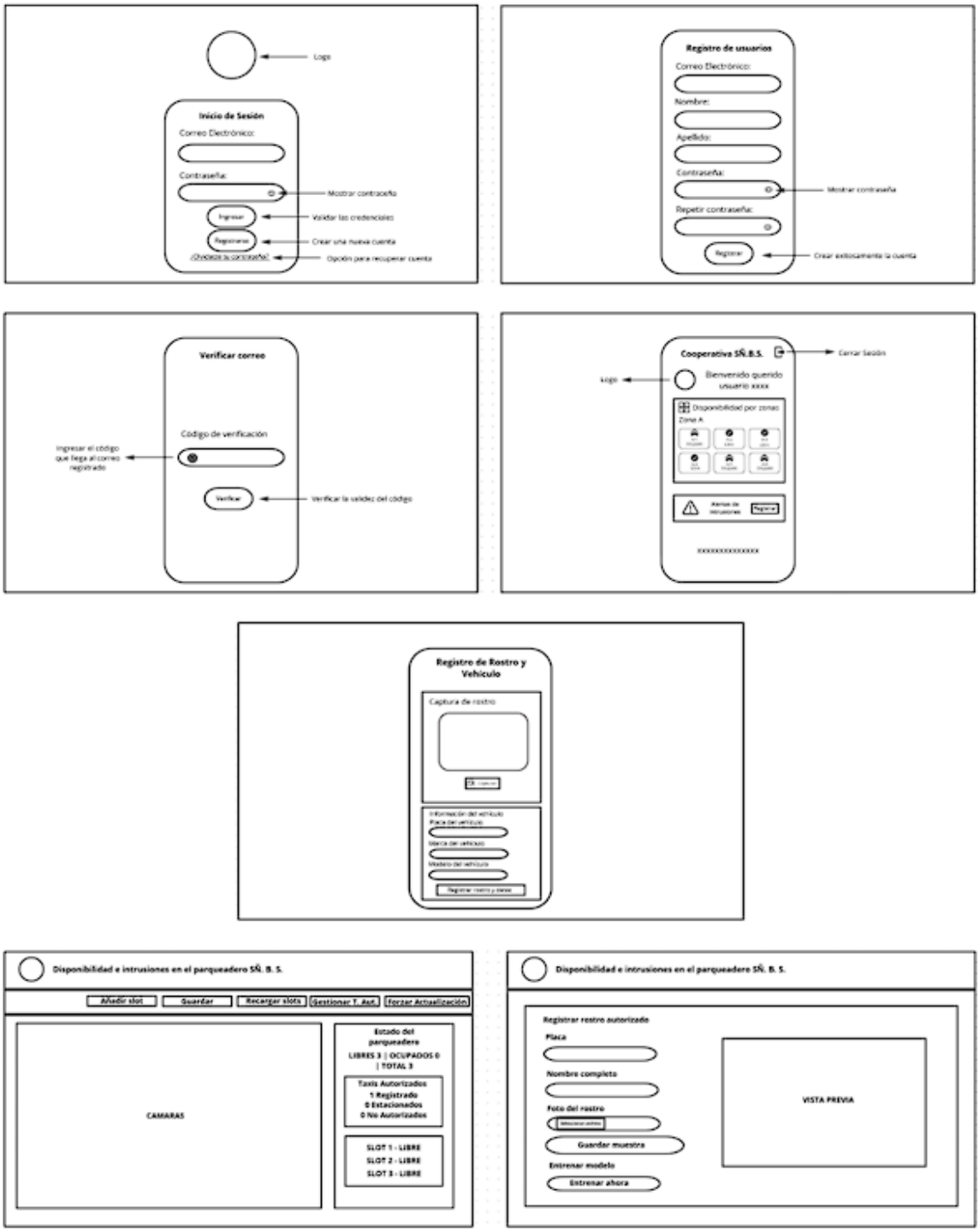


Figura 10: Diagrama de interfaces del sistema

El tablero Kanban cambia sus actividades correspondientes en esta fase como se muestra en la Figura 11.

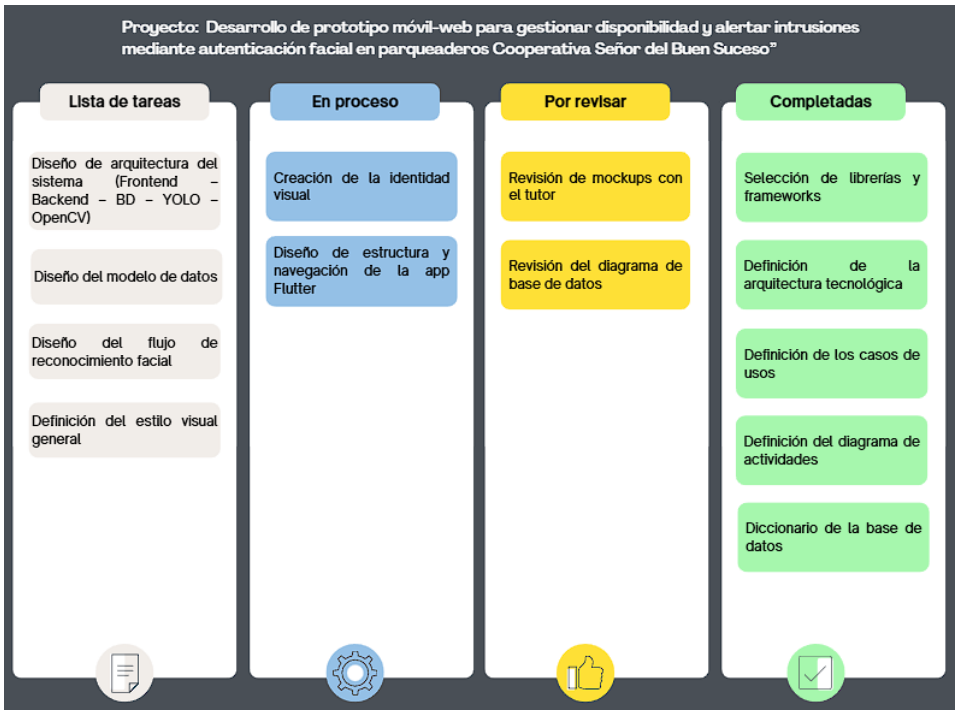


Figura 11: Tablero Kanban Fase 2

3.7.3 Fase 3: Implementación

Desarrollo del sistema

Arquitectura (módulos)

La arquitectura del sistema se diseñó siguiendo un modelo cliente-servidor distribuido, compuesto por módulos que interactúan entre sí mediante servicios y APIs. Los principales módulos que conforman el sistema son los siguientes:

- Módulo de Autenticación Facial: encargado de la captura, procesamiento y validación de los rostros de los conductores, usando algoritmos de reconocimiento facial para verificar la identidad del usuario al momento del ingreso o retiro del vehículo.
- Módulo de Gestión de Parqueaderos: responsable del control de la disponibilidad de espacios de estacionamiento en tiempo real, permitiendo registrar entradas, salidas y estados de ocupación.
- Módulo de Alertas de Seguridad: encargado de la generación automática de notificaciones ante intentos de acceso no autorizado y activándose cuando la autenticación facial no coincide con los registros existentes.
- Módulo de Gestión de Usuarios: administra la información de los socios y conductores autorizados de la cooperativa, incluyendo datos personales y registros faciales.
- Módulo de Base de Datos: centraliza el almacenamiento y recuperación de la información generada por el sistema, garantizando integridad de los datos.
- Módulo de Interfaz de Usuario: permite la interacción entre el sistema y los usuarios a través de pantallas intuitivas y accesibles desde dispositivos móviles y navegadores web.

Lenguaje de desarrollo

Para el desarrollo del prototipo de aplicación móvil-web se seleccionaron lenguajes de programación y tecnologías capaces de integrar de manera eficiente la interfaz de usuario, el procesamiento inteligente de imágenes y la gestión estructurada de la información. El sistema fue implementado utilizando Dart, Python y MySQL, asignando a cada tecnología un rol específico dentro de la arquitectura general del sistema.

El frontend de la aplicación se desarrolló utilizando el lenguaje Dart mediante el framework Flutter, permitió la creación de una interfaz gráfica multiplataforma, compatible con dispositivos móviles y con entornos web. Flutter facilitó el diseño de interfaces responsivas y de alto rendimiento, garantizando consistencia visual, fluidez en la navegación y una experiencia de usuario adecuada. Además, el uso de Dart ayudó a reducir los tiempos de desarrollo y mantenimiento al trabajar sobre una única base de código.

Para los módulos de reconocimiento facial y detección de placas, se utilizó Python, gracias a su amplia adopción en el ámbito de la computer vision y la inteligencia artificial. Python permitió implementar algoritmos especializados para la captura, análisis y comparación de imágenes, facilitando la identificación del conductor y la verificación del vehículo asociado. La disponibilidad de librerías especializadas y su facilidad de integración con servicios backend hacen a Python una herramienta muy útil para el procesamiento avanzado requerido por el sistema.

La gestión de la información se realizó a través del sistema gestor de base de datos MySQL, permitió almacenar de manera estructurada los datos de usuarios, registros faciales, placas vehiculares, estados de ocupación de los parqueaderos y eventos de seguridad. MySQL fue seleccionado por su estabilidad, fiabilidad y compatibilidad con aplicaciones web y móviles, además de su capacidad para garantizar integridad y persistencia de los datos.

La combinación de Dart con Flutter para el desarrollo del frontend, Python para los módulos inteligentes de reconocimiento y MySQL como base de datos permitió construir un sistema escalable y alineado con los objetivos de fiabilidad planteados.

Patrón de diseño

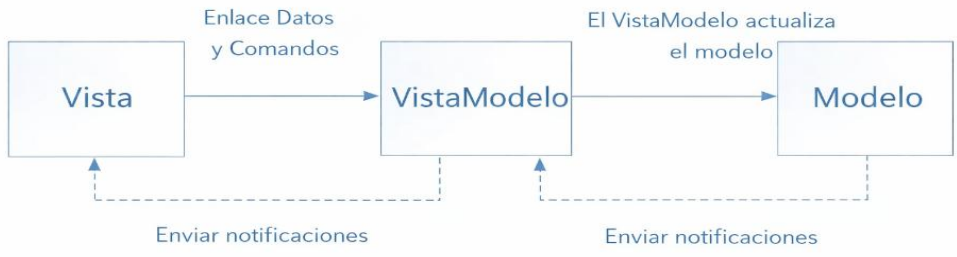


Figura 12: Patrón de diseño

Como se observa en la Figura 12, se aplicó el patrón de diseño Modelo–Vista–VistaModelo (Model–View–ViewModel, MVVM), el cual permitió desacoplar de manera significativa la interfaz de usuario de la lógica de negocio y del manejo de datos. El uso de

este patrón contribuyó a mejorar la organización del código, facilitar el mantenimiento del sistema y favorecer su escalabilidad.

El patrón MVVM se estructura en tres principales componentes: Modelo, Vista y VistaModelo, cada uno con responsabilidades definidas dentro del sistema.

El Modelo fue la capa encargada de gestionar los datos y la lógica de negocio del sistema. En esta aplicación, el modelo se encarga de definir las estructuras de datos relacionadas con los usuarios, registros de parqueo, resultados del reconocimiento facial, detección de placas y eventos de seguridad. Además, esta capa mantiene la comunicación con la base de datos MySQL y garantiza la integridad y consistencia de la información almacenada.

La Vista corresponde a la interfaz de usuario desarrollada en Flutter, responsable de mostrar la información al usuario de forma clara e intuitiva, así como de capturar las interacciones realizadas desde los dispositivos móviles o navegadores web. La vista no contiene lógica de negocios, solo se limita a mostrar los datos proporcionados por el VistaModelo y emitir eventos derivados de las acciones del usuario, asegurando una experiencia de uso fluida.

La VistaModelo (ViewModel) actúa como intermediario entre el modelo y la vista, siendo el encargado de procesar la lógica de presentación y gestionar el estado de la aplicación. La VistaModelo recibe las solicitudes de la vista, interactúa con los modelos y con los servicios backend desarrollados en Python, y expone los datos procesados para que la vista pueda actualizarlos automáticamente. Esta separación permitió un manejo eficiente del estado de la aplicación y una mejor respuesta ante cambios en tiempo real, como la actualización de la disponibilidad de parqueaderos o la generación de alertas de seguridad.

La implementación del patrón MVVM permitió que cada componente del sistema evolucione de forma independiente y así facilitar la incorporación de nuevas funcionalidades como la realización de pruebas unitarias y la reutilización del código. De este modo, el uso de MVVM contribuyó a la construcción de un sistema ordenado y alineado con los principios de buenas prácticas en el desarrollo de aplicaciones móviles y web.

Código (Backend & FrontEnd)

Pasos que se realizaron para implementar el sistema.

Implementación del módulo de conexión con la cámara IP

Se integraron librerías especializadas como OpenCV y Dlib para capturar video en tiempo real, localizar los puntos faciales clave y calcular el parámetro EAR (Eye Aspect Ratio).

Se desarrolló la configuración de una aplicación backend mediante el uso de la librería Pydantic en Python. Se implementó una clase de modificaciones que definió

parámetros operativos fundamentales, tales como la localización del directorio de datos, el prefijo de las rutas de la API y el número total de espacios disponibles.

Además, se integró la URL de un flujo de video RTSP con credenciales de acceso para la gestión de cámaras IP y se configuró un prefijo de entorno para facilitar la gestión de variables externas. El proceso concluyó con la instanciación de un objeto global para proveer estos valores de forma consistente a todo el sistema como se observa en la Figura 13.

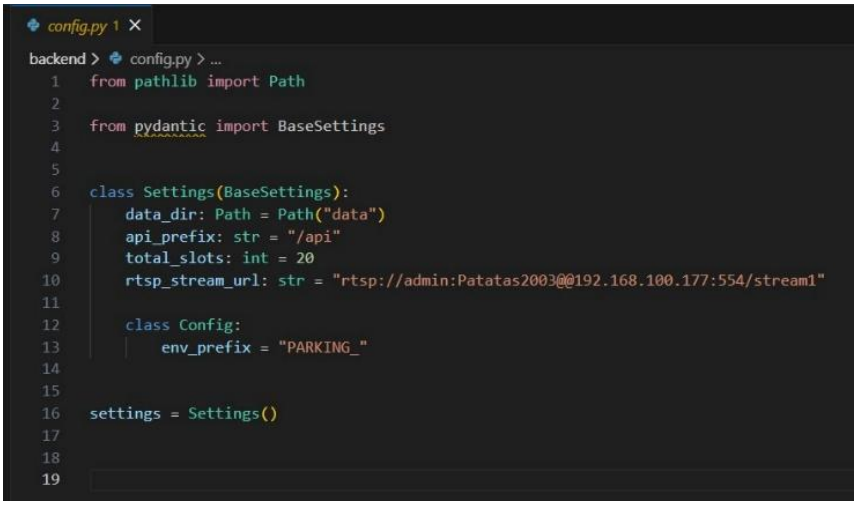
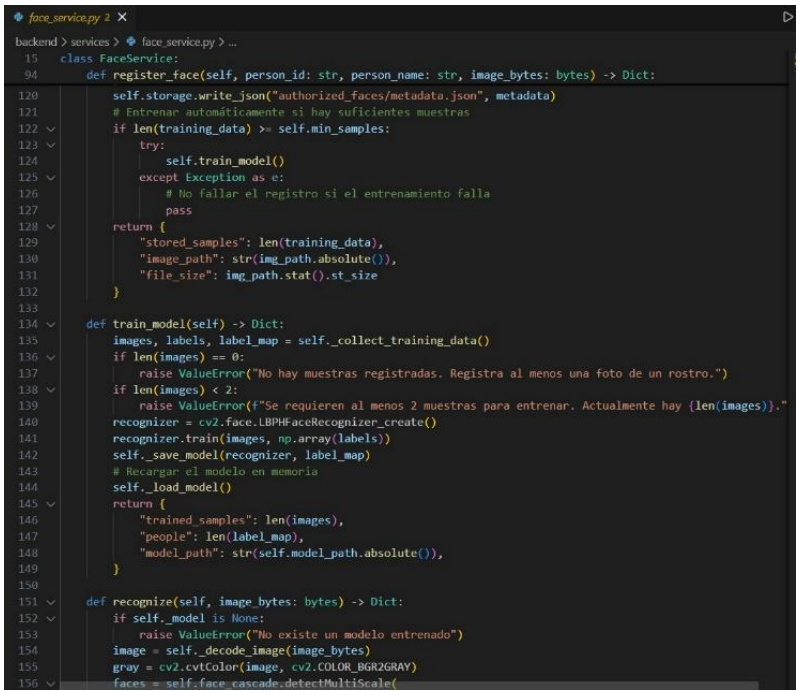


Figura 13: Conexión con la cámara IP

Implementación del módulo de reconocimiento de rostros

Se implementó el reconocimiento facial como un servicio de procesamiento biométrico, el sistema integró una funcionalidad para registrar nuevos rostros, la cual almacenó metadatos y automáticamente un reentrenamiento del modelo al alcanzar un umbral mínimo de muestras.

Además, se desarrolló un proceso de entrenamiento utilizando el algoritmo LBPH de OpenCV, validando la existencia de datos suficientes y persistiendo el modelo actualizado en el disco como se observa en la Figura 14.

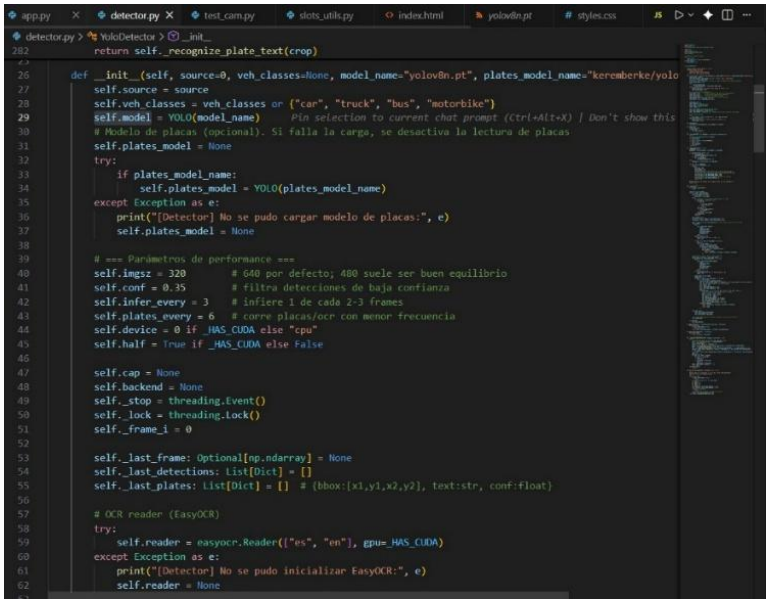


```python
class FaceService:
    def register_face(self, person_id: str, person_name: str, image_bytes: bytes) -> Dict:
        self.storage.write_json("authorized_faces/metadata.json", metadata)
        # Entrenar automáticamente si hay suficientes muestras
        if len(training_data) >= self.min_samples:
            try:
                self.train_model()
            except Exception as e:
                # No fallar el registro si el entrenamiento falla
                pass
        return {
            "stored_samples": len(training_data),
            "image_path": str(img_path.absolute()),
            "file_size": img_path.stat().st_size
        }

    def train_model(self) -> Dict:
        images, labels, label_map = self._collect_training_data()
        if len(images) == 0:
            raise ValueError("No hay muestras registradas. Registra al menos una foto de un rostro.")
        if len(images) < 2:
            raise ValueError(f"Se requieren al menos 2 muestras para entrenar. Actualmente hay {len(images)}.")
        recognizer = cv2.face.LBPHFaceRecognizer_create()
        recognizer.train(images, np.array(labels))
        self._save_model(recognizer, label_map)
        # Recargar el modelo en memoria
        self._load_model()
        return {
            "trained_samples": len(images),
            "people": len(label_map),
            "model_path": str(self.model_path.absolute()),
        }

    def recognize(self, image_bytes: bytes) -> Dict:
        if self._model is None:
            raise ValueError("No existe un modelo entrenado")
        image = self._decode_image(image_bytes)
        gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
        faces = self.face_cascade.detectMultiScale(
```

Figura 14: Modulo de reconocimiento facial con OpenCV

Implementación del módulo para detección de placas vehiculares

Se desarrolló la lógica del detector vehicular basada en modelos YOLOv8, fue orientada a la identificación de vehículos y placas. Se configuró un sistema capaz de cargar modelos pre-entrenados para detectar clases específicas como automóviles, integrando de forma opcional un modelo especializado en la localización de placas y un lector de OCR para la extracción de texto, se observa la estructura en la Figura 15.

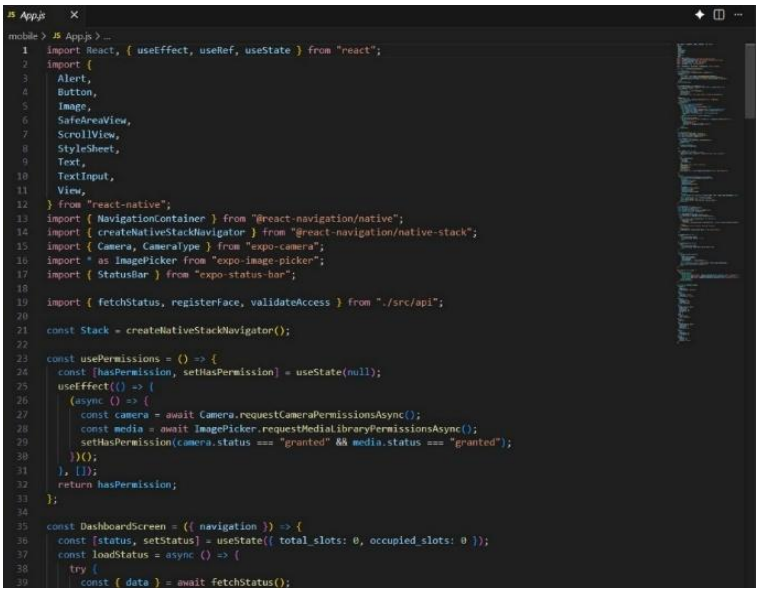


```python
        return self._recognize_plate_text(crop)

    def __init__(self, source=0, veh_classes=None, model_name="yolov8n.pt", plates_model_name="keremberke/yolo
        self.source = source
        self.veh_classes = veh_classes or {"car", "truck", "bus", "motorbike"}
        self.model = YOLO(model_name)     Pin selection to current chat prompt (Ctrl+Alt+X) | Don't show this
        # Modelo de placas (opcional). Si falla la carga, se desactiva la lectura de placas
        self.plates_model = None
        try:
            if plates_model_name:
                self.plates_model = YOLO(plates_model_name)
        except Exception as e:
            print("[Detector] No se pudo cargar modelo de placas:", e)
            self.plates_model = None

        # === Parámetros de performance ===
        self.imgsz = 320        # 640 por defecto; 480 suele ser buen equilibrio
        self.conf = 0.35        # filtra detecciones de baja confianza
        self.infer_every = 3    # infiere 1 de cada 2-3 frames
        self.plates_every = 6   # corre placas/ocr con menor frecuencia
        self.device = 0 if _HAS_CUDA else "cpu"
        self.half = True if _HAS_CUDA else False

        self.cap = None
        self.backend = None
        self._stop = threading.Event()
        self._lock = threading.Lock()
        self._frame_i = 0

        self._last_frame: Optional[np.ndarray] = None
        self._last_detections: List[Dict] = []
        self._last_plates: List[Dict] = []  # {bbox:[x1,y1,x2,y2], text:str, conf:float}

        # OCR reader (EasyOCR)
        try:
            self.reader = easyocr.Reader(["es", "en"], gpu=_HAS_CUDA)
        except Exception as e:
            print("[Detector] No se pudo inicializar EasyOCR:", e)
            self.reader = None
```

Figura 15: Módulo de detección de placas vehiculares

Módulo de navegación y visualización web

Se desarrolló la interfaz de usuario de la aplicación móvil utilizando React Native y Expo mediante Flutter. Se implementó una arquitectura de navegación basada en un stack para gestionar las transiciones entre pantallas, como se ve en la Figura 16.

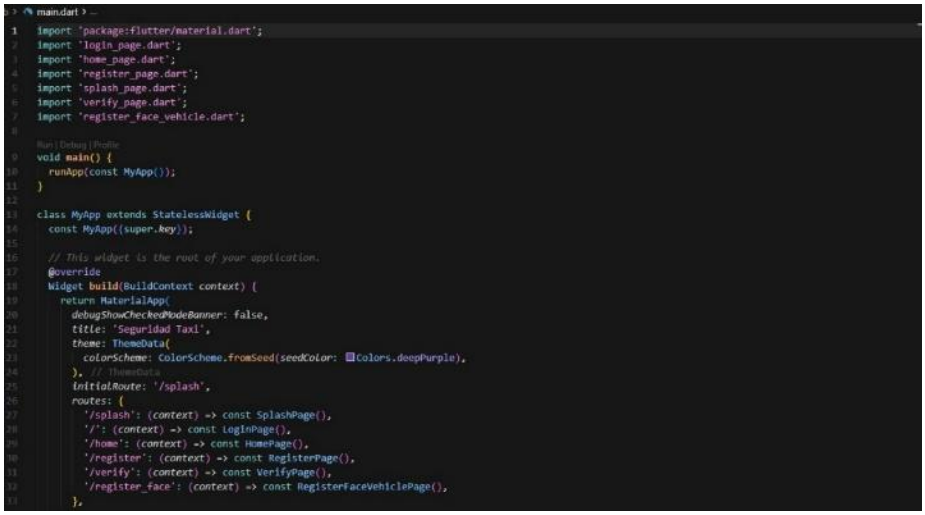


```
import React, { useEffect, useRef, useState } from "react";
import {
  Alert,
  Button,
  Image,
  SafeAreaView,
  ScrollView,
  StyleSheet,
  Text,
  TextInput,
  View,
} from "react-native";
import { NavigationContainer } from "@react-navigation/native";
import { createNativeStackNavigator } from "@react-navigation/native-stack";
import { Camera, CameraType } from "expo-camera";
import * as ImagePicker from "expo-image-picker";
import { StatusBar } from "expo-status-bar";

import { fetchStatus, registerFace, validateAccess } from "./src/api";

const Stack = createNativeStackNavigator();

const usePermissions = () => {
  const [hasPermission, setHasPermission] = useState(null);
  useEffect(() => {
    (async () => {
      const camera = await Camera.requestCameraPermissionsAsync();
      const media = await ImagePicker.requestMediaLibraryPermissionsAsync();
      setHasPermission(camera.status === "granted" && media.status === "granted");
    })();
  }, []);
  return hasPermission;
};

const DashboardScreen = ({ navigation }) => {
  const [status, setStatus] = useState({ total_slots: 0, occupied_slots: 0 });
  const loadStatus = async () => {
    try {
      const { data } = await fetchStatus();
```

Figura 16: Módulo de navegación y visualización web

Módulo de aplicación móvil, mediante Flutter

Se implementó la estructura base de la aplicación mediante un widget de tipo StatelessWidget, configurando un esquema de diseño basado en Material Design. Se definió un sistema de rutas para gestionar el flujo entre las distintas vistas, incluyendo pantallas de inicio (Splash), login, registro de usuarios y una sección especializada para la vinculación de rostros con vehículos como se observa en la Figura 17.



```
import 'package:flutter/material.dart';
import 'login_page.dart';
import 'home_page.dart';
import 'register_page.dart';
import 'splash_page.dart';
import 'verify_page.dart';
import 'register_face_vehicle.dart';

void main() {
  runApp(const MyApp());
}

class MyApp extends StatelessWidget {
  const MyApp({super.key});

  // This widget is the root of your application.
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      debugShowCheckedModeBanner: false,
      title: 'Seguridad Taxi',
      theme: ThemeData(
        colorScheme: ColorScheme.fromSeed(seedColor: Colors.deepPurple),
      ), // ThemeData
      initialRoute: '/splash',
      routes: {
        '/splash': (context) => const SplashPage(),
        '/': (context) => const LogInPage(),
        '/home': (context) => const HomePage(),
        '/register': (context) => const RegisterPage(),
        '/verify': (context) => const VerifyPage(),
        '/register_face': (context) => const RegisterFaceVehiclePage(),
      },
```

Figura 17: Módulo de aplicación móvil

Pantallas

Diseño e implementación de interfaces de usuario

Se desarrollaron las interfaces gráficas utilizando HTML, CSS y JavaScript para la página web, integradas a través del framework Flutter. La Figura 18, muestra la interfaz de login, registro e interfaz final dirigida al usuario.

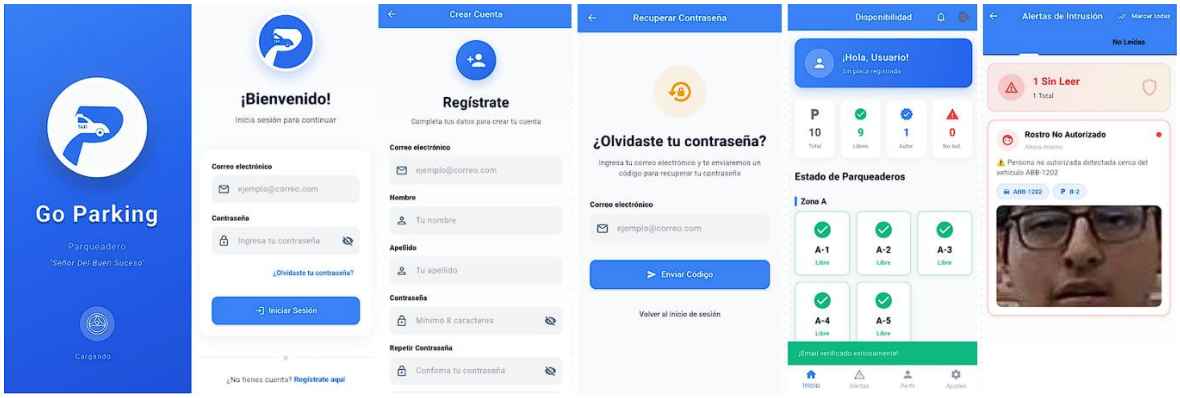


Figura 18: App móvil

Detección de placas (Página Web)

En la Figura 19, se observa el diseño final de la interfaz web para la administración correspondiente del módulo de detección para placas vehiculares.

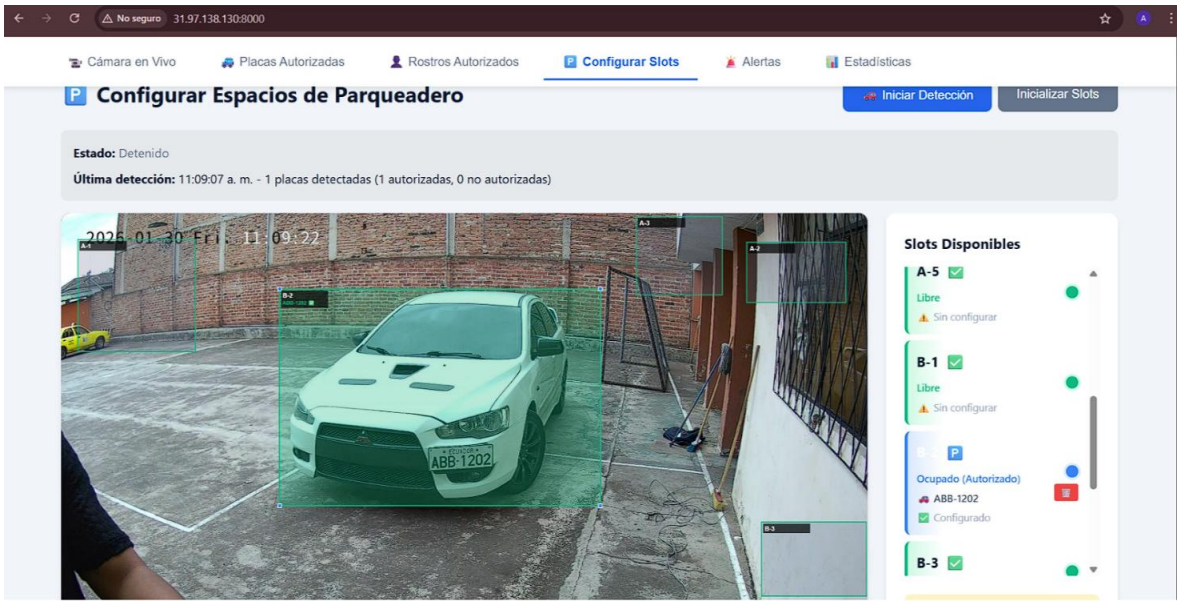


Figura 19: Página web - detección de placas

Reconocimiento facial (Página Web)

La Figura 20, presenta el diseño final de la interfaz web para la administración correspondiente del módulo de reconocimiento facial para las alertas de intrusiones.

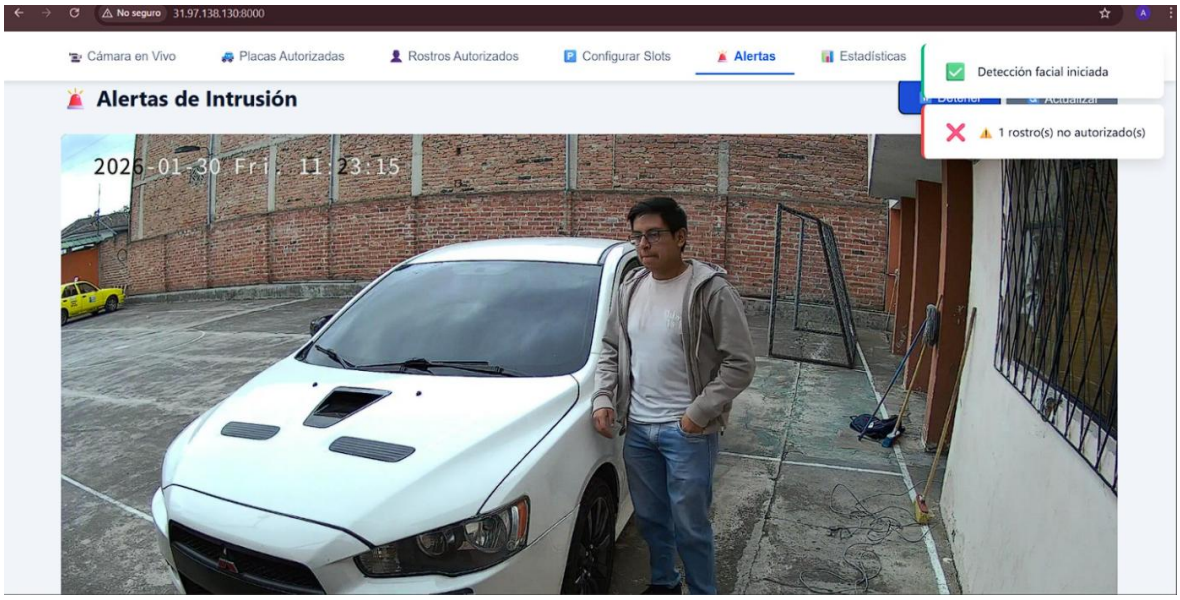


Figura 20: Página web - detección de rostros

El tablero Kanban cambia sus actividades correspondientes en esta fase como se especifica en la Figura 21.

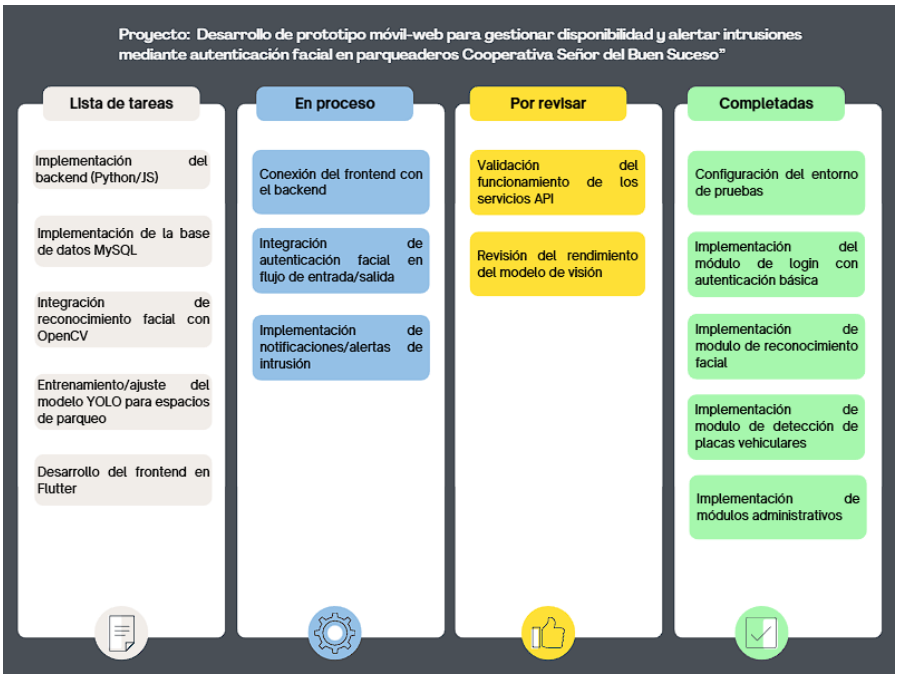


Figura 21: Tablero Kanban Fase 3

3.7.4 Fase 4: Pruebas y evaluación del sistema

Una vez concluida la fase de implementación del prototipo móvil-web, se procedió a la fase de pruebas, cuyo objetivo principal fue verificar el correcto funcionamiento, fiabilidad y calidad del sistema que fueron definidos durante las fases previas del tablero Kanban. Esta fase permitió validar que las funcionalidades desarrolladas cumplan con los criterios de calidad establecidos y que el sistema responda correctamente ante escenarios reales de uso en la Cooperativa de Taxis “Señor del Buen Suceso”.

La metodología Kanban facilitó la gestión de esta fase mediante la visualización de tareas de prueba en estados como “pendiente, en ejecución y finalizado”, permitiendo un control continuo del avance y la detección inmediata de incidencias.

Previo a la ejecución de las pruebas, se diseñó una estrategia estructurada para evaluar el comportamiento del sistema bajo condiciones controladas y escenarios reales simulados. Para ello se consideraron los siguientes aspectos:

- Uso del sistema por parte de usuarios de prueba con perfiles similares a los conductores de la cooperativa.
- Simulación del ingreso y retiro de vehículos en los parqueaderos.
- Pruebas del módulo de autenticación facial para la verificación del conductor autorizado.
- Activación del sistema de alertas ante intentos de acceso no autorizado.
- Verificación de la actualización en tiempo real de la disponibilidad de espacios de parqueo.

Esta planificación permitió asegurar una evaluación integral de las funcionalidades críticas del sistema.

Técnicas de prueba aplicadas

Durante la fase de pruebas se emplearon las siguientes técnicas:

Pruebas funcionales

Permitieron comprobar que cada funcionalidad del sistema funcione de acuerdo con lo esperado, verificando el cumplimiento de los requerimientos funcionales.

Pruebas de uso

Se evaluó la interacción del usuario con la aplicación móvil-web, observando la facilidad de uso, buen flujo de navegación y comprensión de las alertas generadas por el sistema.

Simulaciones controladas

Se realizaron pruebas simulando situaciones reales, como el retiro del vehículo por un usuario no autorizado, con el fin de verificar la correcta detección de intrusiones mediante autenticación facial y la generación inmediata de alertas.

Métrica de evaluación utilizada

Para garantizar una evaluación objetiva y estandarizada de la calidad del sistema, se utilizó la norma ISO/IEC 25010:2023, específicamente la característica de Adecuación Funcional, perteneciente al modelo de calidad del producto de software.

Planificación de las pruebas

¿Qué se va a hacer?

En esta fase se realizarán pruebas técnicas y funcionales al prototipo de aplicación móvil-web desarrollado para la gestión de parqueaderos y autenticación facial. Las pruebas estarán orientadas a evaluar el comportamiento del sistema bajo diferentes condiciones de uso, con énfasis en:

- Rendimiento del sistema
- Capacidad de respuesta
- Estabilidad del backend
- Correcto registro de eventos y accesos
- Funcionamiento de los servicios de autenticación y disponibilidad

¿Cómo se va a hacer?

La planificación y control de las pruebas se realizará utilizando un tablero visual para organizar las actividades en las siguientes columnas:

- Pendiente
- En proceso
- En revisión
- Finalizado

Para la ejecución de las pruebas de carga y rendimiento se empleará la herramienta Apache JMeter, mediante la cual se simularán múltiples solicitudes concurrentes al sistema, evaluando su comportamiento en escenarios controlados.

¿Quiénes participan?

Responsable de pruebas:

Estudiante investigador

Muestra técnica:

Servicios del backend del sistema

Registros y datos:

- Resultados generados por JMeter
- Logs del sistema
- Métricas de rendimiento
- Registros de errores y tiempos de respuesta

CAPÍTULO IV. RESULTADOS Y DISCUSIÓN

4.1 Resultados

Evaluación de la completitud funcional

La completitud funcional permitió verificar si la aplicación móvil-web incorpora todas las funcionalidades necesarias para la gestión segura y eficiente de los parqueaderos. Se evaluó que el sistema permita:

- Registrar y autenticar usuarios.
- Visualizar la disponibilidad de parqueaderos en tiempo real.
- Autenticar facialmente al conductor al momento de retirar el vehículo.
- Detectar intentos de acceso no autorizado.
- Generar alertas automáticas ante intrusiones.
- Registrar eventos de ingreso y salida de vehículos.

La evaluación se realizó mediante una tabla de comprobación funcional, donde cada funcionalidad fue clasificada según su resultado obtenido y su estado de cumplimiento.

Evaluación de la adecuación funcional

La adecuación funcional permitió determinar si las funcionalidades implementadas aportan valor real a los usuarios y cumplen con el propósito del sistema. Se evaluó si:

- El sistema contribuye a mejorar la seguridad de los vehículos estacionados.
- La autenticación facial es útil y confiable para validar al conductor autorizado.
- Las alertas generadas son oportunas, claras y comprensibles.
- El flujo de interacción es coherente con las operaciones diarias de la cooperativa.

Para esta evaluación se aplicaron cuestionarios estructurados, diseñados en función de las subcaracterísticas de la norma ISO/IEC 25010.

Una vez evaluada cada una de las funcionalidades se procederá a explicar en la Tabla 14 acerca de la comprobación de completitud funcional.

Tabla 14: Comprobación de completitud funcional

ID	Funcionalidad	Entrada	Procedimiento	Resultado esperado	Resultado obtenido
CF01	Registro e inicio de sesión	Correo electrónico y contraseña	El usuario accede a la aplicación e ingresa sus credenciales	El sistema valida el acceso y permite ingresar al sistema	Usuario registrado e ingreso correctamente
CF02	Registro de ingreso de vehículo	Datos del vehículo y conductor	Registrar el ingreso del vehículo al parqueadero	El sistema registra el ingreso y actualiza la disponibilidad	Vehículo registrado correctamente

ID	Funcionalidad	Entrada	Procedimiento	Resultado esperado	Resultado obtenido
CF03	Visualización de disponibilidad	Acceso a la vista principal	Consultar los espacios disponibles en tiempo real	Se muestran los espacios ocupados y disponibles correctamente	Se visualiza correctamente
CF04	Autenticación facial del conductor	Captura facial	Validar la identidad del conductor al retirar el vehículo	El sistema reconoce al conductor autorizado	Autentificación facial correctamente
CF05	Detección de intrusión	Rostro no autorizado	Intentar retirar un vehículo sin coincidencia facial	El sistema detecta acceso no autorizado	La Detección de intrusos se realizó correctamente
CF06	Generación de alertas	Evento de intrusión	Activación automática de alerta	Se genera una alerta inmediata al administrador	Se enviaron alertas correctamente
CF07	Registro de eventos	Datos de ingreso y salida	Almacenar eventos del sistema	Los eventos se guardan correctamente en la base de datos	Eventos registrados correctamente

Evaluación de madurez (Ausencia de Fallos)

Se evalúa la madurez del sistema mediante pruebas de carga con Apache JMeter, simulando 10 usuarios concurrentes, ejecutando 10 iteraciones cada uno, para un total de 111 peticiones HTTP a los diferentes endpoints del sistema.

Los resultados obtenidos durante la ejecución de la prueba demuestran la estabilidad del prototipo. Tal como se observa en la Figura 22, el sistema procesa exitosamente el 100% de las solicitudes sin registrar ningún fallo, evidencia en la columna "Error %" un valor de 0.00% en todos los endpoints evaluados.

Figura 22: Prueba de Madurez

En la Tabla 15, se resume los resultados de la prueba de madurez correspondiente.

Tabla 15: Prueba de madurez

Label	# Samples	Average (s)	Min (ms)	Max (ms)	Std. Dev. (ms)	Error %	Throughput
GET - Estado Parking	40	0.348	3	2102	454.61	0.00%	4.3/sec
GET - Estado Camara	40	0.076	3	911	199.02	0.00%	4.3/sec
GET - Lista Alertas	31	0.268	157	622	124.71	0.00%	5.6/sec
TOTAL	111	0.228	3	2102	327.22	0.00%	11.9/sec

En la prueba de madurez, se evaluó el comportamiento de endpoints clave del sistema: Estado Parking, Estado Cámara y Lista de Alertas. En total se registraron 111 solicitudes, obteniendo un tiempo promedio global de 228 ms, con 0% de errores, evidenciando estabilidad en la ejecución del flujo probado. El endpoint con menor latencia fue Estado Cámara (76 ms promedio), mientras que el mayor valor máximo registrado fue de 2102 ms en Estado Parking. Adicionalmente, el sistema alcanzó un throughput total de 11.9 solicitudes/segundo, lo cual refleja una capacidad adecuada de respuesta bajo la carga aplicada en esta evaluación.

Evaluación de disponibilidad

En esta evaluación se observó el grado en que el sistema con sus módulos está operativo cuando se lo requiere, con métricas de 24 horas como se observa en la Figura 23.

Figura 23: Prueba de disponibilidad

Evaluación de tolerancia a fallos

En la evaluación del sistema para operar correctamente ante fallos se sobrecarga a 200 usuarios concurrentes en un mapeo seguido de 10 segundos durante una duración máxima de 5 minutos en todos los endpoints de los módulos como se presenta en la Figura 24.

Figura 24: Prueba de tolerancia a fallos

La Tabla 16, muestra los resultados de la prueba de tolerancia a fallos.

Tabla 16: Prueba de tolerancia a fallos

Label	# Samples	Average (s)	Min (ms)	Max (ms)	Std. Dev. (ms)	Error %	Throughput
GET Parking	98	126.321	6	381025	171810.86	33.67%	15.2/min
GET Alertas	65	8.952	190	345430	146460.22	26.15%	10.1/min
GET Camara	49	17.321	7	344980	49434.93	4.08%	7.6/min
TOTAL	212	89.384	6	381025	150328.32	24.53%	32.9/min

En la prueba de tolerancia a fallos, se sometió el sistema a un escenario de mayor exigencia, evaluando GET Parking, GET Alertas y GET Cámara, con un total de 212 solicitudes. El tiempo promedio global se incrementó a 89 384 ms (≈ 89.3 s) y el sistema alcanzó tiempos máximos de hasta 381 025 ms, evidenciando degradación del rendimiento en condiciones adversas. Además, se registró un 24.53% de errores, donde el endpoint más afectado fue GET Parking (33.67% Error), seguido de GET Alertas (26.15% Error), mientras que GET Cámara presentó mejor comportamiento con 4.08% Error. Estos resultados permiten identificar los puntos de mayor vulnerabilidad del sistema ante carga y condiciones de falla, aportando información clave para optimización y control de resiliencia.

Monitoreo

La Figura 25, detalla los resultados consumo de CPU, RAM, disco y red.

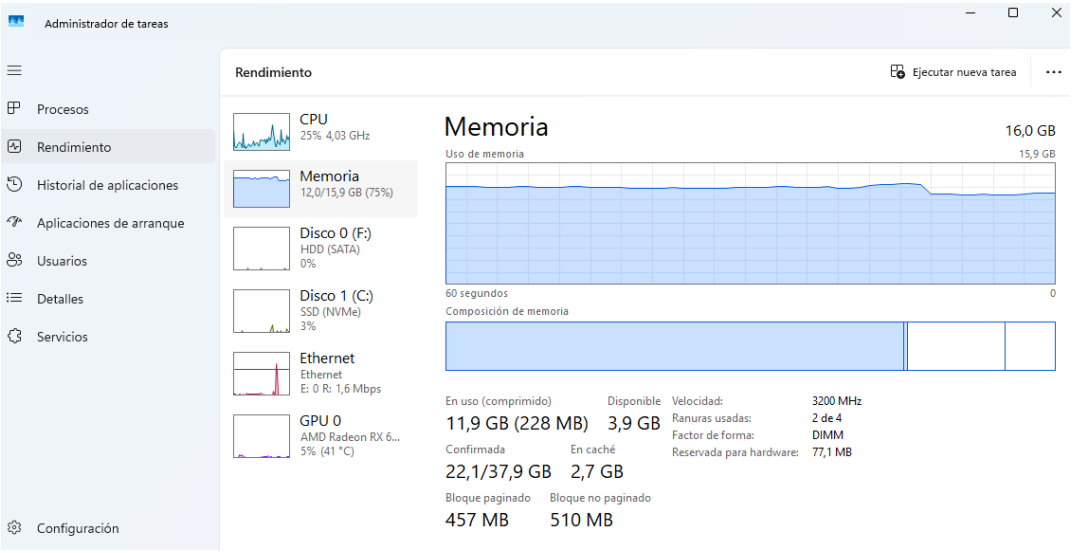


Figura 25: Recursos utilizados

El tablero Kanban cambia sus actividades correspondientes en esta fase como se ve en la Figura 26.

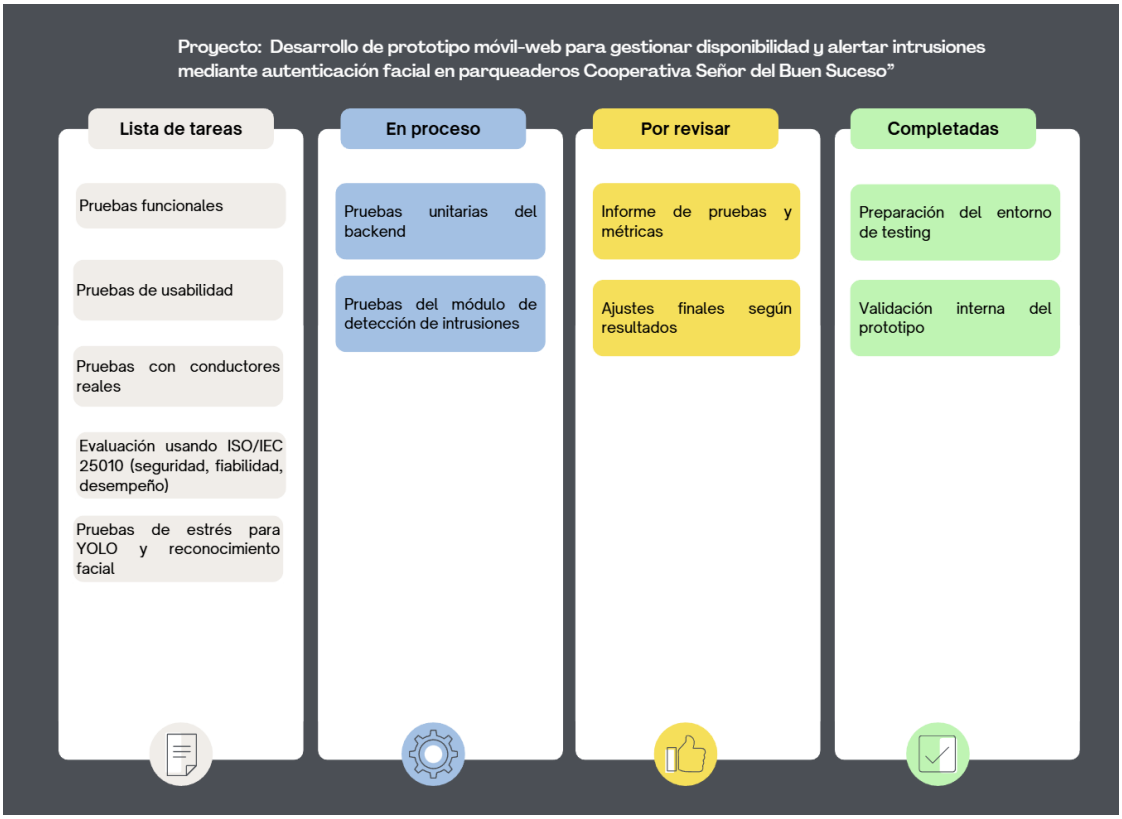


Figura 26: Tablero Kanban fase 4

Ausencia de fallos

En la prueba de madurez se ejecutaron 111 solicitudes sobre tres servicios principales del sistema. El rendimiento global presentó un tiempo promedio de 228 ms y 0% de errores, evidencia un comportamiento estable en el escenario evaluado. El servicio con mejor tiempo de respuesta fue Estado Cámara (76 ms promedio), mientras que Estado Parking registró el

mayor pico (2102 ms máximo), reflejando variaciones puntuales. En términos de capacidad, el sistema alcanzó un throughput total de 11.9 req/s, manteniendo consistencia operativa durante la ejecución. Tal y como se observa en la Figura 27.

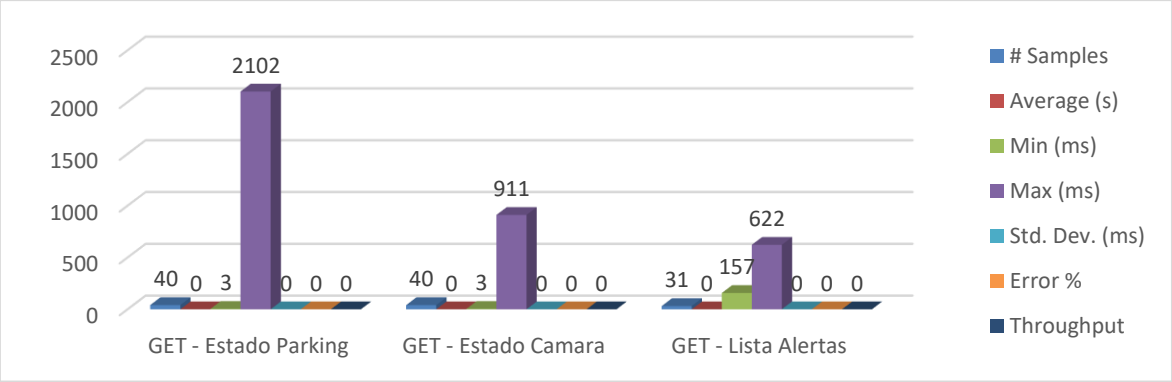


Figura 27: Grafica de madurez en ausencia a fallos

Disponibilidad del sistema

En la prueba de disponibilidad, se validó la continuidad del servicio mediante el endpoint Ping Sistema, registrando 6 solicitudes con un promedio de 773 ms, sin errores (0% Error). El tiempo máximo alcanzó 3045 ms, mientras que el mínimo fue de 5 ms, variaciones asociadas a condiciones del entorno (red/recursos). En términos de capacidad, el sistema mantuvo un throughput de 1.2 solicitudes por minuto, confirmando que el servicio permaneció accesible durante el periodo de verificación como se presenta en la Tabla 17.

Tabla 17: Prueba de disponibilidad

Label	# Samples	Average (s)	Min (ms)	Max (ms)	Std. Dev. (ms)	Error %	Throughput
Ping Sistema	6	0.773	5	3045	1138.83	0.00%	1.2/min
TOTAL	6	0.773	5	3045	1138.83	0.00%	1.2/min

Tolerancia a fallos

En la prueba de tolerancia a fallos, el endpoint GET Parking fue el más crítico, porque registró el mayor tiempo promedio de respuesta (126,321 s) y el pico máximo más alto (381.025 ms), evidenciando alta variabilidad e inestabilidad. En comparación, GET Alertas y GET Cámara presentaron promedios menores (8,952 s y 17,321 s, respectivamente), aunque también alcanzaron picos máximos elevados (345.430 ms y 344.980 ms), esto indica que bajo ciertas condiciones pueden producirse demoras extremas (Ver Figura 28).

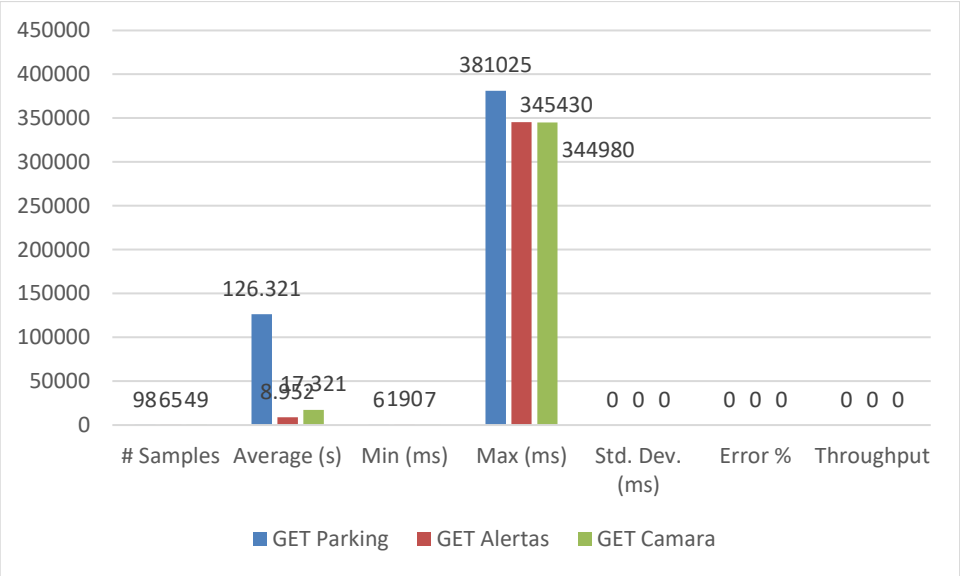


Figura 28: Grafica de tolerancia a fallos

4.2 Discusión

Los resultados esperados del desarrollo de una aplicación móvil-web para la gestión de parqueaderos con autenticación facial pueden analizarse a partir de investigaciones recientes que han abordado problemáticas similares mediante el uso de inteligencia artificial, visión por computador y tecnologías móviles.

En relación con la gestión de disponibilidad de espacios, diversos estudios han demostrado que los sistemas basados en visión artificial permiten mejorar significativamente la eficiencia del estacionamiento. Aguilar-Alvarado et al. desarrollaron un sistema utilizando OpenCV para la detección de espacios disponibles, evidenciando mejoras en la automatización del proceso y reducción de errores humanos [17]. Este enfoque se alinea con la presente investigación, aunque el uso de modelos más avanzados como YOLOv8 representa una mejora sustancial en términos de precisión y velocidad de procesamiento en tiempo real.

De manera similar, investigaciones más recientes han incorporado modelos de aprendizaje profundo para optimizar la detección de espacios. De Luelmo et al. proponen un sistema híbrido que combina deep learning con reglas heurísticas, logrando una mayor robustez en entornos dinámicos [18]. En comparación, la propuesta de esta tesis se enfoca en una arquitectura más integrada orientada a dispositivos móviles y entornos reales de cooperativas, lo cual puede facilitar su implementación práctica.

En cuanto al componente de seguridad, el reconocimiento facial ha sido ampliamente validado como mecanismo de autenticación confiable. Amezquita-Sánchez et al. desarrollaron un sistema basado en redes neuronales convolucionales que permite autenticar usuarios en tiempo real, alcanzando altos niveles de precisión [19]. Estos resultados respaldan la viabilidad del uso de autenticación facial en el contexto de parqueaderos, como

se plantea en esta investigación, donde se busca verificar la identidad del conductor al momento de retirar el vehículo.

Asimismo, algunos estudios han optado por integrar múltiples tecnologías de identificación para reforzar la seguridad. Un sistema propuesto recientemente combina reconocimiento facial con identificación de placas vehiculares, logrando mejorar la detección de accesos no autorizados y optimizar la gestión del parqueo [20]. En comparación, la presente propuesta se centra en la autenticación facial, lo cual simplifica la implementación y reduce costos, aunque podría representar una limitación frente a soluciones más robustas que integran múltiples factores de autenticación.

Desde una perspectiva aplicada, implementaciones reales han demostrado el impacto positivo de los sistemas automatizados de estacionamiento. Un caso de estudio en un centro comercial en Colombia evidenció una reducción significativa en los tiempos de espera mediante el uso de reconocimiento automático de vehículos [21]. Estos resultados permiten inferir que una solución similar, como la planteada en esta tesis, podría mejorar la eficiencia operativa en la Cooperativa de Taxis “Señor del Buen Suceso”.

Finalmente, revisiones sistemáticas recientes destacan que la integración de tecnologías como IoT, inteligencia artificial y aplicaciones móviles constituye la tendencia actual en sistemas de estacionamiento inteligente, mejorando tanto la eficiencia como la seguridad [22]. En este sentido, la propuesta de una aplicación móvil-web con autenticación facial se alinea con estas tendencias, aportando además un enfoque específico en cooperativas de transporte, un contexto poco explorado en la literatura científica.

CAPÍTULO V. CONCLUSIONES Y RECOMENDACIONES

5.1 Conclusiones

- Con base en la revisión y comparación realizada, se identificó que es totalmente viable integrar una solución móvil-web con técnicas de reconocimiento facial para un contexto como el de un parqueadero. La investigación permitió escoger tecnologías que se ajustan al prototipo (por compatibilidad, facilidad de integración y rendimiento), dejando claro que el reconocimiento facial puede utilizarse como un mecanismo útil de verificación de identidad cuando se controla bien el entorno de captura (iluminación, distancia y ángulo).
- Se logró implementar un prototipo funcional que integra la disponibilidad en tiempo real y la autenticación facial del conductor, incorporando además un sistema automático de alertas cuando se detectan intentos de acceso no autorizado. En las pruebas realizadas, el flujo principal de uso (registro, validación, consulta de disponibilidad y generación de alertas) se ejecutó de manera estable, demostrando que la propuesta puede apoyar directamente a la cooperativa en el control del parqueadero y en la mejora de la seguridad operativa.
- La evaluación de la fiabilidad, realizada según los criterios de la ISO/IEC 25010:2023, permitió verificar que el prototipo mantiene un comportamiento adecuado bajo condiciones normales de operación, con una disponibilidad del 100% y sin errores en las pruebas de "Estado Parking" y "Estado Cámara" (Error %: 0.00%). Sin embargo, las pruebas de tolerancia a fallos mostraron que, en escenarios de alta carga, el sistema empieza a evidenciar limitaciones, con un 33.67% de error en la prueba de GET Parking y un 24.53% de error en GET Cámara, lo que indica una caída en el rendimiento y la capacidad de respuesta. Estos resultados subrayan la necesidad de futuras mejoras, como la optimización de servicios, la reducción de la carga de datos y ajustes en la infraestructura, para asegurar que el sistema sea más robusto y escalable en una implementación completa.

5.2 Recomendaciones

- Se recomienda optimizar la arquitectura del sistema para mejorar su rendimiento en escenarios de alta carga, mediante la implementación de técnicas como el uso de servicios en la nube escalables, balanceo de carga y optimización de consultas a la base de datos. Esto permitirá reducir los porcentajes de error observados en las pruebas de tolerancia a fallos y garantizar una mayor estabilidad en entornos reales.
- Se recomienda implementar una validación adicional para los usuarios autorizados, de modo que, una vez autenticados, puedan acceder a una vista en vivo (grabación en tiempo real) del vehículo dentro del parqueadero. Esto permitiría que el conductor pueda confirmar visualmente el estado de su carro cuando lo necesite, reforzando la seguridad y aumentando la confianza en el sistema, especialmente en horarios de mayor riesgo o cuando se genere una alerta.

- Se recomienda ampliar el sistema en futuras versiones mediante la integración de tecnologías complementarias, como el reconocimiento de placas vehiculares o sensores IoT, con el fin de incrementar los niveles de seguridad y automatización. Asimismo, se sugiere realizar pruebas en entornos reales prolongados que permitan evaluar el comportamiento del sistema en condiciones operativas continuas y validar su escalabilidad

BIBLIOGRÁFIA

[1] J. X. Uquillas López, «MOVILIDAD EN LA CIUDAD DE RIOBAMBA EN TIEMPOS DE CUARENTENA POR EL COVID - 19,» Riobamba: Universidad Nacional de Chimborazo, Riobamba, 2021.

[2] S. S. Channamallu, «A review of smart parking systems,,» ELSEVIER, Arlington, 2023.

[3] C. G. Gheorghe, M. Duguleana, G. Bobox y C. C. Postelnicu, «ScienceDirect,» 31 Octubre 2024. [En línea]. Available: https://www.sciencedirect.com/science/article/pii/S1526149224003011. [Último acceso: 2025].

[4] H. Gónzales, «Algoritmos de detección de anomalías con redes profundas. Revisión para detección de fraudes bancarios,» ResearchGate, Francia, 2021.

[5] N. Fadel, «Facial Recognition Algorithms: A Systematic Literature Review,» MDPI, Tabuk, 2025.

[6] K. S. Zurita, «FRAMEWORKS PARA EL DESARROLLO DE APLICACIONES MÓVILES,» Pontifica Universidad Católica del Ecuador, Quito, 2020.

[7] L. R. Quisaguano, M. S. Pallasco, A. A. Andaluz y M. N. Martínez, «Desarrollo Hibrido con Flutter,» Ciencia Latina, Cotopaxi, 2022.

[8] G. Luna, «Desarrollo de una plataforma e-learning para reforzar los procesos de aprendizaje del centro de estudios : desarrollo de una aplicación móvil de la plataforma e-learning para el centro de estudios,» BIBDIGITAL, Quito, 2025.

[9] ARC42, «quality arc42 org,» Quality Model, Noviembre 2023. [En línea]. Available: https://quality.arc42.org/articles/iso-25010-update-2023.

[10] ISO, «ISO ORG,» Febrero 2023. [En línea]. Available: https://www.iso.org/obp/ui/en/#!iso:std:78176:en. [Último acceso: 2025].

[11] ISO 25000, «ISO25000,» iso25000, 2024. [En línea]. Available: https://iso25000.com/index.php/normas-iso-25000/iso-25010.

[12] J. Wang, L. Zhang, A. Li, X. You y H. Cheng, «Efficient Participant Contribution Evaluation for Horizontal and Vertical Federated Learning,» IEEE Xplore, Kuala Lumpur, 2022.

[13] Mkt, «ENGEMAN,» 26 Junio 2024. [En línea]. Available: https://blog.engeman.com/es/que-es-disponibilidad-y-como-calcularla-en-mantenimiento/. [Último acceso: 2025].

[14] F. Kastensmidt, A. Romanovsky, D. Liang y Z. Koťasek, «ScienceDirect,» 2020. [En línea]. Available: https://www.sciencedirect.com/topics/computer-science/fault-tolerance. [Último acceso: 2025].

[15] M. Castillo y E. J. Guaña, «Kanban: Una metodología ágil para la gestión eficiente,» ISSN, Quito, 2024.

[16] L. Castellano Lendínez, «KANBAN. METODOLOGÍA PARA AUMENTAR LA EFICIENCIA DE LOS PROCESOS,» Universidad de Jaén, Valencia, 2019.

[17] M. C. y. L. O. J. Aguilar-Alvarado, Sistema de visión artificial para la detección de espacios de estacionamiento, 2021.

[18] J. G. y. R. G. S. P. de Luelmo, Combining deep learning methods and rule-based systems for automatic parking space detection, 2024.

[19] D. M. y. C. V. J. P. Amezquita-Sánchez, Deep learning-based face recognition for automotive security applications, 2024.

[20] S. S. y. A. P. R. Kumar, A deep learning-powered smart parking system based on facial recognition and license plate analysis, 2024.

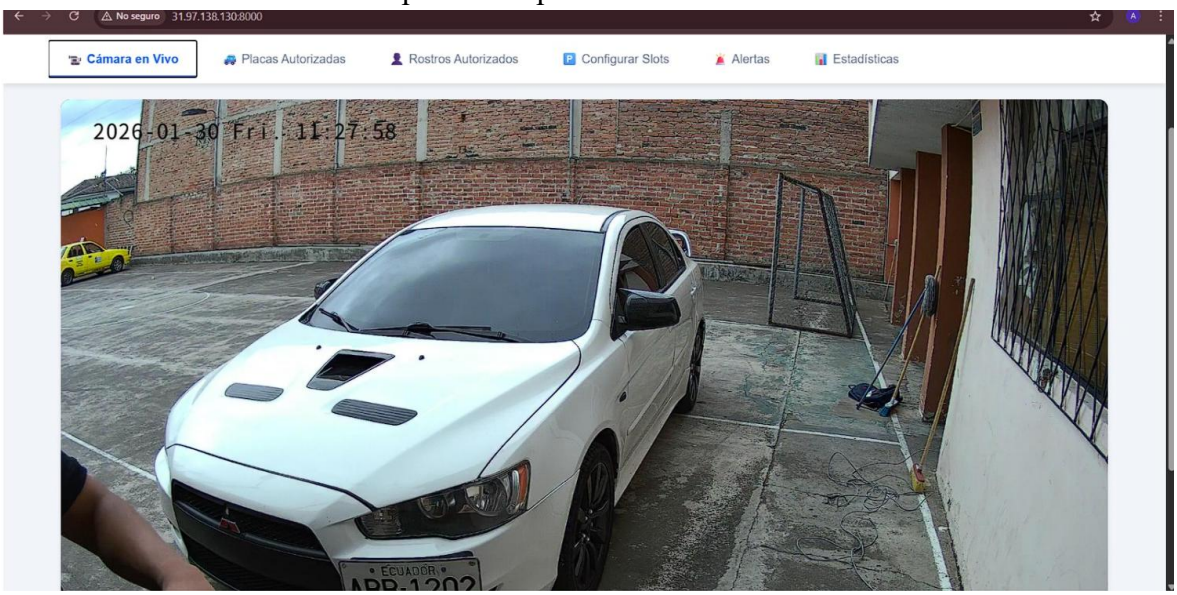
[21] *Sistema inteligente de estacionamiento en Chipichape: caso de estudio,* 2024.

[22] F. Reyes-Salazar, IoT and new technologies to improve smart parking efficiency: A systematic review, 2024.

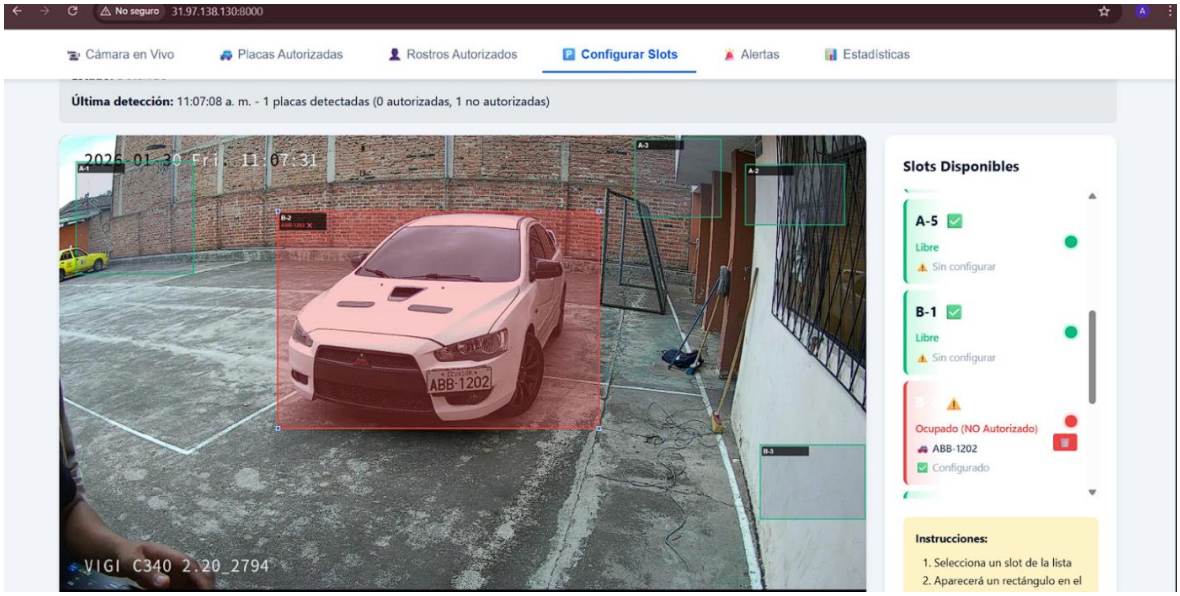
ANEXOS

Anexo 1: Instalación de la cámara IP en Parqueadero “Señor del Buen Suceso”

Anexo 2: Evidencia de la vista previa del panel web



Anexo 3: Evidencia de reconocimiento de placas vehiculares con una no autorizada



Anexo 4: Script de la Base de datos (SQL para creación de esquemas y tablas)

```
CREATE TABLE `authorized_plates_mobile` (
  `id` int(11) NOT NULL,
  `plate_number` varchar(20) NOT NULL,
  `owner_email` varchar(100) NOT NULL,
  `owner_name` varchar(100) NOT NULL,
  `is_active` tinyint(1) DEFAULT 1,
  `created_at` timestamp NULL DEFAULT current_timestamp(),
   `updated_at` timestamp NULL DEFAULT current_timestamp() ON UPDATE current_timestamp()
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

CREATE TABLE `notification_settings` (
  `id` int(11) NOT NULL,
  `user_id` int(11) NOT NULL,
  `intrusion_alerts` tinyint(1) DEFAULT 1,
  `email_notifications` tinyint(1) DEFAULT 1,
  `push_notifications` tinyint(1) DEFAULT 1
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

CREATE TABLE `parking_slots` (
  `slot_id` varchar(10) NOT NULL,
  `zone` varchar(1) NOT NULL,
  `is_occupied` tinyint(1) DEFAULT 0,
  `plate_number` varchar(20) DEFAULT NULL,
  `occupied_since` datetime DEFAULT NULL,
```

```
  `last_update` timestamp NULL DEFAULT current_timestamp() ON UPDATE current_timestamp()
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

CREATE TABLE `users` (
  `id` int(11) NOT NULL,
  `email` varchar(100) NOT NULL,
  `fcm_token` varchar(255) DEFAULT NULL,
  `fcm_platform` varchar(20) DEFAULT NULL,
  `password` varchar(255) NOT NULL,
  `first_name` varchar(50) NOT NULL,
  `last_name` varchar(50) NOT NULL,
  `verification_code` varchar(6) DEFAULT NULL,
  `is_verified` tinyint(1) DEFAULT 0,
  `reset_code` varchar(6) DEFAULT NULL,
  `reset_code_expires` datetime DEFAULT NULL,
  `created_at` timestamp NULL DEFAULT current_timestamp()
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

CREATE TABLE `user_alerts` (
  `id` int(11) NOT NULL,
  `user_id` int(11) NOT NULL,
  `alert_type` enum('intrusion_rostro','intrusion_placa') NOT NULL,
  `description` text DEFAULT NULL,
  `image_path` text DEFAULT NULL,
  `plate_number` varchar(20) DEFAULT NULL,
  `slot_id` varchar(10) DEFAULT NULL,
  `is_read` tinyint(1) DEFAULT 0,
  `created_at` timestamp NULL DEFAULT current_timestamp()
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

CREATE TABLE `user_faces` (
  `id` int(11) NOT NULL,
  `user_id` int(11) NOT NULL,
  `face_image` longtext NOT NULL,
  `created_at` timestamp NULL DEFAULT current_timestamp()
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;


CREATE TABLE `user_plates` (
  `id` int(11) NOT NULL,
  `user_id` int(11) NOT NULL,
  `plate_number` varchar(20) NOT NULL,
  `is_active` tinyint(1) DEFAULT 1,
```

```
  `created_at` timestamp NULL DEFAULT current_timestamp()
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

CREATE TABLE `user_tokens` (
  `id` int(11) NOT NULL,
  `user_id` int(11) NOT NULL,
  `token` varchar(255) NOT NULL,
  `device_info` text DEFAULT NULL,
  `fcm_token` varchar(255) DEFAULT NULL,
  `expires_at` datetime NOT NULL,
  `created_at` timestamp NULL DEFAULT current_timestamp()
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
```