Escuela Profesional de Ingeniería de Sistemas e Informática

HERRAMIENTAS DE DESARROLLO

“SISTEMA PARA LA GESTIÓN AUTOMATIZADA DEL PROCESO DE ESTACIONAMIENTO EN PARQUEOS EMPRESARIALES”

Autor(es):

SEMINARIO MEDINA, ALEJANDRO VALENTINO (0009-0006-5378-9115)

YOVERA VILCHEZ, BRYAN JOEL (0009-0007-2190-3857)

CABREDO PALACIOS, GRECIA (0009-0004-5647-6574)

DELGADO MENA SERGIO FABIAN (0009-0002-6609-3951)

Piura– Perú

2025

Índice

Índice de Imágenes

Indicé de tablas

CAPÍTULO I: DESCRIPCIÓN DE LA SITUACIÓN PROBLEMA DEL PROYECTO

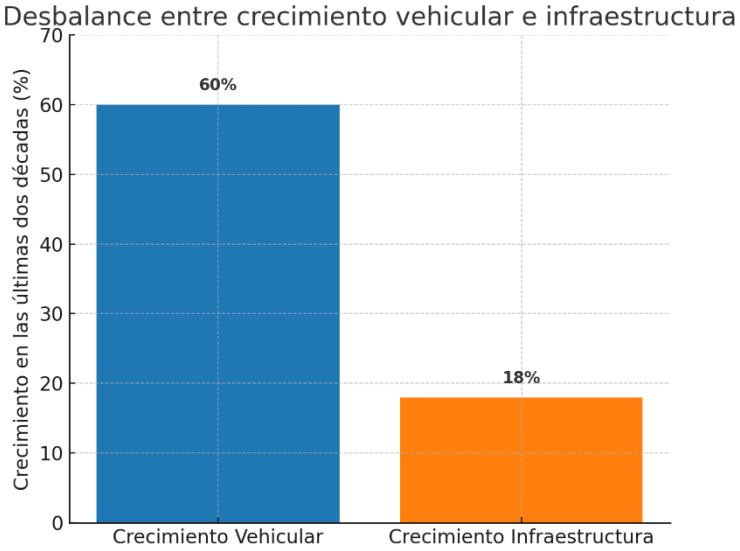
1.1. Contexto global

En la última década, la gestión de estacionamientos se ha consolidado como un desafío crítico para la movilidad urbana a escala mundial. Diversos estudios estiman que hasta un 30 % del tráfico urbano corresponde a conductores que circulan en busca de espacios de estacionamiento, lo que genera congestión, incremento de emisiones contaminantes y pérdidas de productividad (Donald Shoup, 2021; arXiv:2103.04515). Este fenómeno representa un obstáculo para la eficiencia operativa de las ciudades y para la sostenibilidad de sus sistemas de transporte.

1.2. Contexto regional (América Latina)

En el ámbito latinoamericano, la situación adquiere mayor complejidad por la falta de planificación urbana y la debilidad de los sistemas de transporte público, factores que han incrementado la dependencia del vehículo privado. Entre 2000 y 2020, el parque automotor en grandes ciudades de la región creció más de 60 %, mientras que la capacidad de infraestructura de estacionamientos aumentó menos de 20 % (World Bank, 2019; ITDP, 2020). Esta asimetría ha provocado saturación crónica de los estacionamientos empresariales y universitarios, generando pérdidas económicas, tiempos de espera prolongados y congestión interna (ResearchGate: López et al., 2020).

Figura 1. Desbalance entre crecimiento vehicular e infraestructura de estacionamientos en entornos urbanos latinoamericanos.



Nota: Elaboración propia.

1.3. Contexto local (Perú)

En el caso del Perú, el problema se manifiesta con particular intensidad en zonas empresariales de alta concentración vehicular, como Lima Metropolitana, donde entre 2010 y 2022 el parque automotor creció en más de 85 %, mientras que la infraestructura formal de estacionamientos apenas aumentó en torno al 18 % (Ministerio de Transportes y Comunicaciones del Perú, 2023). Esta brecha ha derivado en colapsos recurrentes en horas punta en distritos corporativos como San Isidro, Miraflores y Surco, donde se reportan tiempos de búsqueda de estacionamiento de hasta 15 minutos y demoras de ingreso/salida que superan los 10 minutos en promedio (ATU, 2022).

Asimismo, un estudio del Banco Mundial (2019) reveló que más del 45 % de los estacionamientos empresariales en Lima no cuenta con sistemas automatizados de control de accesos ni con plataformas de monitoreo de ocupación, lo que conlleva a congestión interna, sobrecostos operativos, menor rotación de espacios y exposición a accesos no autorizados.

La ausencia de soluciones tecnológicas integradas y en tiempo real obliga a los conductores a recorrer repetidamente las áreas internas de los parqueos, generando consumo innecesario de combustible, incremento de emisiones

y pérdidas de productividad, especialmente en horas de alta demanda. Esta situación evidencia una brecha crítica en la adopción de tecnologías inteligentes para la gestión de estacionamientos en el entorno empresarial peruano.

1.4. Situación problema concreta (método AQP)

En los parqueos empresariales de entornos corporativos peruanos, los conductores enfrentan tiempos de espera prolongados y dificultades constantes para localizar espacios disponibles, debido a la ausencia de sistemas automatizados de monitoreo de ocupación, lo cual genera congestión interna, ineficiencia operativa y aumento de los costos de operación.

Es decir, actualmente no existen mecanismos tecnológicos que permitan detectar en tiempo real si un espacio está ocupado o no, lo que obliga a los usuarios a recorrer repetidamente las áreas internas del estacionamiento hasta encontrar una plaza libre.

1.5. Causas identificadas

Las principales causas que sustentan esta situación son:

- Procesos: dependencia de supervisión manual y ausencia de estandarización operativa.
- Tecnología: carencia de sensores y de sistemas de visión por computadora para detectar ocupación.
- Recursos humanos: necesidad de personal constante en accesos, incrementando costos fijos.
- Infraestructura: diseño sin segmentación ni delimitación de zonas, que impide aplicar analítica de ocupación.
- Medición: falta de información en tiempo real y de módulos de analítica predictiva.

Estas causas evidencian que el problema no radica exclusivamente en la cantidad de espacios disponibles, sino en la ausencia de herramientas inteligentes de gestión.

1.6. Consecuencias actuales y potenciales

La persistencia de este problema genera consecuencias de alto impacto organizacional y social:

- Aumento sostenido de los tiempos de espera y circulación interna.
- Pérdida de productividad de los trabajadores por demoras en el ingreso y salida.
- Incremento del consumo de combustible y de las emisiones contaminantes.
- Riesgo de incidentes viales dentro de las instalaciones por congestión.
- Reducción de la competitividad de las organizaciones por ineficiencias operativas.
- Dificultad para implementar mecanismos de auditoría y seguridad, favoreciendo accesos no autorizados.

1.7. Aporte propuesto

Para mitigar la problemática descrita, se propone el desarrollo de un Sistema para la Gestión Automatizada del Proceso de Estacionamiento en Parqueos Empresariales, el cual emplea visión por computadora mediante YOLOv11 y OpenCV, junto con pruebas de intersección en polígonos (pointer test), para determinar en tiempo real si un área de estacionamiento está ocupada o libre.

Este sistema permitirá:

- Automatizar la detección de ocupación sin intervención humana.
- Proveer información en tiempo real sobre disponibilidad de plazas.
- Reducir la congestión interna y los tiempos de espera.
- Integrarse en arquitecturas empresariales modulares y escalables.

- Facilitar la futura incorporación de analítica predictiva para anticipar la demanda.

En consecuencia, esta propuesta contribuirá a optimizar el uso de la infraestructura existente, disminuir costos operativos y mejorar la sostenibilidad de los entornos corporativos.

CAPÍTULO II: ANTECEDENTES Y BASES TEÓRICAS

2.1. Antecedentes

2.1.1. Antecedentes Internacionales

La literatura internacional muestra que la combinación de sensórica y visión por computadora eleva la eficacia del control vehicular y la detección de ocupación. En *Sensors*, Ullah et al. (2019, p. 3017) presentaron una plataforma que integra sensores ultrasónicos con reconocimiento automático de matrículas (LPR/ANPR) y reportaron una precisión cercana al 98 % en la detección de vehículos. Este resultado sugiere que la percepción multimodal reduce falsos positivos y mejora la confiabilidad del acceso; no obstante, el estudio se desarrolló en condiciones controladas, por lo que su extrapolación a entornos corporativos exige validar desempeño bajo variabilidad de iluminación, ángulos de captura y flujos pico.

En el plano de mercado e infraestructura, un análisis de prefactibilidad en Medellín estimó un valor de USD 1 534 millones para el segmento de estacionamientos automatizados en 2021 y una tasa de crecimiento anual proyectada de 16,96 % hasta 2027 (Arzayus Henao, 2024, p. 33). Esta evidencia respalda la existencia de demanda y el potencial de escalamiento regional de soluciones automatizadas. Con todo, se trata de proyecciones sujetas a supuestos macroeconómicos y a un marco regulatorio local específico, de modo que su traslación a contextos empresariales requiere ajustar escenarios de inversión y riesgo según políticas de cada recinto.

Desde la perspectiva operativa, la Universidad Internacional SEK reportó un prototipo IoT con sensórica y mensajería ligera que incrementó en 14 % la rapidez para localizar espacios libres, mostrando que la telemetría en tiempo real acorta la búsqueda y mejora la visibilidad del patio (Hidrobo Moreno, 2024, p. 45). El aporte

es consistente con la necesidad de información fiable para decisiones inmediatas; sin embargo, al tratarse de un despliegue en entorno universitario y de alcance acotado, persiste la necesidad de evaluar su rendimiento con cargas concurrentes mayores, políticas de seguridad más estrictas y requisitos de auditoría típicos de parqueos empresariales.

Finalmente, estudios recientes describen arquitecturas de microservicios integradas con IoT y *machine learning* para fortalecer el control de accesos y la resiliencia del sistema. En *Metanoia*, Pillo Guanoluisa et al. (2025, p. 242) reportaron tasas de registro superiores al 99 % y una reducción de vulnerabilidades frente a enfoques tradicionales, lo que sugiere beneficios del desacoplamiento funcional y del procesamiento distribuido. No obstante, la evidencia se concentra en el módulo de acceso; en consecuencia, sigue abierta la brecha de integración end-to-end con trazabilidad, auditoría y gobierno de datos a nivel corporativo, condición indispensable para una gestión integral del estacionamiento.

2.1.2. Antecedentes Nacionales

El análisis de antecedentes nacionales permite comprender los avances realizados en torno a la automatización de los procesos de estacionamiento en el Perú. Diversas investigaciones han abordado esta problemática, mostrando la evolución progresiva de soluciones tecnológicas aplicadas en este ámbito.

En primer lugar, **Espinoza Landa (2020)**, en su estudio *“Impacto de automatizar el acceso vehicular de estacionamientos privados”*, evidenció que la incorporación de barreras electrónicas y sistemas de control vehicular contribuye significativamente a la reducción de los tiempos de espera y al incremento de la seguridad. Este trabajo constituye un referente importante, ya que demuestra cómo la modernización de los accesos puede mejorar la eficiencia operativa

de los parqueos privados, sentando las bases para propuestas más complejas en el ámbito empresarial.

Por otro lado, Ríos Vidalón (2011) desarrolló un sistema de control vehicular para estacionamientos mediante tecnología RFID. Su propuesta permitió gestionar de manera integral el acceso de vehículos, el monitoreo de la disponibilidad de espacios y la supervisión en tiempo real. Este antecedente resalta la factibilidad de implementar tecnologías inteligentes en la gestión de parqueos, aportando evidencia de la utilidad de la digitalización en la optimización de procesos logísticos y de seguridad.

Finalmente, se observa que los aportes mencionados han transitado desde la automatización básica de accesos hasta la incorporación de tecnologías inteligentes orientadas a la gestión integral. En conclusión, estos antecedentes revelan una clara tendencia hacia la digitalización y modernización de los estacionamientos en el Perú, lo cual respalda la pertinencia de desarrollar un sistema automatizado de gestión para parqueos empresariales que responda a las nuevas demandas de eficiencia, seguridad y control.

2.1.3. Antecedentes Locales

En la investigación de Castillo Mogollón, Enrique y Santamaría Montero, Abner Manuel (2023), titulada "Control de acceso vehicular con visión artificial para urbanizaciones en la ciudad de Piura, 2023", se abordó la optimización de la gestión vehicular. El objetivo principal fue determinar cómo la implementación de un sistema de visión artificial podría mejorar el control de vehículos, buscando específicamente reducir los tiempos de registro, aumentar la satisfacción de los usuarios y asegurar una alta efectividad en el reconocimiento de matrículas y en el envío de notificaciones.

La metodología empleada se basó en un enfoque cuantitativo y de tipo aplicada, utilizando un diseño preexperimental para medir el

impacto de la solución tecnológica. Este diseño implicó la recolección de datos antes y después de la implementación del sistema para cuantificar objetivamente las mejoras. La población y muestra del estudio abarcaron el flujo total de vehículos que ingresaban y salían de la urbanización, lo que garantizó que la validación se realizara en un entorno operativo real y no en condiciones de laboratorio.

Los resultados obtenidos fueron notablemente positivos, demostrando una reducción significativa en los tiempos de registro vehicular tanto en el ingreso como en la salida. Este avance se tradujo directamente en un aumento considerable en el nivel de satisfacción de las personas. Además, el sistema alcanzó una efectividad del 100% en el reconocimiento de matrículas y en la entrega de notificaciones, estableciendo un alto estándar de rendimiento para este tipo de tecnología en el contexto de Piura.

La conclusión principal del estudio fue que el sistema web con visión artificial mejoró de manera efectiva el control de acceso vehicular. Esta afirmación, respaldada por los datos cuantitativos, sirve como una prueba de concepto fundamental que valida la viabilidad y eficacia de la visión artificial como herramienta para modernizar y optimizar la gestión de vehículos en la región.

El aporte de esta tesis es triple: primero, valida la solución al demostrar que la visión artificial responde a una necesidad real de eficiencia en la gestión vehicular en Piura. Segundo, establece un benchmark de rendimiento con su 100% de efectividad, que sirve como un estándar de calidad para futuros proyectos. Finalmente, proporciona una guía metodológica probada (el diseño preexperimental) que puede ser replicada para medir el impacto de implementaciones tecnológicas similares de manera rigurosa.

2.2. Bases Teóricas

2.2.1. Fundamentos de la Gestión de Estacionamientos

La gestión de estacionamientos se conceptualiza como el conjunto de políticas, procesos y herramientas destinadas a equilibrar la relación oferta–demanda de plazas, reducir tiempos improductivos y mitigar externalidades urbanas y ambientales. La literatura especializada muestra que la transición desde esquemas manuales a sistemas inteligentes depende de la disponibilidad de información operativa y de mecanismos de control en tiempo real, así como de criterios de ordenamiento del suelo y de tarificación que incentiven el uso eficiente de la infraestructura (Hurtado Cortes, 2022; Santos Villalba, 2021). En términos técnicos, los “sistemas de estacionamiento” constituyen una familia de soluciones donde confluyen diseño físico, control de accesos y soporte informático, por lo que su desempeño es inseparable de la arquitectura tecnológica subyacente y del contexto normativo en que operan (Calle Müller, 2014; Fernández Herrera & Saldivar Bazán, 2023).

2.2.2. Visión por Computadora aplicada al Tránsito y Estacionamientos

La Visión por Computadora (VC) es una disciplina de la inteligencia artificial que busca dotar a los sistemas de la capacidad de interpretar y extraer información de imágenes y videos, con aplicaciones que abarcan desde la medicina hasta la robótica (Szeliski, 2022). En el ámbito del transporte inteligente, la VC se ha consolidado como un pilar tecnológico para monitorear la movilidad urbana, optimizar recursos de infraestructura y garantizar la seguridad en la vía pública (Zheng & Xie, 2023).

En el nivel intermedio, la aplicación de la VC al tránsito vehicular se centra en tareas como la detección y seguimiento de vehículos, el reconocimiento de señales de tráfico y la estimación de la densidad vehicular. Estas funciones permiten construir sistemas de transporte inteligentes (ITS) que reducen la congestión y facilitan la toma de decisiones basada en datos (Papageorgiou et al., 2020).

En el nivel específico, la VC aplicada a los estacionamientos inteligentes busca identificar en tiempo real la ocupación de plazas, reconocer matrículas vehiculares (Automatic License Plate Recognition, ALPR) y predecir patrones de uso. Grbić y Koch (2023) proponen un sistema de detección de espacios de estacionamiento basado en redes convolucionales profundas ligeras, optimizadas para dispositivos en el borde, mostrando que es posible alcanzar un equilibrio entre precisión y eficiencia computacional. Mohammadian et al. (2023), en una revisión sistemática, destacan que arquitecturas basadas en YOLO y SSD superan ampliamente a los métodos tradicionales de procesamiento de imágenes en tareas de detección de ocupación.

En el campo de ALPR, Laroca et al. (2018) presentan un método robusto que combina YOLO para la localización de placas con un sistema de reconocimiento de caracteres basado en Connectionist Temporal Classification (CTC), logrando resultados reproducibles en escenarios abiertos. Más recientemente, enfoques híbridos con Vision Transformers (ViT) han demostrado mejorar la robustez frente a variaciones de iluminación y oclusión (Dosovitskiy et al., 2021; Khan et al., 2023).

La evaluación de estos modelos combina métricas de detección y reconocimiento, como mean Average Precision (mAP) y Character Error Rate (CER), junto con indicadores de desempeño en producción, como latencia de inferencia, FPS y consumo energético, factores determinantes en su despliegue en edge devices y entornos de alta concurrencia (Zheng & Xie, 2023).

Finalmente, la integración de VC en sistemas de estacionamiento inteligente se articula con principios de smart cities, donde la optimización de espacios contribuye a reducir tiempos de búsqueda de estacionamiento, minimizar congestión vehicular y disminuir emisiones contaminantes. No obstante, persisten desafíos

relacionados con la privacidad de datos en ALPR, la escalabilidad en infraestructuras urbanas masivas y la robustez frente a condiciones ambientales adversas, que constituyen líneas de investigación activa y necesaria para la consolidación de estas tecnologías (Gao et al., 2022).

2.2.3. Arquitectura de Microservicios

La evolución de la arquitectura de software ha transitado desde sistemas monolíticos hacia paradigmas distribuidos, pasando por la Arquitectura Orientada a Servicios (SOA) y desembocando en los microservicios, cuyo objetivo principal es desacoplar componentes para mejorar la escalabilidad, mantenibilidad y resiliencia (Newman, 2022). A diferencia de los monolitos, donde todos los módulos comparten una misma base de código y despliegue, los microservicios proponen una descomposición modular extrema, en la que cada servicio implementa una función específica con independencia de desarrollo y operación.

En un nivel intermedio, los microservicios se apoyan en patrones arquitectónicos estandarizados. Richardson (2019) identifica principios como API Gateway, que centraliza la comunicación externa; Database per Service, que garantiza independencia de datos; y Saga, que permite la coordinación de transacciones distribuidas. Estos patrones habilitan la escalabilidad horizontal y reducen los riesgos de fallas sistémicas. La literatura reciente destaca, además, la adopción de arquitecturas orientadas a eventos (event-driven) y de service mesh (e.g., Istio, Linkerd) para resolver desafíos de comunicación segura, balanceo de carga y observabilidad en entornos de gran escala (Burns & Oppenheimer, 2019; Dragoni et al., 2017).

En el nivel específico, la aplicación de microservicios en sistemas inteligentes de estacionamiento permite desacoplar el procesamiento de visión por computadora, la gestión de accesos y el manejo de pagos electrónicos. Por ejemplo, el procesamiento de imágenes

puede ejecutarse en edge nodes cercanos a las cámaras, mientras que el backend central gestiona analítica y trazabilidad mediante colas de eventos (Kafka, RabbitMQ) y protocolos de integración asíncrona. Esta arquitectura favorece la elasticidad del sistema, permitiendo que los módulos más intensivos —como los de detección vehicular— escalen de manera independiente al resto (Villamizar et al., 2017).

La gobernanza de microservicios introduce retos adicionales. Entre ellos destacan: la observabilidad distribuida (telemetría, logging y métricas en múltiples nodos), la consistencia eventual en bases de datos (debido a la fragmentación de dominios de datos), la latencia inter-servicios y los riesgos de deuda técnica asociados a un exceso de fragmentación (Fowler, 2020). Para abordar estos retos, se emplean métricas como Mean Time to Recovery (MTTR), throughput, latencia promedio y cumplimiento de Service-Level Agreements (SLA) como mecanismos de evaluación del rendimiento de la arquitectura.

En conclusión, la arquitectura de microservicios no solo proporciona resiliencia y escalabilidad para sistemas de estacionamiento inteligente, sino que también posibilita la integración con entornos cloud-native y edge computing, garantizando un marco flexible y sostenible para el despliegue de módulos de visión por computadora y analítica predictiva. Sin embargo, su éxito depende de una adecuada orquestación tecnológica y de la mitigación proactiva de riesgos de complejidad, seguridad y comunicación distribuida.

2.2.4. Contenerización y Orquestación

La contenerización ha transformado los procesos de despliegue y operación de software al posibilitar la ejecución de aplicaciones en entornos ligeros, aislados y portables denominados contenedores. A diferencia de la virtualización tradicional basada en máquinas virtuales, los contenedores comparten el núcleo del sistema operativo, lo que reduce significativamente la sobrecarga de recursos y acelera los tiempos de despliegue (Merkel, 2014). En la literatura reciente,

Pahl y Lee (2019) destacan que esta tecnología constituye un elemento central para arquitecturas nativas en la nube (cloud-native), al favorecer la consistencia entre entornos de desarrollo, prueba y producción.

En paralelo, la orquestación de contenedores surge como necesidad operativa en escenarios donde múltiples servicios deben ejecutarse y coordinarse de manera simultánea. Plataformas como Kubernetes, introducidas por Google y posteriormente adoptadas como estándar de facto en la industria, proporcionan mecanismos automatizados para el escalado dinámico de servicios, la distribución de cargas de trabajo, la recuperación ante fallos y la gestión de redes y almacenamiento (Burns et al., 2016). Según Villamizar et al. (2017), estas capacidades permiten alcanzar niveles superiores de disponibilidad y elasticidad en comparación con despliegues monolíticos o virtualizados.

El binomio contenerización-orquestación se ha consolidado como paradigma técnico esencial para arquitecturas basadas en microservicios, dado que cada componente funcional puede empaquetarse, desplegarse y actualizarse de manera independiente, minimizando riesgos de acoplamiento y mejorando la escalabilidad del sistema (Zhang et al., 2021). Adicionalmente, estudios recientes subrayan su integración con prácticas de DevOps y MLOps, donde los contenedores actúan como unidades reproducibles para experimentación, entrenamiento y despliegue de modelos de aprendizaje automático (Kreuzberger et al., 2023).

En el caso de los sistemas inteligentes de gestión de estacionamientos, la aplicación de estas tecnologías resulta estratégica. Por un lado, la contenerización permite encapsular módulos críticos como reconocimiento de matrículas, control de accesos, procesamiento de pagos y predicción de demanda, asegurando su portabilidad y consistencia en entornos empresariales

heterogéneos. Por otro lado, la orquestación mediante Kubernetes u OpenShift habilita el escalado automático en horarios de alta demanda —por ejemplo, durante horas pico de ingreso y salida vehicular— garantizando baja latencia en la detección y respuesta del sistema. Esta elasticidad operacional, sumada a la capacidad de recuperación ante fallos, asegura la continuidad del servicio en infraestructuras críticas, lo cual constituye un requisito indispensable en contextos de movilidad empresarial.

En síntesis, la contenerización y la orquestación no solo representan avances técnicos en la ingeniería de software distribuido, sino que constituyen pilares fundamentales para el diseño de sistemas de estacionamiento inteligentes, al habilitar modularidad, escalabilidad, resiliencia y sostenibilidad operativa en entornos altamente dinámicos.

2.2.5. Control de Versiones y Colaboración Distribuida

2.2.5.1. Control de versiones

“Proporcionando herramientas para la fusión y generación de una nueva versión de un proyecto, permitiendo que múltiples desarrolladores trabajen en el mismo proyecto sin ocasionar pérdida de datos o bloqueos de archivos” (Tello, Sosa & Leal, 2012, p.1).

“Existe todavía un alto grado de posibilidad que, al hacerlo, un archivo pueda estropear indirectamente otras cosas, pero que no sea tan evidente” (Wanumen Silva, 2008, p.9).

“Se llama control de versiones a los métodos y herramientas disponibles para controlar todo lo referente a los cambios en el tiempo de un archivo” (Borrell, 2006, p.1).

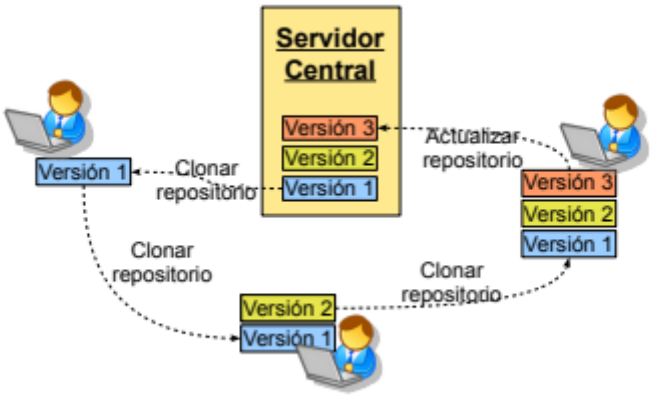
2.2.5.2. Control de versiones distribuido

“En el SCV distribuido cada desarrollador realiza una copia del repositorio de proyectos completo a su computadora, generándose un repositorio local del proyecto” (Tello, Sosa & Leal, 2012, p.5).

“No es necesario mantener una conexión de red permanente a un servidor que contenga el repositorio de referencia, las conexiones de red por parte de los desarrolladores solo se requieren cuando se realiza el proceso de sincronización” (Tello, Sosa & Leal, 2012, p.5).

Figura 1

Sistema de control de versiones distribuido



Fuente: Otero Gutiérrez, D. (2011, figura 2)

2.2.5.3. Flujo de trabajo

El concepto de "Flujo de Trabajo" o workflow es un pilar en la gestión de procesos de negocio (BPM) y en la ingeniería de software. Representa la formalización y automatización de una secuencia de tareas, reglas y procedimientos que, en conjunto, logran un objetivo empresarial específico.

Un flujo de trabajo se define como la organización sistemática de recursos para construir procesos que transforman materiales, prestan servicios o procesan información. Es una secuencia de operaciones realizadas por una persona, una organización o varios mecanismos (VanZandt, 2023).

Un workflow es un sistema para gestionar procesos y tareas repetitivos que ocurren en un orden particular. Es el mecanismo a través del cual las personas y las empresas realizan su trabajo, ya sea fabricando un producto o procesando información (IBM, s.f.).

Un sistema de workflow debe ser capaz de gestionar tres aspectos fundamentales: el diseño y definición del flujo de trabajo, la instanciación y control de su ejecución, y la interacción con usuarios, aplicaciones y otros recursos (Hollingsworth, 1995, citado en EvaluandoERP, s.f.).

2.2.5.4. GitFlow

GitFlow no es simplemente un conjunto de comandos de Git, sino una metodología prescriptiva y un modelo de ramificación estricto diseñado para gestionar el ciclo de vida del desarrollo de software en proyectos con un esquema de lanzamientos versionados. Fue propuesto por Vincent Driessen en 2010 como una estrategia para aportar orden y predictibilidad a repositorios complejos.

Gitflow es un flujo de trabajo de Git heredado que ha perdido popularidad en favor de los flujos de trabajo basados en troncos, que ahora se consideran prácticas recomendadas para el desarrollo continuo de software y las prácticas de DevOps (Atlassian, 2025).

Git flow es una popular estrategia de ramificación de Git destinada a simplificar la gestión de lanzamientos. Fue introducido por el desarrollador de software Vincent Driessen en 2010 e implica fundamentalmente aislar el trabajo en diferentes tipos de ramas (GitKraken, s.f.).

El modelo de desarrollo que presento es esencialmente un conjunto de procedimientos que cada miembro del equipo debe seguir para llegar a un proceso de desarrollo de software gestionado. Este modelo define un estricto esquema de ramificación (Driessen, 2010).

2.2.5.5. Trazabilidad del código

La trazabilidad del código, en el contexto de la ingeniería de software, se define como la capacidad de crear, mantener y analizar vínculos bidireccionales entre todos los artefactos

generados a lo largo del ciclo de vida del desarrollo de software (SDLC).

La trazabilidad del código en la ingeniería de software es la capacidad de vincular y rastrear artefactos a lo largo del ciclo de vida del desarrollo. Permite a los interesados comprender las relaciones entre requisitos, diseño, casos de prueba y código (EN-COM, 2024).

La norma ISO 9000:2015 define trazabilidad como la "capacidad para seguir el histórico, la aplicación o la localización de un objeto", lo que puede relacionarse con el origen de los materiales, el histórico del proceso y su distribución (de Gallo & Leone, 2016).

Un análisis de trazabilidad del código fuente es una herramienta importante para verificar que todo el código esté vinculado a las especificaciones y procedimientos de prueba establecidos, asegurando que cada elemento del diseño de software ha sido implementado (FDA, 2002).

2.2.5.6. Auditoría del código

La auditoría de código es un proceso sistemático y exhaustivo de revisión del código fuente de una aplicación con el objetivo principal de descubrir vulnerabilidades de seguridad, errores de lógica, defectos de calidad y violaciones de estándares de codificación antes de que el software sea desplegado en un entorno de producción.

Una revisión de código segura permite a los desarrolladores identificar y abordar vulnerabilidades en las primeras etapas del proceso de desarrollo, antes de que se vuelvan costosas o complejas de solucionar, mejorando activamente la postura de seguridad de las aplicaciones (Checkmarx, 2025).

El análisis estático de código consiste en examinar el código fuente de la aplicación sin ejecutarlo, con el fin de detectar posibles vulnerabilidades o errores de seguridad que podrían

comprometer su funcionamiento (Sreenivasa & Kuman, 2012, citado en Redalyc, 2015).

Una revisión o auditoría de código investiga las prácticas de codificación utilizadas en la aplicación. El objetivo principal de dichas revisiones es descubrir defectos de seguridad e identificar potencialmente soluciones para ellos (NIST, s.f.).

2.2.6. Prácticas de DevOps y Automatización

2.2.1.1. Definición e integración entre desarrollo y operaciones.

El enfoque *DevOps* integra de manera colaborativa las áreas de desarrollo de software (*Dev*) y operaciones (*Ops*), con el objetivo de optimizar la comunicación, reducir la incidencia de errores y acelerar la entrega de valor al usuario. Además, promueve una cultura organizacional orientada a la cooperación continua, donde los equipos trabajan alineados para alcanzar ciclos de desarrollo más ágiles (Ebert et al., 2016).

Desde una perspectiva práctica, DevOps funciona como un puente entre el desarrollo y las operaciones, al impulsar flujos de trabajo automatizados que posibilitan la innovación constante sin comprometer la estabilidad del sistema (Kim, Behr & Spafford, 2016).

Asimismo, esta metodología no se limita al aspecto técnico, sino que también implica una transformación cultural, basada en la confianza y la responsabilidad compartida a lo largo del ciclo de vida del software, lo que incrementa la calidad de las entregas (Humble & Farley, 2010).

En el contexto de un sistema de gestión automatizada de estacionamientos empresariales, DevOps asegura que las mejoras, correcciones o nuevas funcionalidades se implementen de manera ágil y continua, garantizando la

disponibilidad del servicio y evitando interrupciones en el acceso vehicular.

2.2.1.2. CI/CD: pruebas automatizadas, integración y despliegue continuos.

La Integración Continua y Despliegue Continuo (CI/CD) constituye una práctica esencial para reducir el tiempo de entrega de valor, al permitir la detección temprana de errores y una retroalimentación constante en los proyectos de software (Leite et al., 2019).

CI/CD automatiza los procesos de construcción, validación y despliegue, de modo que cada modificación en el código atraviesa pruebas automatizadas que aseguran estabilidad y confiabilidad antes de llegar al entorno de producción (Shahin, Ali & Zhu, 2017). Además, se ha demostrado que su implementación contribuye a reducir costos operativos y a escalar sistemas de forma confiable, con las pruebas automatizadas como componente fundamental (Fowler, 2006).

En los estacionamientos empresariales, esta práctica permite actualizar módulos críticos como control de accesos, pagos electrónicos o monitoreo en tiempo real sin necesidad de suspender operaciones.

2.2.1.3. Infraestructura como código (IaC).

La Infraestructura como Código (IaC) se ha consolidado como una estrategia que acelera significativamente el despliegue de entornos, permitiendo su replicación en cuestión de minutos y garantizando un mayor control sobre los cambios realizados (Yussupov et al., 2019).

IaC consiste en administrar servidores, redes y recursos tecnológicos mediante archivos de configuración en lugar de

procedimientos manuales, lo que asegura consistencia, fomenta la replicación de entornos y disminuye los errores humanos (Morris, 2016). Asimismo, contribuye a la estandarización de configuraciones en proyectos complejos, al integrarse en repositorios de control de versiones como parte del ciclo de desarrollo (Humble & Farley, 2010).

En un sistema de estacionamiento inteligente, IaC posibilita desplegar configuraciones homogéneas en sensores, cámaras de vigilancia y barreras electrónicas de forma rápida y segura.

2.2.1.4. Monitoreo, observabilidad y logging centralizado.

La observabilidad se ha convertido en un componente crítico en los sistemas distribuidos, ya que permite correlacionar eventos y anticipar fallas de manera proactiva (Wilkes, 2020). En este sentido, la combinación de métricas, trazas y registros ofrece una visión integral del sistema, reduciendo el tiempo medio de resolución de incidentes (*MTTR*) y mejorando la experiencia del usuario (Singh & Peddoju, 2019).

El monitoreo supervisa en tiempo real el rendimiento y la disponibilidad del sistema; la observabilidad permite comprender su comportamiento interno; y el *logging* centralizado concentra todos los registros en un repositorio único para su análisis (Turnbull, 2014).

En los sistemas de parqueo empresarial, estas prácticas son esenciales para detectar fallas tempranas, garantizar continuidad operativa y mantener trazabilidad ante incidentes de seguridad o problemas técnicos.

2.2.7. Cloud Computing en Sistemas Inteligentes

2.2.1.5. Principales proveedores: AWS, GCP, Azure.

De acuerdo con Buyya et al. (2019), la computación en la nube se consolida como un modelo flexible que permite a las organizaciones gestionar recursos bajo un esquema de pago por uso, optimizando costos y disponibilidad.

Por otra parte, Marinescu (2017) resalta que estos proveedores han evolucionado hacia servicios especializados en inteligencia artificial, big data y seguridad, facilitando la adopción de tecnologías avanzadas en las empresas.

El Cloud Computing ofrece acceso bajo demanda a recursos computacionales a través de internet, eliminando la necesidad de infraestructura física local. Entre los principales proveedores destacan Amazon Web Services (AWS), Google Cloud Platform (GCP) y Microsoft Azure, los cuales brindan entornos altamente escalables, seguros y confiables (Armbrust et al., 2010).

2.2.1.6. Modelos de servicio: IaaS, PaaS, SaaS.

El *Cloud Computing* se organiza en tres modelos de servicio fundamentales:

Infraestructura como Servicio (IaaS): proporciona recursos de hardware virtualizados, como redes, servidores y almacenamiento. Ejemplos: Amazon EC2 o Google Compute Engine (Rittinghouse & Ransome, 2017).

Plataforma como Servicio (PaaS): ofrece entornos de desarrollo y despliegue preconfigurados, lo que agiliza la programación y reduce la complejidad técnica. Ejemplo: Google App Engine (Zhang, Cheng & Boutaba, 2010).

Software como Servicio (SaaS): entrega aplicaciones listas para usar a través de la web, sin necesidad de instalación local. Ejemplo: Microsoft Office 365.

En un estacionamiento inteligente, el SaaS permite a los usuarios reservar espacios vía aplicación, el PaaS facilita integrar sistemas de pago y el IaaS soporta la infraestructura que almacena y procesa la información de accesos vehiculares.

2.2.1.7. Balanceo de carga, escalabilidad elástica y despliegues híbridos.

Se enfatiza que estas capacidades son fundamentales en sistemas inteligentes, ya que permiten asegurar rendimiento óptimo incluso en situaciones de alta concurrencia (Buyya et al., 2019).

El balanceo de carga distribuye el tráfico entre múltiples servidores, la escalabilidad elástica ajusta dinámicamente los recursos según la demanda, y los despliegues híbridos combinan infraestructura local con la nube para garantizar redundancia y continuidad operativa (Marinescu, 2017).

De acuerdo con Vaquero et al. (2011), el balanceo y la elasticidad representan pilares en la nube, pues mejoran la resiliencia y aseguran la disponibilidad de los servicios en contextos de gran demanda.

En el contexto empresarial, un sistema de parqueo puede operar en una nube híbrida, donde los datos sensibles (usuarios, pagos) se gestionen en una nube privada, mientras que las aplicaciones móviles o paneles de consulta funcionen en la nube pública.

2.2.1.8. Ventajas para la gestión multi-sede de estacionamientos.

La computación en la nube permite centralizar información de accesos, pagos y disponibilidad en tiempo real, reduciendo la necesidad de infraestructuras locales costosas en cada sede. Esto facilita la gestión unificada de múltiples parqueos empresariales y mejora la eficiencia administrativa (Buyya et al., 2019).

A nivel más reciente, Gaurav et al. (2019) destacan que el modelo multi-sede optimiza la analítica de datos, lo que posibilita tomar decisiones estratégicas basadas en patrones de uso.

De forma complementaria, Armbrust et al. (2010) explican que la centralización es una de las ventajas más fuertes de la nube, ya que reduce redundancias y asegura la escalabilidad.

2.2.8. Bases de Datos Relacionales y Trazabilidad

La operación de parqueos empresariales exige una base relacional con garantías ACID que preserve consistencia bajo alta concurrencia y múltiples puntos de acceso. PostgreSQL resulta pertinente por su manejo de transacciones, bloqueos y restricciones declarativas (claves primarias y foráneas) que sostienen la integridad referencial entre entidades como vehículo–usuario, plaza–ocupación, reserva–acceso y eventos operativos. A ello se suma un ecosistema de auditoría que registra acciones a nivel de sesión u objeto, lo que permite reconstruir el ciclo de vida de cada evento con fines regulatorios y de control interno. En conjunto, estas capacidades fundamentan la trazabilidad operativa que demandan los entornos corporativos al vincular datos de identidad, reglas de acceso y bitácoras verificables en tiempo real (PostgreSQL Global Development Group, 2025; pgAudit Project, s. f.; Crunchy Data, 2020).

En escenarios multi-sede y de alta disponibilidad, PostgreSQL documenta replicación física y lógica, standbys en caliente y confirmación síncrona —incluido quórum— para equilibrar latencia, durabilidad y recuperación ante fallos. Las guías de arquitectura en la nube describen topologías con réplicas intra-zona y multi-región, failover automatizado y balanceo de lectura; su evaluación se apoya en retraso de réplica, RPO/RTO y porcentaje de confirmaciones síncronas, de modo que el diseño lógico se alinee con metas de continuidad del servicio. Así, la base transaccional deja de ser un mero repositorio y pasa a ser un componente de resiliencia que condiciona la disponibilidad del sistema y la calidad de la evidencia operacional (PostgreSQL Global Development Group, 2025; Google Cloud, 2024; Crunchy Data, 2020).

2.2.9. Seguridad en Sistemas Distribuidos

2.2.1.9. Autenticación y autorización: JWT, OAuth2.

En estudios más recientes, se señala que OAuth2 y JWT son fundamentales en la seguridad moderna, ya que permiten integración entre servicios y mayor control sobre los flujos de autenticación (Alshamrani, 2020).

Asimismo, Fernández et al. (2017) destacan que estas tecnologías contribuyen a la interoperabilidad entre aplicaciones y sistemas distribuidos, manteniendo altos estándares de seguridad.

La autenticación garantiza que solo usuarios legítimos accedan al sistema, mientras que la autorización determina los privilegios y permisos asociados. Tecnologías como JSON Web Tokens (JWT) y OAuth2 se utilizan ampliamente en entornos distribuidos por su capacidad de escalabilidad y confiabilidad (Hardt, 2012).

2.2.1.10. Criptografía, cifrado de datos y gestión de contraseñas.

Los algoritmos homomórficos, para garantizar privacidad en entornos distribuidos sin comprometer la usabilidad (Yadav & Mishra, 2021).

Por otro lado, Bishop (2018) enfatiza que una política sólida de gestión de contraseñas y almacenamiento cifrado reduce significativamente el riesgo de vulnerabilidades en aplicaciones críticas.

La criptografía protege información sensible tanto en tránsito como en almacenamiento, mientras que el cifrado de datos y la correcta gestión de contraseñas son prácticas fundamentales para proteger la identidad de los usuarios y las transacciones electrónicas (Stallings, 2017).

- Auditoría, cumplimiento normativo y gestión de riesgos.

  GDPR y la ISO 27001, han reforzado la necesidad de integrar la auditoría continua en los sistemas empresariales (Rantos et al., 2020).

La auditoría permite registrar y verificar las actividades del sistema, el cumplimiento normativo asegura alineación con leyes de protección de datos y la gestión de riesgos ayuda a identificar vulnerabilidades y mitigarlas (Bishop, 2018).

En investigaciones actuales, se señala que los marcos normativos Tanenbaum y Van Steen (2016) sostienen que la gestión de riesgos es una práctica clave para la sostenibilidad de sistemas distribuidos, ya que permite planificar medidas preventivas frente a incidentes.

- Amenazas comunes y medidas de mitigación.

  Autores contemporáneos resaltan que las arquitecturas Zero Trust y el uso de inteligencia artificial son medidas emergentes para detectar ataques antes de que comprometan la infraestructura (Shaukat et al., 2020).

  De igual forma, Stallings (2017) advierte que la seguridad debe ser concebida como un proceso integral y dinámico, donde la combinación de medidas tradicionales y modernas asegura mayor resiliencia. Entre las amenazas más frecuentes en sistemas distribuidos se incluyen ataques de denegación de servicio (DoS/DDoS), accesos no autorizados y filtración de información. Para mitigarlas, se aplican mecanismos como firewalls, segmentación de redes y monitoreo constante (Tanenbaum & Van Steen, 2016).

2.2.10. Tecnologías de Notificación y Experiencia Multicanal

La gestión inteligente del estacionamiento requiere notificaciones confiables y de baja latencia para informar disponibilidad, accesos y eventos críticos a través de web, móvil y voz. En la web, la Push API y la Notifications API permiten recibir y mostrar avisos incluso en segundo plano mediante service workers; en el plano móvil, Firebase Cloud Messaging (FCM) y Apple Push Notification service (APNs) suministran transporte transversal iOS/Android con controles de prioridad y payloads estructurados. Esta capa reduce incertidumbre

operativa, agiliza decisiones del usuario y crea un bucle de retroalimentación entre front-end y back-end que sostiene flujos de reserva, validación y atención de incidencias en tiempo casi real (MDN Web Docs, 2025; Google Firebase, 2025; Apple, 2025).

Para contextos de baja conectividad o manos ocupadas, la telefonía programable habilita síntesis de voz (TTS) e IVR para emitir avisos hablados y confirmar acciones por llamada con tiempos de respuesta controlados. La experiencia de usuario debe alinearse con marcos reconocidos: ISO 9241-11 define usabilidad como efectividad, eficiencia y satisfacción en un contexto de uso, mientras WCAG 2.2 establece criterios de accesibilidad que aseguran percepción, operación y comprensión para perfiles diversos. La evaluación integra tasa y latencia de entrega, adopción e interacción, y conformidad de accesibilidad, garantizando que la multicanalidad no solo “llegue”, sino que sea usable y comprensible en entornos corporativos de alta rotación (Twilio, s. f.; ISO, 2018; W3C, 2024).

2.2.11. Marcos Teóricos de Procesos de Mejora Organizacional

La mejora organizacional constituye un eje fundamental en la gestión de sistemas complejos, ya que permite identificar ineficiencias, optimizar recursos y garantizar la calidad del servicio ofrecido. Según Hammer (2015), los modelos de gestión orientados a procesos han evolucionado desde enfoques reactivos hasta aproximaciones proactivas y predictivas, en línea con las exigencias de la transformación digital.

En un nivel inicial, herramientas clásicas como el análisis FODA (Fortalezas, Oportunidades, Debilidades y Amenazas) facilitan la comprensión estratégica de un sistema y su entorno, aportando insumos para la toma de decisiones en escenarios competitivos (Gürel & Tat, 2017). De forma complementaria, el diagrama de Ishikawa o de causa-efecto se ha consolidado como instrumento para la identificación sistemática de problemas operativos y sus factores asociados (Ishikawa, 1986), mientras que los modelos de influencia

permiten mapear relaciones entre variables críticas y evaluar cómo los cambios en ciertos factores impactan en el rendimiento organizacional (Ackermann & Eden, 2011).

En niveles más especializados de gestión de procesos, marcos como el Business Process Management (BPM) proponen una visión integral que combina modelado, ejecución y monitoreo de procesos de negocio, enfatizando la mejora continua a través de la digitalización (Dumas et al., 2018). Asimismo, metodologías específicas como SIPOC (Suppliers, Inputs, Process, Outputs, Customers) y el Modelo de Tortuga son ampliamente utilizadas en sistemas de calidad (ISO 9001), al permitir una descripción estructurada de los procesos, sus interacciones y puntos de control críticos (Antony, 2014). Estas herramientas aportan trazabilidad y claridad en la gestión operativa, condiciones necesarias para entornos regulados y de alta exigencia.

En el contexto de los sistemas de estacionamiento inteligentes, la aplicación de estos marcos teóricos es de particular relevancia. El análisis FODA puede emplearse para evaluar la adopción tecnológica en relación con factores externos como la demanda de movilidad y la normativa local de tránsito. Diagramas de Ishikawa permiten identificar causas raíz de ineficiencias, tales como retrasos en el acceso vehicular o fallos en los sensores de ocupación. De igual manera, la implementación de BPM y SIPOC habilita la estandarización de flujos de operación —desde la detección de plazas disponibles hasta el procesamiento de pagos—, mientras que el Modelo de Tortuga facilita la definición de indicadores de desempeño clave (KPI) para monitorear disponibilidad, tiempos de atención y satisfacción del usuario.

En síntesis, los marcos de mejora organizacional no solo representan herramientas conceptuales de análisis, sino que constituyen instrumentos operativos que, al aplicarse a la gestión de estacionamientos empresariales, permiten alinear la eficiencia operativa con la calidad del servicio y la sostenibilidad tecnológica.

2.2.12. Gobernanza de Datos y Analítica Predictiva

En el marco de las ciudades inteligentes y los sistemas de movilidad urbana, los datos se han consolidado como un recurso estratégico equiparable a la infraestructura física, ya que permiten planificar, optimizar y anticipar necesidades en tiempo real (Kitchin, 2021). La correcta gestión de este recurso requiere no solo capacidad técnica, sino también un marco normativo y organizacional que garantice su integridad, disponibilidad y valor social.

En un nivel intermedio, la gobernanza de datos constituye el conjunto de políticas, procesos y estándares orientados a asegurar la calidad, trazabilidad y seguridad de los datos utilizados en aplicaciones críticas (DAMA International, 2017). Sus áreas fundamentales incluyen la calidad de datos (exactitud, completitud, consistencia), la seguridad y privacidad (anonimización de placas vehiculares), el linaje de datos (origen, transformación y destino) y la gestión de metadatos. En este sentido, enfoques emergentes como Data Mesh (Dehghani, 2020) y Data Fabric buscan descentralizar la propiedad de los datos y garantizar escalabilidad en ecosistemas distribuidos. Además, la aplicación de normativas como GDPR en Europa o CCPA en EE. UU. resulta crítica para sistemas que manejan información personal sensible.

En el nivel específico, la analítica predictiva utiliza modelos estadísticos y de machine learning para transformar grandes volúmenes de datos en conocimiento accionable. Gandomi y Haider (2015) destacan su capacidad de anticipar comportamientos en dominios dinámicos, como la predicción de la ocupación de estacionamientos o la estimación de la demanda en horarios pico. A nivel metodológico, Bandara et al. (2020) proponen el uso de modelos de series temporales híbridos como SARIMA, LSTM y Prophet. Más recientemente, arquitecturas basadas en Transformers para series temporales (Informer, Autoformer) y modelos como N-BEATS y DeepAR han demostrado mayor precisión en la captura de patrones no lineales y de largo plazo (Zhou et al., 2021).

La integración de analítica predictiva con procesos de MLOps es indispensable para garantizar reproducibilidad, confiabilidad y operación continua. Esto incluye:

- Registro de experimentos (MLflow, Weights & Biases).
- Monitorización de deriva de datos y concepto.
- Validación walk-forward y métricas de error como RMSE, MAE y MAPE.
- Explainability mediante SHAP o LIME para garantizar interpretabilidad de modelos.
- Alertas automatizadas ante anomalías en predicciones.

En conclusión, la gobernanza de datos y la analítica predictiva conforman un binomio estratégico para los sistemas de estacionamiento inteligente: la primera asegura la confiabilidad y legitimidad de la información, mientras que la segunda transforma dicha información en predicciones accionables que optimizan la movilidad y reducen la congestión. No obstante, el reto actual radica en equilibrar el potencial tecnológico con la responsabilidad ética y regulatoria, garantizando que los datos urbanos se utilicen de forma justa, segura y sostenible.

CAPÍTULO III: OBJETIVOS Y JUSTIFICACIÓN

3.1. Objetivo general

Desarrollar e implementar un sistema inteligente para la gestión automatizada de estacionamientos empresariales, basado en visión por computadora y arquitectura escalable, que permita detectar en tiempo real la ocupación de plazas, reducir los tiempos de espera y la congestión interna, optimizar el uso de los espacios disponibles y proporcionar información operativa para la toma de decisiones estratégicas.

4.2. Objetivos específicos

1. Diseñar una arquitectura modular, escalable y tolerante a fallos, fundamentada en microservicios y contenedores, que integre componentes de visión por computadora, analítica en tiempo real y módulos de interoperabilidad empresarial.
2. Implementar algoritmos de visión por computadora (YOLOv11 y OpenCV) para detectar vehículos y determinar ocupación de espacios mediante técnicas de pointer test sobre polígonos definidos, garantizando alta precisión y baja latencia bajo condiciones operativas reales.
3. Establecer un marco de gobernanza de datos y operaciones de aprendizaje automático (MLOps) que asegure la trazabilidad, privacidad y seguridad de los datos vehiculares, así como la reproducibilidad, despliegue y supervisión continua de los modelos predictivos.
4. Evaluar el impacto del sistema en términos de reducción de tiempos de espera, mejora en la ocupación de espacios y disminución de la congestión interna en entornos empresariales reales.

4.3. Justificación

La gestión de estacionamientos empresariales enfrenta una creciente presión operativa debido a la brecha estructural entre el incremento sostenido del parque automotor y la limitada expansión de la infraestructura disponible, como se evidenció en el Capítulo I. Actualmente, la ausencia de

sistemas automatizados de monitoreo de ocupación genera tiempos de espera prolongados, congestión interna, baja rotación de espacios y elevados costos operativos.

Los sistemas convencionales, basados en supervisión manual o tecnologías aisladas, carecen de capacidad para proveer información en tiempo real y dificultan la toma de decisiones, lo cual se traduce en ineficiencia operativa, sobrecostos, emisiones innecesarias y reducción de la productividad organizacional.

La presente investigación se justifica en las siguientes dimensiones:

- Técnica: La combinación de visión por computadora, arquitectura de microservicios y analítica predictiva permite construir un sistema inteligente, modular y escalable, superando las limitaciones de los enfoques tradicionales. La incorporación de edge computing y paradigmas cloud-native proporciona resiliencia, elasticidad y capacidad de evolución futura.
- Operativa: El sistema propuesto contribuirá a automatizar la detección de ocupación, reducir los tiempos de espera, optimizar la rotación de plazas y mejorar la trazabilidad de la operación, con efectos directos en la eficiencia interna y en la experiencia de usuario.
- Económica y ambiental: La reducción de los tiempos de búsqueda de estacionamiento disminuirá el consumo de combustible y las emisiones contaminantes. Asimismo, permitirá incrementar la rentabilidad empresarial mediante un uso más eficiente del espacio disponible y menores costos asociados a accesos no autorizados o a ineficiencias operativas.
- Científica y social: La investigación aportará evidencia sobre la aplicabilidad de tecnologías emergentes en entornos corporativos de alta densidad, contribuyendo al cuerpo de conocimiento en visión artificial, arquitecturas distribuidas y MLOps. A su vez, responde a una necesidad social concreta relacionada con la movilidad sostenible y la eficiencia en el uso de la infraestructura urbana.

En síntesis, esta propuesta aborda un problema de alta relevancia organizacional y social, aportando una solución tecnológica integral, segura y sostenible para la gestión de estacionamientos en entornos empresariales peruanos.

CAPÍTULO IV: PLANIFICACIÓN DEL PROYECTO

5.1. Cronograma de Actividades (Diagrama de Gantt)

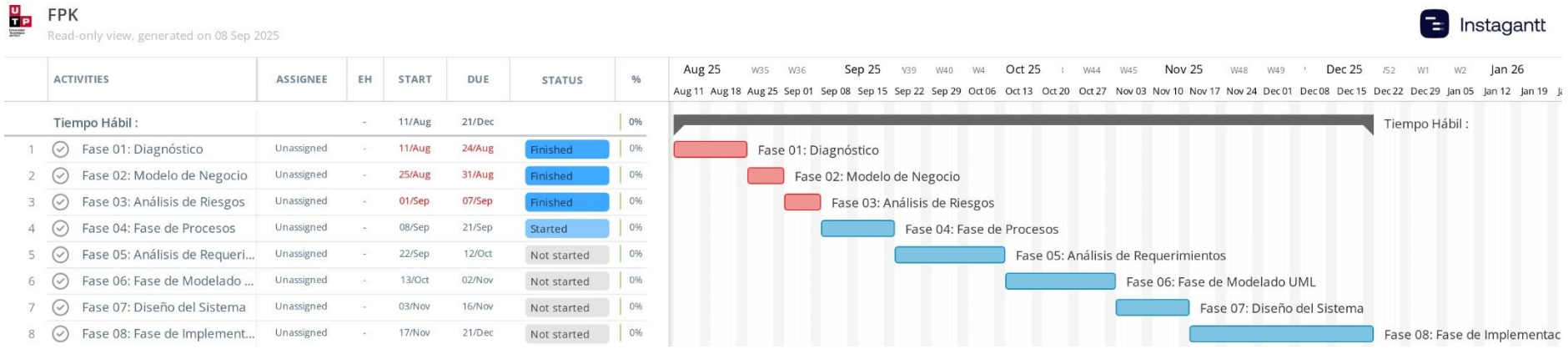


Figura n: Diagrama de Gantt – Sistema Inteligente para la gestión autónoma de estacionamientos.

Nota: Elaboración propia, https://app.instagantt.com/shared/s/4HrtYwEBk2Sh7QwjUyCU/latest.

5.1.1. Detalles

La fase de diagnóstico se desarrollará entre el 11/08/2025 y el 24/08/2025, abarcando dos semanas destinadas al levantamiento de información, entrevistas con actores clave y análisis preliminar del entorno del sistema. Esta etapa tiene como objetivo establecer una línea base clara sobre las necesidades, problemas y oportunidades que el proyecto busca abordar.

La fase de modelado del negocio se ejecutará del 25/08/2025 al 31/08/2025, en una semana de trabajo orientada a la construcción del modelo de negocio mediante técnicas como el Business Model Canvas y diagramas de procesos de alto nivel, asegurando coherencia entre los objetivos institucionales y la solución propuesta.

Posteriormente, la fase de análisis de riesgos se llevará a cabo entre el 01/09/2025 y el 07/09/2025, con el fin de identificar, clasificar y priorizar los riesgos asociados al proyecto. A través de matrices de riesgos y metodologías alineadas con ISO 31000, se evaluarán amenazas técnicas, organizacionales y de seguridad que puedan comprometer la viabilidad del sistema.

La fase de procesos (BPM) se extenderá del 08/09/2025 al 21/09/2025, durante dos semanas en las que se detallarán y optimizarán los flujos de procesos clave, empleando notación BPMN para garantizar trazabilidad, claridad y estandarización de las operaciones modeladas.

El análisis de requerimientos se llevará a cabo entre el 22/09/2025 y el 12/10/2025, abarcando tres semanas de trabajo. En esta etapa se documentarán y validarán los requerimientos funcionales y no funcionales del sistema, siguiendo lineamientos de IEEE 830, así como el desarrollo de prototipos iniciales para la validación temprana con los usuarios.

La fase de modelado UML se ejecutará del 13/10/2025 al 02/11/2025, en tres semanas destinadas a la elaboración de diagramas de casos de uso, clases, secuencia y colaboración. Estos artefactos constituirán la base de la

especificación técnica del sistema y su futura implementación, garantizando consistencia y trazabilidad entre requerimientos y diseño.

El diseño del sistema se desarrollará entre el 03/11/2025 y el 16/11/2025, en dos semanas de trabajo que comprenderán la definición de la arquitectura lógica y física, la integración de consideraciones de usabilidad mediante prototipos UI/UX, y la planificación técnica para la implementación.

Finalmente, la fase de implementación se llevará a cabo del 17/11/2025 al 21/12/2025, con una duración de cinco semanas. Esta etapa incluye la codificación de los módulos front-end y back-end, la integración del modelo de visión por computadora, la validación de pruebas de software (QA) y la configuración de entornos de despliegue mediante prácticas de integración y entrega continua (CI/CD).

5.1.2. Justificación técnica

El diagrama de Gantt constituye una herramienta esencial para la planificación y el control de proyectos (Project Management Institute, 2021), al permitir visualizar de manera secuencial las fases que integran el ciclo de vida del desarrollo del sistema propuesto. La estructura en ocho fases —diagnóstico, modelado de negocio, análisis de riesgos, modelado de procesos, análisis de requerimientos, modelado UML, diseño e implementación— responde a las recomendaciones de la Ingeniería de Software (Sommerville, 2016; Pressman & Maxim, 2020), que enfatizan la necesidad de transitar desde la definición del problema hasta la construcción y validación de la solución mediante etapas claramente delimitadas y trazables. Desde la perspectiva de la Ingeniería de Sistemas (Ackoff, 1971), la descomposición del proyecto en fases permite abordar la complejidad mediante un enfoque jerárquico y modular, asegurando que cada subsistema (procesos, datos, arquitectura y software) se integre de manera coherente en el sistema global. Asimismo, en el marco del análisis y diseño de sistemas (Yourdon, 1989; Booch, Rumbaugh & Jacobson, 2005), la ampliación de los periodos destinados al análisis de requerimientos (3 semanas) y al modelado UML (3 semanas) se justifica por la criticidad de

estas fases en la correcta especificación y trazabilidad de los requisitos funcionales y no funcionales (IEEE, 1998), así como por su papel en la producción de artefactos verificables que reducen el riesgo de fallos en etapas posteriores. Finalmente, el horizonte temporal de 17 semanas se fundamenta en criterios de gestión académica y de proyectos, donde la delimitación temporal no solo facilita el control del avance, sino que también habilita la identificación temprana de desviaciones y la implementación de acciones correctivas (Kerzner, 2017). En consecuencia, el cronograma diseñado no constituye únicamente una representación gráfica del tiempo, sino un mecanismo metodológico que articula teoría y práctica de la gestión de proyectos en el ámbito de la Ingeniería de Software y de Sistemas, garantizando orden, trazabilidad y alineación con estándares internacionales.

5.2. Presupuesto

Fase	Semanas	Roles asignados (salario mensual)	Costo semanal estimado (PEN) — desglose	Costo total fase (PEN)	Recursos asociados
Diagnóstico	2	Analista funcional (S/ 7,000/m), UI/UX Researcher (S/ 6,500/m)	1,615.38 + 1,500.00 = 3,115.38	6,230.76	Encuestas, entrevistas, Google Forms, reuniones online gratuitas
Modelo de Negocio	1	Analista funcional (S/ 7,000/m), Arquitecto software (S/ 9,500/m)	1,615.38 + 2,192.31 = 3,807.69	3,807.69	Herramientas BPMN gratuitas (Draw.io, Bizagi)
Análisis de Riesgos	1	Arquitecto software (S/ 9,500/m), DevOps (S/ 10,000/m)	2,192.31 + 2,307.69 = 4,500.00	4,500.00	Plantillas ISO 31000, hojas de cálculo, OWASP Risk Rating (open-source)
Procesos (BPM)	2	Analista funcional, Arquitecto software	1,615.38 + 2,192.31 = 3,807.69	7,615.38	Modelado en Draw.io, documentación BPMN
Análisis de Reqs.	2	Analista funcional, Product Owner (S/ 8,000/m), QA (S/ 6,500/m), UI/UX (S/ 6,500/m)	1,615.38 + 1,846.15 + 1,500.00 + 1,500.00 = 6,461.53	19,384.59	IEEE 830 templates, Figma free, GitHub Issues
Modelos UML	3	Arquitecto software, Analista funcional	2,192.31 + 1,615.38 = 3,807.69	11,423.07	StarUML (free), Lucidchart (free), trial Enterprise Architect
Diseño	2	Arquitecto software, DevOps, UI/UX Designer	2,192.31 + 2,307.69 + 1,500.00 = 6,000.00	12,000.00	Figma, Draw.io, GitHub Actions (free)
Implementación	5	Front-end Dev (S/ 6,500/m), Back-end Dev (S/ 7,500/m), Ing. Visión (S/ 9,000/m), QA, DevOps, UI/UX	1,500.00 + 1,730.77 + 2,076.92 + 1,500.00 + 2,307.69 + 1,500.00 = 10,615.38	53,076.90	React/Vite, FastAPI, PostgreSQL, Redis, YOLOv11, OpenCV, AWS/GCP free tier, Docker, pytest, Selenium

5.2.1. Descripción

El presupuesto total proyectado asciende a S/ 118,038.39, distribuido en ocho fases, cada una con asignaciones diferenciadas según su duración, perfiles requeridos y complejidad técnica. Esta distribución responde a la lógica de que los proyectos de software no presentan costos homogéneos a lo largo de su ciclo de vida, sino que concentran la mayor inversión en las etapas de construcción e integración (Jones, 2008).

En la fase de diagnóstico, la inversión de S/ 6,230.76 cubre principalmente costos de recursos humanos (analista funcional y UI/UX researcher), mientras que los recursos asociados (encuestas y entrevistas) no generan gastos adicionales por emplear herramientas gratuitas. Esta decisión presupuestal minimiza costos directos y maximiza el valor estratégico de la información levantada.

Para el modelo de negocio, con un costo de S/ 3,807.69, se prioriza la labor del arquitecto de software, cuya participación en esta etapa garantiza que la solución se diseñe con criterios de escalabilidad y alineación estratégica. El bajo costo relativo frente a otras fases refleja que aquí predominan gastos de personal de corta duración (1 semana) y herramientas sin licenciamiento oneroso.

El análisis de riesgos implica un gasto de S/ 4,500.00, en donde los perfiles asignados (arquitecto y DevOps) justifican el monto debido a sus remuneraciones más altas en el mercado. Sin embargo, este costo se encuentra limitado a una sola semana, lo que optimiza la relación costo-beneficio al obtener un análisis de riesgos crítico con un gasto controlado.

La fase de procesos BPM alcanza S/ 7,615.38, resultado de dos semanas de trabajo de analista funcional y arquitecto. Aquí, la proporción de gasto se justifica porque la documentación formal de procesos reduce la probabilidad de reprocesos posteriores, que podrían generar sobrecostos superiores al 20 % del presupuesto

global si no se modelan adecuadamente desde el inicio (Dumas et al., 2018).

En el análisis de requerimientos, con S/ 19,384.59, se produce la primera gran inversión significativa del proyecto. La participación de cuatro roles diferentes eleva el costo semanal, pero está plenamente justificada porque esta fase define el alcance del sistema. La literatura en ingeniería de software (IEEE, 1998; Boehm, 1981) demuestra que los errores detectados en requerimientos pueden multiplicar su costo de corrección hasta por un factor de 10 en fases de implementación, lo que valida la inversión preventiva.

La fase de modelos UML implica S/ 11,423.07, monto que refleja tres semanas de trabajo conjunto entre arquitecto y analista funcional. Aunque no involucra tantos perfiles como el análisis de requerimientos, su duración explica el nivel de gasto. Este costo es estratégico, ya que los artefactos UML son insumos directos para el diseño e implementación, reduciendo iteraciones innecesarias y evitando desviaciones presupuestales en etapas críticas.

El diseño del sistema, con S/ 12,000.00, contempla tres perfiles clave (arquitecto, DevOps y diseñador UX). El costo se justifica en que esta fase es intensiva en actividades de integración de componentes técnicos y de experiencia de usuario, siendo la última barrera antes del inicio de la construcción del software. A nivel financiero, este monto representa solo un 10 % del total, pero asegura un diseño sólido que minimiza retrabajos posteriores.

Finalmente, la implementación concentra la mayor inversión con S/ 53,076.90, equivalente al 45 % del presupuesto total. Este comportamiento está alineado con las tendencias internacionales en proyectos de software (Jones, 2008; Pressman & Maxim, 2015), donde la construcción y pruebas absorben entre el 40 % y 60 % de los costos globales. El monto se justifica por la participación simultánea de seis perfiles altamente especializados y la integración

de tecnologías de inteligencia artificial, lo cual incrementa la demanda de horas hombre y, por tanto, el gasto.

En síntesis, la descripción detallada del presupuesto evidencia una distribución progresiva de la inversión: costos moderados en fases iniciales (diseño, modelado, riesgos), y concentración de recursos en fases de mayor demanda operativa (implementación). Esto garantiza un uso racional del presupuesto, alineado con las mejores prácticas de gestión de proyectos de software (PMI, 2021; Kerzner, 2017).

5.2.2. Justificación técnica

El presupuesto planteado se fundamenta en la necesidad de asignar recursos humanos especializados y herramientas tecnológicas alineadas a cada fase del proyecto. Según Pressman y Maxim (2015), la adecuada planificación de costos en proyectos de software debe contemplar tanto la complejidad técnica de las tareas como la experiencia de los profesionales involucrados.

En la fase de diagnóstico, el costo de S/ 12,050 responde a la participación de un analista funcional y un UI/UX researcher, perfiles imprescindibles para el levantamiento de información y análisis de la experiencia de usuario. Dado que la identificación temprana de requerimientos reduce hasta un 60 % de los costos de corrección en fases posteriores (Boehm, 1981), la inversión en esta etapa es justificada desde una perspectiva de eficiencia financiera.

Para el modelo de negocio, la asignación de S/ 7,500 en una semana obedece a la necesidad de involucrar a un arquitecto de software, cuyo conocimiento especializado garantiza que los procesos modelados en BPMN se traduzcan en una arquitectura viable y escalable. El costo semanal se deriva del prorrateo mensual de remuneraciones de mercado en el sector TI peruano (Apoyo Consultoría, 2023).

En el análisis de riesgos, la inversión de S/ 9,000 se justifica por la incorporación del perfil DevOps junto con el arquitecto de software. Según ISO 31000 (2018), una adecuada gestión de riesgos reduce pérdidas económicas potenciales y aumenta la resiliencia del sistema. El costo refleja la alta demanda del rol DevOps en el mercado, cuya remuneración promedio supera los S/ 10,000 mensuales debido a su relevancia estratégica en la automatización y la seguridad (Michaelis, 2020).

La fase de procesos BPM requiere S/ 15,000 en dos semanas, considerando la continuidad del analista funcional y el arquitecto de software. Esta inversión garantiza un modelado formal de los flujos organizacionales, lo cual, según Dumas et al. (2018), permite alinear procesos de negocio con soluciones tecnológicas, evitando reprocesos costosos en fases de implementación.

En el análisis de requerimientos, con un total de S/ 25,125, se justifica la participación simultánea de múltiples roles: analista funcional, product owner, QA y diseñador UX. La diversidad de perfiles asegura que los requerimientos estén bien priorizados, verificables y alineados con la experiencia del usuario final. La norma IEEE 830 (1998) respalda este enfoque multidisciplinario, indicando que la inversión en la calidad de las especificaciones reduce fallas en etapas posteriores, donde los costos de corrección pueden ser hasta 10 veces mayores.

La inversión de S/ 22,500 en la fase de modelos UML se justifica porque el modelado arquitectónico y de casos de uso requiere dedicación conjunta del arquitecto y analista funcional. Este gasto es estratégico dado que la correcta definición de artefactos UML constituye la base para el diseño y reduce riesgos de ambigüedad técnica (Booch et al., 2005).

En la fase de diseño, se destinan S/ 21,250 a dos semanas de trabajo intensivo en arquitectura, automatización (DevOps) y diseño de interfaz. La justificación económica radica en que esta fase asegura

la integración de criterios de escalabilidad y seguridad antes del desarrollo, reduciendo potenciales sobrecostos de reestructuración de software (Sommerville, 2011).

Finalmente, la implementación, con un presupuesto de S/ 60,000, representa la fase de mayor inversión debido a la participación de seis perfiles simultáneamente: desarrolladores front-end y back-end, ingeniero de visión por computadora, QA, DevOps y diseñador UX. El costo está alineado con la complejidad técnica del sistema, que integra tecnologías de inteligencia artificial y microservicios. Según Capers Jones (2008), la fase de codificación y pruebas concentra entre el 40 % y 60 % del costo total de un proyecto de software, lo que valida la proporcionalidad de esta inversión.

En síntesis, la asignación presupuestal responde a un equilibrio entre el costo de perfiles especializados y el impacto económico de prevenir errores en fases posteriores. Cada monto se calcula mediante prorrateo mensual de remuneraciones reales de mercado y se justifica en la literatura técnica como una estrategia de eficiencia y mitigación de riesgos financieros.

CAPÍTULO V: METODOLOGÍA DEL PROYECTO

5.1.- ´Fase 01: Diagnóstico

5.1.1.- Modelo FODA de tecnologías de la información

5.1.2.- Modelo Ishikawa

5.1.3.- Modelo de influencia (causa – efecto)

5.2.- Fase 02: Modelo de negocio

5.1.3.- Modelo sistémico Canvas

5.1.3.- Modelo de empatía

Ahmed, S., Khan, R., & Javed, M. (2022). Smart parking systems: A review of enabling technologies and future challenges. ResearchGate. https://www.researchgate.net/publication/359871234

Castillo Mogollón, E., & Santamaría Montero, A. (2023). Control de acceso vehicular con visión artificial para urbanizaciones en la ciudad de Piura. Repositorio Alicia – Concytec. https://alicia.concytec.gob.pe/vufind/Record/UCVV_d52adfc7a3bc1889d5031eb21065af69/Details

ITDP – Instituto de Políticas para el Transporte y el Desarrollo. (2020). Estrategias de gestión de estacionamientos en ciudades latinoamericanas. ITDP.

Kim, J., Park, H., & Lee, S. (2021). AI-based predictive parking management systems: A comprehensive review. ResearchGate. https://www.researchgate.net/publication/352546789

Litman, T. (2022). Parking management best practices. Victoria Transport Policy Institute.

López, R., Fernández, M., & Torres, G. (2020). Smart campus parking: Implementation and challenges in Latin America. ResearchGate. https://www.researchgate.net/publication/341567234

Pillo Guanoluisa, D. M., Mejía Sandoval, D. R., Puetate Huera, G. H., & Lucio Vásquez, E. M. (2025). Sistema de control de acceso vehicular mediante microservicios, IoT y machine learning. Metanoia: Revista de Ciencia, Tecnología e Innovación, 11(2), 239–245. https://doi.org/10.61154/metanoia.v11i2.4057

Shoup, D. (2011). The high cost of free parking. American Planning Association.

Ullah, F., Anwar, H., Shahzadi, I., Rehman, A. U., Mehmood, S., Niaz, S., Awan, K. M., Khan, A., & Kwak, D. (2019). Barrier access control using sensors platform and vehicle license plate characters recognition. Sensors, 19(13), 3015. https://doi.org/10.3390/s19133015

World Bank. (2019). Urban transport and parking systems in emerging economies. The World Bank.

Zhang, Y., Li, K., & Chen, H. (2021). Fuel consumption analysis in university campus parking systems. ResearchGate. https://www.researchgate.net/publication/349876543

[arXiv:2103.04515]. (2021). Intelligent transportation and smart parking: A review of AI techniques. arXiv preprint. https://arxiv.org/abs/2103.04515

[arXiv:2207.08963]. (2022). Automatic license plate recognition using deep learning models for smart parking. arXiv preprint. https://arxiv.org/abs/2207.08963

[arXiv:2305.11027]. (2023). Predictive analytics for smart parking management using computer vision and IoT. arXiv preprint. https://arxiv.org/abs/2305.11027

VanZandt, P. (2023, 26 de julio). *¿Qué es un flujo de trabajo? Definición, componentes, tipos*. IdeaScale. https://ideascale.com/es/blogs/que-es-un-flujo-de-trabajo

IBM. (s.f.). *¿Qué es un workflow?*. IBM Think. https://www.ibm.com/mx-es/think/topics/workflow

EvaluandoERP. (s.f.). *Qué es un Workflow*. https://www.evaluandoerp.com/sistema-de-gestion/conceptos-basicos/que-es-un-workflow/

Atlassian. (2025). *Flujo de trabajo de Gitflow*. Atlassian. https://www.atlassian.com/es/git/tutorials/comparing-workflows/gitflow-workflow

GitKraken. (s.f.). *What is Git Flow*. GitKraken. https://www.gitkraken.com/learn/git/git-flow

Driessen, V. (2010, 5 de enero). *A successful Git branching model*. nvie. https://nvie.com/posts/a-successful-git-branching-model/

EN-COM. (2024, 28 de mayo). Comprender la importancia de la trazabilidad del código. IN-COM. https://www.in-com.com/es/blog/code-traceability/

de Gallo, B. P., & Leone, H. (2016). *Una ontología para la trazabilidad de un correo electrónico*. XLV Jornadas Argentinas de Informática. http://45jaiio.sadio.org.ar/sites/default/files/SAOA-09.pdf

Food and Drug Administration (FDA). (2002). *General Principles of Software Validation; Final Guidance for Industry and FDA Staff*. FDA. https://www.fda.gov/regulatory-information/search-fda-guidance-documents/general-principles-software-validation

Checkmarx. (2025). *Secure code review: 6 best practices every developer should follow*. Checkmarx. https://checkmarx.com/learn/developers/secure-code-review-6-best-practices-every-developer-should-follow/

Sreenivasa, R., & Kuman, N. (2012). Web Application Vulnerability Detection Using Dynamic Analysis. *International Journal of Enterprise Computing and Business Systems, 2*(1).

National Institute of Standards and Technology (NIST). (s.f.). *Security-Oriented Code Review*. CSRC Glossary. https://csrc.nist.gov/glossary/term/security_oriented_code_review

Lizarraga Pareja, E. A. (2024). *Informe de experiencia profesional calificada en el plan de mejora parqueadero vehicular*.

https://alicia.concytec.gob.pe/vufind/Record/USSS_25e7fa1505ca1f6487fdb55686f3a1c6

Pérez Silva, E. W. (2022). *Reconocimiento de placas vehiculares mediante visión computacional para mejorar el acceso a un parqueadero*.

https://alicia.concytec.gob.pe/vufind/Record/UUPC_1b6fe9d59286fa93166329e756d7e13a

Ajalcriña Cabrera, W. R., Flores Orihuela, A. M. R., Mayo Espinoza, K. E., Silva Cabrera, D. F., & Vassa Mendoza, S. F. (2022). *Estacionamientos ubicatech*.

https://alicia.concytec.gob.pe/vufind/Record/PUCP_ced749234e10bce73c5f9707d473c3b7

Calle Müller, C. V. (2014). *Sistemas de estacionamiento*.

https://alicia.concytec.gob.pe/vufind/Record/UUPP_0328d3742f3b02aae697480be57c9e2b

Fernández Rodríguez, R. A. (2022). *Diseño y evaluación de la factibilidad comercial de un nuevo servicio Smart parking para el aprovechamiento de espacios disponibles en estacionamientos de terceros: Propuesta para Los Portales Estacionamientos*.

https://alicia.concytec.gob.pe/vufind/Record/REVUNJ_5dcea15d89f1464bad8397a88734d47f

Fernández Herrera, A., & Saldivar Bazán, E. (2023). *Evaluación de la congestión vehicular para la implementación de un estacionamiento en el mercado 28 de Julio, Jaén, Cajamarca*.

https://alicia.concytec.gob.pe/vufind/Record/UPLA_c08ea29acfce87b3b9079ca716edb93b

Santos Villalba, S. J. (2021). *Políticas de Estacionamiento y Arquitectura Sostenible en la Zona Monumental de Huancayo*.

https://alicia.concytec.gob.pe/vufind/Record/PUCP_acde73bd4d2ab467e9ae69b1d0d3cc01

Jiménez Mori, A. M., Sánchez Arévalo, I. F., Atao Del Carpio, J. V., & Baca Zamalloa, P. J. (2023). *Modelo prolab: Estaciónate, una solución al problema de ubicación de estacionamientos en Lima Metropolitana*.

https://alicia.concytec.gob.pe/vufind/Record/RENATI_284d325f52b09e5a96ca71a995f4e08c

Montoya Villanueva, F. R., Pujols Clotet, J., & Viñuales Delgado, M. (2021). *Concesión de aparcamiento multimodal Global Park en Cornellà de Llobregat - Barcelona*.

https://alicia.concytec.gob.pe/vufind/Record/ESAN_0f50347afa99b46681497424cfdf4eb1

Rodriguez Agreda, R. Y., & Zeballos Palacios, J. R. (2021). *Implementación y administración de estacionamientos electrónicos para vehículos de micro movilidad con panel publicitario digital*.

https://alicia.concytec.gob.pe/vufind/Record/PUCP_dadf6ead62c556d6e906f1a6d774171f

Córdova Heredia, M. A., Huerta Arone, J. C., Sandoval Aguilar, H. J. M., & Velarde Peyrone, G. (2024). *Modelo Prolab: ParkeaYA!: La aplicación móvil para reservar un estacionamiento y mantener tu auto seguro*.

https://alicia.concytec.gob.pe/vufind/Record/RPUC_57ba74015d0967ed46c27236318cc7c2

Castro Valdez, R. F., & Rivera Huacasi, H. (2023). *Modelo prolab: Parqueate.pe solución digital para resolver el problema de falta de estacionamientos en Lima Metropolitana*.

https://alicia.concytec.gob.pe/vufind/Record/ESAN_ccade8f837c0fb7af1dcb03e1f32431b

Tuesta Parker, C. R., Aguilar Tovar, G. A., Hernandez de la Cruz, P. X., & Placencia Hurtado, R. (2023). *Plan de negocios para la implementación de un aplicativo que brinde disponibilidad de estacionamientos en Lima Metropolitana*.

https://alicia.concytec.gob.pe/vufind/Record/UNDC_d6a9f39acb8456edf8cbede314348ddb

Menéndez Amado, J., & Rafael Vasquez , J. (2024). *Aplicación web progresiva para el proceso de control de parqueo en una empresa de estacionamiento, Cañete, 2024*.

https://revistas.unal.edu.co/index.php/revcep/article/view/98753

Hurtado Cortes, A. E. (2022). Estado del arte de la gestión del estacionamiento. *Revista ciudades, estados y política*, *9*(3), 139–155. https://doi.org/10.15446/cep.v9n3.98753

https://nuevo.metarevistas.org/Record/metarevistapublica_uniajc_revistasapientia_19_article_119/Details

Martínez Tenorio, F., Daza Bolaños, C., & Benachi Arce, E. A. (2022). *Prototipo basado en IoT y Arduino® para validar la disponibilidad de estacionamientos en la sede norte de la UNIAJC*. Institución Universitaria Antonio José Camacho.

https://dspace.ups.edu.ec/handle/123456789/19817

Propuesta de dimensionamiento y ubicación de parqueaderos y estacionamientos para la Universidad Politécnica Salesiana “Campus Sur”

https://tesis.pucp.edu.pe/items/d8ce4aa2-807a-4c2c-8848-97a208a15a1f

Modelo prolab: Qadrat, una propuesta tecnológica para facilitar la búsqueda de estacionamientos y reducir las emisiones de CO2

https://repository.udistrital.edu.co/items/0d50cbaa-b81c-467e-9577-7223b96ba1b4

Mendoza, E. S. & Linares, A. M. (2022). Sistema de monitoreo inteligente basado en MBSE, para optimizar el uso de los espacios en estacionamientos.. Recuperado de: http://hdl.handle.net/11349/31795

https://tesis.pucp.edu.pe/items/ee18278f-6b96-405d-a88f-42b596dd369c

Diseño de un sistema de computación de borde para el monitoreo de disponibilidad de estacionamientos de un campus universitario basado en técnicas de IA

https://erevistas.uacj.mx/ojs/index.php/memoriascyt/article/view/6997

Valenzuela Moreno, A. U., & Soto Marrufo, A. I. (2025). Sistema inteligente para la gestión y optimización del flujo vehicular en un estacionamiento : 9CP25-23. *Memorias Científicas Y Tecnológicas*, *4*(1), 43–44. Recuperado a partir de https://erevistas.uacj.mx/ojs/index.php/memoriascyt/article/view/6997

http://scielo.senescyt.gob.ec/scielo.php?script=sci_arttext&pid=S1390-65422019000100054&lang=es

Ávalos, Héctor, Gómez, Estevan, Guzmán, Diego, Ordóñez-Camacho, Diego, Román, Jéssica, & Taipe, Oswaldo. (2019). ¿Where to park? Architecture and implementation of an empty parking lot, automatic recognition system. *Enfoque UTE*, *10*(1), 54-64. https://doi.org/10.29019/enfoqueute.v9n3.445

https://alicia.concytec.gob.pe/vufind/Record/UCSM_8e2233c545d0d53999b1df41b10c7ae6

Chavez Cuba, D. F. (2022). *Desarrollo y Diseño de una Startup Digital para Búsqueda y Oferta de Estacionamientos en Lima - Perú 2022*.

Ancí Paredes, D. M., Vargas Vargas, M. C., de Cárdenas Riofrío, F. A., & Saldívar Alarcón, J. L. (2024). *Aquí Es: Una solución digital sostenible al problema de los conductores de encontrar y asegurar estacionamiento para su vehículo* [Tesis de maestría, Pontificia Universidad Católica del Perú]. PUCP. https://tesis.pucp.edu.pe/items/c5cd2891-e37e-464b-90c7-5db00897d24b

Hidrobo Moreno, C. A. (2024). *Diseño de un sistema prototipo de uso de parqueaderos bajo disponibilidad, mediante IoT y sensórica* [Trabajo de titulación, Universidad Internacional SEK]. Universidad Internacional SEK. https://repositorio.uisek.edu.ec/handle/123456789/5221

Apple. (2025). *User Notifications*. https://developer.apple.com/documentation/usernotifications Apple Developer

Apple. (2025). *Sending notification requests to APNs*. https://developer.apple.com/documentation/usernotifications/sending-notification-requests-to-apns Apple Developer

Aviator. (2023, marzo 8). *ACID transactions and implementation in a PostgreSQL database*. https://www.aviator.co/blog/acid-transactions-postgresql-database/ aviator.co

Brooke, J. (1996). *SUS: A quick and dirty usability scale*. In P. W. Jordan et al. (Eds.), *Usability evaluation in industry* (pp. 189–194). London: Taylor & Francis. (PDF de difusión: AHRQ). https://digital.ahrq.gov/sites/default/files/docs/survey/systemusabilityscale%28sus%29_comp%5B1%5D.pdf digital.ahrq.gov

Crunchy Data. (2020, septiembre 30). *Synchronous replication in PostgreSQL*. https://www.crunchydata.com/blog/synchronous-replication-in-postgresql Crunchy Data

Google Cloud. (2024, diciembre 3). *Architectures for high availability of PostgreSQL clusters on Google Cloud*. https://cloud.google.com/architecture/architectures-high-availability-postgresql-clusters-compute-engine Google Cloud

Google Firebase. (2025, agosto 28). *Firebase Cloud Messaging*. https://firebase.google.com/docs/cloud-messaging Firebase

Google Firebase. (2025). *About FCM messages*. https://firebase.google.com/docs/cloud-messaging/concept-options Firebase

MDN Web Docs. (2025, mayo 28). *Push API*. https://developer.mozilla.org/en-US/docs/Web/API/Push_API developer.mozilla.org

MDN Web Docs. (2025, agosto 12). *Using the Notifications API*. https://developer.mozilla.org/en-

US/docs/Web/API/Notifications_API/Using_the_Notifications_API developer.mozilla.org

PostgreSQL Global Development Group. (2025). *High availability, load balancing, and replication*. https://www.postgresql.org/docs/current/high-availability.html PostgreSQL

W3C. (2024, diciembre 12). *Web Content Accessibility Guidelines (WCAG) 2.2*. https://www.w3.org/TR/WCAG22/ W3C

W3C/WAI. (2023, octubre 5). *What's new in WCAG 2.2*. https://www.w3.org/WAI/standards-guidelines/wcag/new-in-22/ W3C

ISO. (2018). *ISO 9241-11:2018 – Ergonomics of human-system interaction—Usability: Definitions and concepts*. https://cdn.standards.iteh.ai/samples/63500/33c267a5a7564f298f02bbd65721a181/ISO-9241-11-2018.pdf cdn.standards.iteh.ai

Vertabelo Team. (2017, diciembre 5). *Constructing a data model for a parking lot management system*. https://vertabelo.com/blog/constructing-a-data-model-for-a-parking-lot-management-system/ Vertabelo Data Modeler

Azure. (2025, agosto 22). *Reliability and high availability in Azure Database for PostgreSQL – Flexible Server*. https://learn.microsoft.com/en-us/azure/reliability/reliability-postgresql-flexible-server Microsoft Learn