Desarrollo de una Aplicación Web para la Optimización de la Gestión Hotelera

César Antonio Ríos Gonzalez

Universidad del Valle

Escuela de Ingeniería en Sistemas y Computación

Tuluá

2026

Desarrollo de una Aplicación Web para la Optimización de la Gestión Hotelera

César Antonio Ríos Gonzalez

Código 202059959

rios.cesar@correounivalle.edu.co

Director

Ing. Hector Fabio Ocampo

hector.ocampo@correounivalle.edu.co

Universidad del Valle

Escuela de Ingeniería en Sistemas y Computación

Tuluá

2026

Tabla de Contenido

Índice de cuadros

Índice de figuras

Resumen

El sector hotelero especialmente en hoteles pequeños y medianos, se enfrentan diversos desafíos operativos relacionados con la gestión manual de reservas y la falta de integración con plataformas digitales de reservas en línea. En muchos casos, los hoteles deben administrar manualmente las reservas que llegan desde diferentes canales, lo que dificulta mantener actualizada la disponibilidad de habitaciones en tiempo real.

Esta situación puede generar problemas frecuentes como sobreventa de habitaciones, duplicidad de reservas o errores en la facturación. Además, la ausencia de un sistema centralizado que integre toda la información afecta la eficiencia de la operación interna del hotel y puede impactar negativamente la experiencia del huésped.

En respuesta a esta problemática, se diseñó e implementó un sistema web de gestión hotelera orientado a hoteles pequeños y medianos del Valle del Cauca, Colombia. El sistema fue desarrollado bajo una arquitectura cliente-servidor con API REST, utilizando React.js en el frontend, Node.js con Express.js en el backend y PostgreSQL como base de datos. Los requisitos fueron especificados bajo el estándar IEEE 830 y el proyecto fue gestionado mediante la metodología Kanban.

El sistema integra módulos funcionales de gestión de reservas, registro de huéspedes, control de habitaciones, facturación y generación de reportes. Como aporte diferenciador, se implementó la integración con plataformas externas de reservas en línea (OTAs) a través del channel manager Channex, mediante un modelo basado en webhooks que permite la sincronización automática de disponibilidad y la recepción de reservas en tiempo real, eliminando la necesidad de registro manual y reduciendo el riesgo de sobreventas.

La validación del sistema se realizó mediante pruebas unitarias automatizadas sobre los módulos críticos del backend (autenticación, reservas, check-in, check-out, seguridad e integración con OTAs), y pruebas funcionales con usuarios finales utilizando una escala Likert de cinco niveles. Los resultados obtenidos evidenciaron una alta aceptación del sistema: usabilidad, funcionalidad, eficiencia, satisfacción general, e integración con OTAs. Más del 90 % de las respuestas se concentraron en las categorías “De acuerdo” y “Totalmente de acuerdo”, sin registrar ninguna respuesta negativa.

Los resultados confirman que el sistema desarrollado es funcional, confiable y adecuado para su implementación en el contexto hotelero real, cumpliendo con los objetivos planteados y sentando las bases para una evolución futura hacia un producto comercial escalable.

Palabras clave: Gestión hotelera, Sistema PMS, Aplicación web, Integración OTA, Automatización de reservas, Arquitectura REST, Transformación digital.

Capítulo 1

Introducción

1.1. Formulación del problema

¿Cómo diseñar e implementar un sistema de gestión hotelera que facilite los procesos operativos y administrativos en hoteles, mejorando la eficiencia operativa y la satisfacción del cliente?

1.2. Descripción del problema

En la actualidad la industria hotelera enfrenta grandes desafíos críticos como la ineficiencia operativa, falta de integración de datos, errores en la gestión hotelera generalmente procesos esenciales que garantizan un servicio de calidad en el hotel. No obstante, muchos hoteles pequeños y medianos continúan utilizando procesos manuales, estos procesos requieren un esfuerzo considerable por parte del personal y pueden estar sujetos a errores humanos que afectan la experiencia del cliente, y disminuye su competitividad en el mercado. Según O'Fallon y Rutherford, "los hoteles que carecen de tecnología adecuada enfrentan dificultades en la optimización de sus operaciones, lo que impacta directamente en la satisfacción del cliente y la rentabilidad"[1]

Uno de los principales problemas que en los hoteles que no cuentan con un sistema de gestión hotelera es la alta probabilidad de erros en la asignación de habitaciones y la gestión de reservas. Los procesos manuales pueden provocar una sobreventa, reservas duplicadas o incorrectas. Kasavana y Cahill destacan que "la automatización en la industria hotelera reduce significativamente los errores humanos y mejora la eficiencia operativa"[2] Además, la falta de integración entre las diferentes áreas del hotel dificulta la recopilación y el análisis de datos, limitando la capacidad de los gerentes para tomar decisiones informadas y mejorar la rentabilidad del negocio [3]

La ausencia de un sistema de gestión hotelero y la falta de herramientas que permitan un control detallado de reportes financieros pueden derivar en problemas contables Ivanov sostiene que "la tecnología es un factor clave para la optimización financiera en el hotelería, ya que facilita el seguimiento de los recursos y mejora la rentabilidad"[4] Asimismo, sin una solución que permita una gestión eficiente del inventario y el mantenimiento de las instalaciones, los costos operativos pueden aumentar debido a fallas en la infraestructura y desperdicio de recursos [5]

En un entorno altamente competitivo, donde la digitalización es clave para la supervivencia, los hoteles que no adoptan tecnologías avanzadas enfrentan serias desventajas. La falta de un sistema integral que unifique la gestión financiera, operativa y de reservas puede derivar en pérdidas económicas, insatisfacción del cliente y una menor capacidad para adaptarse a las demandas del mercado.

1.3. Objetivos

1.3.1. Objetivo general

Diseñar e implementar un sistema de gestión hotelera que permita automatizar y optimizar procesos operativos y administrativos

1.3.2. Objetivos específicos

1. Identificar los procesos actuales de gestión hotelera para identificar problemas recurrentes
2. Definir los requisitos funcionales y no funcionales del sistema, asegurando que cumpla con las necesidades operativas y administrativas del negocio.
3. Diseñar una arquitectura de software que permita la integración de módulos clave como reservas, facturación y reportes.
4. Implementar un sistema web utilizando una metodología de desarrollo ágil, garantizando flexibilidad y calidad en el producto final.
5. Realizar pruebas de verificación del sistema para validar que cumple con los requisitos funcionales y de diseño establecidos.

1.4. Resultados esperados

Tabla 1.1: Relación entre los objetivos específicos y los resultados esperados

Objetivo específico	**Resultado**
Identificar los procesos actuales de gestión hotelera para detectar problemas recurrentes.	Se identificaron y analizaron los procesos operativos y administrativos actuales de hoteles pequeños, permitiendo reconocer problemas recurrentes como la falta de sincronización con OTAs, procesos manuales y demoras en la gestión de reservas. Estos resultados se documentan en el Capítulo 4 – Análisis y levantamiento de requerimientos, donde se presentan los resultados de la encuesta aplicada y su respectivo análisis.
Definir los requisitos funcionales y no funcionales del sistema, asegurando que cumpla con las necesidades operativas y administrativas del negocio.	Se definieron y documentaron de manera formal los requisitos funcionales y no funcionales del sistema, siguiendo el estándar IEEE 830, garantizando que el desarrollo estuviera alineado con las necesidades reales del entorno hotelero. Este resultado se presenta en el Capítulo 4 – Análisis y levantamiento de requerimientos, en la sección correspondiente a la especificación de requisitos.
Diseñar una arquitectura de software que permita la integración de módulos clave como reservas, facturación y reportes.	Se diseñó una arquitectura de software basada en el modelo cliente-servidor, que integra de forma modular los principales componentes del sistema, incluyendo reservas, facturación, usuarios y reportes, asegurando escalabilidad y facilidad de mantenimiento. Este resultado se evidencia en el Capítulo 5 – Diseño de la arquitectura del sistema.
Implementar un sistema web utilizando una metodología de desarrollo ágil, garantizando flexibilidad y calidad en el producto final.	Se implementó un sistema web funcional para la gestión hotelera, desarrollado de forma incremental y modular, cumpliendo con los requisitos definidos y aplicando principios de metodologías ágiles. El sistema resultante permite la gestión de reservas, check-in, check-out, facturación, pagos e integración con OTAs. La implementación se presenta en el Capítulo 6 – Implementación del sistema.

Objetivo específico	**Resultado**
Realizar pruebas de verificación del sistema para validar que cumple con los requisitos funcionales y de diseño establecidos.	Se realizaron pruebas unitarias, de integración y pruebas con usuarios finales, validando el correcto funcionamiento del sistema en distintos escenarios operativos. Los resultados demostraron que el sistema cumple con los requisitos definidos y ofrece una experiencia satisfactoria para los usuarios. Este resultado se documenta en el Capítulo 8 – Pruebas y resultados.

1.5. Metodología

1.5.1. Metodología de Investigación

La metodología de investigación para este proyecto se fundamenta en los Niveles de Maduración Tecnológica (TRL, por sus siglas en inglés), los cuales permiten evaluar el desarrollo y validación de tecnologías en diferentes etapas:

1. Análisis preliminar: Identificación del problema y análisis de requerimientos, considerando estudios previos sobre gestión hotelera.
2. Prueba de concepto: Diseño inicial de la solución, definiendo funcionalidades clave y evaluando la factibilidad técnica.
3. Desarrollo y validación: Construcción del sistema en un entorno controlado, seguido de pruebas técnicas y de usuario.
4. Implementación: Evaluación de la herramienta en un entorno real para garantizar su efectividad y escalabilidad.

1.5.2. Metodología de Desarrollo de Software

Una metodología de desarrollo de software es un conjunto estructurado de principios, prácticas y procesos que guían la creación de un software. Estas metodologías se utilizan para planificar, estructurar y controlar el proceso de desarrollo de software de manera efectiva. El objetivo principal de una metodología de desarrollo es asegurar que el producto final cumpla con los requisitos del cliente y sea entregado dentro del plazo y presupuesto establecido. Existen varias metodologías, cada una con características, ventajas y desventajas particulares, y su elección depende de factores como el tamaño y complejidad del proyecto, los plazos de entrega, y la flexibilidad requerida durante el desarrollo.

Tabla 1.2: Comparación de las diferentes metodologías de desarrollo de software

Metodología	**Características**	**Ventajas**	**Desventajas**
Cascada	Proceso secuencial que sigue fases específicas (análisis, diseño, desarrollo, pruebas, implementación).	Estructura clara y fácil de gestionar en proyectos pequeños.	Rigidez, difícil adaptarse a cambios durante el desarrollo.
Melé	Metodología ágil basada en sprints, con entregas incrementales y feedback constante.	Flexibilidad, mejora continua y enfoque en el cliente.	Requiere alta disciplina del equipo y roles específicos (Scrum Master).
Kanban	Visualización del flujo de trabajo, priorización de tareas y eliminación de cuellos de botella.	Ideal para proyectos dinámicos, enfoque en productividad.	No hay estructura estricta, puede ser difícil para equipos sin experiencia en autogestión.

La metodología Kanban es las más adecuada para el proyecto debido a su enfoque flexible y centrado en la productividad. Teniendo en cuenta que el proyecto está orientado a la gestión hotelera el cual es un entorno dinámico y cambiante, la capacidad de adaptarse rápidamente a nuevos requerimientos y mejorar continuamente los procesos es clave. la flexibilidad de Kanban permite una gestión más eficiente de los recursos en entornos cambiantes, lo que es fundamental en la industria hotelera. [6]
Por lo anterior Kanban se adapta bien a la creación de una herramienta que se va a actualizar y permite mejorar progresivamente conforme se vayan obteniendo nuevos requisitos del cliente y obtener un feedback
Ventajas de Kanban para el Proyecto:

- Visualización del flujo de trabajo: permite tener un control claro de las tareas y su estado en todo momento, lo cual es crucial cuando se gestionan múltiples procesos en paralelo como reservas, registros de huéspedes y facturación.
- Priorización de tareas: Kanban facilita la identificación de tareas más urgentes y su rápida implementación, lo cual es vital en entornos donde las operaciones diarias requieren rapidez y precisión.

1.5.3. Gestión del Proyecto con Kanban y Trello

La aplicación de Kanban se realizó mediante Trello, un tablero digital, en el cual se representó el flujo de trabajo del proyecto a través de columnas que reflejaban el estado de cada tarea. El tablero se estructuró inicialmente con las siguientes columnas:

- Lista de tareas: contenía todas las funcionalidades, mejoras y tareas identificadas a partir del análisis de requerimientos y de los objetivos del proyecto.
- En proceso: incluía las tareas que estaban siendo desarrolladas activamente, tanto en backend como en frontend.
- Hecho: incluía las tareas finalizadas y validadas, listas para su uso o documentación.
- Historias de usuario: columna destinada a la definición de historias de usuario, redactadas a partir de los requerimientos funcionales del sistema.
- HU hecas: incluía las historias de usuario finalizadas y validadas.

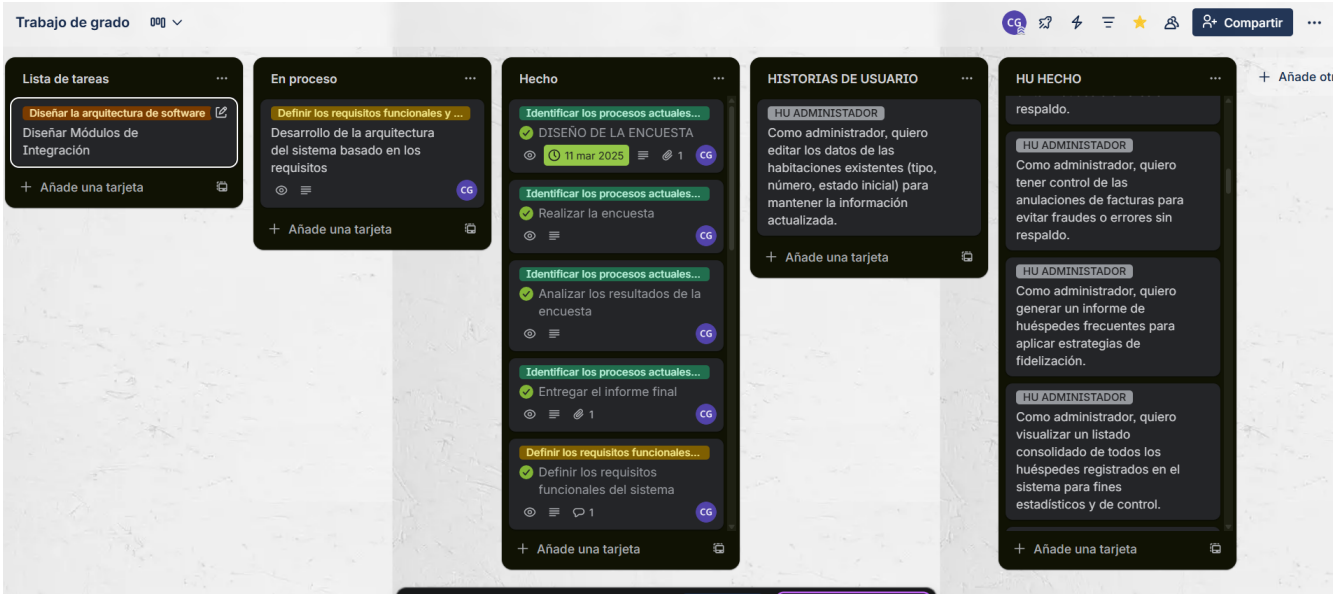

Figura 1.1: Tablero trello

Cada funcionalidad del sistema fue gestionada como un ticket Kanban, el cual incluía una descripción clara de la tarea, el módulo afectado (reservas, check-in, facturación, integración OTA, seguridad, entre otros) Esta gestión permitió dividir el proyecto en tareas pequeñas y manejables, facilitando el seguimiento del avance y la detección temprana de bloqueos o retrasos.

Durante el desarrollo del proyecto, el tablero fue actualizado de manera constante. A medida que se avanzaba en la implementación del sistema y se obtenía retroalimentación, se crearon nuevas historias de usuario y nuevas tareas, reflejando la naturaleza incremental del desarrollo. Por ejemplo, al diseñar y aplicar la encuesta de análisis de procesos y posteriormente la encuesta de validación con usuarios finales, se generaron tareas específicas que incluían la elaboración del instrumento, la aplicación de la encuesta y el análisis de resultados, adjuntando los respectivos enlaces e informes dentro de los tickets correspondientes.

Figura 1.2: Ejemplo de ticket

El uso de Kanban permitió también priorizar tareas críticas, como el desarrollo del módulo de reservas, la gestión del check-in y check-out, y la integración con plataformas OTAs mediante Channex, asegurando que estas funcionalidades estuvieran implementadas y validadas antes de avanzar a etapas posteriores del sistema

En conclusión, la aplicación de la metodología Kanban permitió una gestión eficiente del proyecto, favoreciendo la visualización del progreso, la creación continua de nuevas historias de usuario, la adaptación a cambios y la mejora progresiva del sistema. Esta metodología resultó adecuada para el desarrollo del sistema de gestión hotelera, ya que facilitó la organización del trabajo y garantizó la entrega de un producto funcional y alineado con las necesidades del entorno hoteler

Capítulo 2

Alcance del proyecto

2.1. Alcance

El desarrollo de este proyecto tiene como objetivo el diseño y la creación de una aplicación web para la gestión integral de operaciones en hoteles pequeños y medianos ubicados principalmente en la región del Valle del Cauca, Colombia. Esta herramienta está dirigida a abordar los desafíos operativos comunes del sector hotelero, específicamente en aquellos establecimientos que carecen de sistemas automatizados para la gestión de sus operaciones diarias.
La aplicación web está enfocada en automatizar y agilizar procesos críticos como la gestión de reservas, el registro de huéspedes, el control de habitaciones, la facturación electrónica y los informes operativos. Uno de los aspectos más importantes de este sistema es la automatización de la gestión de reservas, administrar la disponibilidad y asignación de habitaciones en tiempo real, asegurando que no haya reservas duplicadas ni conflictos de asignación. Este enfoque reduce significativamente la cantidad de errores humanos que a menudo ocurren en los sistemas de entrada manual [7].
Además, se integrará con plataformas de reservas en línea, conocidas como OTAs (Online Travel Agencies). Las OTAs son sistemas de intermediación digital que facilitan la conexión entre los hoteles y los clientes potenciales, permitiendo la realización de reservas en tiempo real. Según Gómez y Torres, "Las OTAs constituyen una herramienta esencial en la modernización de la gestión hotelera, permitiendo la conexión directa entre los establecimientos y los clientes potenciales a través de plataformas digitales"[8]. Esta integración permitirá optimizar el proceso de reservas y reducir la dependencia de métodos tradicionales, contribuyendo a una mayor eficiencia operativa. El sistema estará compuesto por varias funcionalidades clave, que incluyen:

- Gestión de reservas: los clientes podrán realizar sus reservas en línea, mientras que los administradores del hotel podrán gestionar la disponibilidad y asignación de habitaciones en tiempo real, asegurando que no haya duplicación de reservas ni conflictos en la asignación.

  Esta funcionalidad contribuye a una mejor organización de las operaciones diarias y asegura que las habitaciones sean gestionadas de manera eficiente, maximizando la ocupación del hotel y minimizando la posibilidad de errores en la asignación de las mismas [7]

- Registro de huéspedes: el sistema permitirá realizar el registro de huéspedes de manera rápida y sencilla, recopilando información como nombre, documento de identidad, detalles de contacto y preferencias, lo que facilitará un proceso de check-in ágil.

- Control de habitaciones: a través de este módulo, los administradores podrán conocer en todo momento el estado de las habitaciones (ocupadas, disponibles, en mantenimiento), facilitando una mejor organización de las operaciones y maximizando la ocupación del hotel.
- Facturación y pagos: la aplicación generará facturas electrónicas conforme a la normativa colombiana, y permitirá procesar pagos en línea, optimizando el proceso de cobro y reduciendo el riesgo de errores en los registros financieros.
- Generación de informes: el sistema permitirá a los administradores generar informes detallados sobre las operaciones del hotel, incluyendo ingresos, ocupación, gastos y más, para tomar decisiones informadas basadas en datos actualizados.

  la generación automática de informes detallados sobre ingresos, ocupación, gastos y otros aspectos operativos permite a los administradores tomar decisiones informadas basadas en datos actualizados. Estos informes se generan en tiempo real, eliminando la necesidad de generar reportes manualmente, proceso propenso a errores y que consume mucho tiempo [9]
- Integración con plataformas de reservas en línea (OTAs): esta funcionalidad conecta la aplicación con OTAs, permitiendo la realización de reservas en tiempo real y ampliando el alcance comercial del hotel. La integración con OTAs facilita la conexión entre los establecimientos y clientes potenciales, contribuyendo a una mayor visibilidad y optimización del proceso de reserva

Además, se incorporará una funcionalidad de personalización del sistema, lo que permitirá que cada hotel pueda adaptar la plataforma a sus necesidades particulares, ya sea modificando los tipos de habitaciones, ajustando las tarifas según la temporada, o añadiendo servicios especiales que se ajusten a la demanda local.

Capítulo 3

Marco referencial

En este capítulo se describen los conceptos y definiciones necesarias para el entendimiento de las estrategias a implementar en el desarrollo del proyecto.

3.1. Marco Teórico

El marco teórico de este proyecto se fundamenta en diversas teorías relevantes al desarrollo de sistemas tecnológicos y su aplicación en la gestión hotelera. Estas teorías proporcionan un contexto conceptual para entender los elementos clave del problema y su solución..

- Teoría de Sistemas: la teoría de sistemas establece que cualquier organización, como un hotel, puede ser vista como un conjunto de componentes interrelacionados que trabajan en armonía para alcanzar objetivos específicos. La implementación de una aplicación web para la gestión hotelera mejora la interacción entre subsistemas (reservas, facturación, inventarios) para optimizar el funcionamiento global. [10]
- Teoría de las Restricciones (TOC): identificar y gestionar las limitaciones es esencial para optimizar el rendimiento del sistema hotelero. La TOC sugiere que la solución tecnológica debe enfocarse en resolver las restricciones operativas, como la falta de automatización o errores manuales en la gestión de reservas y pagos. [11]
- Teoría de la Calidad: esta teoría, basada en la mejora continua y la satisfacción del cliente, respalda la importancia de implementar una solución que garantice un servicio de alta calidad, reduciendo errores y optimizando la experiencia del usuario. [12]
- Teoría de la Relatividad Organizacional: propone que la efectividad de un sistema depende de su capacidad de adaptarse al entorno cambiante. La aplicación web debe ser escalable, adaptable y alineada con las necesidades específicas del hotel.
- Teoria de la innovación Disruptiva: esta teoria sugiere que las nuevas tecnologías transforman sectores establecidos. La implementacion de un sistema de gestión hotelera mejorar significativamente la manera que los hoteles gestion sus operaciones. [13]

3.2. Estado del arte

El desarrollo de sistemas de gestión hotelera ha evolucionado significativamente en las últimas décadas, impulsado por la necesidad de optimizar procesos operativos, mejorar la eficiencia administrativa.

Sistemas de Gestión Hotelera (PMS):
Plataformas como Opera PMS y Cloudbeds son ampliamente utilizadas a nivel global. Estas ofrecen funcionalidades avanzadas para la gestión de reservas, facturación y generación de reportes en tiempo real, facilitando la administración integral de establecimientos hoteleros. No obstante, presentan ciertas limitaciones como altos costos de implementación y una curva de aprendizaje significativa para el personal operativo [14]

Tecnologías emergentes en la gestión hotelera
La integración de tecnologías emergentes como la inteligencia artificial (IA) y los chatbots ha revolucionado la industria hotelera. Estas tecnologías permiten automatizar la atención al cliente, gestionar reservas de manera eficiente y analizar grandes volúmenes de datos para la toma de decisiones

Tendencias en la arquitectura de software
El diseño de arquitecturas de software escalables y modulares es una tendencia clave en el desarrollo de sistemas de gestión hotelera. En este sentido, se ha promovido el uso de arquitecturas basadas en microservicios, que facilitan la integración de nuevos módulos sin afectar la operatividad del sistema principal. Además, el enfoque en plataformas web responsivas permite la accesibilidad desde dispositivos móviles, lo cual es fundamental para la gestión en tiempo real. [15]

Evaluación de sistemas de gestión
Diversos estudios han evaluado la eficacia de los sistemas de gestión hotelera existentes, destacando la importancia de la usabilidad y la eficiencia operativa como factores críticos para el éxito. La incorporación de métricas de rendimiento y análisis de datos ha permitido identificar áreas de mejora en los procesos internos de los hoteles, optimizando la experiencia del usuario final. [16]

Ciberseguridad en sistemas de gestión hotelera
Los sistemas de gestión hotelera deben implementar protocolos de seguridad robustos, como el cifrado de datos, la autenticación multifactor y la gestión de vulnerabilidades, para proteger la información sensible de los clientes, es fundamental que los hoteles adopten medidas proactivas para proteger la información de sus clientes y mantener la integridad de sus operaciones frente a posibles amenazas cibernéticas. [17]

La necesidad de un sistema de gestión hotelera flexible, escalable y adaptados a las necesidades específicas de cada establecimiento, la integración de tecnologías emergentes y el diseño de arquitecturas modulares son factores determinantes para el desarrollo de soluciones innovadoras en este ámbito.

3.3. Antecedentes

El desarrollo de sistemas de gestión hotelera ha sido un área de constante innovación, impulsada por la necesidad de mejorar la eficiencia operativa, la industria hotelera, como parte fundamental del sector turístico, ha adoptado diversas tecnologías para enfrentar diferentes desafíos que se presentan.
A pesar de la diversidad de soluciones disponibles, muchas de ellas presentan limitaciones que afectan su adopción, especialmente en hoteles pequeños y medianos. Estas limitaciones incluyen altos costos de implementación, falta de personalización, complejidad en el uso y problemas de escalabilidad. Además el análisis de datos en tiempo real, aspectos que no siempre son abordados por los sistemas tradicionales.

Proyectos similares en la gestión hotelera:

- Opera PMS: Opera PMS es uno de los sistemas de gestión hotelera más utilizados a nivel mundial. Ofrece funcionalidades avanzadas como la gestión de reservas, facturación y generación de reportes en tiempo real. Sin embargo, su alto costo de implementación y la complejidad de uso representan barreras significativas para pequeños y medianos hoteles. [14]
- Cloudbeds: plataforma que integra la gestión de reservas, administracións y la contabilidad. Aunque es más accesible aún presenta desafíos relacionados con la personalización y la dependencia de la conectividad a internet para su funcionamiento óptimo
- Hotelerum: Hotelerum es una solución enfocada en la optimización de reservas en línea. Aunque es eficaz en la gestión de reservas, no cubre completamente otros aspectos críticos como la facturación y la generación de informes operativos detallados. [18]
- Soluciones Personalizadas: algunos medianos y pequeños hoteles han optado por desarrollar soluciones personalizadas que se adaptan a sus necesidades específicas. Estos sistemas permiten una mayor flexibilidad, pero pueden presentar problemas de escalabilidad y mantenimiento a largo plazo debido a la falta de soporte técnico especializado. [15]

Tabla 3.1: Comparación de proyectos antecedentes

Proyecto	**Pros**	**Contras**
Opera PMS	Funcionalidades avanzadas, gestión en tiempo real	Alto costo, curva de aprendizaje significativa.
Cloudbeds	Integración completa de gestión y contabilidad	Limitada personalización, dependencia de internet
Hotelerum	Optimización de reservas en línea, visibilidad aumentada	Gestión limitada de facturación e informes operativos
Soluciones Personalizadas	Adaptación específica a necesidades del hotel	Problemas de escalabilidad y mantenimiento técnico.

Estos antecedentes destacan la necesidad de un sistema que no solo integre funcionalidades clave de gestión, sino que también ofrezca personalización, accesibilidad y escalabilidad, abordando las limitaciones observadas en las soluciones actuales. La oportunidad radica en desarrollar una plataforma que combine lo mejor de las soluciones existentes con tecnologías emergentes, adaptándose a las necesidades específicas de los hoteles medianos y pequeños en el Valle del Cauca y ofreciendo una alternativa más asequible y flexible.

3.4. Marco Conceptual

El marco conceptual define los términos y conceptos fundamentales relacionados con el área de negocio y el proyecto:

- Gestión Hotelera: proceso integral que incluye la administración de reservas, inventarios, finanzas y atención al cliente.
- Aplicación Web: plataforma tecnológica accesible desde navegadores, diseñada para optimizar procesos operativos y administrativos.
- Automatización: uso de tecnologías para minimizar la intervención manual en procesos rutinarios, reduciendo errores y mejorando la eficiencia.
- Satisfacción del Cliente: indicador clave en la industria hotelera, directamente influenciado por la calidad del servicio y la rapidez en los procesos.
- Área de Negocio: el enfoque de este proyecto se centra en el sector hotelero , específicamente en la mejora de sus procesos internos mediante herramientas tecnológicas.

Capítulo 4

Análisis del negocio y levantamiento de requerimientos

4.1. Contexto y caracterización del entorno

La operación diaria de los hoteles pequeños y medianos depende en gran medida de la correcta gestión de procesos administrativos y operativos que involucran áreas como reservas, recepción, control de habitaciones y facturación. En este contexto, los sistemas de gestión hotelera o Property Management System (PMS) se han consolidado como herramientas fundamentales para integrar y automatizar actividades. permitiendo administrar reservas, disponibilidad de habitaciones, procesos de check-in y check-out, así como la generación de facturación y reportes operativos. La implementación adecuada de estos sistemas contribuye directamente a mejorar la eficiencia operativa y la calidad del servicio ofrecido a los huéspedes. [19]

En el caso de hoteles con una capacidad reducida (frecuentemente inferior a 20 habitaciones) la gestión de estas actividades suele recaer en un número limitado de personas, generalmente personal de recepción y administración. En muchos establecimientos pequeños aún se apoyan en registros manuales, hojas de cálculo o mediante herramientas informáticas no integradas. Diversos estudios sobre adopción de tecnologías de información en el sector hotelero señalan que las limitaciones de recursos y capacidades organizacionales en hoteles pequeños dificultan la implementación de sistemas tecnológicos integrados, lo que puede generar errores en la asignación de habitaciones, retrasos en los procesos de atención al cliente y problemas en el control de la información operativa [20].

Asimismo, el crecimiento del uso de plataformas de reservas en línea (Online Travel Agencies u OTAs), como Booking o Expedia, ha incrementado la complejidad operativa de los hoteles independientes, ya que la gestión manual de estas plataformas puede producir inconsistencias entre la disponibilidad real del establecimiento y la información publicada en línea. [21], la integración tecnológica entre sistemas internos y plataformas externas se ha convertido en un factor clave para la competitividad de pequeñas empresas turísticas, especialmente en destinos emergentes donde la digitalización del sector aún se encuentra en proceso de consolidación.

4.2. Metodología de levantamiento de información

El levantamiento se realizó mediante una encuesta estructurada aplicada a tres (3) participantes con roles directamente vinculados a la operación del hotel: dos recepcionistas y un administrador, todos pertenecientes a un hotel pequeño (menos de 20 habitaciones). El tamaño reducido de la muestra responde a un criterio de pertinencia operativa: los participantes cubren los dos roles que concentran la totalidad de las interacciones con los procesos que el sistema busca automatizar. Sommerville [22] establece que en proyectos de software para organizaciones pequeñas, la participación directa de los operadores del sistema es más determinante para la calidad de los requisitos que el tamaño muestral, siempre que los roles cubiertos sean representativos del dominio del problema.

El enfoque de la encuesta fue identificar:

1. Procesos actuales del hotel soportados por herramientas digitales (PMS).
2. Procesos manuales que aún requieren intervención humana.
3. Problemas recurrentes en reservas, check-in/check-out y facturación.
4. Uso de OTAs y dificultades de integración.
5. Nivel de importancia percibida frente a una integración OTA y automatización.

La encuesta se diseñó con preguntas cerradas (selección múltiple) y una sección final abierta de sugerencias, con el fin de obtener tanto datos cuantificables como observaciones cualitativas útiles para el análisis.

4.3. Resultados de la encuesta

4.3.1. Perfil de participantes y contexto del hotel

Los participantes encuestados se distribuyeron de la siguiente manera:

- Recepcionista: 2 participantes
- Administrador: 1 participante

Todos los participantes afirmaron trabajar en un hotel de tamaño pequeño (menos de 20 habitaciones), lo cual coincide con el enfoque y alcance del proyecto. Respecto a las herramientas actuales, los tres participantes indicaron que el hotel utiliza un software de gestión hotelera (PMS) como sistema principal para apoyar la operación diaria.

4.3.2. Procesos principales soportados por el sistema actual

Los encuestados reportaron que el PMS actualmente gestiona de forma digital:

- Gestión de reservas
- Facturación y pagos
- Control de habitaciones y mantenimiento
- Reportes y análisis de datos

En un caso adicional se mencionó la gestión del personal, lo que sugiere que algunos establecimientos incorporan funciones ampliadas dependiendo del PMS empleado o de las prácticas internas.

4.3.3. Procesos que aún se realizan de forma manual

A pesar del uso del PMS, los participantes señalaron procesos que aún se gestionan de forma manual:

- **Gestión de limpieza y mantenimiento:** reportado por los tres participantes.
- **Comunicación con plataformas de reservas (OTAs):** reportado por los tres participantes.

La persistencia de procesos manuales evidencia que la digitalización del establecimiento no es completa. Buhalis y O'Connor [23] advierten que esta fragmentación tecnológica es uno de los problemas más frecuentes en hoteles pequeños e independientes: el PMS gestiona operaciones internas, pero carece de integración con los canales externos de distribución, obligando al personal a ejecutar actualizaciones manuales con alto riesgo de error.

4.3.4. Problemas frecuentes identificados

A continuación se presentan los principales problemas identificados por área operativa:

1. **Gestión de reservas:** la problemática predominante fue la falta de sincronización con plataformas externas (OTAs), señalada por dos de los tres participantes. Un tercer participante reportó adicionalmente la ocurrencia de doble reserva o sobreventa de habitaciones. Estos hallazgos son coherentes con lo documentado por Buhalis y Law [24], quienes identifican la desconexión entre el sistema interno del hotel, los canales de distribución en línea y la falta de sincronizan disponibilidad en tiempo real generan inconsistencias que se traducen en sobreventa y conflictos de asignación difíciles de resolver manualmente.
2. **Check-in y check-out:** los tres participantes coincidieron en señalar que los procesos de recepción son lentos y generan esperas innecesarias. Esto confirma la necesidad de automatizar el flujo operativo de recepción.
3. **Facturación y pagos:** se identificaron dos tipos de dificultades: errores en cálculos y cobros (dos participantes) y falta de integración de la facturación con otros módulos del sistema (un participante). Abukhalifeh y Pratt [25] señalan que la automatización de la gestión de cuentas y liquidación de folios es una de las funcionalidades centrales que distingue a un PMS maduro de soluciones parciales, ya que garantiza coherencia entre los datos de reserva, consumo y cobro sin depender de cálculos manuales.

4.4. Especificación de requisitos

Los requisitos funcionales y no funcionales de este proyecto se documentan siguiendo el estándar IEEE 830 [26], cuyo propósito es describir las características que debe tener una especificación de requisitos de software de calidad: correcta, no ambigua, completa, consistente, verificable, modificable y trazable..

4.4.1. Requisitos funcionales (RF)

Módulo 1. Gestión de reservas

- RF-01. El sistema debe permitir al recepcionista o administrador registrar reservas manualmente (ingresando nombre del cliente, documento de identificación, fecha de entrada, fecha de salida, tipo o número de habitación y número de huéspedes) y visualizar la disponibilidad de habitaciones en tiempo real.
- RF-02. El sistema debe permitir modificar o cancelar reservas existentes, manteniendo un historial de cambios.

Módulo 2. Registro de huéspedes

- RF-03. El sistema debe permitir registrar la información básica del huésped (nombre, tipo documento, documento, correo, numero de celular, ciudad).
- RF-04. El sistema debe permitir realizar el proceso de check-in y check-out de manera digital.
- RF-05. El sistema debe guardar el historial de estancias por cada huésped.

Módulo 3. Control de habitaciones

- RF-06. El sistema debe mostrar el estado de todas las habitaciones (disponible, ocupada, en mantenimiento).
- RF-07. El sistema debe permitir actualizar el estado de las habitaciones de forma manual o automática según check-in/check-out.
- RF-08. El sistema debe permitir asociar servicios adicionales (ej. minibar, lavandería) a una habitación.

Módulo 4. Facturación y pagos

- RF-09. El sistema debe generar facturas asociadas a reservas y consumos registrados.
- RF-10. El sistema debe almacenar un historial de transacciones por cada reserva.

Módulo 5. Reportes

- RF-11. El sistema debe generar informes automáticos de ocupación, ingresos y gastos.
- RF-12. El sistema debe permitir filtrar los informes por fechas, tipos de habitación y fuente de reserva.
- RF-13. El sistema debe exportar los informes a formatos como PDF y Excel.

Módulo 6. Integración con OTAs

- RF-14. El sistema debe registrar reservas realizadas desde las OTAs en el módulo de reservas.
- RF-15. El sistema debe sincronizar automáticamente la disponibilidad de habitaciones con las OTAs conectadas.
- RF-16. El sistema debe actualizar tarifas en tiempo real en todas las plataformas conectadas.

Módulo 7. Personalización

- RF-17. El sistema debe permitir personalizar los tipos de habitación disponibles.
- RF-18. El sistema debe permitir modificar tarifas por temporada o promociones.
- RF-19. El sistema debe permitir configurar servicios especiales y valores adicionales.

Módulo 8. Gestión de usuarios y seguridad

- RF-20. El sistema debe permitir crear, modificar y eliminar usuarios del sistema.
- RF-21. El sistema debe asignar roles y permisos diferenciados a cada usuario.
- RF-22. El sistema debe registrar auditorías de acciones realizadas por cada usuario.

4.4.2. Requisitos no funcionales (RNF)

Rendimiento

- RNF-01. El sistema debe cargar la página principal en menos de 5 segundos con una conexión promedio.
- RNF-02. Las acciones clave (registrar huésped, generar factura) deben ejecutarse en menos de 10 segundos.

Usabilidad

- RNF-03. El sistema debe tener una interfaz amigable y comprensible para usuarios no técnicos.

- RNF-04. Los elementos de navegación (botones, menús y formularios) deben ser visibles y etiquetados claramente.
- RNF-05. El sistema debe mantener consistencia visual y estructural en todas sus pantallas.

Seguridad

- RNF-06. Solo usuarios autenticados podrán acceder al panel administrativo.
- RNF-07. Las contraseñas deben almacenarse cifradas en la base de datos.
- RNF-08. El sistema debe cerrar sesión automáticamente tras 30 minutos de inactividad.

Mantenibilidad

- RNF-09. El sistema debe tener el código organizado por módulos o componentes reutilizables.
- RNF-10. El código debe estar comentado para facilitar futuras modificaciones.

Disponibilidad y confiabilidad

- RNF-12. En caso de error, el sistema debe mostrar mensajes claros para el usuario.

Compatibilidad

- RNF-14. No se requiere compatibilidad con aplicaciones móviles o nativas.

Capítulo 5

Diseño del sistema

5.1. Introducción

El diseño del sistema de gestión hotelera Hotel Manager PMS establece la estructura técnica sobre la cual se construye toda la lógica operativa del establecimiento: reservas, control de habitaciones, check-in/check-out, facturación, reportes, administración y seguridad, incluyendo la integración con plataformas externas de reservas en línea (OTAs).

Fowler [15] establece que la separación de la aplicación en capas bien definidas (presentación, lógica de negocio y persistencia) es uno de los principios fundamentales para construir sistemas de información mantenibles y escalables. Este principio arquitectónico guía todas las decisiones de diseño adoptadas en el presente proyecto.

En este capítulo se presentan: la arquitectura general del sistema en tres capas, la justificación y arquitectura detallada del backend y del frontend, el esquema de base de datos relacional, el diseño de los módulos funcionales principales y el diseño de la interfaz de usuario con sus respectivos mockups de pantallas representativas.

5.2. Tecnologías y componentes del sistema

El sistema Hotel Manager PMS fue construido sobre el stack PERN (PostgreSQL, Express, React, Node.js), complementado con librerías especializadas para cada funcionalidad:

- **Frontend:** React, Bootstrap + CSS, Axios
- **Backend:** Node.js con Express.js, API REST,autenticación JWT, middlewares de seguridad por roles.
- **Base de datos:** PostgreSQL.
- **Generación de reportes:** PDFKit para documentos PDF y SheetJS para exportación en Excel.
- **Integración OTA:** Channex.io como channel manager intermediario con plataformas como Booking.com, Airbnb y otras, vía webhooks HTTP.

5.3. Arquitectura general del sistema

El sistema sigue una arquitectura cliente-servidor de tres capas, ampliamente adoptada en el desarrollo de aplicaciones web modernas [15]. Esta separación garantiza que cada capa evolucione de forma independiente, facilita las pruebas y el mantenimiento, y permite escalar cada componente sin afectar a los demás.

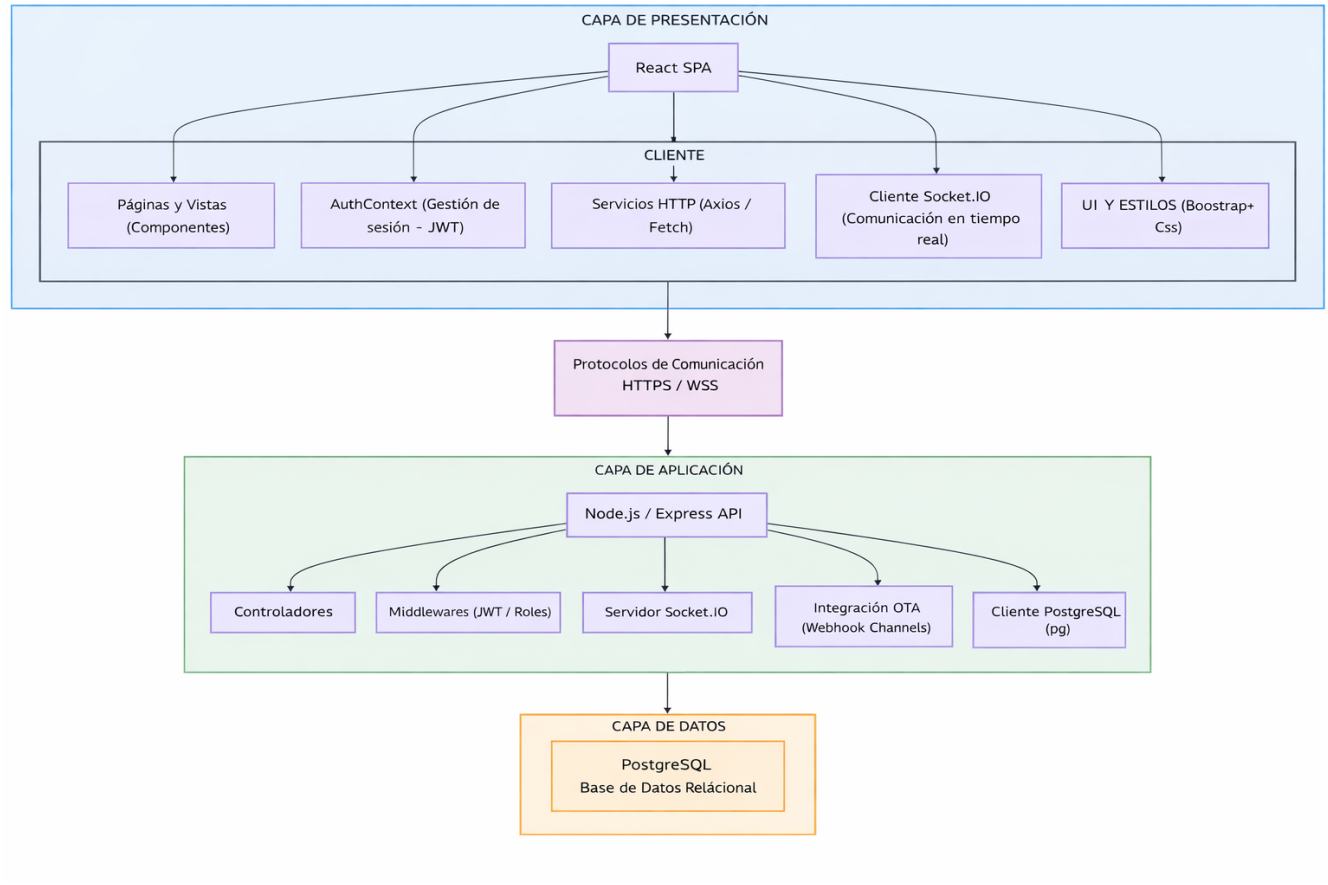

Figura 5.1: Arquitectura general del sistema

Como se observa en la Figura 5.1, el sistema está organizado en tres capas claramente diferenciadas, comunicadas de forma secuencial:

5.3.1. Capa de presentación (cliente)

La capa superior corresponde a la interfaz de usuario, implementada como una aplicación de página única (SPA) desarrollada en React. Esta capa se ejecuta completamente en el navegador web del usuario y está compuesta por cinco elementos principales:

- **Páginas:** Representan las secciones funcionales del sistema (Dashboard, Reservas, Recepción, Habitaciones, Facturación, Reportes y Administración, etc) construidas como componentes reutilizables de React.

- **AuthContext (Gestión de sesión – JWT):** Módulo de contexto global que almacena el estado de autenticación del usuario (token JWT, nombre y rol), disponible en toda la aplicación sin necesidad de pasar propiedades entre componentes.
- **Servicios HTTP (Axios / Fetch):** Capa de servicios que encapsula todas las llamadas HTTP hacia el backend. Las páginas y vistas no realizan llamadas directas; en su lugar, importan los servicios correspondientes a su dominio. Este diseño sigue el Principio de Responsabilidad Única (SRP) de SOLID [27].
- **UI y Estilos (Bootstrap + CSS):** Bootstrap 5 combinado con CSS personalizado proporciona el sistema de diseño visual uniforme en todas las pantallas del sistema.

5.3.2. Protocolos de comunicación: HTTPS / WSS

La comunicación entre la capa de presentación y la capa de aplicación se realiza mediante dos protocolos complementarios:

- **HTTPS:** Para el intercambio de datos síncronos entre el frontend y la API REST del backend. Todas las peticiones llevan el token JWT en el encabezado `Authorization`, verificado por el middleware antes de acceder a cualquier recurso protegido.
- **WSS (WebSocket Secure):** Para la comunicación bidireccional en tiempo real entre el cliente Socket.IO y el servidor Socket.IO del backend, utilizada para la emisión de notificaciones push.

Esta combinación de protocolos permite que el sistema atienda tanto operaciones transaccionales (reservas, pagos, check-in) como eventos en tiempo real (nuevas reservas OTA, alertas operativas) de forma eficiente [28].

5.3.3. Capa de aplicación (servidor)

La capa intermedia corresponde al backend del sistema, desarrollado con Node.js y Express.js. Actúa como el núcleo de la lógica de negocio y está compuesta por los siguientes elementos:

- **Node.js / Express API:** Punto central que recibe y enruta todas las peticiones HTTP entrantes,
- **Controladores:** Módulos que contienen la lógica de negocio del sistema. Procesan las solicitudes recibidas, aplican las reglas de negocio correspondientes y coordinan el acceso a la base de datos.
- **Middlewares (JWT / Roles):** Funciones intermedias que interceptan cada petición antes de llegar al controlador. Verifican la validez del token JWT y comprueban que el rol del usuario autenticado tenga permisos para ejecutar la operación solicitada.
- **Integración OTA (Webhook Channex):** Endpoint especializado que recibe las reservas enviadas desde Channex.io mediante webhooks HTTP y las procesa automáticamente en el sistema.
- **Cliente PostgreSQL (pg):** Módulo de acceso a la base de datos a través de un pool de conexiones centralizado, que gestiona eficientemente los recursos de conexión y garantiza la ejecución de operaciones dentro de transacciones ACID.

5.3.4. Capa de datos

La capa inferior es la base de datos PostgreSQL, que almacena toda la información operativa del hotel de forma persistente. Su estructura relacional con tablas interdependientes garantiza la integridad referencial mediante claves foráneas y restricciones, y soporta operaciones transaccionales complejas como el proceso de check-in, el registro de pagos y la generación de facturas [29].

5.4. Justificación del backend

El backend de un sistema de gestión hotelera constituye la capa lógica del software, responsable del procesamiento de datos, reglas de negocio, comunicación con la base de datos y exposición de servicios a través de APIs. La elección adecuada de la tecnología para el backend es esencial para garantizar la escalabilidad, mantenibilidad, rendimiento y seguridad del sistema.
Tecnologías de Backend Evaluadas Se consideraron las siguientes tecnologías ampliamente utilizadas en el desarrollo web moderno:

- Node.js con Express.js
- Django (Python)
- Laravel (PHP)
- Spring Boot (Java)

A continuación, se realiza una comparación con base en criterios clave para proyectos académicos y sistemas modulares:

Tabla 5.1: Tabla comparativa backned

Criterio	**Express.js**	**Django**	**Laravel**	**Spring Boot**
Curva de aprendizaje	Baja	Media	Media	Alta
Rendimiento	Alto	Medio	Medio	Alto
Flexibilidad	Alta	Alta(estructurada)	Alta(estructurada MVC)	Alta(estructurada)
Comunidad y soporte	Muy activa	Activa.	Activa	Activa
Facilidad para Apis	Muy alta	Alta	Alta	Alta
Integracion con frontend	Muy bueno	Buena	Buena	Buena
Requisito despliegue	Livianos	Moderados	Moderados	Elevados

5.4.1. Justificación de la elección: Node.js con Express.js

Tras el análisis comparativo, se seleccionó Node.js con Express.js por las siguientes razones técnicas:

- **Modelo de I/O no bloqueante:** Node.js opera sobre un modelo asíncrono que permite manejar múltiples conexiones concurrentes con bajo consumo de recursos, aspecto clave en un PMS que gestiona simultáneamente peticiones HTTP y conexiones WebSocket de varios usuarios [28].
- **Ecosistema JavaScript unificado:** Al compartir lenguaje con el frontend (React.js), se reduce la carga cognitiva del desarrollo y facilita la reutilización de lógica de validación entre capas [30].

- **Adopción en la industria:** Node.js es el entorno de ejecución más utilizado a nivel global según la encuesta de desarrolladores de Stack Overflow [31], lo que garantiza amplia documentación y soporte comunitario.
- **Ligereza y modularidad:** Express.js permite construir APIs REST de forma modular, alineándose con el patrón de arquitectura en capas adoptado en el proyecto.

5.4.2. Arquitectura interna del backend

El backend implementa el patrón MVC (Modelo-Vista-Controlador) con middlewares de seguridad, organizando el procesamiento de cada petición en etapas claramente definidas.

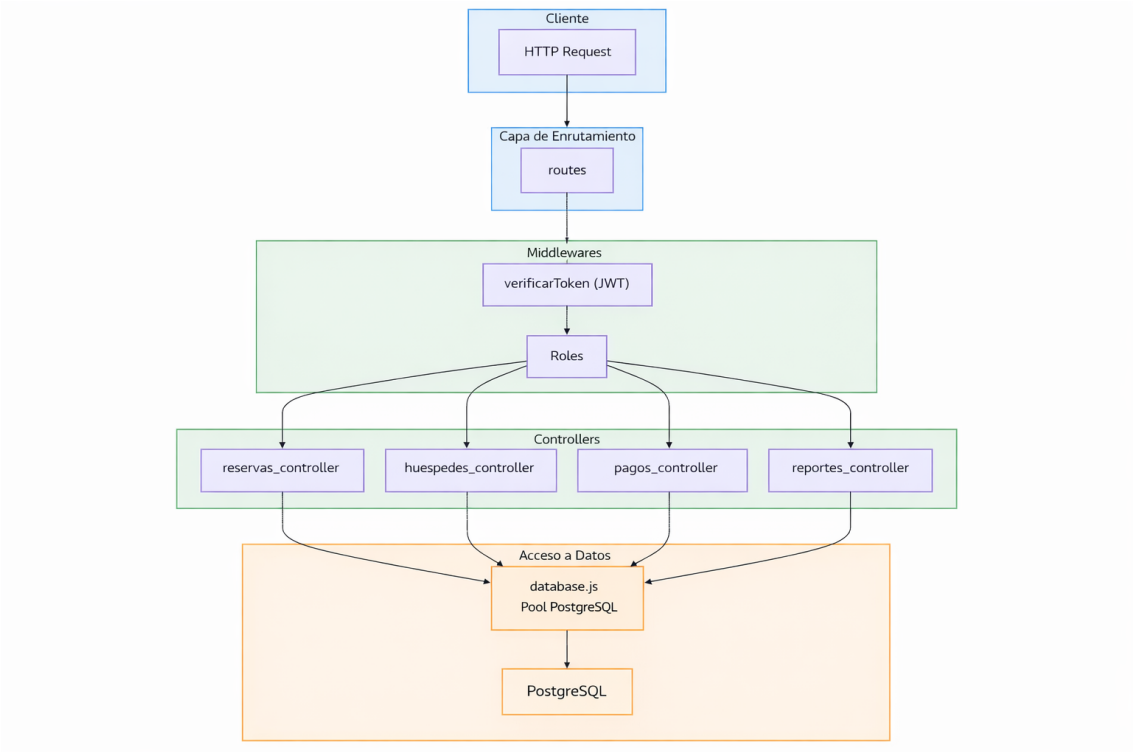

Figura 5.2: Arquitectura del backend

La Figura 5.2 muestra el flujo completo de una petición HTTP desde el cliente hasta la base de datos. El procesamiento sigue las siguientes etapas:

1. **Cliente (HTTP Request):** El cliente envía una petición HTTP con el token JWT adjunto en el encabezado de autorización. Esta petición es gestionada por la instancia de Axios configurada con interceptores automáticos.
2. **Capa de enrutamiento (routes):** Express recibe la petición y la dirige al archivo de rutas correspondiente, que identifica el controlador destino. Los archivos de rutas no contienen lógica de negocio; su única responsabilidad es declarar los verbos HTTP y delegar al controlador apropiado.

3. **Capa de middlewares:** Antes de llegar al controlador, la petición atraviesa dos middlewares de seguridad en cadena:

   - `verificarToken (JWT):` Valida la firma y vigencia del token JWT. Si el token es inválido o ha expirado, retorna HTTP 401 (No autorizado) y la petición no continúa.
   - `Roles:` Verifica que el rol del usuario autenticado tenga permisos para ejecutar la operación solicitada. Si el rol no es suficiente, retorna HTTP 403 (Prohibido).

4. **Controllers:** Una vez superada la capa de seguridad, el controlador correspondiente (reservas_controller, `huespedes_controller`, `pagos_controller` o `reportes_controller`, entre otros) ejecuta la lógica de negocio: valida los datos de entrada, aplica las reglas del dominio hotelero y determina las operaciones a realizar sobre la base de datos.

5. **Acceso a datos / Pool PostgreSQL:** Los controladores acceden a la base de datos exclusivamente a través del módulo `database.js`, que gestiona un pool de conexiones hacia PostgreSQL. Las operaciones críticas (check-in, registro de pagos, emisión de facturas) se ejecutan dentro de transacciones para garantizar la integridad de los datos.

6. **PostgreSQL:** El motor de base de datos persiste y retorna la información solicitada. El resultado viaja de regreso por la misma cadena hasta ser devuelto al cliente como respuesta JSON con el código HTTP apropiado.

Este diseño en capas garantiza que ningún componente asuma responsabilidades que no le corresponden, facilitando las pruebas unitarias de cada capa de forma aislada [27].

5.5. Justificación del Frontend

El frontend de un sistema de gestión hotelera constituye la capa de presentación del software, responsable de la interacción con el usuario, la visualización de la información y la comunicación con el backend a través de APIs. Esta capa es clave para garantizar una experiencia de usuario intuitiva, eficiente y accesible, así como para facilitar el mantenimiento y la escalabilidad de la aplicación en el tiempo.

La elección adecuada de la tecnología de frontend es esencial para asegurar un desarrollo modular, un buen rendimiento en el navegador y una correcta integración con servicios backend modernos [32].

Tecnologías de Frontend Evaluadas

- Angular
- Vue.js
- React.js

A continuación, se presenta una comparación de estas tecnologías con base en criterios relevantes para proyectos académicos y sistemas modulares.

Tabla 5.2: Tabla comparativa frontend

Tecnologias	**Ventajas**	**Desventajas**
Angular	Framework completo, herramientas integradas	Mayor complejidad, curva de aprendizaje
Vue	Fácil de aprender, ligero	Menor adopción en grandes
React.js	Flexible, gran comunidad, componentes reutilizables	Requiere configuración inicial manual

5.5.1. Justificación de la elección: React.js

React.js fue seleccionado como tecnología de frontend por las siguientes razones:

- **Arquitectura basada en componentes:** React permite dividir la interfaz en componentes reutilizables e independientes, facilitando el mantenimiento y la escalabilidad en un sistema con múltiples módulos [32].
- **Renderizado eficiente con Virtual DOM:** React minimiza las actualizaciones directas sobre el DOM real del navegador, mejorando el rendimiento en operaciones frecuentes como la actualización del estado de habitaciones [32].
- **Alta adopción en la industria:** React.js es la biblioteca frontend más utilizada según Stack Overflow [31], garantizando amplia documentación y soporte.

- **Cohesión tecnológica:** Al usar JavaScript tanto en el frontend como en el backend (Node.js), se simplifica el intercambio de datos JSON y se reduce la carga cognitiva del equipo de desarrollo [30].

5.5.2. Arquitectura interna del frontend

El frontend implementa una Service Layer que desacopla las páginas de la comunicación HTTP, siguiendo el Principio de Responsabilidad Única (SRP) [27].

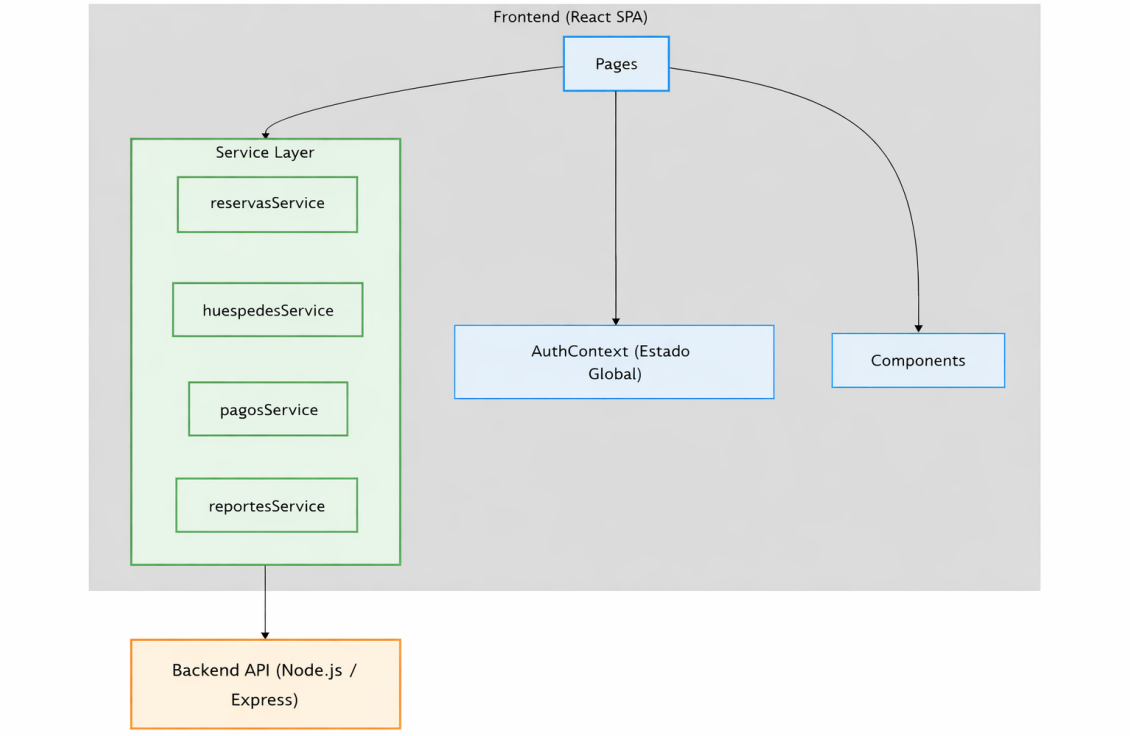

Figura 5.3: Arquitectura del frontend

La Figura 5.3 muestra la organización interna de la SPA React y cómo sus cuatro elementos principales se relacionan entre sí y con el backend:

- **Pages (Páginas):** Constituyen el punto de entrada de cada sección del sistema. Las páginas organizan la presentación de información y las interacciones del usuario, pero no realizan llamadas HTTP directas ni acceden directamente al estado global. En su lugar, delegan estas responsabilidades a los elementos especializados descritos a continuación.
- **Service Layer:** Es el único punto de contacto de la aplicación con el backend. Está compuesta por archivos de servicio dedicados por dominio: (`reservasService`, `huespedesService`, `pagosService` y `reportesService`), entre otros. Cada servicio encapsula las llamadas HTTP (a través de la instancia centralizada de Axios con interceptores JWT) y expone funciones con nombres descriptivos que las páginas pueden invocar sin conocer los detalles del protocolo HTTP. Esta separación garantiza que cambios en los endpoints del backend solo requieran modificaciones en el archivo de servicio correspondiente, sin afectar a las páginas.

- **AuthContext (Estado Global):** Módulo de contexto que centraliza el estado de autenticación (token JWT, usuario, rol). Las páginas lo consumen para determinar qué funcionalidades mostrar según el rol del usuario autenticado (administrador o recepcionista), sin necesidad de pasar esta información como propiedades entre componentes.
- **Components:** Elementos de interfaz reutilizables (Navbar, PrivateRoute, AdminRoute) que son compartidos por múltiples páginas. Los protectores de ruta (`PrivateRoute`, `AdminRoute`) bloquean el acceso a secciones protegidas sin token válido o sin el rol requerido, reforzando la seguridad en el lado del cliente.

La Service Layer se comunica directamente con el Backend API (Node.js / Express), cerrando el ciclo de comunicación del sistema. Este diseño garantiza que las páginas sean responsables exclusivamente de la presentación, y que ningún componente de la interfaz contenga lógica de acceso a datos [27].

5.6. Justificación base de datos

La base de datos es un componente esencial para el sistema de gestión hotelera, ya que almacenará información crítica como datos de huéspedes, reservas, facturación y disponibilidad de habitaciones. Se evaluaron diferentes tecnologías de bases de datos con el objetivo de seleccionar la opción que mejor se adapte a los requerimientos de consistencia, escalabilidad moderada, integridad referencial y consultas complejas.
Bases de datos evaluadas:

- MySQL: Sistema de gestión de bases de datos relacional ampliamente adoptado, con buena documentación y facilidad de uso.
- PostgreSQL: Base de datos relacional avanzada, conocida por su soporte a tipos de datos complejos, transacciones ACID, integridad de datos y extensibilidad.
- MongoDB: Base de datos NoSQL orientada a documentos, adecuada para datos no estructurados y escalabilidad horizontal.

  [33]

A continuación, se presenta una comparación de estas tecnologías con base en criterios relevantes para proyectos académicos y sistemas modulares.

Tabla 5.3: Tabla comparativa base de datos

Tecnologias	**Tipo**	**Ventajas**	**Desventajas**
MySQL	Relacional	Rápido, ampliamente usado, buen rendimiento general	Menor soporte para funciones avanzadas
PostgreSQL	Relacional	Soporte avanzado de transacciones, integridad, funciones SQL potentes, extensibilidad	Ligero aumento de complejidad
MongoDB	No relacional	Flexible, escalable horizontalmente, ideal para datos semi/no estructurados	No garantiza integridad referencial

5.6.1. Justificación de la elección: PostgreSQL

PostgreSQL fue seleccionado como motor de base de datos por las siguientes razones:

- **Estructura relacional del dominio:** El sistema hotelero trabaja con entidades fuertemente relacionadas (huéspedes, reservas, habitaciones, pagos y usuarios), lo que hace que el modelo relacional sea la elección natural para garantizar la coherencia de los datos [29].

- **Transacciones ACID:** Las operaciones críticas del sistema (check-in, registro de pagos, emisión de facturas) requieren atomicidad, consistencia, aislamiento y durabilidad, propiedades que PostgreSQL garantiza de forma nativa [29].
- **Integridad referencial:** PostgreSQL soporta claves foráneas y restricciones que protegen la coherencia entre tablas relacionadas, evitando, por ejemplo, reservas sin huésped asociado o pagos sin reserva existente.
- **Compatibilidad con el pool `pg`:** El backend utiliza la librería `pg` para acceso directo a PostgreSQL mediante un pool de conexiones, con soporte pleno para transacciones y consultas parametrizadas.
- **Licencia open-source:** PostgreSQL es de código abierto con mantenimiento activo, viable tanto para entornos académicos como para implementaciones comerciales en hoteles pequeños y medianos.

5.7. Esquema de base de datos relacional

El diseño del esquema partió de los requisitos funcionales identificados en el Capítulo 4, y cubre los siguientes dominios operativos del hotel:

- Reservas de habitaciones.
- Huéspedes y sus datos personales.
- Facturación y pagos asociados a reservas.
- Servicios adicionales consumidos por los huéspedes.
- Bloqueos de habitaciones por mantenimiento u otros motivos.
- Tarifas y planes de habitación, incluyendo configuraciones por temporada.
- Usuarios del sistema, que gestionan la operación del hotel.
- Integración con canales OTA (como Booking, Expedia) para reservas externas [23].

A continuación se describen las tablas principales y sus funciones:

HUESPEDES

Contiene la información de los clientes del hotel. Guarda datos como: nombre completo, documento de identidad, contacto, fecha de nacimiento, dirección, nacionalidad, etc. Relación: Un huésped puede tener múltiples reservas a lo largo del tiempo.

HABITACIONES

Guarda información de las habitaciones del hotel. Columnas relevantes: número, tipo, capacidad, estado (disponible/ocupada), activo (si está habilitada para reservas).
Estados de habitación:

- Disponible: se puede reservar / walk-in.
- Reservada: existe una reserva activa futura o vigente (sin check-in).
- Ocupada: hay un huésped alojado (walk-in o reserva con check-in).
- Mantenimiento / fuera de servicio: NO se puede reservar ni walk-in.

RESERVAS

Representa la reserva de una habitación por un huésped. Columnas clave: fechas de inicio y fin, estado de la reserva, plan de tarifa, check-in/check-out, facturada o no, origen (manual o canal OTA).
Estados de reserva:

- Reservada: creada sin check-in.

- Ocupada: con check-in (incluye walk-in).
- Finalizada: checkout hecho.
- Cancelada: cancelada.

RESERVA-HUESPEDES
Tabla intermedia que permite asignar varios huéspedes a una reserva.

BLOQUEOS
Guarda información de bloqueos temporales de habitaciones.
Motivos: mantenimiento, limpieza, eventos internos.
Columnas: tipo, motivo, fechainicio, fechafin, estado.

PAGOS
Registra los pagos realizados por reservas. Columnas: tipo, método, monto, referencia, fecha.

SERVICIOS-CONSUMIDOS
Guarda servicios adicionales que los huéspedes pueden consumir durante su estadía.
Columnas: costo, fecha de consumo.

TARIFAS y TIPOS-HABITACION
TIPOS-HABITACION: catálogo de tipos de habitación (suite, doble, single). TARIFAS: precios asignados a cada tipo de habitación por temporada y plan.

OTA-CONFIG
Configuración de integración con canales de reserva externos (OTA) como Booking o Expedia. Columnas: ota-canal, activa, habitaciones-incluidas, planes-incluidos.

USUARIOS
Tabla de usuarios del sistema (recepcionistas, administradores). Columnas: nombre, email, password, rol, estado. Controla acceso y roles dentro del sistema.

CONFIGURACION
Tabla de configuración general del sistema. Columnas: fecha-sistema. Guarda datos globales para la operación del sistema.

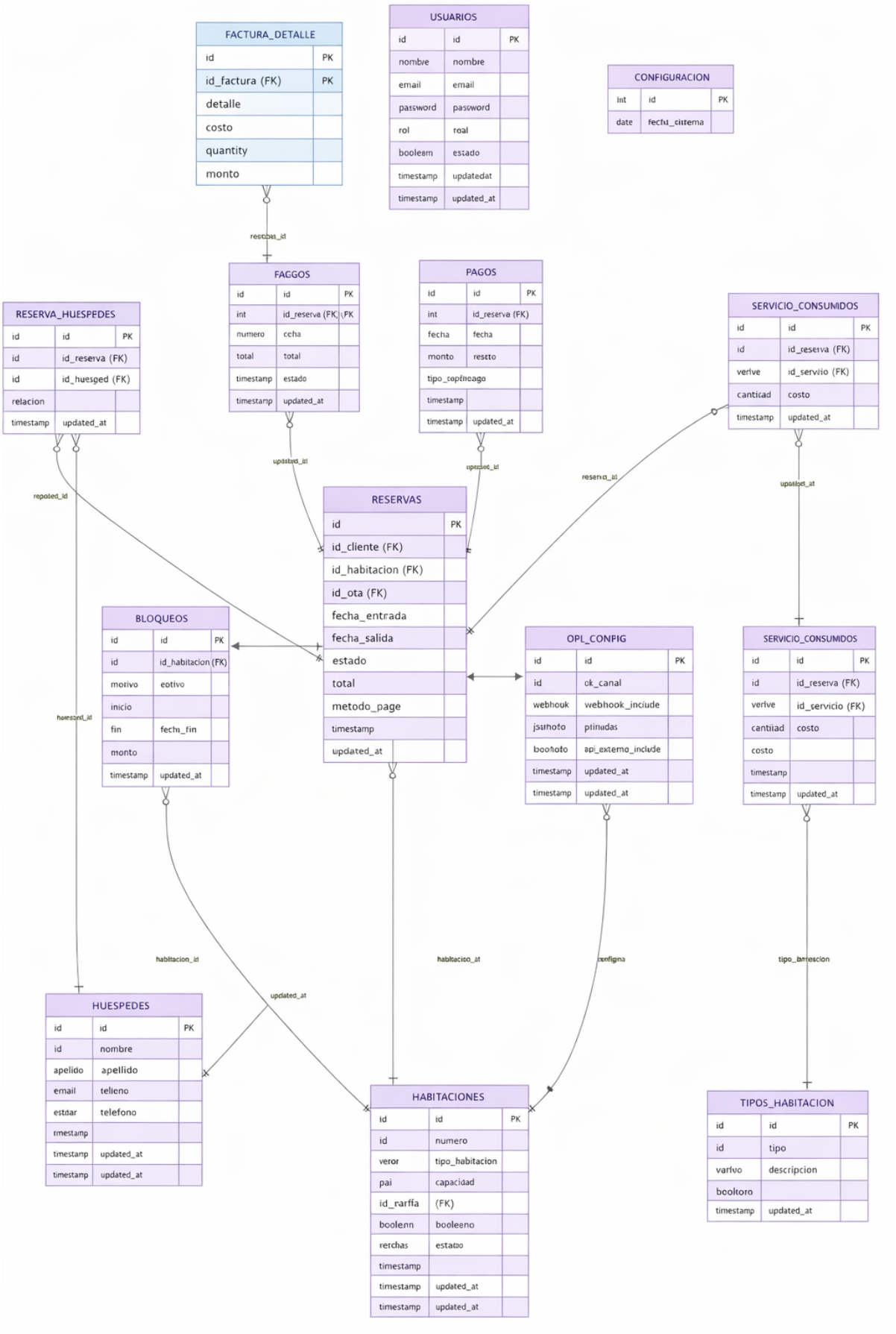

Figura 5.4: Esquema de base de datos relacional

Como se observa en la Figura 8.9, la tabla RESERVAS actúa como nodo central del esquema: referencia directamente a HUESPEDES, HABITACIONES, OTA_CONFIG para identificar el origen de cada reserva. Desde RESERVAS se desprenden las relaciones con CARGOS, PAGOS, FACTURA_DETALLE y SERVICIO_CONSUMIDOS, que conforman el módulo financiero del sistema. RESERVA_HUESPEDES actúa como tabla de intersección entre RESERVAS y HUESPEDES, permitiendo registrar múltiples huéspedes por reserva. BLOQUEOS se vincula directamente a HABITACIONES para gestionar los períodos de indisponibilidad. TIPOS_HABITACION clasifica las habitaciones y determina qué tipos se distribuyen a través de OTA_CONFIG. USUARIOS y CONFIGURACION son tablas independientes que soportan la gestión de accesos y los parámetros globales del sistema.

El sistema está diseñado para:

- Gestionar la ocupación del hotel de manera precisa (habitaciones, bloqueos, reservas).
- Administrar huéspedes y reservas grupales.
- Controlar la facturación y pagos, separando cabecera y detalle.
- Integrar servicios adicionales consumidos durante la estadía.
- Gestionar tarifas y planes dinámicos según tipo de habitación y temporada.
- Sincronizar disponibilidad y reservas externas mediante la integración con canales OTA a través del componente intermediario Channex (Channel Manager).
- Controlar accesos y roles mediante los usuarios del sistema.

Este diseño garantiza la integridad referencial entre todas las tablas del dominio hotelero a través de claves foráneas y transacciones ACID en PostgreSQL [29], asegurando que operaciones críticas como el check-in, el registro de pagos y la emisión de facturas se ejecuten de forma atómica y sin comprometer la coherencia de los datos.

5.8. Diseño de la interfaz de usuario s

Se elaboraron wireframes de las pantallas principales como parte de la fase de diseño de la interfaz de usuario.

5.8.1. Dashboard

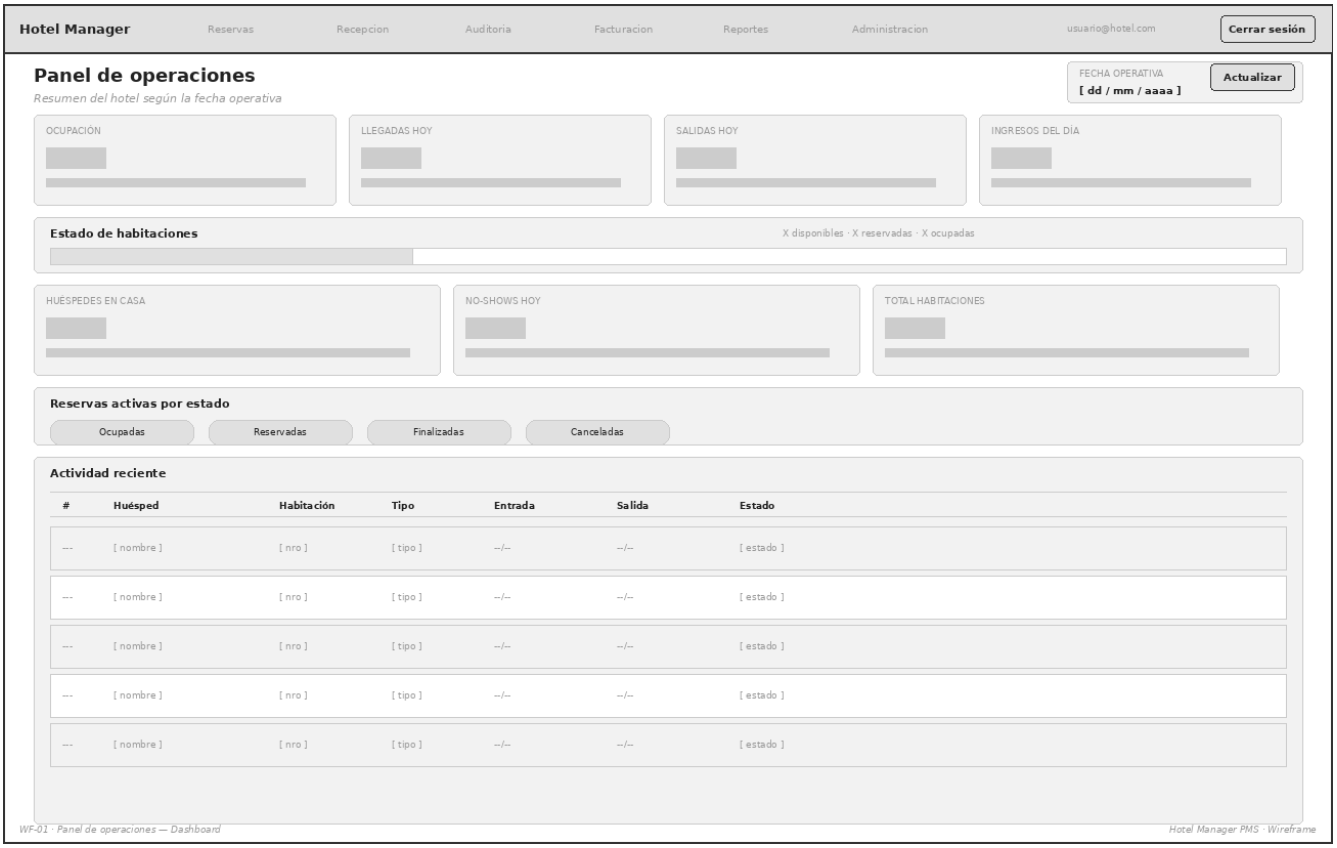

Figura 5.5: Wireframe del Dashboard

La Figura 5.5 presenta el wireframe del panel principal del sistema. Se diseñó como la primera pantalla que ve el usuario al iniciar sesión, con el objetivo de ofrecer un resumen operativo completo del hotel en un solo vistazo.

La barra de navegación superior, presente en todas las pantallas, agrupa los módulos en: Reservas, Recepción, Auditoría, Facturación, Reportes y Administración. En el extremo derecho se ubican el indicador del usuario activo con su rol y el botón de cierre de sesión.

5.8.2. Wireframe del Rack interactivo de habitaciones

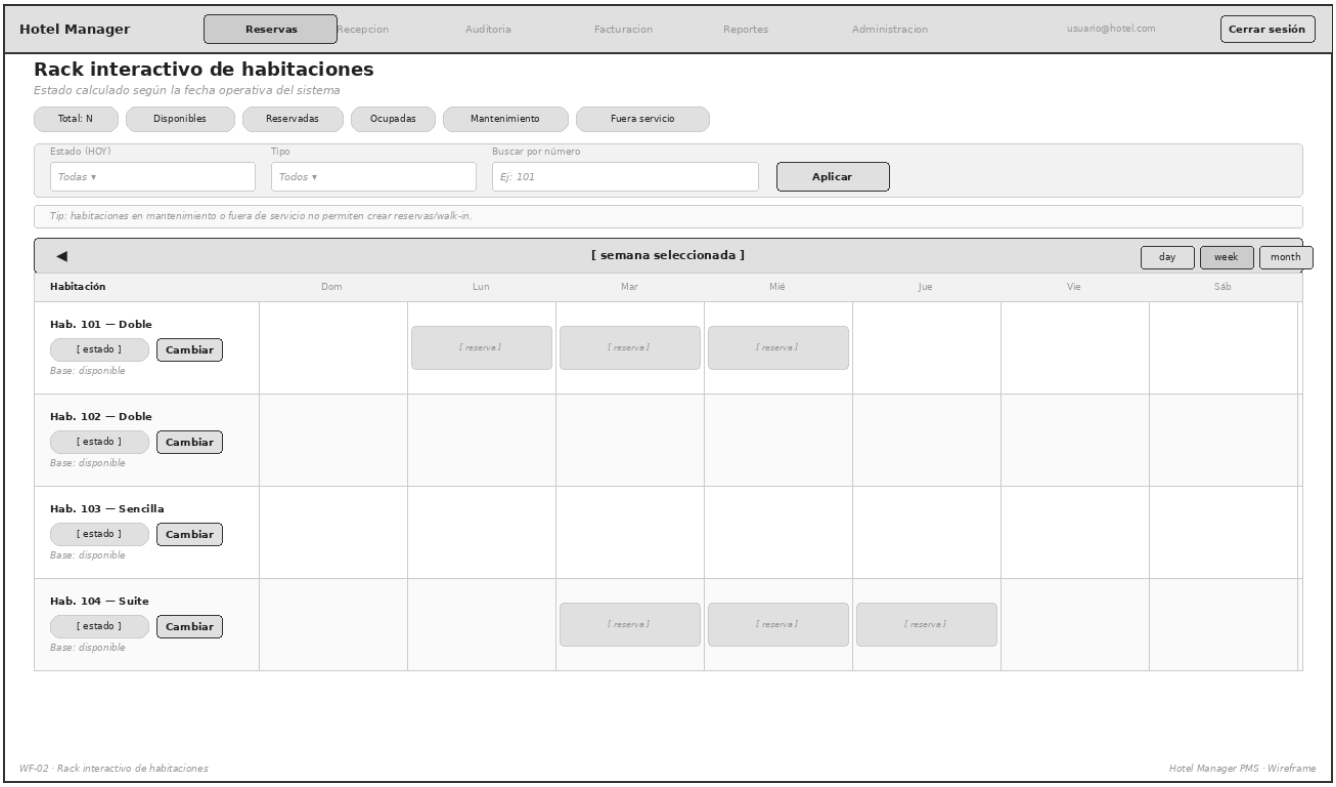

Figura 5.6: Wireframe del Rack de habitaciones

La Figura 5.6 muestra el wireframe del rack interactivo de habitaciones, accesible desde el módulo Reservas. Esta pantalla fue concebida como la herramienta central de control de disponibilidad del hotel, combinando en una sola vista el estado actual de cada habitación con su ocupación a lo largo de la semana.

5.8.3. Wireframe de Registrar Walk-In

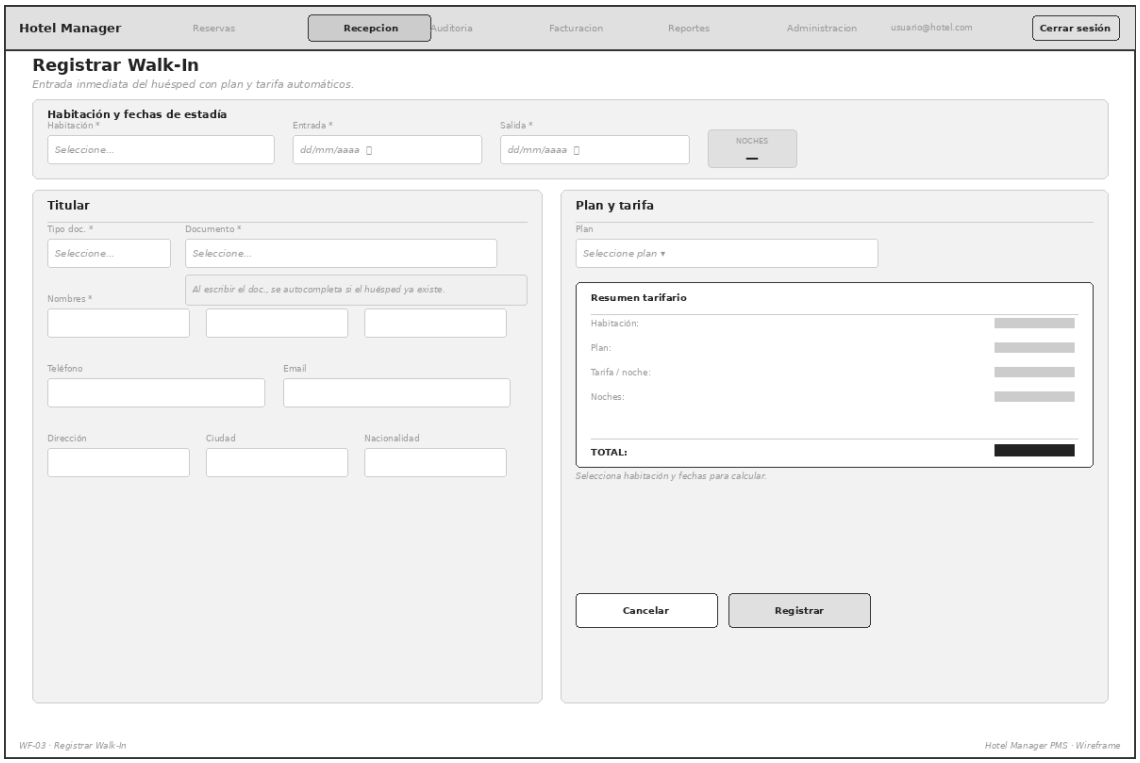

Figura 5.7: Wireframe del Formulario de registro Walk-In

La Figura 5.7 presenta el wireframe del formulario de registro de walk-in, accesible desde el módulo de Recepción. Un walk-in corresponde a la entrada inmediata de un huésped sin reserva previa.

5.8.4. WWireframe del Módulo de reportes

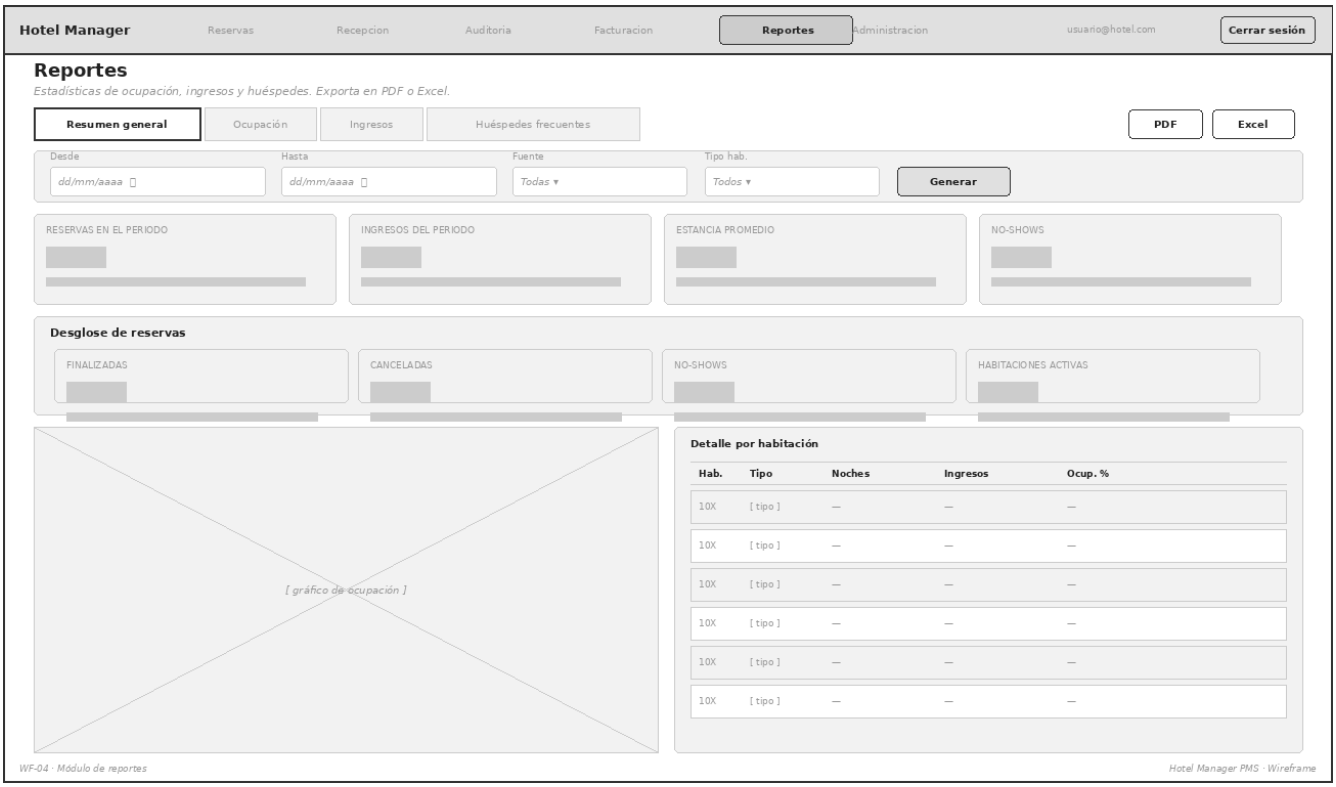

Figura 5.8: Wireframe del Módulo de reportes

La Figura 5.8 muestra el wireframe del módulo de reportes gerenciales, diseñado para ofrecer información analítica sobre la operación del hotel en períodos configurables.

La pantalla se organiza en cuatro pestañas: Resumen general, Ocupación, Ingresos y Huéspedes frecuentes. En la parte superior derecha se ubicaron los botones de exportación en PDF y Excel.

Los wireframes descritos constituyeron la base del diseño de interfaz del sistema. Su elaboración previa al desarrollo permitió definir la distribución de los elementos, la estructura de navegación y los flujos de interacción de los módulos principales, aplicando los principios de usabilidad propuestos por Nielsen [34]: visibilidad del estado del sistema, consistencia de la interfaz, retroalimentación inmediata al usuario y prevención de errores mediante validaciones en formularios. El diseño definitivo implementado respetó en todos los casos la estructura y los flujos definidos en esta fase de prototipado.

Capítulo 6

Implementación

6.1. Introducción

En este capítulo se describe el proceso de implementación del sistema de gestión hotelera propuesto, a partir de la arquitectura y el diseño definidos en el capítulo anterior. Se detallan las decisiones técnicas adoptadas, la organización del backend y frontend, así como la implementación de los principales módulos funcionales del sistema.
La implementación se realizó siguiendo la arquitectura de tres capas definida en el Capítulo 5, los controladores, los middlewares de seguridad y una capa de datos correspondiente a la base de datos PostgreSQL gestionada mediante un pool de conexiones.

6.2. Tecnologías Utilizadas

Autenticación de Usuarios
El sistema implementa un mecanismo de autenticación que garantiza que solo usuarios registrados puedan acceder al panel administrativo. Cada usuario debe iniciar sesión utilizando credenciales válidas, las cuales son verificadas por el backend antes de permitir el acceso al sistema. Las contraseñas son almacenadas de forma cifrada en la base de datos, evitando el almacenamiento de información sensible en texto plano, lo que contribuye a la seguridad del sistema.

Autorización basada en Roles
El sistema implementa un esquema de control de acceso basado en roles, permitiendo diferenciar las funcionalidades disponibles según el perfil del usuario. Los roles definidos en el sistema:

- **Administrador:** acceso completo a todas las funcionalidades del sistema, incluyendo configuraciones, gestión de usuarios, tarifas y reportes.
- **Recepcionista:** acceso limitado a funciones operativas como reservas, check-in/check-out y facturación.

Este enfoque evita accesos no autorizados y garantiza que cada usuario solo pueda ejecutar las acciones correspondientes a su rol.

Uso de Middlewares
Como se aprecia en el diagrama de arquitectura del backend (Figura 5.2), entre la capa de enrutamiento y los controladores se ubica una capa de middlewares que comprueba que el rol del usuario autenticado tenga permisos para la operación solicitada. Estos middlewares se encargan de:

- Verificar la autenticación del usuario antes de acceder a endpoints protegidos.
- Validar permisos según el rol del usuario.
- Proteger rutas críticas del sistema.
- Validar datos de entrada para evitar errores o inconsistencias.

El uso de middlewares permite centralizar la lógica de seguridad, reduciendo la duplicación de código y mejorando la mantenibilidad del sistema.

6.3. Implementación del backend

El backend del sistema fue desarrollado utilizando Node.js junto con el framework Express.js, adoptando una arquitectura modular que facilita la mantenibilidad, escalabilidad y reutilización del código. La aplicación se organizó en diferentes capas, cada una con responsabilidades claramente definidas.
La estructura general del backend se compone de los siguientes elementos:

- **Rutas (Routes):** Definen los endpoints REST y direccionan las solicitudes HTTP hacia los controladores correspondientes.
- **Controladores (Controllers):** Contienen la lógica de negocio del sistema, procesan las solicitudes y coordinan la interacción entre los servicios y la base de datos.
- **Servicios (Services):** Encapsulan procesos específicos y reglas de negocio complejas, promoviendo la reutilización del código.
- **Middlewares:** Implementan validaciones, autenticación, autorización y control de acceso.
- **Configuración:** Maneja la conexión a la base de datos, variables de entorno y parámetros globales del sistema.

Esta organización permite desacoplar las distintas funcionalidades del sistema, de modo que cada componente pueda desarrollarse, modificarse o ampliarse de manera independiente. Esto facilita la incorporación de nuevas características y el mantenimiento del software, ya que los cambios en un módulo tienen un impacto mínimo en los demás componentes del sistema [35].

6.3.1. Implementación de la API REST

La comunicación entre el frontend y el backend se realiza mediante una API REST, utilizando el formato JSON como medio de intercambio de información. Este estilo arquitectónico permite que los sistemas

se comuniquen a través de servicios web ligeros basados en HTTP, facilitando la interoperabilidad entre diferentes aplicaciones y plataformas [36].
La API expone servicios para la gestión de reservas, huéspedes, habitaciones, facturación, usuarios, reportes y configuraciones del sistema.
Cada módulo del sistema cuenta con un conjunto de endpoints que permiten realizar operaciones CRUD (crear, consultar, actualizar y eliminar), garantizando la integridad de la información y el control de los procesos operativos del hotel.
El backend implementa mecanismos de autenticación y autorización basados en roles, permitiendo restringir el acceso a determinadas funcionalidades según el perfil del usuario, como administrador o recepcionista. De esta forma, se garantiza que solo los usuarios autorizados puedan ejecutar acciones críticas dentro del sistema.

6.4. Implementación del Frontend

El frontend fue desarrollado utilizando React.js, una biblioteca de JavaScript orientada a la construcción de interfaces de usuario dinámicas y basadas en componentes. La aplicación se estructuró en componentes reutilizables, páginas y servicios de consumo de la API REST.
Entre los principales elementos del frontend se encuentran:

- **Componentes:** Encargados de representar elementos reutilizables de la interfaz, como formularios, tablas, botones y modales.
- **Páginas:** Representan las diferentes secciones del sistema, como reservas, recepción, facturación, administración y reportes.
- **Servicios de comunicación:** Gestionan las solicitudes HTTP hacia el backend utilizando Axios.

Esta organización basada en componentes y en la separación de responsabilidades permite mantener una estructura de código más modular y escalable. De esta manera, los cambios en la lógica de comunicación con el backend o en la interfaz pueden realizarse de forma independiente, reduciendo el acoplamiento entre las distintas partes de la aplicación y facilitando su mantenimiento y evolución a lo largo del tiempo [37].

6.4.1. Interfaz de usuario y experiencia (UI/UX)

La interfaz del sistema fue diseñada priorizando la usabilidad y la claridad visual, considerando que los usuarios finales no necesariamente cuentan con conocimientos técnicos avanzados. Se implementó un diseño coherente en todas las pantallas, manteniendo una estructura uniforme de menús, colores y tipografías.

El sistema presenta formularios claros, tablas informativas y flujos de navegación intuitivos que permiten a los usuarios ejecutar sus tareas diarias de manera rápida y eficiente.

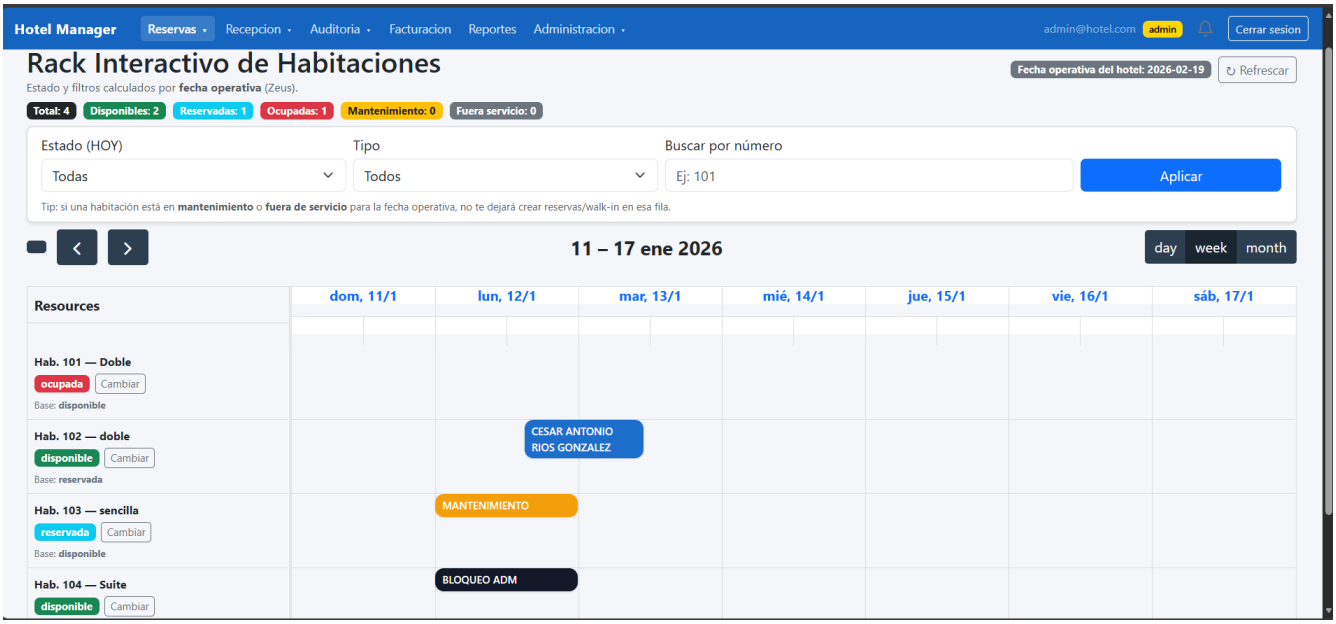

Figura 6.1: Panel principal del sistema.

6.5. Implementación de los Módulos Funcionales

6.5.1. Módulo de Gestión de Reservas

El módulo de gestión de reservas permite a los usuarios registrar reservas manuales, modificar o cancelar reservas existentes y visualizar la disponibilidad de habitaciones en tiempo real. Este módulo cumple con los requisitos funcionales definidos para la gestión de reservas, asegurando que no se produzcan duplicidades ni conflictos en la asignación de habitaciones.

Además, el sistema registra el origen de cada reserva, permitiendo diferenciar entre reservas manuales y reservas provenientes de plataformas externas.

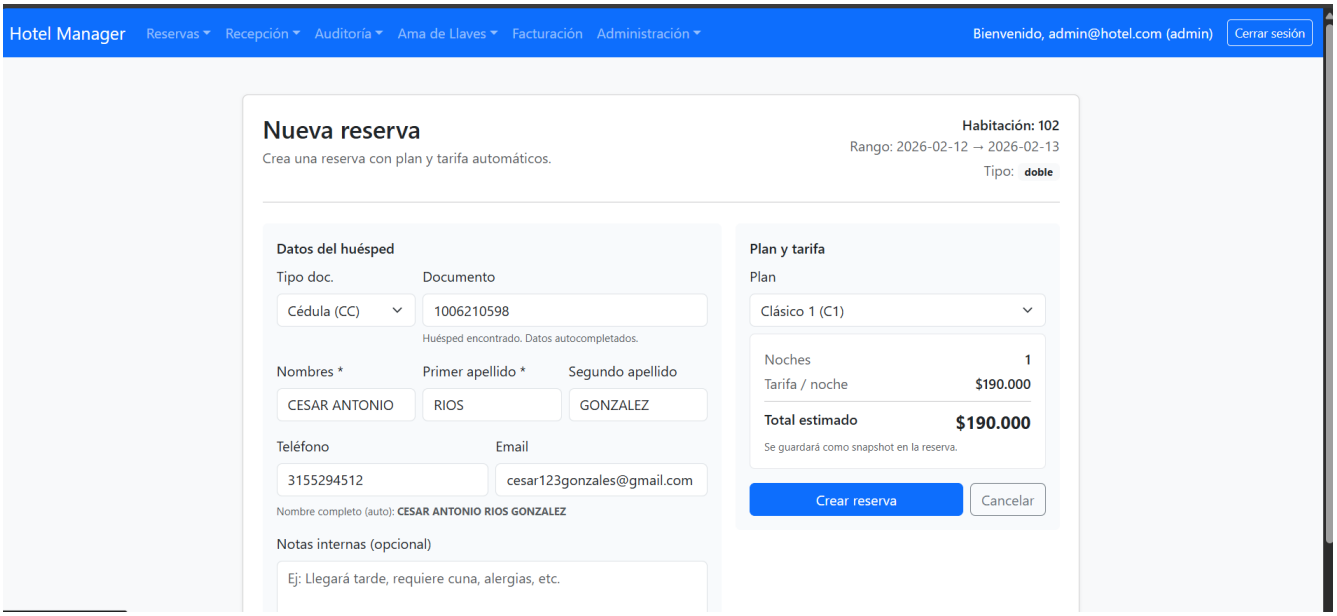

Figura 6.2: Formulario creacion de reservas

6.5.2. Módulo de Registro de Huéspedes

Este módulo permite registrar la información básica del huésped, incluyendo datos personales y de contacto. Asimismo, soporta los procesos de check-in y check-out de forma digital, almacenando el historial de estancias asociadas a cada huésped.

La digitalización de este proceso reduce los tiempos de atención y minimiza errores asociados a registros manuales, mejorando la experiencia tanto del personal del hotel como de los huéspedes.

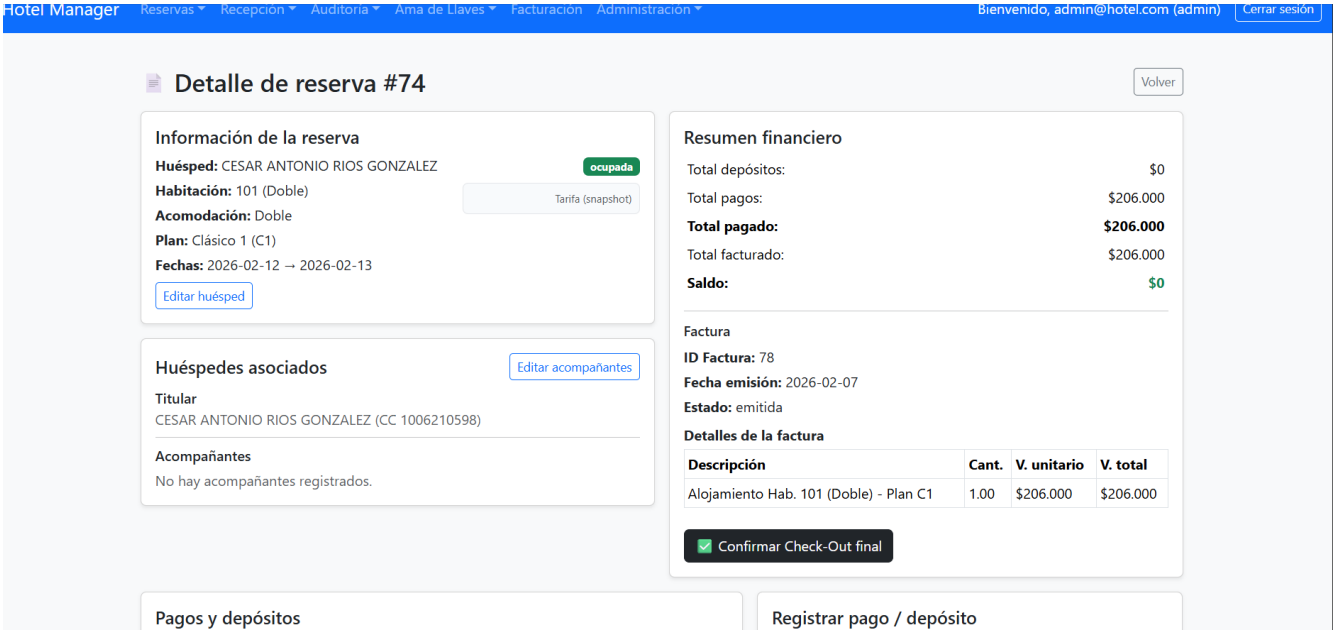

Figura 6.3: Formulario de registro de huésped / check-in.

6.5.3. Módulo de Control de Habitaciones

El módulo de control de habitaciones permite visualizar en todo momento el estado de cada habitación, clasificándolas como disponibles, ocupadas o en mantenimiento. El estado de las habitaciones se actualiza automáticamente según los procesos de reserva y check-in/check-out.

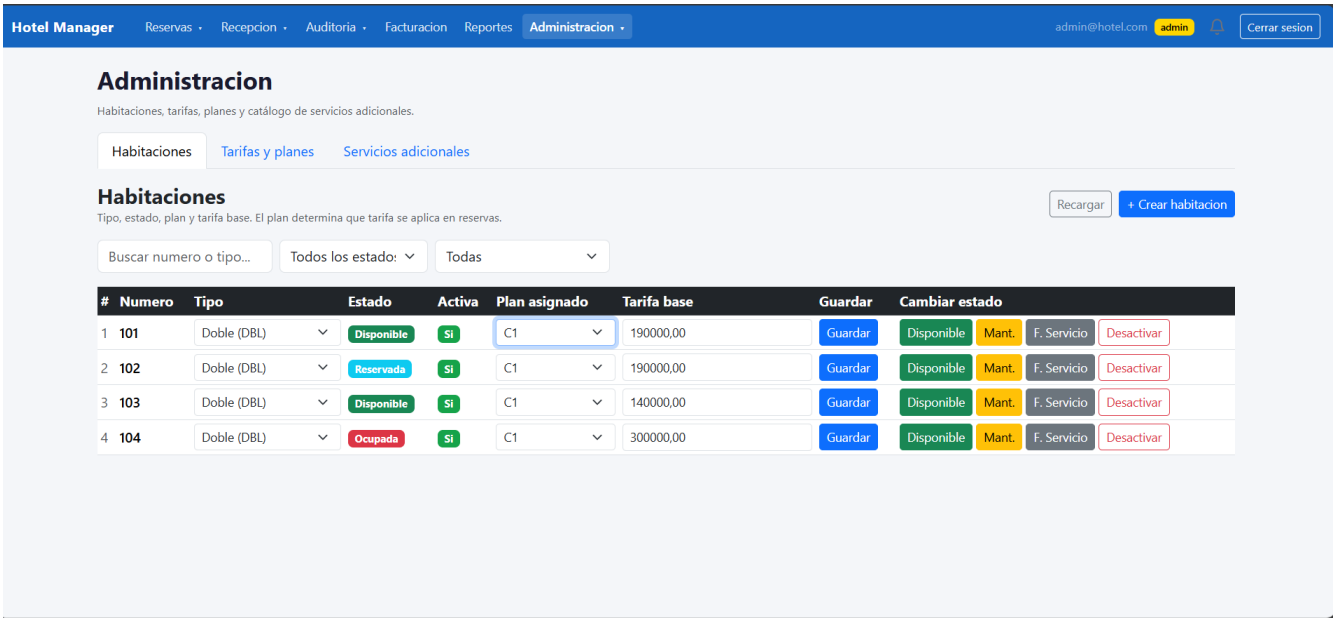

Figura 6.4: Control de habitaciones

Esta funcionalidad permite al personal del hotel tener un control preciso del inventario de habitaciones y facilita la toma de decisiones operativas.

6.5.4. Módulo de Facturación y Pagos

El sistema incluye un módulo de facturación que permite generar facturas asociadas a cada reserva y registrar pagos o depósitos realizados por los huéspedes. Toda la información financiera queda almacenada, permitiendo consultar el historial de transacciones y el estado de pago de cada reserva.

Este módulo contribuye a una gestión financiera más organizada y reduce errores en los cálculos y cobros.

6.5.5. Módulo de Reportes

Se implementó un módulo de generación de reportes que permite obtener información relevante sobre la operación del hotel, como ocupación, ingresos y estadísticas de reservas. Los reportes pueden filtrarse por fechas y exportarse para su análisis administrativo.

Este módulo facilita la toma de decisiones estratégicas basadas en datos reales y actualizados.

6.6. Restricciones Operativas Implementadas

Durante la implementación del sistema de gestión hotelera se definieron y aplicaron diversas reglas de negocio y restricciones operativas, las cuales permiten garantizar la coherencia de la información, evitar errores humanos y asegurar que los procesos del hotel se ejecuten conforme a la lógica real del negocio hotelero.

Restricciones en la Gestión de Reservas

- No se permite crear una reserva para una habitación que se encuentre en estado ocupada, en mantenimiento o fuera de servicio.
- No se permite registrar una reserva que se superponga en fechas con otra reserva activa para la misma habitación.
- No se permite realizar una reserva con fecha de salida anterior o igual a la fecha de entrada.
- Toda modificación o cancelación de una reserva queda registrada para mantener la trazabilidad del proceso.
- Las reservas provenientes de plataformas externas (OTAs) son validadas antes de ser confirmadas en el sistema, evitando duplicaciones.

Estas restricciones reducen significativamente los errores en la asignación de habitaciones y previenen situaciones de sobreventa.

Restricciones en el Proceso de Check-in y Check-out

Para los procesos de check-in y check-out se implementaron las siguientes reglas:

- El sistema solo permite realizar el check-in si la fecha actual coincide con la fecha de inicio de la reserva.
- No se permite realizar un check-in sobre una habitación que ya se encuentre ocupada.
- No se permite realizar el check-out de una habitación que no haya sido previamente registrada como ocupada.
- Al realizar el check-out, el sistema actualiza automáticamente el estado de la habitación a disponible, garantizando coherencia entre reservas y disponibilidad.

Estas reglas aseguran un flujo correcto del proceso de alojamiento y evitan inconsistencias en el estado de las habitaciones.

Restricciones para Reservas Walk-in

El sistema contempla el manejo de reservas tipo walk-in (reservas realizadas el mismo día), bajo las siguientes condiciones:

- Solo se permite registrar una reserva walk-in para la fecha operativa actual.
- No se permite registrar walk-ins para fechas futuras o pasadas.
- La habitación seleccionada debe encontrarse disponible al momento de la creación de la reserva.
- El sistema valida automáticamente la disponibilidad antes de permitir confirmar el walk-in.

Estas restricciones garantizan que las reservas inmediatas se realicen de manera controlada y coherente con la operación diaria del hotel

Restricciones en la Gestión de Habitaciones

En el módulo de control de habitaciones se implementaron las siguientes reglas:

- Una habitación en estado mantenimiento o fuera de servicio no puede ser asignada a reservas ni check-ins.
- El estado de una habitación se actualiza automáticamente en función de los eventos del sistema (check-in, check-out, cancelaciones).
- Solo usuarios con rol de administrador pueden modificar manualmente el estado de una habitación.
- Esto permite una gestión precisa del inventario y evita errores operativos.

Restricciones de Acceso y Seguridad

En términos de seguridad y control de acceso, el sistema implementa las siguientes restricciones:

- Solo usuarios autenticados pueden acceder al sistema.
- Las funcionalidades disponibles dependen del rol asignado al usuario.
- Los usuarios con rol de recepcionista no pueden acceder a configuraciones críticas del sistema.

Capítulo 7

Integración con plataformas externas (OTAs) – Channex

7.1. ¿Qué son las OTAs y por qué integrarlas en un sistema hotelero?

Las OTAs (Online Travel Agencies) son plataformas digitales de intermediación turística que permiten a los hoteles comercializar sus habitaciones y servicios a través de internet, facilitando la conexión entre los establecimientos de alojamiento y los potenciales huéspedes. Entre las OTAs más reconocidas a nivel internacional se encuentran Booking.com, Expedia y Airbnb, las cuales concentran una gran demanda de usuarios y se han consolidado como uno de los principales canales de distribución en la industria hotelera.[38]
No obstante, el uso de OTAs también introduce desafíos operativos relevantes cuando no existe una integración adecuada con el sistema de gestión hotelera (Property Management System – PMS). Entre los principales problemas se encuentran:

- La doble reserva o sobreventa de habitaciones, causada por la falta de sincronización de la disponibilidad.
- La ausencia de actualización en tiempo real del inventario, lo que afecta la confiabilidad de la información mostrada al cliente.
- Cancelaciones o modificaciones de reservas que no se reflejan correctamente en el sistema interno.
- Inconsistencias en las tarifas publicadas entre diferentes canales de venta.

De acuerdo con Buhalis [23], una gestión eficiente de los canales de distribución requiere sistemas integrados que permitan el intercambio automático y en tiempo real de datos, con el fin de garantizar coherencia operativa y mejorar la experiencia del cliente. En este contexto, la integración entre el sistema hotelero y las OTAs se vuelve una necesidad fundamental para optimizar los procesos administrativos, reducir errores humanos, asegurar la exactitud de la información y fortalecer la competitividad del hotel en un mercado altamente digitalizado.

7.2. Justificación integración con plataformas externas de reservas

La integración con plataformas externas de reservas constituye un componente clave dentro de un sistema de gestión hotelera, ya que permite la sincronización automática de disponibilidad, tarifas y reservas provenientes de múltiples canales de venta. Antes de implementar dicha integración, se realizó un análisis comparativo de distintas alternativas tecnológicas, considerando factores como el nivel de automatización, y la adecuación al contexto académico y funcional del proyecto.
Teconlogás evaluadas

- **Integración directa con cada OTA:** Esta alternativa consiste en consumir de forma independiente las APIs proporcionadas por cada plataforma de reservas (Booking.com, Expedia, Airbnb, entre otras). Si bien ofrece un alto nivel de control y automatización, implica una elevada complejidad técnica, ya que cada OTA maneja estándares, protocolos y flujos de integración distintos.
- **Uso de Channel Manager (Channex):**Un Channel Manager actúa como intermediario entre el sistema hotelero y múltiples OTAs, centralizando la gestión de reservas, tarifas y disponibilidad mediante una única API. Channex es una plataforma que ofrece este servicio, facilitando la integración y reduciendo la carga técnica del desarrollo.
- **Uso de un PMS comercial con integración nativa**: Implica la adopción de un sistema de gestión hotelera comercial que ya incluya integración con OTAs, como Opera PMS o Cloudbeds. Aunque estas soluciones ofrecen funcionalidades completas, su costo elevado y su carácter cerrado las hacen poco adecuadas para un proyecto académico orientado al desarrollo propio.

A continuación, se realiza una comparación con base en criterios clave para proyectos académicos y sistemas modulares:

Tabla 7.1: Tabla comparativa de alternativas de integración con OTAs

Tecnologias	**Ventajas**	**Desventajas**
Integración directa con cada OTA	Alto nivel de control sobre la integración. Automatización completa de reservas y disponibilidad.	Alta complejidad técnica. Múltiples APIs con distintos estándares. Mayor tiempo de desarrollo y mantenimiento.
Channel Manager (Channex)	Centraliza la comunicación con múltiples OTAs. API única y estandarizada. Soporte de webhooks en tiempo real. Entorno de pruebas (staging). Buena escalabilidad.	Dependencia de un servicio externo. Funcionalidades limitadas al alcance del proveedor.
PMS comercial con integración nativa	Solución completa y probada. Integración inmediata con OTAs.	Alto costo de licenciamiento. Baja flexibilidad. No favorece el desarrollo propio del sistema.

Channex fue seleccionado como la solución de integración con plataformas externas de reservas debido a que ofrece un equilibrio adecuado entre automatización, complejidad técnica y escalabilidad. Al actuar como intermediario entre el sistema hotelero y múltiples OTAs, permite centralizar la comunicación mediante una única API, simplificando significativamente el desarrollo.

Además, Channex proporciona soporte para webhooks en tiempo real, lo que facilita la actualización inmediata de reservas y disponibilidad. También cuenta con un entorno de pruebas (staging), permitiendo validar la integración sin afectar datos reales, lo cual resulta especialmente valioso en un entorno académico.[39]

Esta elección es coherente con los objetivos del proyecto, ya que permite implementar una integración profesional y realista, sin depender de múltiples APIs externas ni de soluciones comerciales cerradas, favoreciendo el aprendizaje, la mantenibilidad y la escalabilidad del sistema.

7.3. Diseño del Flujo de Integración

Fue diseñado siguiendo un modelo basado en eventos, en el cual las reservas generadas en canales externos se propagan de forma automática hacia el sistema de gestión hotelera. Este enfoque permite una comunicación eficiente y en tiempo real entre los distintos componentes involucrados, reduciendo la intervención manual y minimizando el riesgo de inconsistencias en la información.

El flujo de integración inicia cuando el huésped realiza una reserva a través de una plataforma OTA. Posteriormente, la OTA envía los datos de la reserva a Channex, que actúa como intermediario central. Al recibir esta información, Channex genera un evento de reserva y notifica al sistema interno mediante el envío de un webhook HTTP POST al backend de la aplicación.

Una vez recibido el evento, el backend procesa la información, valida los datos y registra la reserva en la base de datos del sistema. Finalmente, la reserva almacenada se refleja automáticamente en el calendario o rack de reservas, actualizando la disponibilidad y permitiendo una gestión centralizada y sincronizada.

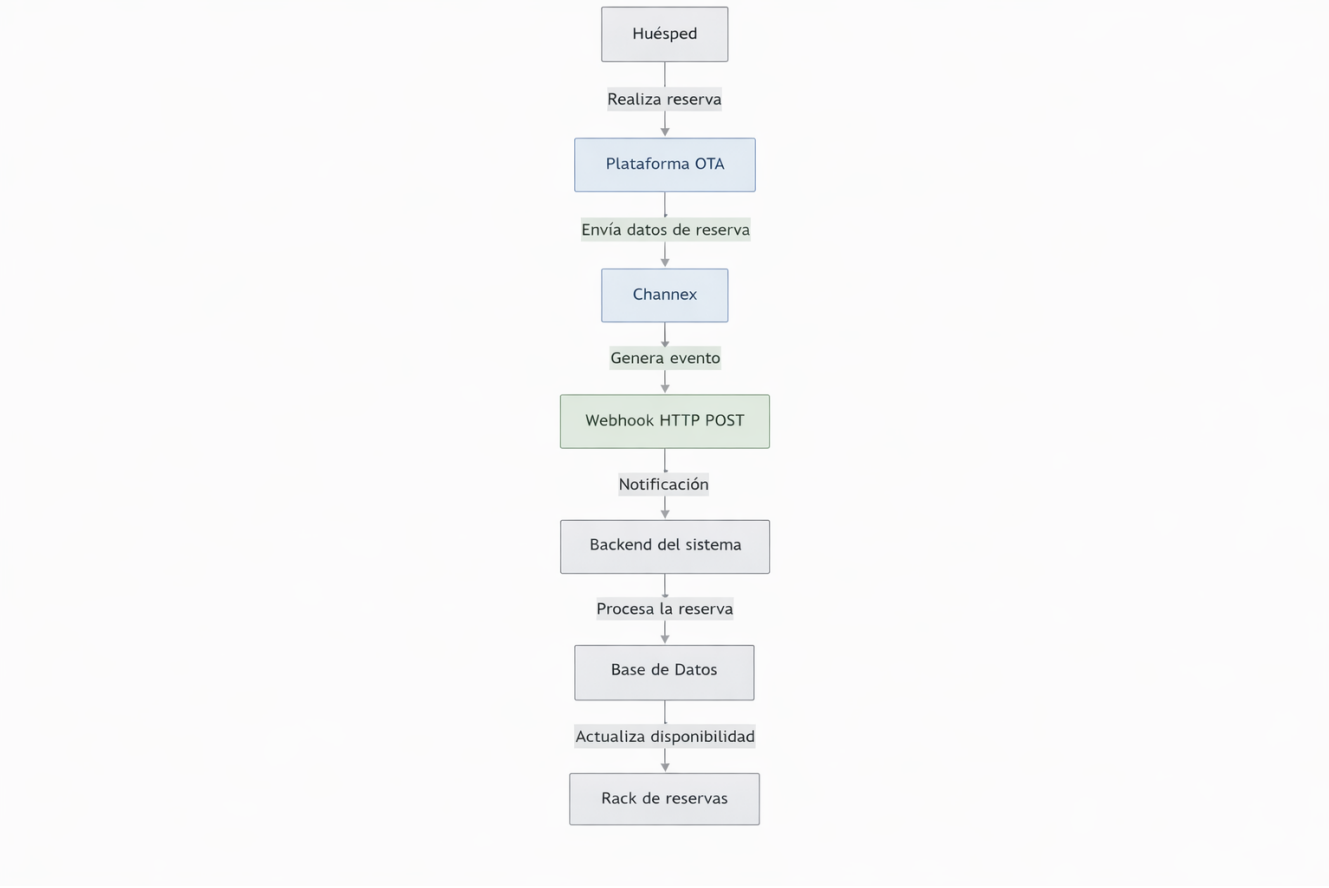

Figura 7.1: Flujo de integración con plataformas OTA

7.4. Implementaciónn técnica

Un endpoint es una URL expuesta por el backend que permite recibir o enviar información a otros sistemas mediante HTTP.
Un webhook es un mecanismo mediante el cual un sistema externo envía información automáticamente cuando ocurre un evento específico, sin necesidad de consultas periódicas. [40]
En este proyecto, Channex utiliza webhooks para notificar al sistema hotelero cuando ocurre un evento relevante, como la creación o modificación de una reserva.
El sistema implementa el siguiente endpoint:

- **POST /api/otas/channex/webhook**

Este endpoint:

Recibe eventos enviados por Channex

Procesa la información de la reserva

Actualiza la base de datos

Refleja los cambios en el calendario (rack)

Para evitar reintentos por parte de Channex, el endpoint responde inmediatamente con un código HTTP 200, mientras el procesamiento se realiza internamente.
Seguridad del Webhook

Para proteger la integración:

Se utiliza una API Key de Channex enviada en los headers HTTP.

Se valida que la solicitud provenga de una fuente confiable.

El endpoint no requiere autenticación JWT, ya que no es accedido por usuarios.

Se validan campos mínimos del payload antes de procesar la información.

Estructura de Datos Recibidos
El webhook recibido contiene información como:

Identificador único de la reserva OTA

Fechas de check-in y check-out

Número de noches

Nombre del huésped

Identificador de la propiedad

Evento asociado (creación o modificación)

Toda la información relevante se almacena en la base de datos, incluyendo el payload completo para auditoría

7.5. Manejo de Errores

Durante el desarrollo y las pruebas de la integración con Channex, se identificaron distintos escenarios en los que la comunicación con la API podía verse afectada, principalmente por errores de autorización, respuestas incompletas o indisponibilidad temporal del servicio. Estos escenarios son comunes en integraciones con servicios externos y, de no ser tratados adecuadamente, pueden provocar la pérdida de información crítica, como las reservas generadas por las OTAs.
Con el fin de garantizar la continuidad operativa del sistema y la integridad de los datos, se implementó una estrategia de fallback orientada a asegurar que ninguna reserva se pierda, incluso ante fallos parciales de la integración. En primer lugar, cuando no es posible obtener el detalle completo de la reserva a través de la API de Channex, el sistema utiliza la información básica contenida directamente en el webhook recibido, la cual incluye los datos mínimos necesarios para identificar la reserva.
En estos casos, la reserva es almacenada en el sistema con información esencial, permitiendo su visualización y seguimiento dentro del módulo de gestión. Adicionalmente, el payload completo del webhook es registrado y almacenado en la base de datos, lo que posibilita futuras validaciones, auditorías o ajustes manuales por parte del administrador del sistema.
Esta estrategia permite mantener la trazabilidad de las reservas y garantiza que el sistema continúe operando de forma confiable, incluso en condiciones adversas de integración con servicios externos.

7.5.1. Restricciones y Validaciones Aplicadas

Durante la integración con OTAs se aplicaron múltiples restricciones:

- No se permite registrar reservas OTA si no existe disponibilidad real.
- Se evita la duplicación de reservas mediante claves únicas.
- Las reservas canceladas desde la OTA se reflejan automáticamente en el sistema.
- Las habitaciones en mantenimiento o fuera de servicio no se asignan a reservas externas.
- El sistema evita sobreventas mediante validación de fechas y estados.

Estas validaciones refuerzan la confiabilidad del sistema y reducen errores operativos.
La integración con plataformas de reservas en línea mediante Channex representa uno de los componentes más relevantes del sistema de gestión hotelera desarrollado. Esta integración permite conectar el sistema interno del hotel con el ecosistema digital actual, automatizando procesos críticos y mejorando la eficiencia operativa.

La solución implementada demuestra que es posible integrar sistemas académicos con plataformas reales del sector hotelero, sentando las bases para una implementación productiva en entornos reales y futuras ampliaciones del sistema.
Para la validación de la integración OTA se utilizó el entorno de pruebas (staging) de Channex, el cual permite simular reservas provenientes de plataformas como Booking.com o Expedia. Estas reservas son creadas dentro del Channel Manager y enviadas al PMS mediante webhooks en tiempo real, replicando el comportamiento de un entorno productivo sin necesidad de tráfico real.”

7.6. Resultados de la integración en el sistema

La integración con Channex se refleja de forma directa en la interfaz del sistema a través de dos vistas: el listado general de reservas y el panel dedicado de Reservas OTA.

7.6.1. Identificación del origen OTA en el listado de reservas

En el módulo de de reservas, el sistema incorpora una columna Fuente que identifica el canal de origen de cada reserva. Las reservas recibidas desde plataformas externas a través de Channex se identifican con el valor channex en dicha columna, junto con una nota descriptiva que especifica la plataforma de origen (Expedia, Airbnb, Booking.com) y el nombre del huésped tal como fue registrado en la OTA.

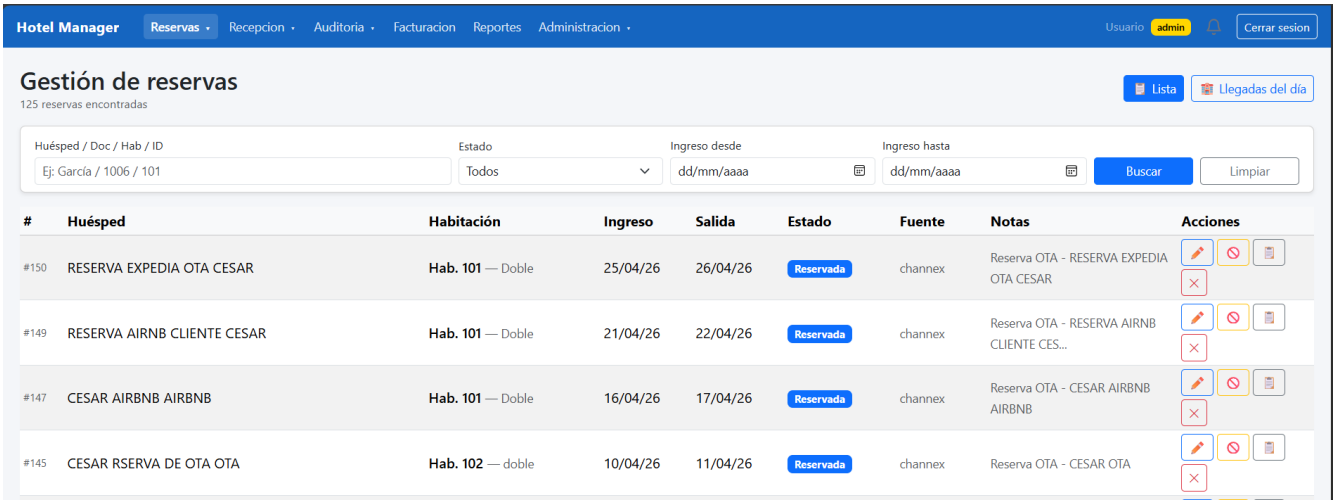

Figura 7.2: Listado de reservas

Como se observa en la Figura 7.2, las reservas fueron recibidas automáticamente desde sus respectivas plataformas y registradas en el sistema sin intervención manual del personal del hotel. La columna Fuente con valor channex confirma que el webhook fue procesado correctamente y la reserva fue almacenada en la base de datos.

7.6.2. Panel de Reservas OTA

El sistema cuenta con un módulo dedicado, accesible desde el menú de reservas, que centraliza la visualización de todas las reservas recibidas desde canales externos. Este panel muestra contadores diferenciados por plataforma (Expedia, Airbnb, Booking.com) y una tabla detallada con los campos: número de reserva interno, canal de origen, ID de reserva asignado por la OTA, nombre del huésped, habitación asignada, fechas de check-in y check-out, número de noches y estado actual.

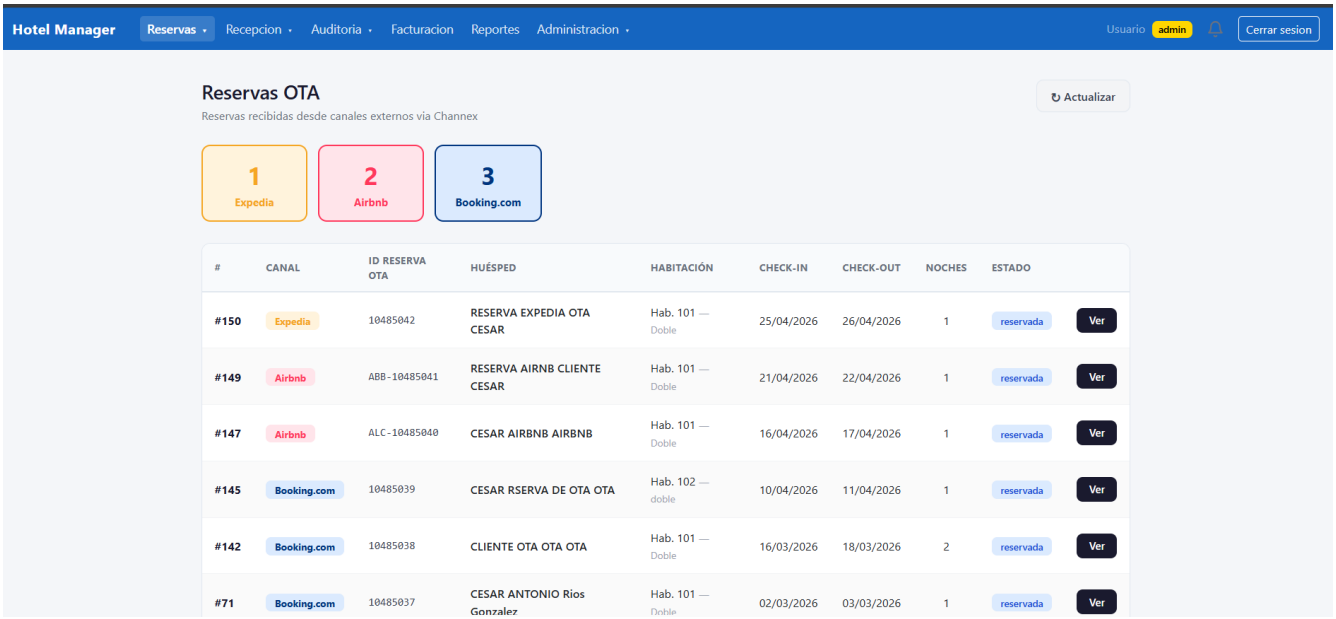

Figura 7.3: Panel de Reservas OTA

La Figura 7.3 evidencia el funcionamiento completo de la integración: el sistema recibió y procesó reservas provenientes de los tres canales configurados. Expedia, Airbnb y Booking permiten identificar el origen de cada reserva. El ID de reserva OTA corresponde al identificador asignado por Channex, lo que garantiza la trazabilidad entre el sistema interno y las plataformas externas.

Capítulo 8

Pruebas y resultados

8.1. Proceso de pruebas

El proceso de pruebas del sistema de gestión hotelera tuvo como propósito verificar la calidad, estabilidad y correcto funcionamiento del software, así como validar que las funcionalidades implementadas cumplen con los requisitos definidos y responden adecuadamente a escenarios reales de uso.
Para lograr este objetivo, se diseñó una estrategia de pruebas dividida en dos grandes enfoques complementarios:

- Pruebas técnicas automatizadas (pruebas unitarias y de integración)
- Pruebas reales con usuarios finales (pruebas funcionales y de usabilidad)

Esta combinación permitió evaluar el sistema tanto desde una perspectiva técnica como desde la experiencia del usuario final, asegurando un producto funcional y alineado con las necesidades del entorno hotelero

8.2. Pruebas Unitarias y Técnicas Automatizadas

8.2.1. Objetivo de las pruebas unitarias

Las pruebas unitarias tuvieron como objetivo validar el correcto funcionamiento de la lógica interna del sistema, verificando que cada módulo cumpla con sus reglas de negocio, restricciones y validaciones, de forma independiente y controlada.
Estas pruebas se enfocaron principalmente en el backend del sistema, ya que allí se concentra la mayor parte de la lógica crítica, como:

- Gestión de reservas
- Check-in y check-out
- Control de estados

- Seguridad y control de acceso
- Pagos y facturación
- Integración con servicios externos (OTAs)

8.2.2. Herramientas utilizadas para las pruebas unitarias

Para la implementación de las pruebas unitarias y de integración se utilizaron las siguientes herramientas:

- Jest: Framework de pruebas para JavaScript, utilizado para definir, ejecutar y organizar los casos de prueba.
- Supertest: Librería utilizada para simular peticiones HTTP reales hacia la API REST.
- Node.js (ES Modules): Entorno de ejecución del backend.
- PostgreSQL: Base de datos real utilizada durante las pruebas,
- JWT (JSON Web Tokens): Para pruebas de autenticación y autorización.
- Axios (mockeado en pruebas OTA): Para simular respuestas de servicios externos.

Las pruebas se realizaron sobre una base de datos real, no simulada, lo cual permitió validar el comportamiento del sistema en condiciones cercanas a un entorno productivo permitiendo validar transacciones, bloqueos y consistencia de datos.

8.3. Resultado de pruebas automatizadas

Módulos evaluados mediante pruebas unitarias

Las pruebas unitarias se organizaron por módulos del sistema, permitiendo una validación clara y estructurada.

8.3.1. Módulo de Autenticación (Login)

Se realizaron pruebas para validar:

- Inicio de sesión con usuario inexistente
- Inicio de sesión con contraseña incorrecta
- Inicio de sesión exitoso y generación de token JWT

Objetivo: Garantizar la seguridad del sistema y el correcto control de acceso.

Tabla 8.1: Tabla de casos de prueba – Autenticación

Caso de prueba	Datos de entrada	Resultado esperado
Usuario no exite	Email inexistente	Error
Contraseña incorrecta	Password inválida	Erro
Login válido	Credenciales correctas	Token JWT generado

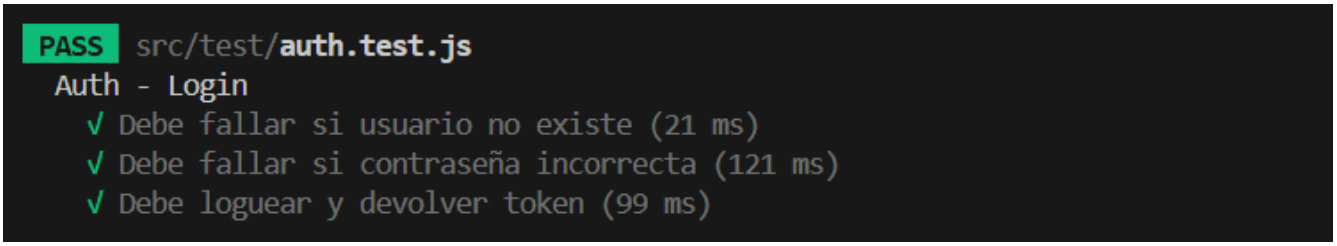

Figura 8.1: Test login

Flujo validado
Inicio de sesión → validación de credenciales → generación de token JWT → control de acceso por rol.

Se confirmó que el sistema protege el acceso, diferencia errores correctamente y genera tokens válidos para sesiones autenticadas.

8.3.2. Módulo de Gestión de Reservas

Se probaron escenarios como:

- Rechazo de reservas con rangos de fechas inválidos
- Creación exitosa de reservas válidas
- Detección de choque de reservas (evitar sobreventa)

Objetivo: Asegurar la integridad de la disponibilidad de habitaciones.

Tabla 8.2: Tabla de casos de prueba – Gestión de reservas

Caso de prueba	Datos de entrada	Resultado esperado
Fechas inválidas	Fecha fin ¡inicio	Error
Reserva válida	Habitación disponible	Reserva creada
Choque de reservas	Rango ocupado	Error

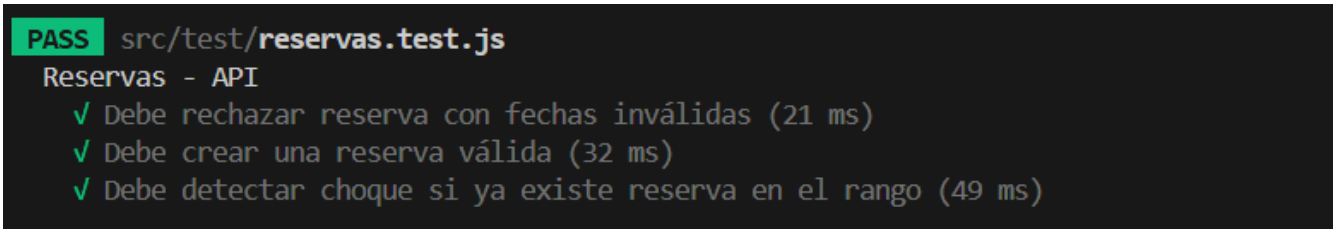

Figura 8.2: Test reserva

Flujo validado
Solicitud de reserva → validación de fechas → validación de disponibilidad → creación de reserva.

Se garantizó la no sobreventa de habitaciones, validando correctamente rangos de fechas y estados de reserva.

8.3.3. Módulo de Check-in

Se validaron las siguientes reglas de negocio:

- Rechazo de check-in sin documento del huésped titular
- Cambio de estado de la reserva a “ocupada”
- Cambio de estado de la habitación a “ocupada”
- Rechazo de check-in en habitaciones en mantenimiento

Objetivo: Garantizar que el proceso de check-in se realice únicamente bajo condiciones válidas.

Tabla 8.3: Tabla de casos de prueba – Check in

Caso de prueba	Datos de entrada	Resultado esperado
Sin documento	Titulo incompleto	Error
Check-in válido	Datos completos	Check-in completo y habitacion ocupada
Habitación en mantenimeinto	Estado inválido	Error

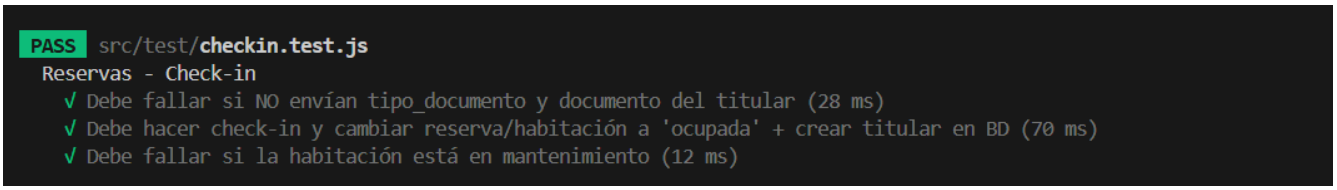

Figura 8.3: Test Check-in

Flujo validado
Reserva → validación documental → actualización de estados → asignación de huéspedes.

Se comprobó que el sistema impide check-in inválidos y mantiene coherencia entre reservas, huéspedes y habitaciones.

8.3.4. Módulo de Check-out

Se probaron escenarios como:

- Rechazo de check-out sin factura generada
- Liberación correcta de la habitación
- Cambio del estado de la reserva a "finalizada"

Objetivo: Asegurar que el cierre de la estancia cumpla con las reglas operativas y financieras.

Tabla 8.4: Tabla de casos de prueba – Check out

Caso de prueba	Datos de entrada	Resultado esperado
Sin factura	Factura inexistente	Error
Check-out válido	Factura emitida	Reserva finalizada

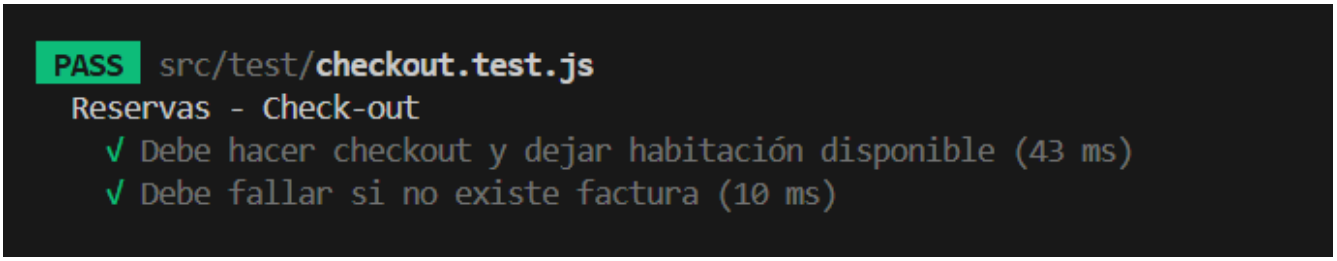

Figura 8.4: Test check out

Flujo validado
Reserva ocupada → validación de factura → liberación de habitación → cierre de reserva.

Se aseguró que el check-out solo se complete bajo condiciones financieras válidas

8.3.5. Módulo de Seguridad y Control de Acceso

Se realizaron pruebas sobre:

- Middleware de verificación de token JWT
- Restricción de acceso por roles (solo administrador)

Objetivo: Verificar el control de acceso y la protección de rutas críticas.

Tabla 8.5: Tabla de casos de prueba – Seguridad

Caso de prueba	Datos de entrada	Resultado esperado
Sin token	Header vacio	Error
Token válido	JWT correcto	Acceso permitido
Rol inválido	No admin	Error

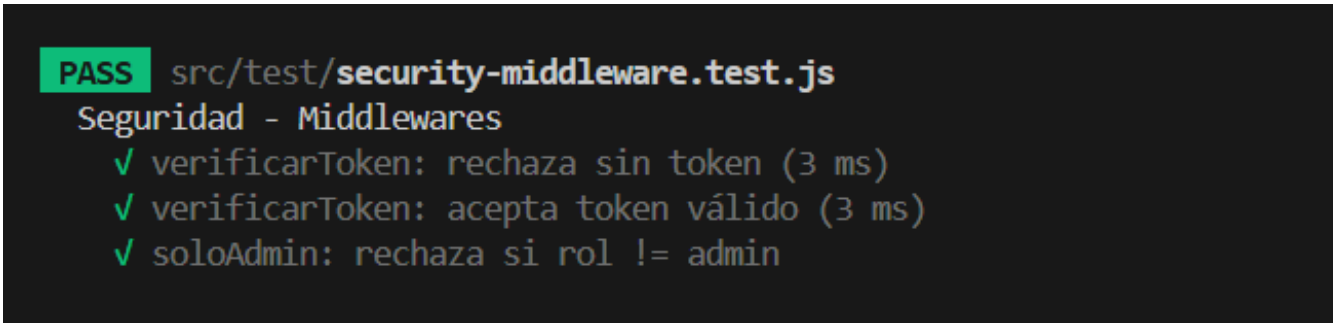

Figura 8.5: Test seguridad

Flujo validado
Solicitud → verificación de token → validación de rol → autorización.

Se confirmó un control de acceso robusto, fundamental para la seguridad del sistema.

8.3.6. Modulo de Integración OTA – Webhook Channex

Se probaron escenarios como:

- Recepción de webhook desde Channex
- Respuesta 200 obligatoria ante cualquier evento
- Manejo de errores de autorización (Unauthorized)
- Estrategia de fallback sin interrupción del sistema

Objetivo: Garantizar la robustez del sistema frente a integraciones externas

Tabla 8.6: Tabla de casos de prueba – Seguridad

Caso de prueba	Datos de entrada	Resultado esperado
Webhook válido	booking-revision	Exito
Error externo	Unauthorized	Sistema no falla

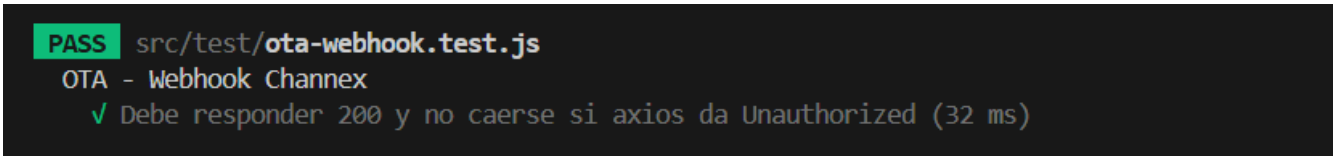

Figura 8.6: Test OTA

Flujo validado
OTA → Channex → Webhook → Backend → Base de datos → Rack.

Se garantizó tolerancia a fallos externos, evitando caídas del sistema ante errores de terceros.

8.4. Resultado pruebas usuarios finales

Diseño de la prueba
Con el fin de evaluar el sistema desde la perspectiva del usuario final, se diseñó una encuesta estructurada orientada a medir la usabilidad, funcionalidad, eficiencia, satisfacción general e integración con plataformas OTA del sistema de gestión hotelera.

La encuesta fue aplicada a tres (3) usuarios finales, quienes interactuaron previamente con el sistema ejecutando procesos reales de operación hotelera, tales como creación de reservas, check-in, registro de pagos, check-out y visualización de reservas provenientes de plataformas externas (OTAs).

El tamaño de la muestra responde directamente al alcance del sistema: Hotel Manager PMS fue diseñado para establecimientos hoteleros pequeños, de hasta 20 habitaciones, cuya estructura operativa habitual se compone de un administrador y uno o dos recepcionistas, tal como se estableció en el Capítulo 2 del presente documento. En este tipo de organizaciones, la totalidad del personal que interactúa con el sistema de gestión es reducida por naturaleza, lo cual hace que una muestra de tres usuarios represente el 100 % de los perfiles operativos reales del establecimiento destino [34].

Nielsen [34] establece que en pruebas de usabilidad con usuarios representativos, entre tres y cinco participantes son suficientes para identificar la mayoría de los problemas de uso en sistemas con perfiles de usuario homogéneos y bien definidos. En este caso, los perfiles evaluados (administrador y recepcionista) corresponden exactamente a los roles del sistema, lo que garantiza que las respuestas reflejan la experiencia real de los usuarios objetivo del software. Esta decisión metodológica es coherente con los trabajos de Sommerville [22], quien señala que el tamaño de la muestra en pruebas de aceptación debe estar determinado por la representatividad de los perfiles involucrados y no exclusivamente por el volumen numérico de participantes.
Estructura de la encuesta La encuesta se diseñó utilizando una escala tipo Likert de 5 niveles, donde:

- Totalmente en desacuerdo
- En desacuerdo
- Neutral
- De acuerdo
- Totalmente de acuerdo

Este tipo de escala permite cuantificar percepciones subjetivas y obtener resultados comparables y analizables de forma estadística.

La encuesta estuvo compuesta por cinco secciones, descritas a continuación:

Sección Dimensión evaluada
Sección A Usabilidad del sistema
Sección B Funcionalidad
Sección C Eficiencia
Sección D Satisfacción general
Sección E Integración con plataformas OTA

8.4.1. Sección A – Usabilidad del sistema

Esta sección evaluó aspectos relacionados con la facilidad de uso, claridad visual y comprensión de la interfaz, incluyendo:

- Intuición de la interfaz para realizar tareas
- Claridad en la navegación entre pantallas
- Comprensión de botones y opciones

Objetivo: Validar que el sistema pueda ser utilizado sin capacitación extensa.

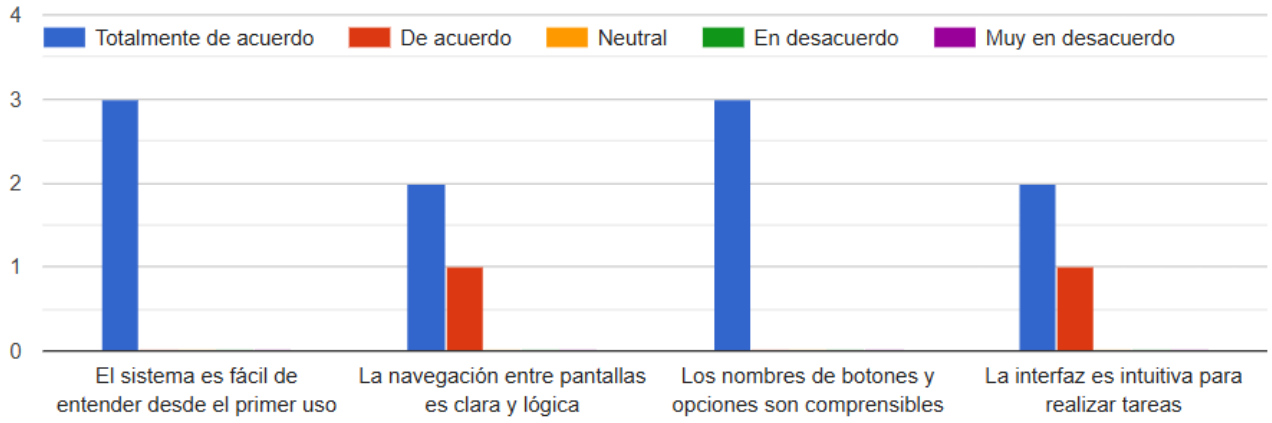

Figura 8.7: Resultados usabilidad

8.4.2. Sección B – Funcionalidad

Se evaluó el correcto funcionamiento de los procesos principales del sistema:

- Creación de reservas
- Proceso de check-in
- Registro de pagos
- Proceso de check-out

Objetivo: Verificar que el sistema cumpla con los procesos operativos del entorno hotelero.

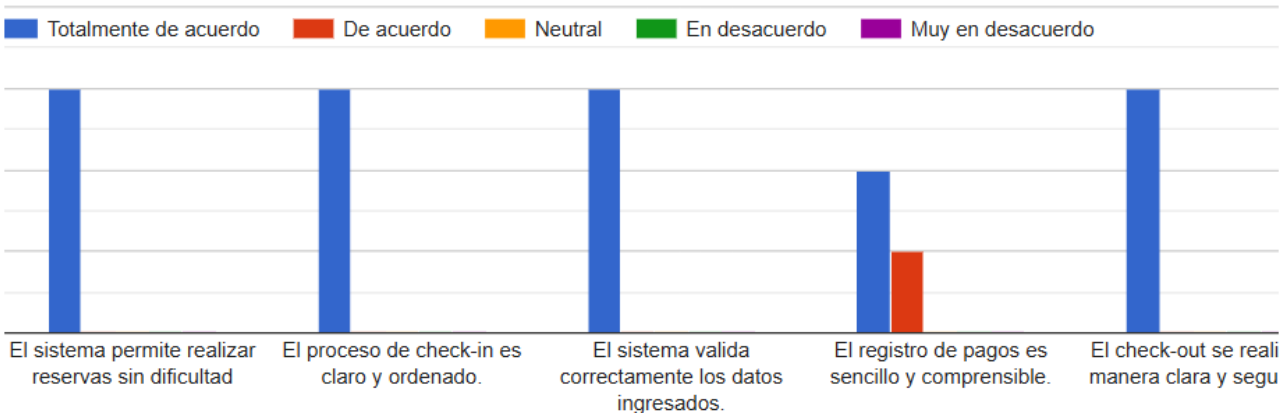

Figura 8.8: Resultados funcionalidad

8.4.3. Sección C – Eficiencia

Esta sección midió el desempeño del sistema en términos de tiempo y estabilidad:

- Tiempo de respuesta del sistema
- Fluidez en la ejecución de acciones
- Ausencia de errores o bloqueos

8.4.4. Sección D – Satisfacción general

Se evaluó la percepción global del usuario respecto al sistema:

- Comodidad durante el uso
- Utilidad del sistema en un hotel real
- Disposición a usar el sistema de forma diaria

8.4.5. Sección E – Integración con plataformas OTA

Dado que la integración con OTAs constituye uno de los aportes principales del proyecto, se incluyó una sección específica para evaluar:

- Comprensión del proceso de llegada automática de reservas
- Correcta visualización de reservas OTA
- Integración ordenada con reservas manuales

- Prevención de conflictos de disponibilidad
- Reducción de errores manuales
- Confianza en la gestión de reservas externas
- Análisis estadístico de los resultados

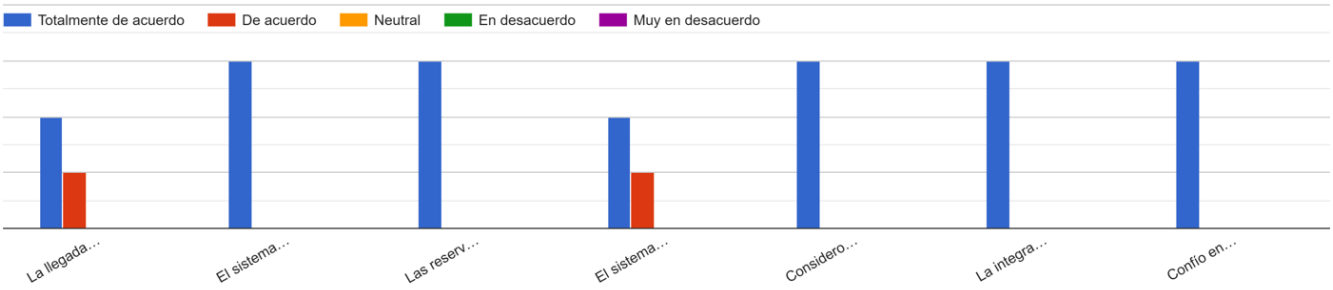

Figura 8.9: Resultado OTA

Para el análisis de resultados, las respuestas se transformaron a valores numéricos (1 a 5), donde 5 corresponde a “Totalmente de acuerdo”.

Tabla 8.7: Resultados prueba usuario finales

Sección	**Promedio obtenido**
Usabilidad	4.75
Funcionalidad	4.92
Eficiencia	4.83
Satisfacción general	5.00
Integración OTA	4.85

Estos resultados evidencian una alta aceptación general del sistema, con especial énfasis en la satisfacción global y la integración con plataformas externas.

Distribución de respuestas

Más del 90 % de las respuestas se concentraron en las categorías “De acuerdo” y “Totalmente de acuerdo”.
No se registraron respuestas en las categorías “En desacuerdo” o “Totalmente en desacuerdo”.
Las observaciones cualitativas reforzaron la percepción positiva del sistema.

Análisis cualitativo
Las respuestas abiertas permitieron identificar ventajas clave del sistema, tales como:

- Automatización del proceso de reservas.
- Reducción de errores manuales.

- Mejora en la eficiencia operativa de la recepción.
- Facilidad de aprendizaje y uso.
- Integración clara y confiable con OTAs.

Los usuarios no reportaron dificultades significativas durante el uso del sistema, lo que refuerza la solidez del diseño y la implementación.

8.5. Conclusión general de las pruebas

Las pruebas realizadas con usuarios finales permitieron evaluar el sistema de gestión hotelera en términos de usabilidad, funcionalidad, eficiencia, satisfacción e integración con plataformas OTA. La encuesta fue aplicada a tres usuarios (administrador y recepcionistas), quienes ejecutaron procesos reales como reservas, check-in, pagos y check-out. Este tamaño de muestra es adecuado, ya que representa la totalidad de los perfiles operativos del sistema en hoteles pequeños. Los resultados evidenciaron una alta aceptación del sistema, con promedios cercanos a 5 en todas las categorías evaluadas y más del 90 % de respuestas positivas, sin registros negativos. Asimismo, los usuarios destacaron beneficios como la automatización de reservas, la reducción de errores manuales, la mejora en la eficiencia operativa, la facilidad de uso y la integración confiable con plataformas externas. En conjunto, estos resultados confirman que el sistema es funcional, confiable y adecuado para su implementación en pequeños establecimientos hoteleros, cumpliendo con los objetivos de validación planteados.

Capítulo 9

Conclusiones

El desarrollo del Sistema de Gestión Hotelera evidenció que la automatización y centralización de los procesos operativos en hoteles pequeños y medianos no solo es técnicamente viable, sino necesaria para reducir errores críticos como la sobreventa, la duplicidad de reservas y la falta de control en la operación diaria. A partir del análisis inicial del contexto hotelero, se evidenció que muchos de los problemas no se originan en la falta de personal, sino en la ausencia de sistemas integrados que respalden la toma de decisiones operativas en tiempo real.

Uno de los hallazgos más relevantes del proyecto fue que el levantamiento estructurado de requisitos, definido bajo el estándar IEEE 830, permitió traducir procesos operativos reales (reservas, check-in, check-out, control de habitaciones y pagos) en reglas de negocio claras y verificables. Esta decisión evitó ambigüedades frecuentes en el desarrollo y facilitó la validación posterior del sistema mediante pruebas unitarias y funcionales. Sin embargo, se identificó como limitación inicial que algunos requisitos solo pudieron refinarse durante la implementación, lo que confirma la necesidad de enfoques iterativos en entornos reales.

En términos de arquitectura, la adopción del modelo de tres capas con comunicación REST fue una de las decisiones técnicas más críticas del proyecto. Esta elección permitió separar claramente la lógica de negocio del backend y la interfaz del frontend, lo cual resultó determinante para resolver conflictos técnicos detectados durante las pruebas, como el manejo simultáneo de reservas manuales y reservas provenientes de OTAs. La modularidad de la arquitectura facilitó ajustes puntuales sin afectar el sistema completo, especialmente en el módulo de reservas y disponibilidad. No obstante, se identificó que una arquitectura orientada a microservicios podría ser una evolución futura para escenarios de mayor escala.

Un aporte diferenciador del sistema fue la integración con plataformas de reservas externas (OTAs) a través del channel manager Channex. Desde el punto de vista técnico, esta integración permitió validar en la práctica conceptos clave como webhooks, idempotencia, manejo de eventos incompletos y tolerancia a fallos externos. Desde la perspectiva operativa, se logró reducir el riesgo de sobreventas y eliminar la necesidad de registrar manualmente reservas externas. Los usuarios finales manifestaron confianza en la gestión de reservas OTA y reconocieron una mejora en la eficiencia operativa de la recepción, lo cual es coherente con los resultados técnicos obtenidos durante las pruebas de integración.

Los resultados de las pruebas técnicas, contrastados con la evaluación de usuarios, son coherentes entre sí. La integración OTA operó de forma estable durante las pruebas, la sincronización automática de

reservas externas redujo los errores de disponibilidad, y la evaluación con usuarios arrojó resultados positivos en todas las dimensiones medidas, con predominio de respuestas de "De acuerdoz "Totalmente de acuerdo.en usabilidad, funcionalidad y satisfacción general.

Las pruebas unitarias, de integración y funcionales permitieron identificar comportamientos críticos del sistema ante escenarios reales, como intentos de check-in fuera de rango, pagos en estados no permitidos y conflictos de disponibilidad. Estos hallazgos no solo validaron el correcto funcionamiento del sistema, sino que también evidenciaron áreas de mejora, como el endurecimiento de validaciones y la estandarización de respuestas ante errores externos. Si el proyecto se desarrollara nuevamente, una de las principales mejoras sería incorporar pruebas automatizadas desde etapas más tempranas del desarrollo para reducir reprocesos.

Finalmente, el objetivo de este trabajo era desarrollar un sistema de gestión hotelera alineado con las necesidades reales del sector, aplicando buenas prácticas de ingeniería de software, validación técnica rigurosa y evaluación con usuarios finales. Los resultados obtenidos, tanto en las pruebas técnicas como en la evaluación con usuarios, permiten afirmar que dicho objetivo fue alcanzado. El sistema tiene el potencial de evolucionar hacia un producto dirigido a hoteles pequeños y medianos, incorporando mejoras en rendimiento, analítica y nuevas integraciones externas. El proyecto no concluye con la entrega, sino que abre una oportunidad concreta para su adopción en entornos productivos reales del sector hotelero.

Capítulo 10

Trabajos futuros

El Sistema de Gestión Hotelera desarrollado en este trabajo de grado fue diseñado bajo una arquitectura modular y escalable, lo que permite su evolución progresiva conforme a nuevas necesidades del sector hotelero y avances tecnológicos. A partir de los resultados obtenidos durante la implementación y las pruebas realizadas, se identifican diversas líneas de trabajo futuro que permitirían ampliar las capacidades del sistema y fortalecer su impacto operativo.

Una primera línea de evolución corresponde a la optimización y ampliación de la integración con plataformas de reservas en línea (OTAs). Si bien el sistema actualmente se integra mediante un channel manager para la recepción automática de reservas, en trabajos futuros se podría incorporar la sincronización bidireccional completa de tarifas, disponibilidad y restricciones, así como la gestión automática de cancelaciones y modificaciones en tiempo real. Esto permitiría reducir aún más el riesgo de sobreventas y mejorar la consistencia de la información entre el sistema hotelero y las plataformas externas.

Otra posible expansión del sistema es la implementación de mecanismos de apoyo a la toma de decisiones mediante análisis inteligente de datos. A partir de la información histórica almacenada (reservas, ocupación, tarifas y temporadas), se podrían incorporar algoritmos de análisis predictivo que ayuden a estimar niveles de ocupación, sugerir ajustes de tarifas según la demanda o identificar patrones de comportamiento de los huéspedes. Estas funcionalidades estarían orientadas a mejorar la gestión estratégica del hotel, manteniendo el enfoque administrativo del sistema.

En relación con la experiencia de usuario, un trabajo futuro relevante es el desarrollo de una aplicación móvil complementaria, orientada principalmente al personal operativo del hotel. Esta aplicación permitiría consultar el estado de las habitaciones, registrar check-in y check-out, visualizar reservas activas y recibir notificaciones en tiempo real, facilitando la operación diaria sin depender exclusivamente del acceso desde un navegador web.

Asimismo, el sistema podría ampliarse mediante la incorporación de un módulo de autoservicio para huéspedes, enfocado en funcionalidades como pre check-in, consulta de información de la reserva y solicitudes de servicios durante la estancia. Este módulo contribuiría a reducir la carga operativa del personal de recepción y a mejorar la experiencia del cliente.

Desde el punto de vista técnico, se plantea como trabajo futuro la evolución de la arquitectura hacia

un modelo más distribuido, permitiendo separar ciertos módulos críticos (reservas, facturación, integración OTA) como servicios independientes. Esto facilitaría el escalamiento del sistema en escenarios de mayor demanda y mejoraría la mantenibilidad a largo plazo.

Finalmente, se propone como línea de trabajo futuro la validación del sistema en entornos reales de operación durante periodos prolongados, incorporando un mayor número de usuarios y hoteles piloto. Esta validación permitiría medir el impacto del sistema en indicadores reales como tiempos de atención, reducción de errores operativos y eficiencia en la gestión de reservas, aportando evidencia empírica adicional sobre su utilidad y efectividad.

En conjunto, estos trabajos futuros representan una evolución natural del sistema desarrollado, manteniendo su enfoque en la automatización de procesos administrativos, la integración con plataformas externas y la mejora continua de la eficiencia operativa en hoteles pequeños y medianos.

Anexos

Los siguientes anexos contienen la documentación visual, y evidencia de los entregables.

Anexo A:
Encuesta sobre procesos actuales en hoteleria: https://forms.gle/vdQpxWaQ4bbAyN1h9

Anexo B:
Repositorios al código fuente: https://github.com/Cesarriosg/sistema-gestion-hotel-tg.git

Anexo C:
Tablero trello: https://trello.com/b/LQEb9Vtc

Anexo D:
Encuesta sobre verificación del sitema: https://forms.gle/jVj2YjAtZSgf6oTx7

Capítulo 11

Bibliografía

[1] M. J. O'Fallon and D. G. Rutherford, *Hotel Management and Operations*. Wiley, 5th ed., 2010.

[2] M. L. Kasavana and J. J. Cahill, *Managing Technology in the Hospitality Industry*. AHLEI, 2011.

[3] R. Law, D. Leung, A. Lo, and R. Leung, "Distribution channel in hospitality and tourism: Revisiting disintermediation from the perspectives of hotels and travel agencies," *International Journal of Contemporary Hospitality Management*, vol. 25, no. 1, pp. 147–163, 2013.

[4] S. Ivanov, *Hotel Revenue Management: From Theory to Practice*. Zangador, 2014.

[5] M. Sigala and D. Buhalis, *IT and Internet Development in Hospitality and Tourism*. Routledge, 2002.

[6] N. N. Taleb, *Antifragile: Things That Gain from Disorder*. Random House, 2018.

[7] P. González and S. Pérez, "Optimización de procesos operativos en hoteles mediante el uso de software especializado," *Journal of Hotel Management*, vol. 29, no. 4, pp. 72–86, 2019.

[8] A. Gómez and M. Torres, "La integración de las otas en la gestión hotelera: un análisis de impacto," *Revista de Tecnología Turística*, vol. 11, no. 1, pp. 34–45, 2022.

[9] F. Vázquez and J. Hernández, "Big data y su impacto en la gestión hotelera: el futuro de la toma de decisiones," *International Journal of Hotel and Tourism Management*, vol. 18, no. 1, pp. 34–45, 2020.

[10] L. v. Bertalanffy, *General System Theory: Foundations, Development, Applications*. George Braziller, 1968.

[11] E. M. Goldratt, *Theory of Constraints*. North River Press, 1990.

[12] W. E. Deming, *Out of the Crisis*. MIT Press, 1986.

[13] C. M. Christensen, *The Innovator's Dilemma: When New Technologies Cause Great Firms to Fail*. Harvard Business Review Press, 1997.

[14] M. Kasavana, *Managing Front Office Operations*. Educational Institute of the American Hotel & Lodging Association, 9th ed., 2013.

[15] M. Fowler, *Patterns of Enterprise Application Architecture*. Addison-Wesley, 2002.

[16] S. Ivanov and C. Webster, "Adoption of robots, artificial intelligence and service automation by travel, tourism and hospitality companies," *International Journal of Contemporary Hospitality Management*, vol. 29, no. 5, pp. 1290–1311, 2017.

[17] I. Academy, "Ciberseguridad en hoteles: Desafíos y estrategias," 2023.

[18] J. Murphy, A. Hofacker, and R. Mizerski, "Consideraciones de activación para comunicaciones de marketing móvil," *Revista de Marketing Interactivo*, vol. 26, no. 4, pp. 271–281, 2012.

[19] H. M. Moyeenudin, S. J. Parvez, R. Anandan, and K. Narayanan, "Data management with property management system in hotel industry," *International Journal of Engineering and Technology*, vol. 7, no. 2.21, pp. 327–330, 2018.

[20] M. Muñoz Osores, M. Taito Jara, and J. Fernández Palma, "Digital adoption of the hotel industry: a comparative study for chile and peru," *Revista Academia & Negocios*, vol. 9, no. 1, pp. 39–50, 2023.

[21] J. Moreno and M. López, "La automatización en el sector hotelero: un estudio sobre la gestión de reservas y facturación," *Revista de Tecnologías Aplicadas al Turismo*, vol. 15, no. 2, pp. 105–123, 2018.

[22] I. Sommerville, *Software Engineering*. Boston: Pearson Education, 10th ed., 2016.

[23] D. Buhalis, P. O'Connor, and R. Leung, "IT strategy in the hotel industry in the digital era," *Sustainability*, vol. 14, no. 17, p. 10705, 2022.

[24] D. Buhalis and R. Law, "Progress in information technology and tourism management: 20 years on and 10 years after the internet the state of etourism research," *Tourism Management*, vol. 29, no. 4, pp. 609–623, 2008.

[25] A. N. Abukhalifeh and T. J. Pratt, "Hotel property management system," in *Encyclopedia of Tourism Management and Marketing* (D. Buhalis, ed.), pp. 604–607, Edward Elgar Publishing, 2022.

[26] IEEE, "Ieee recommended practice for software requirements specifications," Tech. Rep. IEEE Std 830-1998, Institute of Electrical and Electronics Engineers, 1998.

[27] R. C. Martin, *Agile Software Development: Principles, Patterns, and Practices*. Upper Saddle River, NJ: Prentice Hall, 2003.

[28] S. Tilkov and S. Vinoski, "Node.js: Using javascript to build high-performance network programs," *IEEE Internet Computing*, vol. 14, no. 6, pp. 80–83, 2010.

[29] The PostgreSQL Global Development Group, "Postgresql 15 documentation." `https://www.postgresql.org/docs/15/`, 2023. Accessed: March 2026.

[30] A. Rauschmayer, *Speaking JavaScript: An In-Depth Guide for Programmers*. Sebastopol, CA: O'Reilly Media, 2014.

[31] Stack Overflow, "Stack overflow developer survey 2023." `https://survey.stackoverflow.co/2023/`, 2023. Accessed: March 2026.

[32] A. Griffiths, *Learning React: Modern Patterns for Developing React Apps*. Sebastopol, CA: O'Reilly Media, 2nd ed., 2020.

[33] Red Hat, "Sql vs. nosql databases: What's the difference?." https://www.redhat.com/en/topics/data-management/sql-vs-nosql, 2021. Accessed: March 2026.

[34] J. Nielsen, *Usability Engineering*. Boston, MA: Academic Press, 1994.

[35] L. Bass, P. Clements, and R. Kazman, *Software Architecture in Practice*. Addison-Wesley, 3 ed., 2012.

[36] L. Richardson, M. Amundsen, and S. Ruby, *APIs RESTful: Servicios Web para sistemas distribuidos*. O'Reilly Media, 2013.

[37] R. C. Martin, *Clean Architecture: A Craftsman's Guide to Software Structure and Design*. Prentice Hall, 2017.

[38] RoomRaccoon, "¿qué es un hotel channel manager y por qué lo necesitas?." https://roomraccoon.es/blog/que-es-un-hotel-channel-manager/, 2023. Accedido: marzo de 2025.

[39] SiteMinder, "Integración PMS: Eficiencia en la gestión hotelera." https://www.siteminder.com/es/r/integracion-pms-hotel/, 2025. Accedido: marzo de 2025.

[40] Shiji Group, "Introducción a las API hoteleras: Actualizado para 2023." https://insights.shijigroup.com/es/an-introduccion-al-hotel-apis-2023/, 2023. Accedido: marzo de 2025.