# Desarrollo de un sistema web de gestión hotelera para la administración de clientes, habitaciones, reservas, pagos, paquetes turísticos y pqrs+ del Hotel Cacique Bolívar

Corporación Universitaria Remington.
Facultad de ingeniería.
Ingeniería de Sistemas.

Autores
Juan David Orrego Vélez
José Simón Atehortua Amaya
Juan David Sarrazola Fernández
Tutora
Gloria Amparo Lora Patiño
Proyecto de grado
2026

## Resumen

El presente trabajo de grado trata sobre la elaboración del desarrollo de un sistema web que gestione los requerimientos del Hotel Cacique Bolívar, buscando mejorar los procesos administrativos, la atención al cliente y la gestión de la información. Este trabajo nace de las necesidades que se han identificado en relación con la administración de clientes, habitaciones, reservas, pagos, paquetes turísticos y solicitudes donde no existe una organización y digitalización de la gestión.

Con la finalidad de identificar las necesidades del Hotel y de sus usuarios e identificar sus requerimientos se realizó un procedimiento de levantamiento de información mediante una investigación de campo de enfoque cuantitativo, utilizando una encuesta aplicada a 30 personas. A partir de la información que se levantó se establecieron los requerimientos no funcionales y funcionales del sistema a partir de los tenemos encontrado. Posteriormente a esto, se hizo uso de la metodología ágil Scrum donde organizamos el desarrollo de forma iterativa e incremental.

El sistema fue montado utilizando las tecnologías informáticas Java, Spring Boot, React y MySQL, haciendo uso de diferentes módulos para el control de usuarios, clientes, habitaciones, la gestión de reservas, pagos, la gestión de paquetes turísticos y PQRS+. Para poder estructurar de una manera organizada y coherente el contenido y los requerimientos obtenidos en la etapa anterior, se llevan a cabo modelos UML, diseño de base de datos, diseño de arquitectura, frontends, backend, diseño de interfaz y casos de uso.

Como parte del proceso se efectúan pruebas funcionales, de caja negra y de caja blanca que permiten verificar el correcto funcionamiento de los módulos implementados y la satisfacción de los requerimientos. El sistema fue desplegado en un entorno de producción con el uso de Windows, IIS y Cloudflare Tunnel permitiendo su utilización en condiciones reales.

**Palabras clave**

Gestión hotelera; sistema web; reservas; desarrollo de software; automatización.

## Tabla de contenidos

**Resumen** ..... **2**
**Palabras clave** ..... **3**
**Introducción** ..... **13**
**Marco teórico** ..... **15**
**Sistema de información 2.1.** ..... **15**
**Gestión hotelera 2.2.** ..... **17**
**Sistemas de gestión hotelera (HMS) 2.3.** ..... **19**
**Arquitectura cliente-servidor 2.4.** ..... **22**
**Arquitectura en capas 2.5.** ..... **24**
**Bases de datos relacionales 2.6.** ..... **25**
**Spring Boot 2.7.** ..... **27**
**React 2.8.** ..... **30**
**MySQL 2.9.** ..... **31**
**API REST 2.10.** ..... **33**
**Spring Security 2.11.** ..... **35**
**Lenguaje unificado de modelado (UML) 2.12.** ..... **37**
**Interfaz de usuario (UI) y Experiencia de usuario (UX) 2.13.** ..... **40**
**Pregunta problematizadora 3.1.** ..... **43**
**Justificación 3.2.** ..... **43**
**Objetivos** ..... **45**
**Objetivo Generales** ..... **45**
**Objetivos específicos** ..... **45**
**Metodología** ..... **47**
**Enfoque de investigación 6.1.** ..... **47**
**Tipo de investigación 6.2.** ..... **47**
**Población y muestra 6.3.** ..... **48**
**Técnica e instrumento de recolección 6.4.** ..... **48**
**Procedimiento 6.5.** ..... **49**
**Metodología de desarrollo de software 6.6.** ..... **50**
**Metodología Scrum 6.6.1.** ..... **50**

**Etapas del proyecto 6.6.2 ..... 50**
**Roles Scrum 6.6.3 ..... 51**
**Resultados y discusión ..... 53**
**Análisis de requerimientos 7.1. ..... 53**
**Levantamiento de información 7.1.1. ..... 53**
**Resultados de la encuesta 7.1.2. ..... 56**
**Análisis de los resultados de la encuesta 7.1.3. ..... 63**
**Matriz de hallazgos y requerimientos 7.1.4. ..... 65**
**Necesidades identificadas 7.1.5. ..... 69**
**Historia de Usuario 7.1.6. ..... 71**
**Requisitos de usuarios 7.1.7 ..... 74**
**Requisitos funcionales 7.1.8. ..... 80**
**Requisitos no funcionales 7.1.9 ..... 104**
**Diseño del Sistema 7.2. ..... 112**
**Arquitectura del sistema 7.2.1. ..... 112**
**Diseño de interfaces 7.2.2. ..... 115**
**Mockups y prototipos 7.2.3 ..... 117**
**Navegación del sistema 7.2.4. ..... 127**
**Modelado UML 7.2.5. ..... 128**
**Diseño de base de datos 7.2.6. ..... 210**
**Diagrama de clases 7.2.7. ..... 286**
**Análisis de riesgos 7.2.8 ..... 287**
**Desarrollo e implementación del Sistema 7.3. ..... 295**
**Herramientas de desarrollo 7.3.1 ..... 295**
**Tecnologías utilizadas 7.3.2. ..... 296**
**Implementación de la arquitectura 7.3.3 ..... 299**
**Estructura del proyecto 7.3.4. ..... 301**
**Evidencias del desarrollo 7.3.5. ..... 303**
**Validación y pruebas del Sistema 7.4. ..... 311**
**Plan de pruebas 7.4.1. ..... 311**
**Casos de prueba 7.4.2 ..... 313**

**Pruebas de caja blanca 7.4.3. ..... 319**
**Pruebas de caja negra 7.4.4. ..... 341**
**Resultados generales 7.4.5. ..... 355**
**Despliegue e instalación 7.5. ..... 370**
**Objetivo del despliegue e instalación 7.5.1. ..... 370**
**Arquitectura del despliegue 7.5.2 ..... 370**
**Configuración del servidor Windows 7.5.3 ..... 371**
**Despliegue del frontend mediante IIS 7.5.4. ..... 372**
**Configuración del backend como servicio de Windows 7.5.5. ..... 372**
**Configuración de la base de datos 7.5.6 ..... 373**
**Configuración de IIS y comunicación entre frontend y backend 7.5.7. ..... 374**
**Configuración del acceso mediante Cloudflare 7.5.8. ..... 374**
**Acceso desde la red local 7.5.9. ..... 375**
**Verificación del despliegue 7.5.10. ..... 375**
**Resultado del despliegue 7.5.11. ..... 376**
**Reflexiones ..... 378**
**Reflexión de la Fase I – Análisis 8.1. ..... 378**
**Reflexión de la Fase II – Diseño 8.2. ..... 378**
**Reflexión de la Fase III – Desarrollo 8.3. ..... 378**
**Reflexión de la Fase IV – Pruebas 8.4. ..... 379**
**Reflexión de la Fase V — Despliegue 8.5. ..... 379**
**Reflexión Final del Proyecto 8.6. ..... 379**
**Conclusiones y recomendaciones ..... 381**
**Conclusiones 9.1. ..... 381**
**Recomendaciones 9.2. ..... 383**
**Referencias ..... 385**

## Lista de tablas

Tabla 1 Tabla de roles ..... 51
Tabla 2 Matriz de Implementación del Sistema ..... 65
Tabla 3 Ficha Técnica de Tecnologías y Herramientas ..... 69
Tabla 4 Historia de Usuario ..... 72
Tabla 5 Matriz de Hallazgos y Requerimientos ..... 75
Tabla 6 Priorización de Requisitos Funcionales ..... 81
Tabla 7 Priorización de Requisitos Funcionales ..... 100
Tabla 8 Priorización de Requisitos Funcionales ..... 102
Tabla 9 Cuadro de Requerimientos Transversales de Calidad ..... 105
Tabla 10 Cuadro de Requerimientos Transversales de Calidad ..... 106
Tabla 11 Cuadro de Requerimientos Transversales de Calidad ..... 106
Tabla 12 Cuadro de Requerimientos Transversales de Calidad ..... 107
Tabla 13 Cuadro de Requerimientos Transversales de Calidad ..... 108
Tabla 14 Cuadro de Requerimientos Transversales de Calidad ..... 108
Tabla 15 Cuadro de Requerimientos Transversales de Calidad ..... 109
Tabla 16 Cuadro de Requerimientos Transversales de Calidad ..... 109
Tabla 17 Cuadro de Requerimientos Transversales de Calidad ..... 110
Tabla 18 Matriz de trazabilidad del sistema ..... 110
Tabla 19 Estructura del proyecto ..... 113
Tabla 20 Casos de usos ..... 130
Tabla 21 Escenario de Caso de Uso – Gestión de Clientes ..... 155
Tabla 22 Escenario del Caso de Uso – Gestión de Reservas ..... 159
Tabla 23 Escenario del Caso de Uso – Gestión de Tipos de Reserva ..... 163
Tabla 24 Escenario del Caso de Uso – Gestión de Habitaciones ..... 166
Tabla 25 Escenario del Caso de Uso – Gestión de Tipos de Habitación ..... 170
Tabla 26 Escenario del Caso de Uso – Gestión de Paquetes Turísticos ..... 173
Tabla 27 Escenario del Caso de Uso – Gestión de Tipos de Paquetes Turísticos ..... 178
Tabla 28 Especificación de Mensajes de Respuesta para Escenarios de Uso ..... 182
Tabla 29 Escenario del Caso de Uso – Gestión de Tipos de Pago ..... 186
Tabla 30 Escenario del Caso de Uso – Gestión de PQRS+ ..... 190
Tabla 31 Escenario del Caso de Uso – Gestión de Tipos de PQRS+ ..... 194
Tabla 32 Escenario del Caso de Uso – Gestión de Informes ..... 198
Tabla 33 Escenario del Caso de Uso – Gestión de Consultas ..... 200
Tabla 34 Escenario del Caso de Uso – Administración de Usuarios ..... 203
Tabla 35 Escenario del Caso de Uso – Administración de Perfiles ..... 207
Tabla 36 Diccionario Técnico de Clases y Atributos (Cliente) ..... 212
Tabla 37 Diccionario Técnico de Clases y Atributos (Recepcionista) ..... 215

Tabla 38 Diccionario Técnico de Clases y Atributos (Habitación) .....218
Tabla 39Diccionario Técnico de Clases y Atributos (Tipo de Habitación) .....221
Tabla 40 Diccionario Técnico de Clases y Atributos (Paquetes) .....223
Tabla 41 Diccionario Técnico de Clases y Atributos (Tipo Paquetes).....226
Tabla 42 Diccionario Técnico de Clases y Atributos (Usuario) .....228
Tabla 43 Diccionario Técnico de Clases y Atributos (Perfil).....229
Tabla 44 Diccionario Técnico de Clases y Atributos (Tipo Reserva).....232
Tabla 45 Diccionario Técnico de Clases y Atributos (Reservas).....234
Tabla 46 Diccionario Técnico de Clases y Atributos (PQRS+).....237
Tabla 47 Diccionario Técnico de Clases y Atributos (Tipo PQRS+) .....239
Tabla 48 Diccionario de clase Paquetes.....241
Tabla 49 Diccionario de clase Usuario .....245
Tabla 50 Diccionario de clase Perfil .....248
Tabla 51 Diccionario de clase PQRS+ .....251
Tabla 52 Diccionarios de clase Tipo PQRS+.....254
Tabla 53 Diccionarios de clase Habitaciones .....256
Tabla 54 Diccionarios de clase Tipo habitación.....259
Tabla 55 Diccionarios de clase Tipo paquete .....262
Tabla 56 Diccionarios de clase Clientes .....265
Tabla 57 Diccionarios de clase Recepcionista .....269
Tabla 58 Diccionarios de clase Pago.....272
Tabla 59 Diccionarios de clase Met_pago.....275
Tabla 60 Diccionarios de clase Reservas .....277
Tabla 61 Diccionarios de clase Tipo reserva .....281
Tabla 62 Diccionarios de clase Detalle reserva.....283
Tabla 63 Protocolo de Mensajería para la Mitigación de Errores .....288
Tabla 64 Herramientas de desarrollo utilizadas .....295
Tabla 65 Stack tecnológico del sistema .....297
Tabla 66 Casos de prueba del sistema .....313
Tabla 67 Tabla de Resultado.....327
Tabla 68 Tabla de Casos de Prueba.....328
Tabla 69 Casos de Prueba (Validación Fecha de Creación).....339
Tabla 70 Tabla de Entrada de Gestión de Reserva para la prueba de Caja Negra .....343
Tabla 71 Tabla de Salida de Gestión de Reserva para la prueba de Caja Negra.....344
Tabla 72 Tabla de Entrada de Gestión de Cliente para la prueba de Caja Negra.....346
Tabla 73 Tabla de Salidas de Gestión de Cliente para la prueba de Caja Negra .....347
Tabla 74 Tabla de Entrada de Gestión de Pago para la prueba de Caja Negra.....350
Tabla 75 Tabla de Salidas de Gestión de Pago para la prueba de Caja Negra.....350
Tabla 76 Tabla de Entrada de Gestión de PQRS+ para la prueba de Caja Negra .....353

Tabla 77 Tabla de Salidas de Gestión de PQRS+ para la prueba de Caja Negra.....354
Tabla 78 Resultados de los casos de prueba .....356

## Lista de figuras

Figura 1 Resultados de encuesta: Frecuencia de reserva de habitaciones de hotel ..... 56
Figura 2 Resultados de encuesta: Importancia de la facilidad de reserva en línea ..... 57
Figura 3 Resultados de encuesta: Experiencias de demora o confusión en reservas presenciales ... 58
Figura 4 Resultados de encuesta: Preferencia entre registro digital y registro en papel ..... 58
Figura 5 Resultados de encuesta: Grado de interés en la implementación de un sistema de reservas en línea ..... 59
Figura 6 Diagrama de Casos de Uso: Detalle del Módulo Administrador de Perfil ..... 60
Figura 7 Resultados de encuesta: Preferencias de métodos de pago para estadías hoteleras ..... 60
Figura 8 Resultados de encuesta: Experiencias negativas previas en servicios de reserva y registro ..... 61
Figura 9 Resultados de encuesta: Interés en la reserva de actividades turísticas locales mediante la plataforma ..... 62
Figura 10 Resultados de encuesta: Disposición de los usuarios para brindar retroalimentación pos-estadía ..... 63
Figura 11 Capas del proyecto ..... 114
Figura 12 Interfaz de inicio de sesión ..... 117
Figura 13 Menú ..... 117
Figura 14 Login por medio de cuenta de google ..... 118
Figura 15 Módulo de administrador ..... 118
Figura 16 Módulo de clientes ..... 119
Figura 17 Módulo de gestión de reservas ..... 119
Figura 18 Módulos de crear reservas ..... 120
Figura 19 Modulo de ocupación de las habitaciones ..... 121
Figura 20 Módulo de reserva exitosa ..... 121
Figura 21 Módulo de habitaciones ..... 122
Figura 22 Módulo de usuarios ..... 122
Figura 23 Registro de pago ..... 123
Figura 24 Módulo de PQRS+ ..... 124
Figura 25 Módulo de Crear PQRS+ ..... 125
Figura 26 Módulo de Gestión de PQRS+ ..... 125
Figura 27 Módulo de Paquetes Turístico ..... 126
Figura 28 Módulo de Informes ..... 127
Figura 29 Mapa de navegación ..... 128
Figura 30 Diagrama de casos de uso ..... 139
Figura 31 Diagrama de Casos de Uso de Gestión de Clientes ..... 139
Figura 32 Diagrama de Casos de Uso: Detalle del Módulo Gestión de Reserva ..... 140
Figura 33 Diagrama de Casos de Uso: Detalle del Módulo Gestión de Tipo de Reserva ..... 141
Figura 34 Diagrama de Casos de Uso: Detalle del Módulo Gestión de Habitación ..... 142

Figura 35 Diagrama de Casos de Uso: Detalle del Módulo Gestión de tipo de Habitación .....143
Figura 36 Diagrama de Casos de Uso: Detalle del Módulo Gestión Paquetes Turísticos .....144
Figura 37 Diagrama de Casos de Uso: Detalle del Módulo Gestión de Tipo de Paquetes Turístico .....145
Figura 38 Diagrama de Casos de Uso: Detalle del Módulo Gestión de pago .....146
Figura 39 Diagrama de Casos de Uso: Detalle del Módulo Gestión tipo de pago .....147
Figura 40 Diagrama de Casos de Uso: Detalle del Módulo Gestión de PQRS+.....148
Figura 41 Diagrama de Casos de Uso: Detalle del Módulo Gestión de tipo de PQRS+ .....149
Figura 42 Diagrama de Casos de Uso: Detalle de Gestión de informes.....150
Figura 43 Diagrama de Casos de Uso: Detalle del Módulo Gestión de consultas .....151
Figura 44 Diagrama de Casos de Uso: Detalle del Módulo de Administración de usuarios .....152
Figura 45 Diagrama de Casos de Uso: Detalle del Módulo de Administración de perfiles.....153
Figura 46 Modelo de Datos .....211
Figura 47 Diagrama de clases .....286
Figura 48 Arquitectura implementada del backend. .....300
Figura 49 Arquitectura implementada del frontend. .....301
Figura 50 Proyecto React en VS Code.....303
Figura 51 Base de datos en HeidiSQL.....304
Figura 52 Swagger .....305
Figura 53 Prueba de Postman con el endpoint http:/localhost:8080/api/auth/login .....306
Figura 54 Prueba de Postman con el endpoint http:/localhost:8080/api/clientes .....307
Figura 55 Prueba de Postman con el endpoint http:/localhost:8080/api/reservas .....307
Figura 56 Prueba de Postman con el endpoint http:/localhost:8080/api/pagos.....308
Figura 57 Prueba de Postman con el endpoint http:/localhost:8080/api/pqrs.....309
Figura 58 Consola ejecutando npm run dev .....310
Figura 59 Consola ejecutando mvn spring-boot:run .....311
Figura 60 Parte 1 de código analizado para la prueba .....319
Figura 61 Parte 2 de código analizado para la prueba .....320
Figura 62 Parte final de código analizado para la prueba .....320
Figura 63 Grafo de flujo.....321
Figura 64 Primer Camino Independiente .....323
Figura 65 Segundo Camino Independiente.....324
Figura 66 Tercer Camino Independiente.....325
Figura 67 Cuarto Camino Independiente .....326
Figura 68 Codigo de Prueba (Validar Fecha de Creación).....330
Figura 69 Grafo de Flujo (Validación de Fecha de Creación) .....331
Figura 70 Primer Camino Independiente (Validación Fecha de Creación) .....333
Figura 71 Segundo Camino Independiente (Validación Fecha de Creación).....334
Figura 72 Tercer Camino Independiente (Validación Fecha de Creación).....335

Figura 73 Cuarto Camino Independiente (Validación Fecha de Creación) .....336
Figura 74 Quinto Camino Independiente (Validación Fecha de Creación) .....337
Figura 75 Sexto Camino Independiente (Validación Fecha de Creación) .....338
Figura 76 Diagrama de Gestión de Reserva en Caja Negra .....342
Figura 77 Diagrama de Gestión de cliente en Caja Negra .....345
Figura 78 Diagrama de Gestión de Pagos en Caja Negra .....349
Figura 79 Diagrama de Gestión de PQRS+ en Caja Negra .....352
Figura 80 CP-021 Caso de prueba de inicio de sesión .....367
Figura 81 CP-022 Caso de prueba de inicio de sesión .....367
Figura 82 CP – 025 Caso de prueba de Habitación .....368
Figura 83 CP-028 Caso de prueba de Reservas.....369
Figura 84 Arquitectura de despliegue de la página del hotel.....371
Figura 85 Configuración del acceso mediante Cloudflare .....375

## Introducción

En la actualidad, el cambio digital se ha convertido en un factor importante para la competitividad de las organizaciones, especialmente en sectores como la industria hotelera, donde la experiencia del cliente y la eficiencia operativa son determinantes (Kotler et al., 2017; Zhang et al., 2025). El uso de plataformas digitales a ayudado a que los establecimientos turísticos optimicen lo que vendría siendo los procesos, la forma de interacción con los usuarios y amplíen su alcance en mercados globales (Organización Mundial del Turismo [OMT], 2023).

Por esto, surge la necesidad de que el Hotel Cacique Bolívar implemente una solución tecnológica que le permita adaptarse a las nuevas herramientas digitales de la actualidad, ya que la ausencia de una plataforma digital en la actualidad limita lo que vendría siendo la gestión eficiente de reservas, así mismo como la promoción adecuada de los mismos servicios y paquetes turísticos que se ofrecen, lo cual esto afecta a su competencia y su lugar en el mercador.

Por lo tanto, este proyecto de grado busca como objetivo el desarrollo de una página web completa e integra que facilite la administración de clientes, habitaciones, reservas, pagos, paquetes turísticos y PQRS+, ofreciendo a los usuarios una experiencia interactiva y accesible. Esta solución busca integrar funcionalidades orientadas tanto al cliente como a la gestión interna contribuyendo a la optimización de procesos y a la toma de decisiones basada en información.

Asimismo, el sistema incorpora un módulo de PQRS+ que permite registrar, gestionar y dar seguimiento a las peticiones, quejas, reclamos, sugerencias y felicitaciones de los usuarios. Esta funcionalidad fortalece la comunicación entre el hotel y sus clientes, mejora la trazabilidad de las solicitudes y facilita la toma de decisiones orientadas al mejoramiento continuo del servicio. Además, complementa los módulos de reservas y paquetes turísticos, ofreciendo una solución integral alineada con las necesidades identificadas durante el análisis de requerimientos.

Finalmente, este trabajo no solo responde a una necesidad específica del Hotel Cacique Bolívar, sino que también se enmarca en las tendencias actuales de digitalización en el sector turístico, evidenciando la importancia de las tecnologías de la información como herramienta estratégica para el crecimiento y la sostenibilidad empresarial.

## Marco teórico

**Sistema de información 2.1.** Un sistema de información es un conjunto estructurado de personas, procesos, datos, hardware y software que se relacionan y se interrelacionan para poder recoger, procesar, almacenar y entregar información para facilitar la toma de decisiones, la coordinación y el control en un marco de organización del trabajo (Laudon & Laudon, 2016).

En la actualidad, los sistemas de información son una herramienta estratégica para las organizaciones, ya que pueden contribuir a conseguir procesos más eficientes, mejorar los tiempos de respuesta y favorecer la calidad en la prestación de servicios a los usuarios (Irani, 2002; Aydiner et al., 2019). Asimismo, los sistemas de información pueden proporcionar información que apoye los procesos de toma de decisiones dentro de las organizaciones (Radford, 1974; O'Donnell et al., 2000).

- **Características:** Los sistemas de información disponen de múltiples características que les permiten ofrecer un correcto funcionamiento de esta, entre las que se pueden enumerar:
- **Automatización de procesos:** Permiten realizar de manera automática tareas repetitivas, consumiendo de esta forma menos tiempo y contribuyendo a disminuir el error humano. La automatización de procesos constituye además una de las capacidades mediante las cuales los sistemas de información pueden contribuir a mejorar el desempeño de las organizaciones (Kedziora, 2026).

- **Almacenamiento de información:** Permiten almacenar grandes cantidades de datos de forma estructurada y segura.
- **Acceso a la información en el momento que lo requiera:** Con esta característica, la organización dispone de información actualizada que puede apoyar la toma de decisiones en un determinado instante (Radford, 1974; O'Donnell et al., 2000).
- **Integración de procesos:** Pueden utilizarse por diferentes ámbitos para compartir información mediante una base de datos integrada.
- **Seguridad de la información:** Incorporan mecanismos de autorización, autenticación o respaldo para garantizar la confidencialidad e integridad de los datos.
- **Escalabilidad:** En función del crecimiento de la organización los sistemas de información permiten incorporar nuevas funcionalidades y/o módulos.
- **La importancia de los sistemas de información:** La utilización de sistemas de información de la organización le da una ventaja competitiva debido a optimizar los procesos y procedimientos administrativos y productivos, conseguir el mejor control de la información, ayudar a la reducción de la estructura de los costos operativos, participar en la toma de decisiones, asignar alto grado de calidad a los servicios que se prestan a los usuarios y a los clientes (Laudon & Laudon, 2016).

En efecto, en sectores con un alto volumen de información como el hotelero, la implementación de estos sistemas permite controlar de manera eficiente aspectos como las

reservas, el registro de clientes, la disponibilidad de habitaciones, los pagos y la generación de reportes administrativos.

También la información en tiempo real permite mejorar el proceso de planificación estratégica y satisfacer, de manera efectiva y oportuna, las necesidades de los clientes, todo ello contribuyendo a la mejora de la calidad del servicio que se presta.

- **Uso en las organizaciones:** En la actualidad, los sistemas de información son utilizados en casi todos los sectores de la economía: en empresas comerciales, en entidades financieras, en instituciones educativas, en hospitales y en empresas hoteleras ya que en el presente trabajo, el sistema de información desarrollado tiene por objeto la automatización de los procesos administrativos del Hotel Cacique Bolívar, permitiendo la gestión de clientes, habitaciones, reservas, pagos, paquetes turísticos y reportes vía web; y permitir mejorar el funcionamiento global del hotel (Laudon & Laudon, 2016).

**Gestión hotelera 2.2.** La Gestión Hotelera constituye el compendio de los pasos de carácter administrativo, operativo y estratégico que deben seguirse para conseguir que un hotel funcione adecuadamente. En este sentido, por lo tanto, la gestión hotelera implica la planificación, la organización, la dirección y el control de todas las acciones que se relacionan con el hecho de proporcionar servicios de hosting y, al mismo tiempo, ha de atender las necesidades de los clientes, así como conseguir la rentabilidad del negocio (Kotler et al., 2017).

En la actualidad, la gestión hotelera ha pasado por grandes cambios, influenciada por el impacto de las tecnologías de la información que han permitido gestionar de forma eficiente recursos como habitaciones, clientes, empleados, reservas, pagos y servicios complementarios (Dalgiç et al., 2018).

- **Procesos principales de un hotel:** La operación de un hotel involucra diferentes procesos que deben ejecutarse de manera coordinada para garantizar un servicio eficiente. Entre los procesos más importantes se encuentran:
    - Administración de reservas.
    - Registro de huéspedes (Check-In).
    - Asignación de habitaciones.
    - Control de disponibilidad.
    - Registro de pagos.
    - Facturación.
    - Gestión de paquetes turísticos.
    - Atención de solicitudes y PQRS+.
    - Generación de informes administrativos.

Cada uno de estos procesos requiere un adecuado control de la información para evitar inconsistencias que puedan afectar la atención al cliente (Kotler et al., 2017).

- **Administración hotelera:** La administración hotelera consiste en coordinar los recursos humanos, tecnológicos y financieros del establecimiento con el propósito de

ofrecer un servicio de calidad. Para ello es necesario contar con las herramientas tecnológicas que permitirán el control de las habitaciones disponibles, administrar las reservas, gestionar los pagos y mantener actualizada la información de los huéspedes.

Una adecuada administración también facilita el análisis del desempeño del hotel mediante indicadores e informes que apoyan la toma de decisiones (Kotler et al., 2017).

- **Contribución de la Automatización:** La Automatización de los Procesos del Hotel Cacique Bolívar, es poder disminuir notablemente los tiempos de atención y eliminar errores producidos por un manejo manual de la información, al cabo que mejorar la organización de la información y facilitar el acceso a esta misma, así como mejorar la coordinación de los aspectos del hotel.

En la práctica, en el hotel Cacique Bolívar, la Automatización sustituye a un proceso manual, que es reemplazado por la plataforma web capaz de incluir toda la información relativa a clientes, habitaciones, reservas, pagos y paquetes turísticos, así, por lo tanto, se mejora la efectividad de proceso del hotel.

**Sistemas de gestión hotelera (HMS) 2.3.** Un Sistema de Gestión Hotelera, conocido a nivel internacional como Hotel Management System (HMS), es una aplicación de software que permite gestionar de forma integral los diferentes procesos de un hotel, centralizando la información relacionada con las operaciones del establecimiento (Laudon & Laudon, 2016).

Este tipo de sistemas permite gestionar reservas, alojamientos, habitaciones, pagos, informes y servicios prestados por el hotel, y pueden contribuir a mejorar la operativa y la calidad de la prestación de servicios (Kotler et al., 2017).

- **Características**: Entre las principales características de un sistema de gestión hotelera se destacan:
    - Gestión centralizada de la información.
    - Administración de habitaciones.
    - Control de disponibilidad.
    - Gestión de reservas.
    - Administración de huéspedes.
    - Registro de pagos.
    - Generación de reportes.
    - Administración de usuarios mediante roles.
    - Acceso a la información en tiempo real.
    - Integración con bases de datos.
- **Ventajas**: La instalación de un sistema de gestión hotelera aporta múltiples ventajas tanto para la dirección como para los clientes. Entre las principales ventajas se encuentran:
    - Disminución de los errores administrativos
    - Mayor rapidez en la atención al cliente

- Optimización de los procesos de reservas
- Mejor control de habitaciones libres
- Centralización de la información
- Generación automática de los informes
- Mayor seguridad en el almacenamiento de los datos
- Apoyo en la toma de decisiones Módulos principales

- **Generalmente un sistema de gestión hotelera está conformado por módulos especializados como:**
  - Gestión de usuarios.
  - Gestión de clientes.
  - Gestión de habitaciones.
  - Gestión de reservas.
  - Gestión de pagos.
  - Gestión de paquetes turísticos.
  - Gestión de PQRS+.
  - Consultas.
  - Reportes administrativos.

Cada módulo cumple una función específica dentro del sistema, permitiendo mantener organizada la información y facilitar las operaciones del hotel.

- **Implementación del proyecto:** El trabajo realiza una implementación / un desarrollo de un sistema de información de gestión hotelera del Hotel Cacique Bolívar para su uso web. La aplicación permite realizar la gestión de la información de clientes, habitaciones, reservas, pagos, paquetes turísticos, usuarios y reportes, siendo una herramienta que gestiona la administración del hotel y gestiona la información.

**Arquitectura cliente-servidor 2.4.** La arquitectura tipo cliente-servidor es un modelo de desarrollo de software que consiste en un modelo de distribución de las responsabilidades de una aplicación en dos componentes, cliente y servidor. El cliente es la aplicación que usan las personas para interactuar con el sistema y el servidor se encarga de procesar las solicitudes, realizar la lógica del negocio y gestionar el acceso a la base de datos (Pressman & Maxim, 2020).

Este modelo de arquitectura permite simplemente hacer la separación de responsabilidades y construir aplicaciones que sean escalables, seguras y que sean fáciles de mantener.

- **Cliente**: El cliente es la forma en que los usuarios pueden interactuar con el sistema. En el caso de aplicaciones web, el cliente se suele desarrollar con tecnologías orientadas a la web, como React para ofrecer interfaces que sean dinámicas, intuitivas y que sean responsivas (Pressman & Maxim, 2020).

- **Servidor**: El servidor responde a las solicitudes del cliente, las procesa de acuerdo con las reglas del negocio y devuelve una respuesta adecuada. En nuestro caso, el servidor participante en el proyecto fue desarrollado con Spring Boot, implementando servicios REST para la comunicación con el frontend (Pressman & Maxim, 2020).
- **Comunicarse a través del protocolo HTTP:** La comunicación entre cliente y servidor se realiza a través del protocolo HTTP, aplicando los distintos métodos de las peticiones para consultar, registrar, modificar y borrar información [GET, POST, PUT y DELETE].
- **Ventajas:** Las principales ventajas de esta arquitectura son:
  - Separación de responsabilidades.
  - Escalabilidad.
  - Mantenimiento.
  - Seguridad en el acceso a la información.
  - Reutilización de servicios.
  - Integración con distintos clientes.

En el sistema desarrollado para el hotel Cacique Bolívar, esta arquitectura permite la comunicación mediante servicios REST de la interfaz desarrollada en React con el backend implementado en Spring Boot.

**Arquitectura en capas 2.5.** Esta arquitectura en capas es un modelo de diseño de software que dispone la aplicación en distintos niveles funcionales, con el objetivo de separar las responsabilidades de cada uno de los componentes del sistema, lo que favorece el mantenimiento del sistema (Pressman & Maxim, 2020). En una arquitectura en capas, se obtiene, entre otras características, una mayor reutilización del código mejora la organización del proyecto y se pueden realizar modificaciones al sistema sin que afecten al funcionamiento de las otras capas.

El sistema de gestión hotelera se desarrolló con este tipo de arquitectura en capas, pero está implementada con Spring Boot y contiene las siguientes capas:

- **Capa de presentación (Controller):** es la encargada de enviar las peticiones HTTP, las que vienen desde el frontend, y de devolver, en su caso, las respuestas HTTP.
- **Capa de negocio (Service):** contiene la lógica del negocio y las reglas de control del funcionamiento del sistema.
- **Capa de acceso a datos (Repository):** gestiona la comunicación y las operaciones con la base de datos, utilizando para ello Spring Data JPA.
- **Capa de persistencia (Entity):** contiene las entidades que permiten modelar las tablas de la base de datos y por lo tanto guardar la información necesaria para el sistema (Pressman & Maxim, 2020).
- **Ventajas de la arquitectura por capas:** La utilización de este modelo ofrece como resultado las siguientes ventajas:

- Mejor organización del código fuente
- Separación de responsabilidades
- Mayor facilidad para generar mantenimiento.
- Escalabilidad del sistema.
- Reutilización de componentes.
- Mayor facilidad para realizar pruebas.
- Mejor integración con tecnologías actuales como Spring Boot y React.

La implementación de esta arquitectura propició el desarrollo de un sistema modular, organizado y fácilmente mantenible para la gestión de los procesos de Hotel Cacique Bolívar (Pressman & Maxim, 2020).

**Bases de datos relacionales 2.6.** Un sistema de base de datos relacional es un sistema que almacena información en tablas en las que los datos se relacionan por medio de claves primarias y claves foráneas. Este modelo fue propuesto por Edgar F. Codd en 1970 y en la actualidad es uno de los modelos más utilizados en el desarrollo de aplicaciones empresariales debido a su eficiencia, fiabilidad y facilidad a la hora de administrar grandes volúmenes de información (Codd, 1970).

Cada tabla representa una entidad del sistema y está compuesta por registros que son las filas y loas atributos que son las columnas; las relaciones que se pueden establecer entre las tablas permiten mantener la integridad de la información y evitar la duplicación de los datos (Oracle, 2024b).

- **Modelo relacional:** El modelo relacional organiza la información en estructuras tabulares que representan los distintos elementos del sistema; cada una de las tablas almacena una parte específica de la información y se va a relacionar con el resto mediante claves que obligan a asegurar la consistencia de los datos.

En un hotel el modelo relacional hace fácil la gestión de las entidades clientes, habitaciones, reservas, pagos, usuarios, paquetes turísticos, etc. y permite definir las relaciones que existen entre ellas para mantener bien organizada la información.

- **Las tablas y relaciones:** Las tablas constituyen la estructura fundamental de una base de datos relacional. Cada tabla contiene un conjunto de registros y atributos relativos a una entidad específica del sistema, lo que permite un manejo eficiente y seguro de la información a través de consultas estructuradas (Elmasri & Navathe, 2015). Las relaciones permiten conectar la información de las diferentes tablas, que pueden existir entre ellas de forma:
  - Uno a uno (1:1).
  - Uno a muchos (1:N).
  - Muchos a muchos (N:M).

En este proyecto se utilizan relaciones del tipo uno a muchos, por ejemplo, la relación cliente / reserva o la relación entre tipos de habitaciones y habitaciones.

- **Llaves primarias:** La llave primaria (Primary Key) es el atributo que identifica un registro dentro de una tabla de forma única y cuyo valor no puede ser repetido ni nulo.

- **Ejemplo:**
    - IdCliente
    - IdReserva
    - IdHabitación
    - IdUsuario

- **Llaves foráneas:** Las llaves foráneas (Foreign Key) son aquellas que permiten establecer las relaciones entre las distintas tablas que componen la base de datos (Oracle, 2024b).

- **Ejemplo:**
    - Una reserva está relacionada con un cliente.
    - Una habitación pertenece a un tipo de habitación.
    - Un usuario pertenece a un perfil.

Estas relaciones permiten garantizar la integridad de la información que almacenamos o recogemos.

- **Integridad referencial**: La integridad referencial es un mecanismo que permite garantizar que las relaciones que existan entre tablas se mantengan. Este principio permite que no podamos registrar información que sea referenciales a un registro que no existe (Oracle, 2024b).

**Spring Boot 2.7.** Spring Boot es un marco de trabajo de desarrollo que permite la creación de aplicaciones empresariales usando Java. Se diseñó para facilitar la configuración de los proyectos gracias a la inclusión de configuraciones automáticas,

servidores embebidos y gestión de dependencias, con el objetivo final de minimizar el tiempo de desarrollo (VMware, 2024).

Hoy Spring Boot es una de las tecnologías utilizadas para el desarrollo de aplicaciones web y servicios REST, debido a que proporciona herramientas que facilitan la configuración y el desarrollo de aplicaciones basadas en Spring (Walls, 2022).

- **Características:** Las características más representativas de Spring Boot son:

  - Configuración automática del proyecto.
  - Servidor web embebido (Apache Tomcat).
  - Integración con Spring Security.
  - Integración con Spring Data JPA.
  - Desarrollo de APIs REST.
  - Gestión automática de dependencias mediante Maven.
  - Arquitectura por componentes.

Estas características permiten simplificar la configuración y el desarrollo de aplicaciones, reduciendo la necesidad de realizar configuraciones manuales y facilitando la integración de diferentes componentes dentro de un mismo proyecto (Walls, 2022; VMware, 2024).

- **Ventajas**: Las principales ventajas de utilizar Spring Boot son:

  - Reduce la cantidad de configuración manual.
  - Facilita el mantenimiento del proyecto.
  - Facilita la creación de aplicaciones escalables.

- Propicia la reutilización del código.
- Integra mecanismos de seguridad.
- Facilita la conexión con bases de datos.

- **Arquitectura**: Spring Boot propone una arquitectura estructurada, mediante capas independientes.
  - Las capas suelen ser:
  - Controller
  - Service
  - Repository
  - Entity
  - DTO

Cada una de ellas cumple una función específica en el sistema, permitiendo separar adecuadamente la lógica implementada y facilitar el mantenimiento de la aplicación. Esta separación de responsabilidades es consistente con las prácticas de diseño utilizadas en aplicaciones desarrolladas con Spring (Walls, 2022).

- **Se utilizó en el proyecto**: En el desarrollo del sistema de gestión hotelera se utilizó Spring Boot para implementación del backend de la aplicación. Con esta tecnología se implementaron los servicios REST que permiten gestionar la información de clientes, habitaciones, reservas, pagos, paquetes turísticos y usuarios,

y gestionar la comunicación con la base de datos MySQL mediante Spring Data JPA (VMware, 2024).

**React 2.8.** React es una librería de JavaScript desarrollada por Meta o más conocida Facebook para crear interfaces de usuarios dinámicas e interactivas. Su objetivo fundamental es hacer más sencillo el proceso de construir aplicaciones web basándose en los componentes reutilizables (React, 2024).

En la actualidad React es una de las librerías más adoptadas para el desarrollo del frontend gracias su alto rendimiento y la facilidad a la hora de construir aplicaciones SPA (Single Page Application).

- **Componentes:** Los componentes son la unidad de React. Cada componente es una parte independiente de la interfaz y puede ser usado de manera reutilizable en diversas secciones de la aplicación (React, 2024).

  Esa funcionalidad permite mantener el código y mejora la organización del proyecto.

- **JSX**: JSX o también conocidos como JavaScript XML es una extensión de JavaScript que permite escribir en código HTML dentro de los archivos JavaScript. Su uso permite una creación más rápida y organizada de las interfaces de usuario.

- **Hooks**: Los Hooks permiten utilizar características de React sin tener que recurrir a las clases (React, 2024).

- Los hooks más usados son:
  - useState

- useEffect
- useContext
- useNavigate

Todos ellos pueden ayudarte a manejar el estado y el ciclo de vida de los componentes.

- **SPA (Single Page Application):** React nos da la posibilidad de hacer aplicaciones SPA, las cuales cargan únicamente una sola página y sólo actualizan componentes sin volver a cargar por completo todo el navegador. Esto mejora significativamente la experiencia del usuario y reduce los tiempos de respuesta (React, 2024).
- **Virtual DOM**: React se basa en un Virtual DOM al comparar las distintas modificaciones que se puedan realizar en la interfaz para actualizar lo realmente necesario en el DOM. Esto optimiza el rendimiento de la aplicación y reduce el consumo de recursos.
- **Uso en el proyecto**: React fue utilizado en este proyecto para el desarrollo de la interfaz gráfica del sistema de gestión hotelera que revive la implementación de módulos para el manejo de usuarios, clientes, habitaciones, reservas, pagos, paquetes turísticos y reportes a través de una interfaz moderna, dinámica y responsiva.

**MySQL 2.9.** MySQL es un sistema de administración de bases de datos relacional open source muy conocido en el desarrollo de aplicaciones web. Su función primordial es la de guardar, administrar y consultar grandes cantidades de datos de forma eficaz y segura (Oracle, 2024b).

Gracias a su estabilidad y rendimiento, MySQL es una de las bases de datos más extendidas en las aplicaciones empresariales.

- **Características**
  - Entre sus principales características se encuentran:
  - Código abierto.
  - Alto rendimiento.
  - Soporte para múltiples usuarios.
  - Compatibilidad con diferentes sistemas operativos.
  - Integración con Java mediante JDBC.
  - Soporte para transacciones.
  - Integridad referencial (Oracle, 2024b).
- **Seguridad:** MySQL incorpora diferentes mecanismos para proteger la información almacenada, entre ellos:
  - Gestión de usuarios.
  - Asignación de permisos.
  - Control de acceso.
  - Copias de seguridad.
  - Recuperación de información (Oracle, 2024b).

- **Rendimiento:** La máquina de consultas de MySQL permite ejecutar las consultas de forma rápida incluso si la base de datos está poblada con gran cantidad de registros.

Incorpora además mecanismos de optimización para mejorar el rendimiento de aplicaciones empresariales (Oracle, 2024b).

- **Uso en el proyecto:** En el sistema que se ha desarrollado para el Hotel Cacique Bolívar utilizamos MySQL como sistema gestor de base de datos para almacenar la información de clientes, habitaciones, reservas, pagos, usuarios, perfiles, paquetes turísticos y otras entidades requeridas para el funcionamiento del sistema.

**API REST 2.10.** Una API REST o Representational State Transfer es un recuento de reglas para intercambiar información entre múltiples aplicaciones a través del protocolo HTTP. Este modelo arquitectónico permite que la información circule mediante recursos accesibles a través de URL y datos estructurados en formato típicamente JSON (Fielding, 2000).

Las APIs REST son una de las principales tecnologías que se utilizan en las aplicaciones web modernas gracias a su sencillez, escalabilidad e interoperabilidad.

- **Protocolo HTTP:** El protocolo HTTP permite el intercambio de información entre cliente y servidor mediante peticiones y respuestas. Cada petición efectuada desde el frontend se procesa en el backend, que envía como respuesta la información solicitada.

- **Métodos de HTTP:** Las REST APIs utilizan varios métodos para realizar las operaciones sobre los recursos.
  - Los principales son:
    - **GET:** Obtener información.
    - **POST:** Se utiliza para enviar datos al servidor con el fin de crear un recurso o registro nuevo.
    - **PUT:** Permite modificar o actualizar de manera completa la información de un recurso ya existente.
    - **DELETE:** Se emplea para eliminar un recurso o registro específico en el sistema (Fielding, 2000).

Estos métodos permiten implementar las operaciones en el CRUD (Create, Read, Update y Delete) que se utilizan en la mayoría de las aplicaciones Web.

- **Formato JSON:** JSON o JavaScript Object Notation es el formato que se utiliza para intercambiar información entre cliente y servidor (Fielding, 2000).

Posee una estructura ligera, lo que permite un mejor procesado de datos y el aumento de rendimiento de aplicaciones Web.

- **Uso en el proyecto:** En el proyecto actual se implementaron las REST APIs utilizando Spring Boot, para los servicios que permiten la comunicación entre el frontend en React y la base de datos MySQL. A través de estas API REST se realizan las operaciones de registro de usuarios, la gestión de clientes, la gestión de habitaciones, la gestión de reservas, la gestión de pagos, la gestión de paquetes

turísticos y la consulta de información. Todo ello permite la gestión de la comunicación entre los diferentes componentes del sistema.

**Spring Security 2.11.** Spring Security es un framework que se encuentra dentro del ecosistema Spring y que pone en marcha mecanismos de protección de aplicaciones web, haciendo uso de procesos de autenticación y autorización. Su finalidad es controlar el acceso a los recursos del sistema de manera que sólo puedan ser alcanzados por los usuarios que se hayan previamente autorizado para acceder a esta información o ejecutar ciertas acciones (Spring, 2024).

Este framework permite el uso de diferentes mecanismos de autenticación, así como también proteger las aplicaciones ante vulnerabilidades comunes.

- **Autenticación**: La autenticación es un proceso mediante el cual el sistema comprueba la identidad de un usuario, antes de permitirle el acceso a la aplicación, que en general entraña proveer un nombre de usuario y una contraseña haciendo uso de otros mecanismos de verificación (Spring, 2024). Cuando el usuario introduce sus credenciales, Spring Security debe comprobar, que la información que se encuentra en la base de datos se la misma que la que ha sido introducida. En caso de que esta verificación sea satisfactoria, el usuario podrá tener acceso a las funcionalidades que hacen referencia a su perfil.
- **Autorización:** La autorización permite determinar las acciones que debe poder realizar cada uno de los usuarios del sistema tras ser autenticados. En una aplicación

hotelera es común que diferentes tipos de usuarios dispongan de diferentes conexiones.

- Por ejemplo:
    - El Administrador puede gestionar usuarios, habitaciones, reservas y reportes.
    - El Recepcionista puede registrar clientes, gestionar reservas y consultar información.
    - El Gerente puede consultar informes y controlar la operación del hotel.
    - El Cliente solo puede consultar información de sus reservas.

De esa manera se evita que los usuarios que no tienen autorización para hacerlo puedan modificar información sensible del sistema.

- **Roles de usuario**: Spring Security gestiona los permisos a través del concepto de roles, o bien como representación de los diferentes perfiles que hay en una aplicación.

El sistema que hemos desarrollado para el Hotel Cacique Bolívar definía roles para gestionar el acceso a las funciones que ofrece el sistema, de tal manera que todo usuario solo tiene acceso a las funcionalidades que le están asignadas, en función de las responsabilidades que ejerce.

- **Protección de la información:** Además de lo relacionado con el control de acceso lo que Spring Security ofrece son distintos mecanismos para proteger la información del sistema, entre ellos se encuentran estos (Spring, 2024):

- Cifrar contraseñas mediante los algoritmos de cifrados seguros.
- Protegiendo las rutas restringidas del sistema.
- Controlando las sesiones.
- Validando los accesos.
- Prevenir accesos no autorizados.

Todas estas características permiten reforzar la seguridad de la aplicación y proteger la información en la base de datos.

- **Implementación en el proyecto correspondiente:** En el proyecto de Spring Security se utilizó para implementar el login, controlar el acceso a los distintos módulos del sistema y gestionar los permisos que hacen falta a los perfiles de administrador, gerente, recepcionista y cliente para garantizar la seguridad en el uso de la aplicación.

**Lenguaje unificado de modelado (UML) 2.12.** El Lenguaje de Modelado Unificado (Unified Modeling Language - UML) se define como un estándar para la representación gráfica del análisis y el diseño de los sistemas de software (Object Management Group, 2017). A partir de UML, se puede modelar la estructura y el comportamiento de una aplicación, antes de que se implemente, lo que hace que la comunicación entre los integrantes del equipo de desarrollo sea más ágil.

De este modo, toda la utilización de UML irá favoreciendo en buena medida entender cómo funciona el sistema en el que se está trabajando y, a la vez, se está documentando perfectamente cada uno de sus elementos.

- **Diagramas UML:** Los UML proporciona diferentes tipos de diagramas, cada uno orientado a representar aspectos específicos del sistema.
  - **Entre los diagramas más utilizados se encuentran:**
    - Diagrama de casos de uso.
    - Diagrama de clases.
    - Diagrama de actividades.
    - Diagrama de secuencia.
    - Diagrama de componentes.

Cada uno cumple una función distinta durante las fases de análisis y diseño.

- **Diagramas de casos de uso:** Los diagramas de casos de uso representan las funcionalidades del sistema desde la perspectiva de los usuarios.
- **Permiten identificar:**
  - Actores.
  - Funcionalidades.
  - Interacción entre usuarios y sistema.
- **En este proyecto fueron utilizados para representar procesos como:**
  - Gestión de reservas.
  - Gestión de habitaciones.
  - Gestión de pagos.
  - Gestión de clientes.

- Gestión de usuarios.
- Consultas.
- Reportes.

- **Diagrama de clases:** El diagrama de clases presenta la estructura del sistema a través de clases, atributos, métodos y relaciones que se dan entre esas clases.

Este tipo de diagrama ayuda en la implementación del software, porque es un punto de partida para implementar las entidades a través de programación.

- **Importancia de UML:** La utilización de UML aporta diversos beneficios durante el desarrollo del software:
  - Facilita el análisis del sistema.
  - Permite documentar la arquitectura.
  - Reduce errores durante el desarrollo.
  - Favorece la comunicación entre desarrolladores.
  - Facilita el mantenimiento del software

Sirve como apoyo para el mantenimiento del software (Object Management Group, 2017).

- **Aplicación en el proyecto:** Durante el diseño del sistema de gestión hotelera se utilizaron diagramas UML para representar los casos de uso y el diagrama de clases, permitiendo definir la estructura del software antes de iniciar la fase de implementación.

**Interfaz de usuario (UI) y Experiencia de usuario (UX) 2.13.** La Interfaz de Usuario (UI, por sus siglas en inglés) es el total de los elementos gráficos con el que se relacionan los usuarios a través de una aplicación informática. Son elementos tales como botones, formularios, menús, iconos, tablas, colores y demás elementos visuales que nos permiten utilizar el sistema (International Organization for Standardization, 2019).

Un diseño un poco ajustado de la interfaz permite un buen aprendizaje del sistema y una buena relación del usuario con la aplicación y la Experiencia de Usuario (UX, por sus siglas en inglés) son las percepciones de quien utiliza el sistema al igual que la satisfacción que él o ella obtiene.

Una buena experiencia de usuario nos proporciona procesos simples, intuitivos y eficientes.

- **Usabilidad:** La usabilidad representa la facilidad con la que un usuario puede aprender a utilizar un sistema para cumplir sus objetivos (International Organization for Standardization, 2019).
    - Entre sus principales características se encuentran:
        - Facilidad de aprendizaje.
        - Rapidez de uso.
        - Reducción de errores.
        - Satisfacción del usuario.
- **Accesibilidad:** La accesibilidad tiene como objetivo que las aplicaciones sean utilizadas por el mayor número de personas posible, entendiendo como tales a

usuarios con capacidades o posibles limitaciones de diferente tipo. Para ello se recomienda utilizar interfaces visuales claras, con colores adecuados, textos legibles y elementos con una identificación sencilla.

- **Diseño responsivo:** El diseño web responsivo (*Responsive Web Design*, RWD) es un enfoque de desarrollo que permite a una aplicación web adaptar dinámicamente su interfaz gráfica a los diferentes tamaños y resoluciones de pantalla de múltiples dispositivos, tales como computadoras de escritorio, tabletas y teléfonos inteligentes (Marcotte, 2010). De esta manera, se garantiza la accesibilidad del contenido y se logra una experiencia de usuario (UX) óptima y uniforme, independientemente del hardware o la plataforma que el cliente utilice para acceder al sistema.

## Planteamiento del problema y justificación

La falta de un sistema eficiente de gestión hotelera está resultando en un manejo deficiente de los huéspedes, ya que los registros se mantienen en formato físico, lo que dificulta el acceso inmediato a la información cuando es necesaria. Esto está causando demoras y confusiones al momento de realizar reservas de manera física, lo que impacta negativamente en la experiencia del cliente y la eficiencia operativa del hotel.

De esta forma para comprender esta magnitud de este problema, se ha realizado un análisis de brechas comparando el estado actual de los procesos del hotel Cacique Bolívar frente al modelo de operación optimizando lo vendría siendo el alcance con el software propuesto.

En la actualidad, el hotel Cacique Bolívar fundamenta su operación en procesos manuales y descentralizados lo que conlleva a una alta probabilidad de errores y lentitud considerable en los tiempos de repuestas debido al uso formatos tradicionales de registro. Asimismo, la carencia de un sistema formal para la gestión de PQRS+ deriva en un manejo informal de la información, lo que provoca la perdida de trazabilidad y disminución de la capacidad de respuesta ante las solicitudes de los usuarios.

Ante este escenario, el sistema de gestión web propuesto transformará la operatividad del hotel hacia un modelo digitalizado y eficiente. Esta solución optimizará el flujo del trabajo, permitiendo también consultas instantáneas y eliminando el uso de papel en los procesos críticos. Además, la integración de un módulo especializado de PQRS+ facilitara

una toma de decisiones fundamentada en los datos reales sobre los ingresos, el rendimiento operativo y los niveles de satisfacción del cliente.

En este sentido, la implementación de sistemas de información permite mejorar la eficiencia operativa y fortalecer la toma de decisiones dentro de las organizaciones (Laudon & Laudon, 2016).

**Pregunta problematizadora 3.1.** ¿Cómo puede el Hotel Cacique Bolivar mejorar la gestión de huéspedes y optimizar sus procesos de consulta, reservas y registro, a fin de garantizar una experiencia más fluida y satisfactoria tanto para los huéspedes, evitando así procesos, demoras, perdidas de cliente y confusiones a la hora de realizar reservas y consultas?

**Justificación 3.2.** La justificación de este proyecto radica en la necesidad de adaptación del Hotel Cacique Bolívar a un entorno empresarial cada vez más digitalizado y centrado en la experiencia del cliente. La creación de una página web integral no solo responde a la demanda actual del mercado, sino que también busca mejorar la eficiencia en la gestión de reservas y ofrecer una experiencia más completa y atractiva para los huéspedes.

Además, la centralización de la gestión de reservas y la integración de herramientas de análisis de datos permitirán al hotel tomar decisiones más informadas y optimizar sus recursos. La presencia en línea fortalecerá su visibilidad y reputación en el mercado, atrayendo tanto a nuevos clientes como fidelizando a los ya existentes. En resumen, este proyecto se justifica por su capacidad para mejorar la competitividad del hotel, tanto en

términos de eficiencia operativa como de satisfacción del cliente, en un contexto empresarial marcado por la digitalización y la experiencia del usuario

# Objetivos

## Objetivo Generales

Construir un sistema web orientado a la gestión hotelera para el Hotel Cacique Bolívar para el correcto control de la información de clientes, habitaciones, reservas, pagos, paquetes turísticos y módulo de PQRS+, para mejoramiento de los procesos administrativos, la atención al cliente y la toma de decisiones gracias a la gestión digital de la información.

## Objetivos específicos

**1**.Identificar los requerimientos funcionales y no funcionales del sistema web de gestión hotelera mediante el levantamiento de información con el personal del Hotel Cacique Bolívar.

**2.** Diseñar la arquitectura del sistema, la base de datos, las interfaces de usuario y los modelos UML del sistema web de gestión hotelera, de acuerdo con los requerimientos identificados para la administración de usuarios, clientes, habitaciones, reservas, pagos, paquetes turísticos, PQRS+, reportes y demás funcionalidades definidas en los requerimientos del proyecto.

**3.** Desarrollar el sistema web de gestión hotelera mediante Java, Spring Boot, React y MySQL, de acuerdo con la arquitectura y los módulos definidos en el diseño del sistema.

**4.** Validar el funcionamiento del sistema web de gestión hotelera mediante pruebas de caja negra, caja blanca y pruebas funcionales sobre los módulos implementados.

# Metodología

**Enfoque de investigación 6.1.** El proyecto se desarrolló por un enfoque cuantitativo ya que se llevó a cabo un proceso de recolección y análisis de la información a partir de la encuesta aplicada a los sujetos de la población referida al contexto y problemáticas del hospedaje y la reserva. La utilización de dicho enfoque permite obtener información de tipo medible respecto a las necesidades, preferencias y experiencias de los usuarios en lo que respecta a los procesos de la gestión hotelera. Los datos obtenidos y procedentes de la encuesta se utilizaron como insumo para la toma de decisiones en la identificación de necesidades y para establecer los requisitos a contemplar del sistema web que se desarrolló como producto del proyecto para el Hotel Cacique Bolívar.

**Tipo de investigación 6.2.** La presente investigación es de naturaleza descriptiva y de carácter de campo, descriptivo porque permitió reconocer y describir las necesidades referidas a los procesos de gestión hotelera, fundamentalmente aquellos que se relacionan con clientes, habitaciones, reservas, pagos, paquetes turísticos y solicitudes de usuarios; y de campo porque a partir de la encuesta cuyos resultados se mostraron, se obtuvo la práctica y directa información que sirvió para identificar esas necesidades del sistema, los resultados obtenidos permitieron dar una base de inicio para la definición de los requerimientos y orientar la toma de decisiones que guiaran la definición para el diseño y desarrollo de la solución propuesta.

**Población y muestra 6.3.** Se llevó a cabo la obtención de información mediante la aplicación de una encuesta a aquellas personas que pudieran estar dentro del contexto de los servicios de alojamiento, de reservas o bien de atención al cliente. La muestra que se utilizó estuvo formada por 30 personas que formaron parte de la recogida de información.

Los datos obtenidos nos permitieron conocer distintas percepciones y preferencias a partir de los distintos procesos de reserva, de registro, de pagos, de actividades turísticas y de comunicación con el propio establecimiento, pero que posteriormente nos fueron considerados para la identificación de necesidades y para la definición de los requisitos del sistema.

**Técnica e instrumento de recolección 6.4.** La técnica que se utilizó para obtener la información fue la encuesta, ya que esta técnica contribuye a recoger información directamente y del propio participante sobre las experiencias, necesidades y preferencias que tienen respecto a los servicios de gestión hotelera.

Se utilizó como instrumento de recolección la encuesta a partir de la técnica del cuestionario, el cual está formado por preguntas orientadas a conocer aspectos en función del número de reservas, facilidad de los procesos de reservas, preferencias respecto a los medios digitales, métodos de pago, actividades turísticas, así como los mecanismos de comunicación y retroalimentación.

La información que recogieron fue, después, organizada y analizada en función de la identificación de tendencias y necesidades que se puedan contemplar para el diseño y desarrollo del sistema web.

**Procedimiento 6.5.** La conducción del plano vino dada por las distintas fases que se relacionan de una forma u otra con la investigación. el análisis, el diseño, el desarrollo, validación del producto y el despliegue de este.

Como primera etapa se llevó a cabo el levantamiento de información a fin de determinar las necesidades cerca de los procesos de gestión del Hotel Cacique Bolívar. Para la recolección de esa información se aplicó una encuesta a 30 personas y luego se organizó y analizó la información obtenida.

De la información recolectada se identificaron las necesidades y se definieron los requerimientos funcionales y no funcionales del sistema, posteriormente se realizó el plano de la solución por medio de la definición de la arquitectura, modelos UML, estructura de la base de datos, interfaces y navegación del sistema.

El desarrollo del sistema se llevó a cabo una vez definido el correspondiente diseño del sistema como se estableció para el proyecto. Posteriormente se hizo el proceso de realización de pruebas de caja blanca y de caja negra con el fin de mostrar el correcto funcionamiento de los componentes que se había desarrollado y validar el cumplimiento de los requerimientos establecidos.

Finalmente, se hizo el proceso de despliegue del sistema en el entorno definido para el proyecto, utilizando Windows, IIS, una base de datos MySQL y el túnel Cloudflare para permitir el acceso al sistema.

## Metodología de desarrollo de software 6.6.

**Metodología Scrum 6.6.1.** Para realizar este proyecto, decidimos usar la metodología ágil Scrum. Definimos los roles clave para asegurar que el proceso se gestione bien, la comunicación funcione y alcancemos las metas propuestas.

### Etapas del proyecto 6.6.2

- **Levantamiento de información:** Se realizó la aplicación y análisis de encuestas para identificar las necesidades del hotel y los usuarios.
- **Análisis de re1querimientos:** Se definieron los requisitos funcionales y no funcionales del sistema.
- **Diseño del sistema:** Se elaboraron diagramas de casos de uso, modelo de datos, interfaces y estructura de navegación.
- **Desarrollo:** Se implementó el sistema web utilizando tecnologías orientadas al desarrollo web y bases de datos.
- **Pruebas:** Se realizaron pruebas funcionales y de validación para verificar el correcto funcionamiento del sistema.

- **Implementación:** Se integraron los módulos finales y se desplegó el sistema en un servidor de producción (Windows, IIS y Cloudflare) para su uso real.

**Roles Scrum 6.6.3**

*Tabla 1 Tabla de roles*

| Rol | Nombre | Descripción |
|---|---|---|
| Product Owner | Hotel Cacique Bolívar (Representado por Gloria Amparo Lora Patiño) | Es el responsable de definir y priorizar los requisitos del sistema. Se encarga de especificar las necesidades del producto y así asegurar el desarrollo general para el negocio. Además, supervisa el avance del proyecto y la toma de decisiones. |
| Scrum Master | Juan David Orrego Vélez | Es el facilitador del proceso Scrum. Su función principal es garantizar que la metodología ágil se aplique correctamente, eliminando |

| | | |
|---|---|---|
| | | impedimentos durante el desarrollo y asegurando que el equipo siga las prácticas establecidas. También promueve la comunicación y la mejora continua del equipo. |
| Development Team | Juan David Orrego Vélez, Juan David Sarrazola Fernández, José Simón Atehortua Amaya | Es el equipo encargado de llevar a cabo el desarrollo del sistema. Participa en las actividades de análisis, diseño, implementación y pruebas, incluyendo el desarrollo de la página web. Su objetivo es entregar incrementos funcionales del producto en cada iteración, cumpliendo con los requisitos y estándares de calidad establecidos. |

***Nota***. En esta tabla se muestra los roles de cada participante con la estructura scrum.

# Resultados y discusión

## Análisis de requerimientos 7.1.

### Levantamiento de información 7.1.1.

**1. ¿Con qué frecuencia reserva habitaciones en hoteles?**

a) Nunca

b) Ocasionalmente

c) Regularmente

d) Siempre

**2. ¿Qué tan importante considera la facilidad de reserva en línea al elegir un hotel?**

a) Muy importante

b) Importante

c) Poco importante

d) No importante

**3. ¿Ha experimentado demoras o confusiones al realizar reservas de manera presencial en hoteles?**

a) Sí, a menudo

b) Sí, ocasionalmente

c) No, nunca

d) No estoy seguro

**4. ¿Prefiere el registro de huéspedes realizado de manera digital o en papel?**

a) Digital

b) En papel

c) No tengo preferencia

**5. ¿Le gustaría que el hotel ofreciera un sistema de reserva de habitaciones en línea?**

a) Sí, definitivamente

b) Sí, podría ser útil

c) No, no lo considero necesario

d) No estoy seguro

**6. ¿Consideraría importante recibir un correo electrónico de confirmación de su reserva después de realizarla en línea?**

a) Sí, muy importante

b) Sí, importante

c) No es importante

d) No estoy seguro

**7. ¿Preferiría pagar su estadía en el hotel en efectivo, con tarjeta de crédito/débito o mediante transferencia bancaria?**

a) Efectivo

b) Tarjeta de crédito/débito

c) Transferencia bancaria

d) No tengo preferencia

**8. ¿Ha tenido alguna experiencia negativa relacionada con el registro de huéspedes o la gestión de reservas en hoteles? (por ejemplo, errores en la reserva, falta de disponibilidad, etc.)**

a) Sí

b) No

c) No estoy seguro

**9. ¿Le gustaría que el hotel ofreciera servicios adicionales, como reservas para actividades turísticas locales, a través de su plataforma en línea?**

a) Sí, definitivamente

b) Sí, podría ser útil

c) No, no lo considero necesario

d) No estoy seguro

**10. ¿Qué tan dispuesto estaría a brindar retroalimentación sobre su experiencia en el hotel a través de una encuesta en línea después de su estadía?**

a) Muy dispuesto

b) Dispuesto

c) Poco dispuesto

d) No estoy seguro

**Enlace del formulario:** https://forms.gle/mQPyKbB1u8Y5FiNr9

**Resultados de la encuesta 7.1.2.**

La Encuesta se le realizo a un total de 30 personas.




*Figura 1 Resultados de encuesta: Frecuencia de reserva de habitaciones de hotel*

En la gráfica se demuestra que el 65.2 % dijo que ocasionalmente hacen reservas en hoteles.




*Figura 2 Resultados de encuesta: Importancia de la facilidad de reserva en línea*

En la gráfica se demuestra que el 69.6% dijo que era muy importante la facilidad para hacer reservas en línea en hoteles.




*Figura 3 Resultados de encuesta: Experiencias de demora o confusión en reservas presenciales*

En la gráfica se demuestra que el 47.8% dijo que han experimentado demoras al realizar reservas de manera presencial.

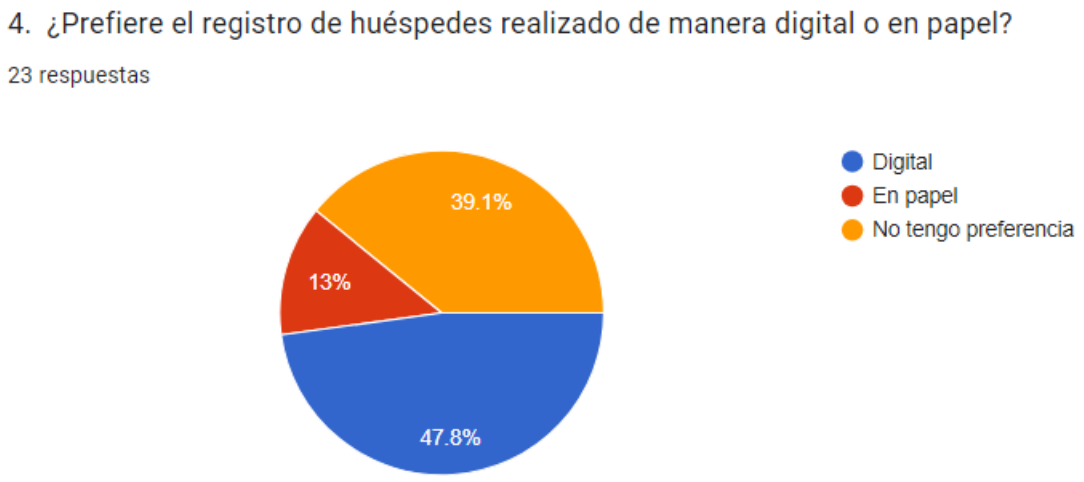



*Figura 4 Resultados de encuesta: Preferencia entre registro digital y registro en papel*

En la gráfica se demuestra que el 47.8 Dijo que prefieren realizar el registro de manera digital en el hotel




*Figura 5 Resultados de encuesta: Grado de interés en la implementación de un sistema de reservas en línea*

En la gráfica se demuestra que el 56.6% dijo que le gustaría que el hotel cacique implementara un sistema de reserva de habitaciones en línea.

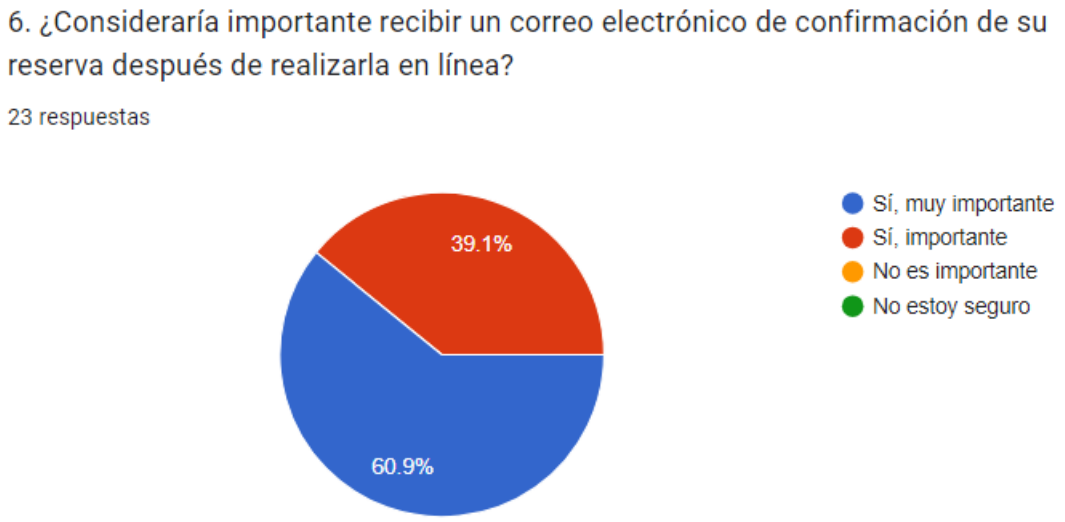



*Figura 6 Diagrama de Casos de Uso: Detalle del Módulo Administrador de Perfil*

En la gráfica se demuestra que el 60.9% considera importante recibir una notificación de confirmación para la reserva en línea.




*Figura 7 Resultados de encuesta: Preferencias de métodos de pago para estadías hoteleras*

En la gráfica se demuestra que el 65.2% dijo que no tenía preferencia a la hora de pagar.

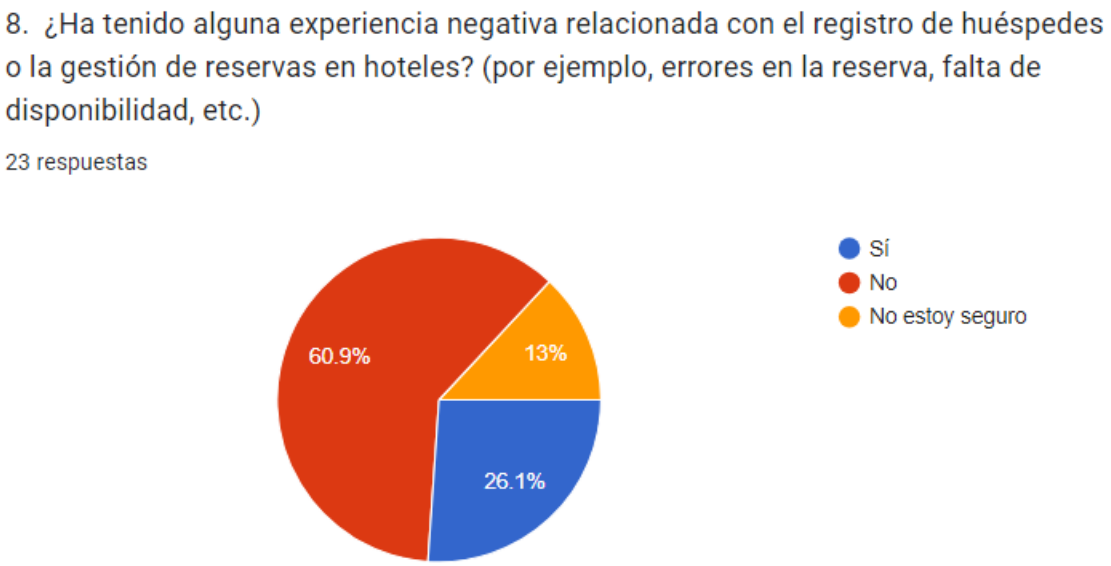



*Figura 8 Resultados de encuesta: Experiencias negativas previas en servicios de reserva y registro*

En la gráfica se demuestra que el 60.9% dijo que no ha tenido experiencias negativas relacionadas con el registro de huéspedes.

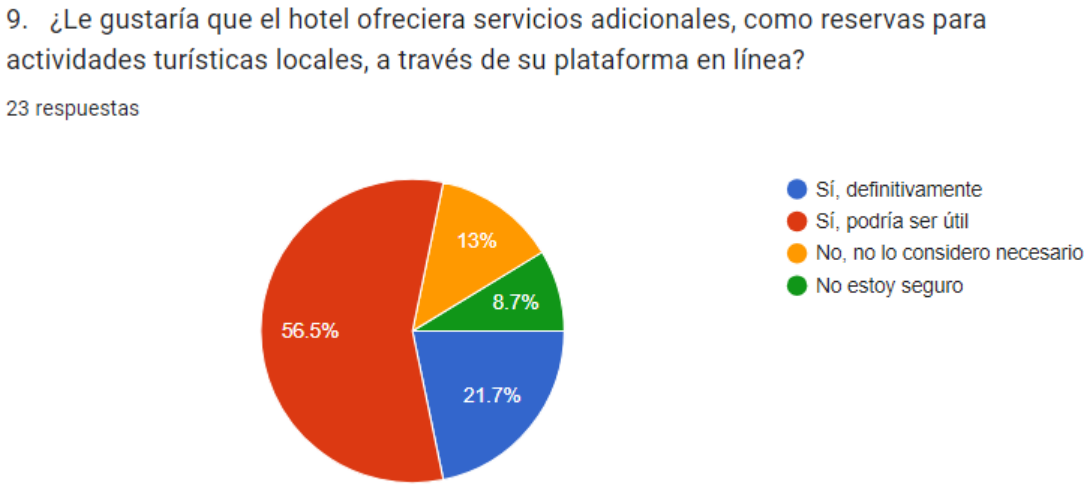



*Figura 9 Resultados de encuesta: Interés en la reserva de actividades turísticas locales mediante la plataforma*

En la gráfica se demuestra que el 56.5% dijo que le gustaría que el hotel ofreciera servicios adicionales tales como paquetes turísticos.

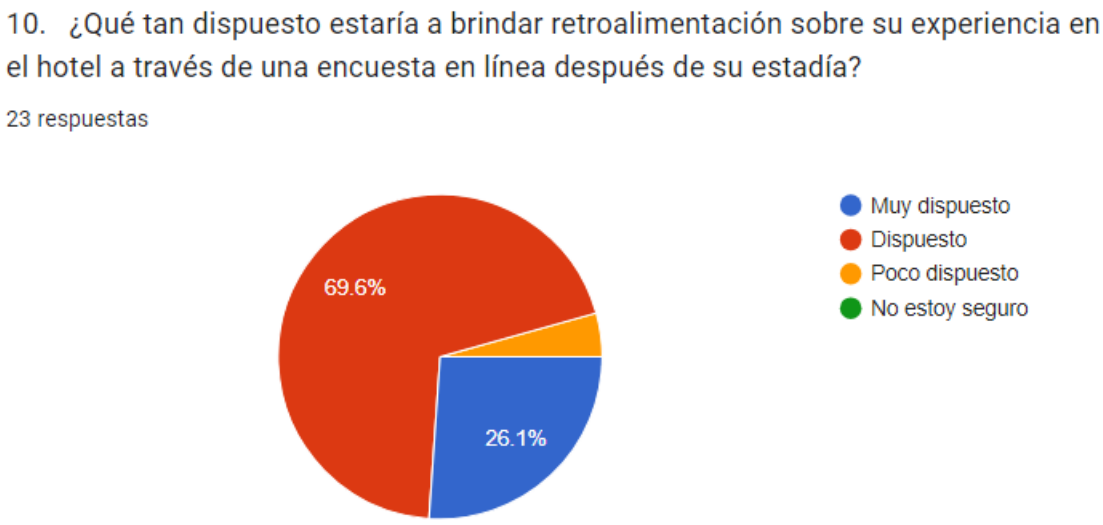



*Figura 10 Resultados de encuesta: Disposición de los usuarios para brindar retroalimentación pos-estadía*

En la gráfica se demuestra que el 69.6% dijo que está dispuesto a brindar su experiencia en el hotel a través de encuestas en línea.

**Análisis de los resultados de la encuesta 7.1.3.** Los resultados logrados permiten establecer una tendencia positiva hacia la digitalización de los procesos vinculados a la gestión hotelera. La asignación del interés en la facilidad de reservas en línea, el interés por contar con plataformas de reservas o la disposición de usuarios de aportar feedback, demuestran la existencia de necesidades que es preciso cubrir a través de mecanismos digitales que permitan establecer un contacto mediante el uso de tecnología entre los clientes de la investigación con el Hotel Cacique Bolívar, como se ha visto en aquellos resultados vinculados a los procesos presenciales o a la agenda de huéspedes en los cuales

se establece que existen oportunidades de mejorar aspectos relacionados con la gestión de la información o la atención prestada al usuario.

**Matriz de hallazgos y requerimientos 7.1.4.**

*Tabla 2 Matriz de Implementación del Sistema*

| Pregunta de la Encuesta | Hallazgo Principal | Requisito Relacionado | Justificación del Proyecto |
|---|---|---|---|
| 1. ¿Con qué frecuencia reserva habitaciones en hoteles? | Identifica el perfil del cliente (frecuente u ocasional). | RU-011 (Gestión de Cliente) / RF-001 (Registrar Cliente). | Justifica la implementación de un módulo de clientes que permita almacenar la información de huéspedes frecuentes para agilizar futuras reservas. |
| 2. ¿Qué tan importante considera la facilidad de reserva en línea al elegir un hotel? | El 69.6 % considera muy importante la reserva en línea. | RU-001 (Gestión de Reservas) / RF-007 (Registrar Reserva). | Valida la necesidad de desarrollar una plataforma web que facilite el proceso de reserva de habitaciones. |

| | | | |
|---|---|---|---|
| 3. ¿Ha experimentado demoras o confusiones al realizar reservas de manera presencial en hoteles? | El 47.8 % ha presentado inconvenientes durante el proceso presencial. | RU-010 (Gestión de Consultas) / RF-084 (Consultar Disponibilidad). | Justifica la implementación de consultas en tiempo real sobre la disponibilidad de habitaciones para reducir errores y tiempos de espera. |
| 4. ¿Prefiere el registro de huéspedes realizado de manera digital o en papel? | El 47.8 % prefiere el registro digital. | RU-011 (Gestión de Cliente) / RF-001 (Registrar Cliente). | Respalda la digitalización del proceso de registro de huéspedes para mejorar la eficiencia del servicio. |
| 5. ¿Le gustaría que el hotel ofreciera un sistema de reserva de habitaciones en línea? | El 56.5 % considera útil contar con un sistema de reservas en línea. | RU-001 (Gestión de Reservas) / RF-009 (Consultar Reserva). | Confirma la necesidad de implementar un sistema web que permita consultar la disponibilidad y |

| | | | |
|---|---|---|---|
| | | | realizar reservas desde cualquier lugar. |
| 6. ¿Consideraría importante recibir un correo electrónico de confirmación de su reserva después de realizarla en línea? | Existe interés en recibir una confirmación de la reserva. | RU-001 (Gestión de Reservas) / RF-008 (Actualizar Reserva). *(o agregar un RF de notificación si realmente lo implementarán).* | Brinda mayor confianza al cliente y mejora la experiencia del proceso de reserva. |
| 7. ¿Preferiría pagar su estadía en el hotel en efectivo, con tarjeta de crédito/débito o mediante transferencia bancaria? | Se evidencia la necesidad de múltiples métodos de pago. | RU-008 (Gestión de Tipo de Pagos) / RF correspondiente al registro de tipos de pago. | Permite ofrecer diferentes alternativas de pago adaptadas a las necesidades de los clientes. |
| 8. ¿Ha tenido alguna experiencia negativa | Los errores en procesos | RU-010 (Gestión de Consultas) / RF-084 | Justifica el control de disponibilidad en |

| | | | |
|---|---|---|---|
| relacionada con el registro de huéspedes o la gestión de reservas? | manuales generan desconfianza. | (Consultar Disponibilidad). | tiempo real para evitar conflictos y sobreventa de habitaciones. |
| 9. ¿Le gustaría que el hotel ofreciera servicios adicionales, como reservas para actividades turísticas locales? | Existe interés por servicios turísticos adicionales. | RU-003 (Gestión de Paquetes Turísticos) / RF-019 (Registrar Paquete Turístico). | Permite ampliar la oferta de servicios mediante paquetes turísticos integrados en la plataforma. |
| 10. ¿Qué tan dispuesto estaría a brindar retroalimentación sobre su experiencia en el hotel? | Los usuarios están dispuestos a compartir su opinión. | RU-013 (Gestión de PQRS+) / RF-055 (Registrar PQRS+). | Facilita el seguimiento de la satisfacción del cliente y apoya el mejoramiento continuo del servicio. |

**Nota:** Esta tabla muestra la correlación entre los resultados obtenidos en la fase de recolección de datos y la definición técnica de los requisitos, validando la necesidad de automatización en los procesos

**Necesidades identificadas 7.1.5.**

*Tabla 3 Ficha Técnica de Tecnologías y Herramientas*

| Necesidades | Características |
|---|---|
| Usuario | Código, nombre de usuario, contraseña, estado de la cuenta. |
| Perfil (Tipo de usuario) | Código, nombre del perfil (Cliente, Recepcionista, Gerente, Administrador), descripción y permisos asociados. |
| Cliente | Código, tipo de documento, número de documento, primer nombre, segundo nombre, primer apellido, segundo apellido, teléfono, correo electrónico, dirección. |
| Habitación | Código o número de habitación, tipo de habitación, capacidad máxima, precio por noche, estado (Disponible, Ocupada, Mantenimiento). |
| Tipo de Habitación | Código, nombre del tipo de habitación (Sencilla, Doble, Suite), descripción. |
| Reserva | Código de reserva, cliente, habitación, fecha de ingreso, fecha de salida, cantidad de huéspedes, estado de la reserva, tipo de pago. |
| Tipo de Reserva | Código, nombre del tipo de reserva, descripción. |

| | |
|---|---|
| Pago | Código de pago, reserva, fecha de pago, valor pagado, método de pago, estado del pago. |
| Tipo de Pago | Código, nombre del tipo de pago (Efectivo, Tarjeta, Transferencia, etc.). |
| Paquete Turístico | Código, nombre, destino, descripción, costo, cantidad máxima de personas, número de noches, habitación asociada (si aplica). |
| Tipo de Paquete Turístico | Código, nombre del tipo de paquete, descripción. |
| PQRS+ | Código de PQRS+, cliente, tipo de PQRS+, fecha de registro, descripción, estado, respuesta. |
| Tipo de PQRS+ | Código, nombre del tipo de PQRS+, descripción. |
| Informes | Informes de clientes, reservas, pagos y PQRS+, filtrados por rango de fechas. |
| Consultas | Consulta de huéspedes, disponibilidad de habitaciones, estado de reservas y seguimiento de PQRS+. |

**Nota:** Esta tabla muestra los requerimientos de información y los atributos esenciales de cada entidad, sirviendo como base técnica para el diseño del modelo de datos y la estructura del sistema.

**Historia de Usuario 7.1.6.** Las historias de usuario son un recurso o técnica que se adhiere a los enfoques ágiles, técnica que se utiliza para modelar las necesidades y las funcionalidades que requieren los usuarios de un sistema. El objetivo de una historia de usuario es expresar un requisito desde la perspectiva del usuario, permite identificar al rol que requiere una funcionalidad, la acción que quiere realizar y el resultado que se espera obtener. Esta forma de representación facilita la comunicación entre los usuarios y el equipo de desarrollo y permite establecer la relación entre las necesidades identificadas y aquellas funcionalidades que van a ser implementadas.

Dentro del ámbito de la ingeniería de requisitos ágil, las historias de usuario se han consolidado como un mecanismo muy utilizado para aprehender requisitos. De hecho, Dieste et al. (2021) afirman que las historias de usuario se utilizan por los grupos ágiles para representar requisitos, y gracias a su estructura son capaces de expresar una necesidad, dado que establece una relación entre el rol de usuario, la acción requerida y el beneficio esperado, lo cual permite que los requisitos puedan expresarse a través de un lenguaje cercano al usuario que, posteriormente, pueden ser analizados y refinados para ser transformados en funcionalidades del sistema.

*Tabla 4 Historia de Usuario*

| ID | Historia de Usuario | Módulo |
|---|---|---|
| **HU-01** | Como administrador, quiero gestionar usuarios (De los clientes) y perfiles para controlar el acceso al sistema. | Clientes / Seguridad |
| **HU-02** | Como cliente, quiero registrarme y gestionar mis datos para agilizar mis futuras reservas. | Clientes |
| **HU-03** | Como recepcionista, quiero consultar y editar clientes por cédula para mantener la base de datos actualizad. | Clientes |
| **HU-04** | Como administrador, quiero gestionar los tipos de habitación (Sencilla, Doble, Suite) y sus precios. | Habitaciones |
| **HU-05** | Como cliente, quiero buscar disponibilidad de habitaciones por fecha y capacidad. | Reservas |
| **HU-06** | Como cliente, quiero realizar una reserva y recibir un comprobante digital. | Reservas |
| **HU-07** | Como recepcionista, quiero cancelar o modificar reservas ante solicitudes del cliente. | Reservas |
| **HU-08** | Como cliente, quiero consultar y reservar paquetes turísticos adicionales. | Paquetes |

| | | |
|---|---|---|
| **HU-09** | Como administrador, quiero registrar y gestionar los pagos de las reservas. | Pagos |
| **HU-10** | Como cliente, quiero registrar una PQRS+ (Queja, Reclamo, Solicitud) sobre mi estancia. | PQRS+ |
| **HU-11** | Como gerente, quiero gestionar las respuestas a las PQRS+ para mejorar el servicio. | PQRS+ |
| **HU-12** | Como administrador, quiero generar reportes de ocupación y financieros por fechas. | Reportes |
| **HU-13** | Como gerente, quiero realizar consultas dinámicas sobre el rendimiento del hotel. | Consultas |

**Nota.** En esta tabla se encuentra todos los requisitos de usuario que se vieron presente para proyecto.

**Requisitos de usuarios 7.1.7**. Por lo que respecta a los requisitos de usuario, estos se definen como aquellos requisitos cuyas necesidades han de cumplir los sistemas para que los usuarios conseguyan los objetivos que se han propuesto alcanzar previa deteminación del contexto de uso. Dichos requisitos permiten convertir las necesidades identificadas durante el levantamiento de información en condiciones que pueden pasar a ser utilizadas posteriormente como ubicaciones iniciales desde las cuales comenzar a diseñar, desarrollar y evaluar la solución.

La norma ISO 25065 establece un marco para la especificación de los requisitos de usuario, e indica que también se incluirán aquellos requisitos vinculados a la interacción entre el usuario y el sistema y aquellos asociados con la calidad de uso y la obtención de resultados de dicha interacción (International Organization for Standardization [ISO], 2020); así, los requisitos de usuario van más allá de la mera exposición de necesidades del usuario sino que permiten establecer con mayor claridad qué es lo que necesita realizar el usuario mediante el uso del sistema y qué resultados espera obtener de dicha interacción.

*Tabla 5 Matriz de Hallazgos y Requerimientos*

| Id Requisito | Nombre del requisito | Descripción del requisito | Usuario | Prioridad |
|---|---|---|---|---|
| **RU-001** | Gestión de Reservas | El sistema permitirá a los clientes realizar reservas de habitaciones, consultar su disponibilidad, modificar o cancelar reservas, y al personal autorizado administrarlas de acuerdo con los permisos asignados. | Cliente, Administrador, Gerente, Recepcionista | Alta |
| **RU-002** | Gestión de Tipos de Reserva | El sistema permitirá administrar los tipos de reserva, registrando, consultando, actualizando e inhabilitando la información de cada tipo de reserva disponible. | Administrador, Recepcionista | Media |

| | | | | |
|---|---|---|---|---|
| **RU-003** | Gestión de Paquetes Turísticos | El sistema permitirá administrar los paquetes turísticos, registrando, consultando, actualizando e inhabilitando la información de los paquetes ofrecidos por el hotel. | Cliente, Administrador, Gerente, Recepcionista | Media |
| **RU-004** | Gestión de Tipos de Paquetes Turísticos | El sistema permitirá administrar los tipos de paquetes turísticos, permitiendo registrar, consultar, actualizar e inhabilitar la información correspondiente. | Administrador, Gerente | Baja |
| **RU-005** | Gestión de Habitaciones | El sistema permitirá administrar las habitaciones del hotel, registrando, consultando, actualizando e inhabilitando su | Gerente, Administrador | Alta |

| | | | | |
|---|---|---|---|---|
| | | información, disponibilidad y estado. | | |
| **RU-006** | Gestión de Tipos de Habitación | El sistema permitirá administrar los tipos de habitación, permitiendo registrar, consultar, actualizar e inhabilitar las categorías disponibles. | Gerente, Administrador | Media |
| **RU-007** | Gestión de Pagos | El sistema permitirá registrar, consultar y administrar los pagos asociados a las reservas realizadas por los clientes. | Gerente, Administrador, Recepcionista | Alta |
| **RU-008** | Gestión de Tipos de Pago | El sistema permitirá administrar los diferentes métodos o tipos de pago utilizados por el hotel. | Gerente, Administrador, Recepcionista | Media |
| **RU-009** | Gestión de Informes | El sistema permitirá generar informes de clientes, reservas, pagos y | Gerente, Administrador | Baja |

| | | | | |
|---|---|---|---|---|
| | | solicitudes (PQRS+) para apoyar el control administrativo y la toma de decisiones. | | |
| **RU-010** | Gestión de Consultas | El sistema permitirá realizar consultas sobre huéspedes, disponibilidad de habitaciones, estado de reservas y reclamos registrados. | Cliente, Administrador, Gerente, Recepcionista | Alta |
| **RU-011** | Gestión de Clientes | El sistema permitirá registrar, consultar, actualizar e inhabilitar la información de los clientes registrados en el hotel. | Cliente, Administrador, Gerente, Recepcionista | Alta |
| **RU-012** | Gestión de Usuarios | El sistema permitirá administrar las cuentas de usuario del sistema, permitiendo registrar, consultar, actualizar e | Administrador | Media |

| | | | | |
|---|---|---|---|---|
| | | inhabilitar usuarios según el rol asignado. | | |
| **RU-013** | Gestión de PQRS+ | El sistema permitirá registrar, consultar, responder y gestionar las peticiones, quejas, reclamos, sugerencias y felicitaciones presentadas por los clientes. | Cliente, Recepcionista, Gerente, Administrador | Baja |
| **RU-014** | Gestión de Tipos de PQRS+ | El sistema permitirá administrar los tipos de PQRS+, registrando, consultando, actualizando e inhabilitando las categorías de solicitudes. | Administrador, Gerente | Baja |
| **RU-015** | Gestión de Perfiles | El sistema permitirá administrar los perfiles de acceso del sistema, definiendo los permisos y | Administrador | Media |

| | | | | |
|---|---|---|---|---|
| | | privilegios de los usuarios según su rol. | | |

***Nota*:** Esta tabla muestra lo que es la trazabilidad entre los hallazgos de la investigación de los campos y la formulación de los requisitos de usuario (RU).

**Requisitos funcionales 7.1.8.** Los requisitos funcionales son las declaraciones que identifican las funciones, comportamientos y servicios que un sistema hay de proporcionar para dar respuesta a las necesidades de los usuarios evidenciadas. Dichos requisitos constituyen el puente para concretar las acciones que debe llevar a cabo el sistema en función de las entradas que recibe, el procesamiento que tiene que efectuar y las salidas o resultados que tiene que devolver. De este modo, los requisitos funcionales constituyen la base para definir el comportamiento que se espera del software y guiar posteriormente las acciones de desarrollo y validación (Malan et al., 2020).

Los requisitos funcionales se encuentran referidos a las acciones o comportamientos que deben encontrarse en el sistema. Por lo tanto, estos también pueden relacionarse con determinadas operaciones como puede ser registrar, consultar, modificar, eliminar, calcular, responder y determinadas interacciones que los usuarios deben llevar a cabo. La definición del propio requisito funcional debe ser tan precisa para poder establecer posteriormente si la funcionalidad en cuestión se encuentra realmente implementada de acuerdo a cómo se había establecido.

*Tabla 6 Priorización de Requisitos Funcionales*

| Id Requisito | Nombre del Requisito | Descripción del requisito | Usuario | Id RU |
|---|---|---|---|---|
| **RF-001** | Registrar Cliente | El sistema permitirá registrar la información de los clientes, incluyendo cédula, nombres, apellidos, teléfono, correo electrónico y observaciones, almacenando los datos de forma segura. | Cliente, Recepcionista, Administrador | RU-011 |
| **RF-002** | Actualizar Cliente | El sistema permitirá actualizar la información del cliente, excepto el número de cédula, garantizando la integridad de los datos registrados. | Recepcionista, Administrador | RU-011 |
| **RF-003** | Consultar Cliente | El sistema permitirá consultar la información de los clientes mediante el número de cédula u otros criterios de búsqueda autorizados. | Recepcionista, Gerente, Administrador | RU-011 |

| | | | | |
|---|---|---|---|---|
| **RF-004** | Inhabilitar Cliente | El sistema permitirá habilitar o inhabilitar un cliente previa confirmación del usuario autorizado, conservando la información registrada para efectos administrativos. | Administrador | RU-011 |
| **RF-005** | Cancelar Operación de Cliente | El sistema permitirá cancelar el proceso de registro o actualización de un cliente sin almacenar los cambios realizados. | Cliente, Recepcionista, Gerente, Administrador | RU-011 |
| **RF-006** | Finalizar Gestión de Clientes | El sistema permitirá finalizar la gestión de clientes y regresar al menú principal o al módulo correspondiente. | Cliente, Recepcionista, Gerente, Administrador | RU-011 |
| **RF-007** | Registrar Reserva | El sistema permitirá registrar una reserva asociando el cliente, la habitación, las fechas de ingreso y salida, el | Cliente, Recepcionista, Administrador | RU-001 |

| | | | | |
|---|---|---|---|---|
| | | tipo de reserva y el método de pago correspondiente. | | |
| **RF-008** | Actualizar Reserva | El sistema permitirá actualizar la información de una reserva existente, excepto el código de identificación de la reserva. | Recepcionista, Administrador | RU-001 |
| **RF-009** | Consultar Reserva | El sistema permitirá consultar la información de las reservas mediante el código de reserva u otros criterios autorizados. | Cliente, Recepcionista, Administrador, Gerente | RU-001 |
| **RF-010** | Inhabilitar Reserva | El sistema permitirá habilitar o inhabilitar una reserva previa confirmación del usuario autorizado, manteniendo la trazabilidad de la información registrada. | Recepcionista, Administrador | RU-001 |
| **RF-011** | Cancelar Operación de Reserva | El sistema permitirá cancelar el proceso de registro o actualización de una reserva | Cliente, Recepcionista, Administrador, Gerente | RU-001 |

| | | | | |
|---|---|---|---|---|
| | | sin guardar los cambios realizados. | | |
| **RF-012** | Finalizar Gestión de Reservas | El sistema permitirá finalizar la gestión de reservas y regresar al menú principal o al módulo correspondiente. | Cliente, Recepcionista, Administrador, Gerente | RU-001 |
| **RF-013** | Registrar Tipo de Reserva | El sistema permitirá registrar un nuevo tipo de reserva indicando su nombre, descripción y estado. | Recepcionista, Administrador | RU-002 |
| **RF-014** | Actualizar Tipo de Reserva | El sistema permitirá actualizar la información de un tipo de reserva previamente registrado. | Recepcionista, Administrador | RU-002 |
| **RF-015** | Consultar Tipo de Reserva | El sistema permitirá consultar la información de los tipos de reserva registrados en el sistema. | Cliente, Recepcionista, Administrador, Gerente | RU-002 |
| **RF-016** | Inhabilitar Tipo de Reserva | El sistema permitirá habilitar o inhabilitar un tipo de reserva | Recepcionista, Administrador | RU-002 |

| | | | | |
|---|---|---|---|---|
| | | previa confirmación del usuario autorizado. | | |
| **RF-017** | Cancelar Operación de Tipo de Reserva | El sistema permitirá cancelar el proceso de registro o actualización de un tipo de reserva sin almacenar los cambios realizados. | Recepcionista, Administrador, Gerente | RU-002 |
| **RF-018** | Finalizar Gestión de Tipos de Reserva | El sistema permitirá finalizar la gestión de los tipos de reserva y regresar al menú principal o al módulo correspondiente. | Cliente, Recepcionista, Administrador, Gerente | RU-002 |
| **RF-019** | Registrar Paquete Turístico | El sistema permitirá registrar la información de los paquetes turísticos, incluyendo código, nombre, costo y destino, almacenando los datos para su posterior consulta y administración. | Administrador, Gerente | RU-003 |

| | | | | |
|---|---|---|---|---|
| **RF-020** | Actualizar Paquete Turístico | El sistema permitirá actualizar la información de un paquete turístico previamente registrado, excepto su código de identificación. | Administrador, Gerente | RU-003 |
| **RF-021** | Consultar Paquete Turístico | El sistema permitirá consultar la información de los paquetes turísticos mediante el código o el nombre del paquete. | Cliente, Recepcionista, Administrador, Gerente | RU-003 |
| **RF-022** | Inhabilitar Paquete Turístico | El sistema permitirá habilitar o inhabilitar un paquete turístico previa confirmación del usuario autorizado, manteniendo la información registrada. | Administrador, Gerente | RU-003 |
| **RF-023** | Cancelar Operación de Paquete Turístico | El sistema permitirá cancelar el proceso de registro o actualización de un paquete turístico sin almacenar los cambios realizados. | Cliente, Recepcionista, Administrador, Gerente | RU-003 |

| | | | | |
|---|---|---|---|---|
| **RF-024** | Finalizar Gestión de Paquetes Turísticos | El sistema permitirá finalizar la gestión de paquetes turísticos y regresar al menú principal o al módulo correspondiente. | Cliente, Recepcionista, Administrador, Gerente | RU-003 |
| **RF-025** | Registrar Tipo de Paquete Turístico | El sistema permitirá registrar un nuevo tipo de paquete turístico indicando su nombre, descripción y estado. | Administrador, Gerente | RU-004 |
| **RF-026** | Actualizar Tipo de Paquete Turístico | El sistema permitirá actualizar la información de un tipo de paquete turístico previamente registrado. | Administrador, Gerente | RU-004 |
| **RF-027** | Consultar Tipo de Paquete Turístico | El sistema permitirá consultar la información de los tipos de paquetes turísticos registrados en el sistema. | Cliente, Recepcionista, Administrador, Gerente | RU-004 |
| **RF-028** | Inhabilitar Tipo de Paquete Turístico | El sistema permitirá habilitar o inhabilitar un tipo de paquete | Administrador, Gerente | RU-004 |

| | | | | |
|---|---|---|---|---|
| | | turístico previa confirmación del usuario autorizado. | | |
| **RF-029** | Cancelar Operación de Tipo de Paquete Turístico | El sistema permitirá cancelar el proceso de registro o actualización de un tipo de paquete turístico sin almacenar los cambios realizados. | Cliente, Recepcionista, Administrador, Gerente | RU-004 |
| **RF-030** | Finalizar Gestión de Tipos de Paquetes Turísticos | El sistema permitirá finalizar la gestión de los tipos de paquetes turísticos y regresar al menú principal o al módulo correspondiente. | Cliente, Recepcionista, Administrador, Gerente | RU-004 |
| **RF-031** | Registrar Pago | El sistema permitirá registrar un pago asociado a una reserva, incluyendo la información del método de pago, fecha, monto, descripción y estado del pago. | Recepcionista, Gerente, Administrador | RU-007 |

| | | | | |
|---|---|---|---|---|
| **RF-032** | Actualizar Pago | El sistema permitirá actualizar la información de un pago previamente registrado, garantizando la integridad de los datos. | Recepcionista, Gerente, Administrador | RU-007 |
| **RF-033** | Consultar Pago | El sistema permitirá consultar la información de los pagos registrados mediante los criterios de búsqueda definidos por el sistema. | Cliente, Recepcionista, Gerente, Administrador | RU-007 |
| **RF-034** | Inhabilitar Pago | El sistema permitirá habilitar o inhabilitar un pago previa confirmación del usuario autorizado, conservando el historial de la transacción. | Recepcionista, Gerente, Administrador | RU-007 |
| **RF-035** | Cancelar Operación de Pago | El sistema permitirá cancelar el proceso de registro o actualización de un pago sin almacenar los cambios realizados. | Cliente, Recepcionista, Gerente, Administrador | RU-007 |

| | | | | |
|---|---|---|---|---|
| **RF-036** | Finalizar Gestión de Pagos | El sistema permitirá finalizar la gestión de pagos y regresar al menú principal o al módulo correspondiente. | Cliente, Recepcionista, Gerente, Administrador | RU-007 |
| **RF-037** | Registrar Tipo de Pago | El sistema permitirá registrar un nuevo tipo o método de pago indicando su nombre, descripción y estado. | Recepcionista, Gerente, Administrador | RU-008 |
| **RF-038** | Actualizar Tipo de Pago | El sistema permitirá actualizar la información de un tipo de pago previamente registrado. | Recepcionista, Gerente, Administrador | RU-008 |
| **RF-039** | Consultar Tipo de Pago | El sistema permitirá consultar la información de los tipos de pago registrados en el sistema. | Cliente, Recepcionista, Gerente, Administrador | RU-008 |
| **RF-040** | Inhabilitar Tipo de Pago | El sistema permitirá habilitar o inhabilitar un tipo de pago previa confirmación del usuario autorizado. | Recepcionista, Gerente, Administrador | RU-008 |

| | | | | |
|---|---|---|---|---|
| **RF-041** | Cancelar Operación de Tipo de Pago | El sistema permitirá cancelar el proceso de registro o actualización de un tipo de pago sin almacenar los cambios realizados. | Cliente, Recepcionista, Gerente, Administrador | RU-008 |
| **RF-042** | Finalizar Gestión de Tipos de Pago | El sistema permitirá finalizar la gestión de los tipos de pago y regresar al menú principal o al módulo correspondiente. | Cliente, Recepcionista, Gerente, Administrador | RU-008 |
| **RF-043** | Registrar Habitación | El sistema permitirá registrar la información de las habitaciones, incluyendo número, tipo, capacidad, estado, precio y características, almacenando los datos de forma segura. | Gerente, Administrador | RU-005 |
| **RF-044** | Actualizar Habitación | El sistema permitirá actualizar la información de una habitación previamente | Gerente, Administrador | RU-005 |

| | | | | |
|---|---|---|---|---|
| | | registrada, manteniendo la integridad de los datos. | | |
| **RF-045** | Consultar Habitación | El sistema permitirá consultar la información de las habitaciones mediante el número, tipo o estado de disponibilidad. | Cliente, Recepcionista, Gerente, Administrador | RU-005 |
| **RF-046** | Inhabilitar Habitación | El sistema permitirá habilitar o inhabilitar una habitación previa confirmación del usuario autorizado, conservando su historial de información. | Gerente, Administrador | RU-005 |
| **RF-047** | Cancelar Operación de Habitación | El sistema permitirá cancelar el proceso de registro o actualización de una habitación sin almacenar los cambios realizados. | Gerente, Administrador | RU-005 |

| | | | | |
|---|---|---|---|---|
| **RF-048** | Finalizar Gestión de Habitaciones | El sistema permitirá finalizar la gestión de habitaciones y regresar al menú principal o al módulo correspondiente. | Gerente, Administrador | RU-005 |
| **RF-049** | Registrar Tipo de Habitación | El sistema permitirá registrar un nuevo tipo de habitación indicando su nombre, capacidad, descripción y tarifa correspondiente. | Gerente, Administrador | RU-006 |
| **RF-050** | Actualizar Tipo de Habitación | El sistema permitirá actualizar la información de un tipo de habitación previamente registrado. | Gerente, Administrador | RU-006 |
| **RF-051** | Consultar Tipo de Habitación | El sistema permitirá consultar la información de los tipos de habitación registrados en el sistema. | Cliente, Recepcionista, Gerente, Administrador | RU-006 |
| **RF-052** | Inhabilitar Tipo de Habitación | El sistema permitirá habilitar o inhabilitar un tipo de | Gerente, Administrador | RU-006 |

| | | | | |
|---|---|---|---|---|
| | | habitación previa confirmación del usuario autorizado. | | |
| **RF-053** | Cancelar Operación de Tipo de Habitación | El sistema permitirá cancelar el proceso de registro o actualización de un tipo de habitación sin almacenar los cambios realizados. | Gerente, Administrador | RU-006 |
| **RF-054** | Finalizar Gestión de Tipos de Habitación | El sistema permitirá finalizar la gestión de los tipos de habitación y regresar al menú principal o al módulo correspondiente. | Gerente, Administrador | RU-006 |
| **RF-055** | Registrar PQRS+ | El sistema permitirá registrar peticiones, quejas, reclamos, sugerencias y felicitaciones realizadas por los clientes, almacenando la información para su seguimiento. | Cliente, Recepcionista, Administrador | RU-013 |

| | | | | |
|---|---|---|---|---|
| **RF-056** | Actualizar PQRS+ | El sistema permitirá actualizar la información de una solicitud PQRS+, registrando el estado y la respuesta correspondiente cuando aplique. | Recepcionista, Gerente, Administrador | RU-013 |
| **RF-057** | Consultar PQRS+ | El sistema permitirá consultar las solicitudes PQRS+ registradas mediante diferentes criterios de búsqueda autorizados. | Cliente, Recepcionista, Gerente, Administrador | RU-013 |
| **RF-058** | Inhabilitar PQRS+ | El sistema permitirá habilitar o inhabilitar una solicitud PQRS+ previa confirmación del usuario autorizado, manteniendo el historial correspondiente. | Administrador | RU-013 |
| **RF-059** | Cancelar Operación de PQRS+ | El sistema permitirá cancelar el proceso de registro o actualización de una solicitud | Cliente, Recepcionista, Gerente, Administrador | RU-013 |

| | | | | |
|---|---|---|---|---|
| | | PQRS+ sin almacenar los cambios realizados. | | |
| **RF-060** | Finalizar Gestión de PQRS+ | El sistema permitirá finalizar la gestión de las solicitudes PQRS+ y regresar al menú principal o al módulo correspondiente. | Cliente, Recepcionista, Gerente, Administrador | RU-013 |
| **RF-061** | Registrar Tipo de PQRS+ | El sistema permitirá registrar un nuevo tipo de PQRS+, indicando su nombre, descripción y estado. | Administrador, Gerente | RU-014 |
| **RF-062** | Actualizar Tipo de PQRS+ | El sistema permitirá actualizar la información de un tipo de PQRS+ previamente registrado. | Administrador, Gerente | RU-014 |
| **RF-063** | Consultar Tipo de PQRS+ | El sistema permitirá consultar la información de los tipos de PQRS+ registrados en el sistema. | Recepcionista, Gerente, Administrador | RU-014 |

| | | | | |
|---|---|---|---|---|
| **RF-064** | Inhabilitar Tipo de PQRS+ | El sistema permitirá habilitar o inhabilitar un tipo de PQRS+ previa confirmación del usuario autorizado. | Administrador, Gerente | RU-014 |
| **RF-065** | Cancelar Operación de Tipo de PQRS+ | El sistema permitirá cancelar el proceso de registro o actualización de un tipo de PQRS+ sin almacenar los cambios realizados. | Administrador, Gerente | RU-014 |
| **RF-066** | Finalizar Gestión de Tipos de PQRS+ | El sistema permitirá finalizar la gestión de los tipos de PQRS+ y regresar al menú principal o al módulo correspondiente. | Administrador, Gerente | RU-014 |
| **RF-067** | Registrar Usuario | El sistema permitirá registrar usuarios del sistema asignando la información personal, credenciales de acceso y el rol correspondiente. | Administrador | RU-012 |

| | | | | |
|---|---|---|---|---|
| **RF-068** | Actualizar Usuario | El sistema permitirá actualizar la información de los usuarios registrados, garantizando la integridad de los datos. | Administrador | RU-012 |
| **RF-069** | Consultar Usuario | El sistema permitirá consultar la información de los usuarios registrados mediante diferentes criterios de búsqueda. | Administrador | RU-012 |
| **RF-070** | Inhabilitar Usuario | El sistema permitirá habilitar o inhabilitar usuarios del sistema previa confirmación del administrador, conservando el historial de la cuenta. | Administrador | RU-012 |
| **RF-071** | Registrar Perfil | El sistema permitirá registrar perfiles de usuario definiendo los permisos y privilegios correspondientes a cada rol del sistema. | Administrador | RU-015 |
| **RF-072** | Actualizar Perfil | El sistema permitirá actualizar la información y los permisos | Administrador | RU-015 |

| | | | | |
|---|---|---|---|---|
| | | asociados a un perfil previamente registrado. | | |
| **RF-073** | Consultar Perfil | El sistema permitirá consultar la información de los perfiles registrados y los permisos asignados a cada uno. | Administrador | RU-015 |
| **RF-074** | Inhabilitar Perfil | El sistema permitirá habilitar o inhabilitar perfiles de usuario previa confirmación del administrador, manteniendo la trazabilidad de los cambios realizados. | Administrador | RU-015 |

**Nota:** Esta tabla muestra cómo se detalla la especificación de los requisitos funcionales, definiendo las acciones permitidas, los datos requeridos y los actores responsables, lo cual garantiza la integridad operativa de cada módulo del sistema.

*Tabla 7 Priorización de Requisitos Funcionales*

| **Requisitos Funcionales (RF) Informes** | | | | |
|---|---|---|---|---|
| **Id** | **Nombre del Requisito** | **Descripción** | **Usuario** | **RU** |
| **RF-075** | Seleccionar Tipo de Informe | El sistema permitirá seleccionar el tipo de informe que se desea generar (clientes, pagos, reservas o PQRS+). | Gerente, Administrador | RU-009 |
| **RF-076** | Filtrar Información del Informe | El sistema permitirá filtrar la información por fechas, estado, cliente u otros criterios antes de generar el informe. | Gerente, Administrador | RU-009 |
| **RF-077** | Exportar Informe | El sistema permitirá exportar los informes generados en formatos compatibles como PDF o Excel para su almacenamiento o impresión. | Gerente, Administrador | RU-009 |
| **RF-078** | Finalizar Gestión de Informes | El sistema permitirá finalizar la gestión de informes y regresar al menú principal del sistema. | Gerente, Administrador | RU-009 |

| | | | | |
|---|---|---|---|---|
| **RF-079** | Generar Informe de Clientes | El sistema permitirá generar informes de los clientes registrados durante un período determinado. | Gerente, Administrador | RU-009 |
| **RF-080** | Generar Informe de Pagos | El sistema permitirá generar informes de los pagos realizados durante un período determinado. | Gerente, Administrador | RU-009 |
| **RF-081** | Generar Informe de Reservas | El sistema permitirá generar informes de las reservas realizadas durante un período determinado. | Gerente, Administrador | RU-009 |
| **RF-082** | Generar Informe de PQRS+ | El sistema permitirá generar informes de las solicitudes PQRS+ registradas durante un período determinado. | Recepcionista, Gerente, Administrador | RU-009 |

**Nota:** Esta tabla muestra cómo se proporciona la información estratégica para la gestión financiera y operativa, facilitando la toma de decisiones.

*Tabla 8 Priorización de Requisitos Funcionales*

| **Requisitos Funcionales (RF) Consultas** | | | | |
|---|---|---|---|---|
| **Id Requisito** | **Nombre del Requisito** | **Descripción del requisito** | **Usuario** | **Id RU** |
| **RF-083** | Consultar Huéspedes | El sistema permitirá consultar la información de los huéspedes registrados en el hotel mediante los criterios de búsqueda autorizados. | Gerente, Administrador | RU-010 |
| **RF-084** | Consultar Disponibilidad | El sistema permitirá consultar la disponibilidad de habitaciones de acuerdo con las fechas y características solicitadas por el usuario. | Gerente, Administrador, Recepcionista, Cliente | RU-010 |
| **RF-085** | Consultar Estado de Reservas | El sistema permitirá consultar el estado de las reservas registradas, indicando si se encuentran activas, canceladas o finalizadas. | Gerente, Administrador | RU-010 |

| | | | | |
|---|---|---|---|---|
| **RF-086** | Consultar Reclamos | El sistema permitirá consultar las solicitudes PQRS+ registradas por los clientes y realizar seguimiento a su estado. | Recepcionista, Cliente, Gerente, Administrador | RU-010 |

**Nota:** Esta tabla muestra como las consultas permiten el acceso inmediato a la información actualizada sobre disponibilidad y estados, de esa manera optimiza los tiempos de respuesta frente a las solicitudes de los usuarios y la administración.

**Requisitos no funcionales 7.1.9.** Los requisitos no funcionales (RNF) son las propiedades de calidad, limitaciones, y condiciones que ha de cumplir un sistema además de dar las funcionalidades demandadas. A la par de los requisitos funcionales, que indican las acciones y los servicios a los que el sistema ha de dar respuesta, los requisitos no funcionales establecen las condiciones en las que deben funcionar esas funcionalidades y permiten establecer como de bien responde el sistema a las expectativas que tienen los usuarios con respecto a él (Olsson et al., 2022).

Los requisitos no funcionales pueden ser muy variados y pueden guardar relación con los distintos atributos de calidad que se encuentren presentes en el software artificial. La seguridad, el rendimiento, la confiabilidad, la disponibilidad, la facilidad de uso, la escalabilidad y la mantenibilidad son características que tienen un efecto directo sobre la calidad de la solución, ya que un sistema informático puede cumplir las funciones de forma correcta y, aún así, ofrecer un producto que sea deficiente cuando no responde con una adecuación suficiente en lo que hace referencia al rendimiento, la seguridad o la facilidad de uso (Rahy, 2021).

Del mismo modo, los requisitos no funcionales pueden imponer restricciones en el diseño y en la implementación del sistema. La definición de los requisitos no funcionales deberá poder comprobar las características esperadas o las que se necesitan, evitando definiciones demasiado generales o ambiguas. La investigación de los requisitos de calidad demuestra la necesidad de definir los requisitos de calidad de manera adecuada, ya que la ambigüedad, la completitud, la consistencia y la corrección afectan a la calidad de los

requisitos, y, en consecuencia, al desarrollo del producto de software (Montgomery et al., 2022).

- **Facilidad de uso ("usability")**

*Tabla 9 Cuadro de Requerimientos Transversales de Calidad*

| ID. Requisito | Descripción del requisite |
|---|---|
| **RNF-001** | Capacitación, antes de intensificar el uso del sistema los usuarios deben conocer su modo de uso. |
| **RNF-002** | Actualizar la información por ingreso o retiro de algún funcionario. |
| **RNF-003** | Diseño adecuado a las necesidades del usuario, para que la web sea intuitiva y sencilla de usar cumpliendo con los siguientes parámetros:<br>Tendrá una interfaz atrayente: formación de los elementos acorde al diseño.<br>La carga de información deberá ser rápida. |

***Nota:*** Esta tabla muestra cómo la calidad enfocada en la usabilidad y la facilidad de uso, muestra como los criterios de diseño de interfaz y formación necesaria para garantizar una curva de aprendizaje mínima y una alta eficiencia operativa.

- **Confiabilidad**

*Tabla 10 Cuadro de Requerimientos Transversales de Calidad*

| ID. Requisito | Descripción del requisito |
|---|---|
| **RNF-004** | Tiempo de respuesta: se espera minimizar el tiempo a un promedio de 20 segundos, con el fin de que no se haga muy pesada la interacción con la Base de Datos manejada vía web. |
| **RNF-005** | Asignar suficiente espacio a la base de datos para soportar las grandes cantidades de información suministradas. |
| **RNF-006** | Configuración adecuada del equipo, para soportar la correcta instalación de la aplicación. |

***Nota*:** Esta tabla muestra cómo se detalla los requerimientos de confiabilidad enfocados en asegurar la estabilidad del sistema, optimizando los tiempos de respuesta.

- **Ambiente de trabajo "Performance"**

*Tabla 11 Cuadro de Requerimientos Transversales de Calidad*

| ID.Requisito | Descripción del requisite |
|---|---|
| **RNF-001** | El sistema debe estar disponible las 24 horas del día |
| **RNF-002** | Debe asegurar la permanente actualización de la base de datos, cuando se registre la información. |

***Nota*:** Esta tabla muestra cómo se define los requerimientos de desempeño enfocados en la alta disponibilidad del sistema y la sincronización de datos en tiempo real.

- **Restricciones de diseño**

*Tabla 12 Cuadro de Requerimientos Transversales de Calidad*

| ID.Requisito | Descripción del requisite |
|---|---|
| **RNF-003** | Tiempo de respuesta: se espera minimizar el tiempo a un promedio de 20 segundos, con el fin de que no se haga muy pesada la interacción con la Base de Datos manejada vía web. |
| **RNF-004** | Asignar suficiente espacio a la base de datos para soportar las grandes cantidades de información suministradas. |
| **RNF-005** | Configuración adecuada del equipo, para soportar la correcta instalación de la aplicación. |

***Nota*:** Esta tabla muestra cómo se define las limitaciones técnicas y los estándares de los diseños necesarios para asegurar la compatibilidad del sistema con los navegadores web modernos.

- **Seguridad**

*Tabla 13 Cuadro de Requerimientos Transversales de Calidad*

| ID.Requisito | Descripción del requisite |
|---|---|
| **RNF-006** | El lenguaje de programación del sistema estara basado en Java utilizando el Framewrok Spring Boot . |
| **RFN-007** | Se requiere de licenciamiento para el desarrollo del software. |

***Nota*:** Esta tabla muestra cómo se define los requisitos de seguridad orientados a garantizar la confidencialidad e integridad de la información.

- **Seguridad y respaldo de la información**

*Tabla 14 Cuadro de Requerimientos Transversales de Calidad*

| ID.Requisito | Descripción del requisite |
|---|---|
| **RNF-008** | Encriptación de las claves. |
| **RNF-009** | Realizar copia de seguridad, automático cada 6 horas, en servidor ubicado en otro espacio fuera de la empresa. |
| **RNF-010** | Los usuarios deberán estar registrados, bajo la modalidad del perfil del sistema. |

***Nota*:** Esta tabla muestra los protocolos de seguridad y contingencia del sistema, incluyendo el cifrado de contraseñas, la gestión de perfiles de usuario y la realización de

copias de seguridad automáticas para garantizar la protección y disponibilidad de la información.

- **Documentación de usuario y sistemas de ayuda**

*Tabla 15 Cuadro de Requerimientos Transversales de Calidad*

| ID.Requisito | Descripción del requisite |
|---|---|
| **RNF-011** | Capacitación a los usuarios del sistema, con el fin de lograr un buen manejo del mismo. |
| **RNF-012** | Manuales de usuario. |

***Nota*:** Esta tabla muestra como la protección de activos digitales, asegurando que el acceso sea controlado mediante perfiles y que la información cuente con duplicidad sistemática fuera de la infraestructura local.

- **Interfaz de Usuario**

*Tabla 16 Cuadro de Requerimientos Transversales de Calidad*

| ID.Requisito | Descripción del requisite |
|---|---|
| **RNF-013** | Será manejado a través de un computador de escritorio. |
| **RNF-014** | Los colores de la aplicación serán gris claro, azul claro y blanco. |
| **RNF-015** | El texto será manejado en color negro, la fuente de la letra será Arial tamaño 12. |

***Nota*:** Esta tabla muestra cómo se establece los lineamientos de la interfaz de usuario de esa manera definiendo como los estándares visuales y de navegación que garantizan una experiencia coherente y profesional.

- **Interfaces de comunicación**

*Tabla 17 Cuadro de Requerimientos Transversales de Calidad*

| ID.Requisito | Descripción del requisite |
|---|---|
| **RNF-016** | El acceso al software será vía web, debe asegurarse la estabilidad y seguridad de la conexión. |
| **RNF-017** | Comunicación con la interfaz de usuario. |

***Nota*:** Esta tabla muestra los requerimientos de comunicación del sistema, definiendo el canal de acceso web y los protocolos necesarios para garantizar una transferencia de datos.

- **Matriz de trazabilidad**

*Tabla 18 Matriz de trazabilidad del sistema*

| Módulo | Historias de Usuario | Requisitos Funcionales | Casos de Uso | Casos de Prueba Asociados |
|---|---|---|---|---|

| | | | | |
|---|---|---|---|---|
| Usuarios y Perfiles | HU-01 | RF-067 – RF-074 | CU-01 Gestión de Usuarios y Perfiles | CP-001, CP-002, CP-003, CP-010, CP-011, CP-022, CP-038, CP-039, CP-040. |
| Clientes | HU-02, HU-03 | RF-001 – RF-006 | CU-02 Gestión de Clientes | CP-004, CP-005, CP-023, CP-024. |
| Habitaciones y Tipo de Habitación | HU-04 | RF-043 – RF-054 | CU-03 Gestión de Habitaciones | CP-006, CP-007, CP-025, CP-026. |
| Reservas y Tipo de Reserva | HU-05, HU-06, HU-07 | RF-007 – RF-018 | CU-04 Gestión de Reservas | CP-008, CP-009, CP-027, CP-028, CP-029. |
| Paquetes Turísticos y Tipo de Paquete | HU-08 | RF-019 – RF-030 | CU-05 Gestión de Paquetes Turísticos | CP-013, CP-014, CP-015, CP-030, CP-031. |
| Pagos y Tipo de Pago | HU-09 | RF-031 – RF-042 | CU-06 Gestión de Pagos | CP-018, CP-034, CP-035. |

| | | | | |
|---|---|---|---|---|
| PQRS+ y Tipo de PQRS+ | HU-10, HU-11 | RF-055 – RF-066 | CU-07 Gestión de PQRS+ | CP-016, CP-017, CP-032, CP-033. |
| Informes | HU-12 | RF-075 – RF-082 | CU-08 Gestión de Informes | CP-019, CP-036. |
| Consultas | HU-13 | RF-083 – RF-086 | CU-09 Gestión de Consultas | CP-020, CP-037. |
| Servicios API REST | Todas las HU relacionadas | API REST del sistema | CU-10 Consumo de API REST | CP-012 |

***Nota*:** Esta tabla muestra cómo se establece la relación entre los requisitos funcionales, las historias de usuario, casos de usos y los casos de prueba.

## Diseño del Sistema 7.2.

**Arquitectura del sistema 7.2.1.** El sistema que fue desarrollado para el Hotel Cacique Bolívar se basa en una arquitectura de tipo cliente-servidor, implementada mediante una estructura en capas utilizando el framework Spring Boot en el backend.

Con esta arquitectura se permite una correcta separación de responsabilidades, facilitando el mantenimiento, la escalabilidad y una buena organización del código.

- **Tipo de Arquitectura**

Se implementó una arquitectura en capas, compuesta por los siguientes niveles:

- **Capa de presentación (Controller)**: Encargada de recibir las solicitudes del usuario (HTTP) y enviar las respuestas correspondientes.
- **Capa de negocio (Service)**: Contiene la lógica del sistema, donde se procesan las reglas de negocio.
- **Capa de acceso a datos (Repository)**: Se encarga de la interacción con la base de datos mediante consultas.
- **Capa de modelo (Entity / DTO)**: Representa las estructuras de datos del sistema.

- **Estructura del Proyecto**

El sistema está organizado en paquetes, siguiendo buenas prácticas de desarrollo en Spring Boot:

*Tabla 19 Estructura del proyecto*

| Paquete | Descripción |
|---|---|
| **Config** | Configuración general del sistema |
| **controller** | Maneja las peticiones HTTP |
| **Dto** | Objetos de transferencia de datos |
| **Entity** | Modelos de base de datos |

| | |
|---|---|
| **exception** | Manejo de errores |
| **mapper** | Conversión entre entidades y DTO |
| **repository** | Acceso a la base de datos |
| **security** | Configuración de seguridad |
| **service** | Lógica de negocio |

**Nota**. En la tabla se presentan las capas del proyecto con breves descripciones.

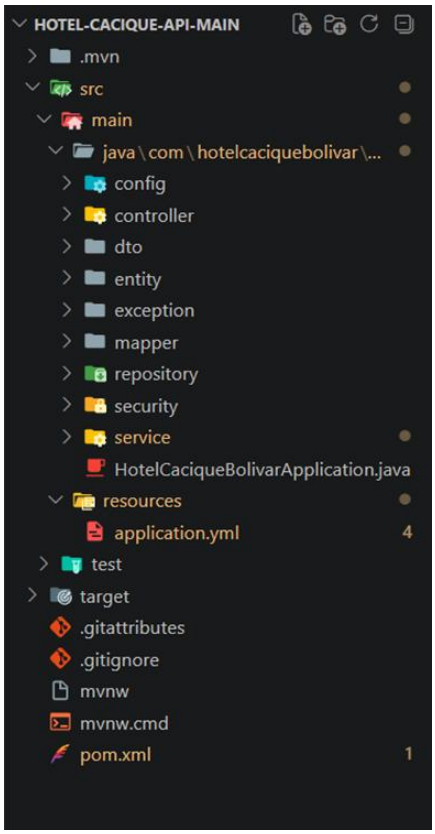



*Figura 11 Capas del proyecto*

**Nota**. En la imagen se muestran las capas del proyecto.

- **Flujo de Funcionamiento del Sistema**

El flujo del sistema funciona de la siguiente manera:

1. El usuario realiza una solicitud desde la interfaz web.
2. La solicitud es recibida por el **Controller**.
3. El **Controller** envía la información al **Service**.
4. El **Service** procesa la lógica del negocio.
5. Si es necesario, el **Repository** consulta la base de datos.
6. La información retorna en forma de respuesta al usuario.

- **Configuración del Sistema**

El sistema utiliza el archivo application.yml, donde se definen:

- Configuración de la base de datos
- Puerto del servidor
- Parámetros del sistema
- Seguridad y conexión

**Diseño de interfaces 7.2.2.** Las interfaces del sistema fueron diseñadas con el objetivo de ofrecer una experiencia de usuario intuitiva y a su vez permitiendo que tanto el administrador como los clientes puedan acceder fácilmente a las diferentes funcionalidades

del sistema. Para su diseño se tuvieron en cuenta criterios de usabilidad, consistencia visual, facilidad de navegación y organización de la información.

- **Las principales interfaces desarrolladas corresponden a:**
    - Inicio de sesión.
    - Panel principal (Dashboard).
    - Gestión de clientes.
    - Gestión de habitaciones.
    - Gestión de reservas.
    - Gestión de pagos.
    - Gestión de paquetes turísticos.
    - Gestión de usuarios.
    - Generación de reportes.

**Mockups y prototipos 7.2.3.**

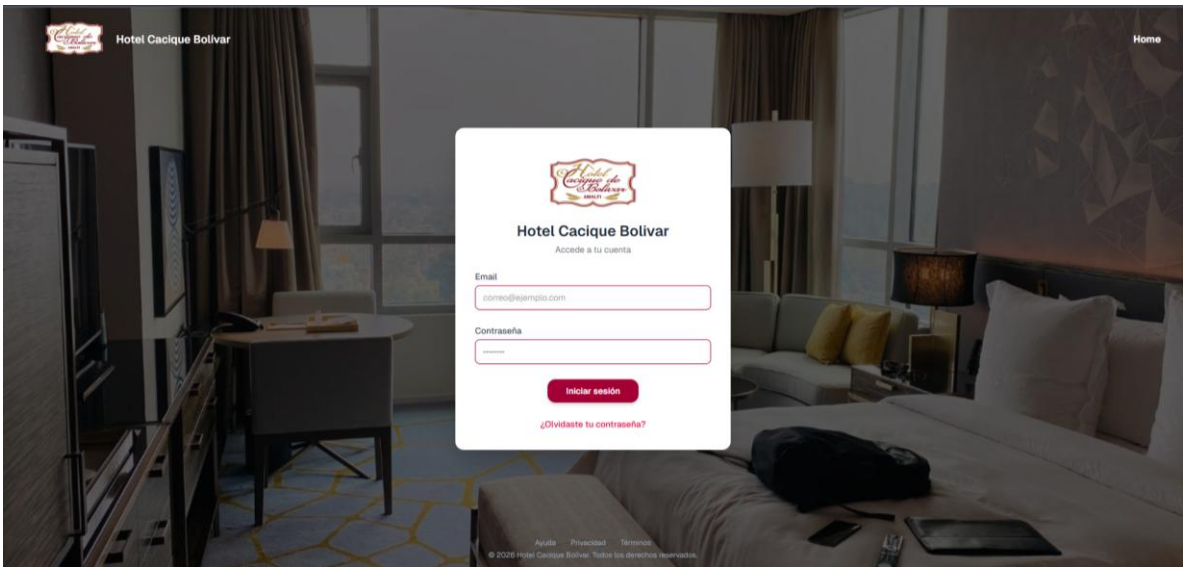



*Figura 12 Interfaz de inicio de sesión*

***Nota.*** Pantalla utilizada para la autenticación de usuarios del dashboard.

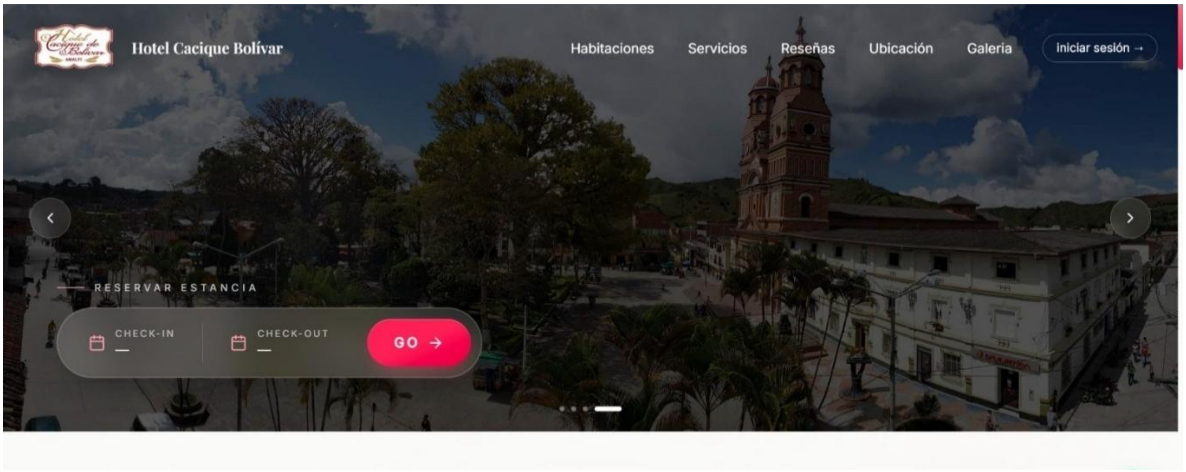



*Figura 13 Menú*

***Nota.*** Pantalla de inicio de la página web.

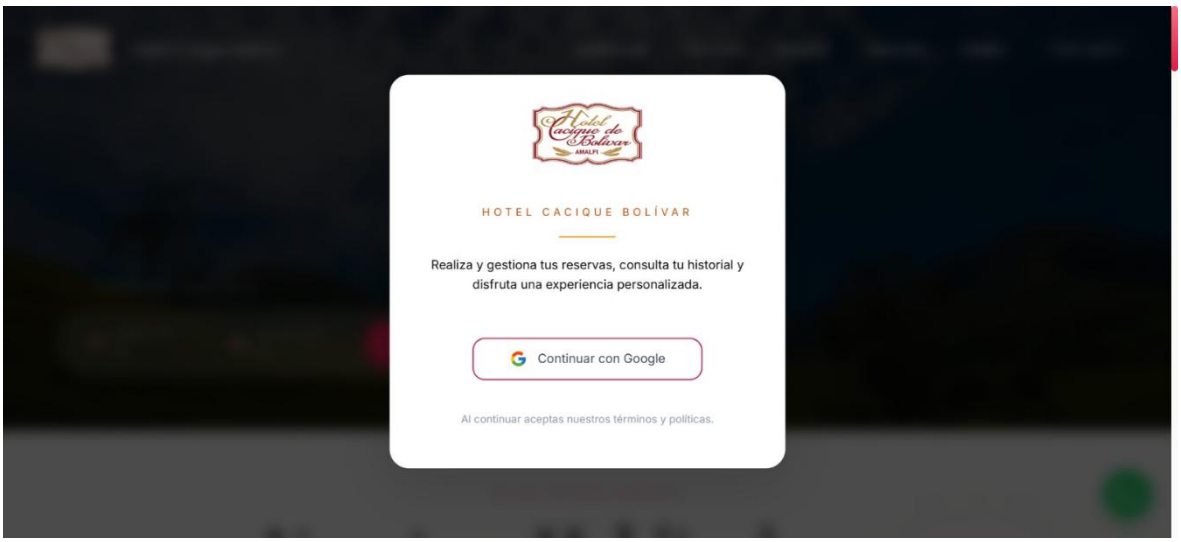



*Figura 14 Login por medio de cuenta de google*

***Nota.*** Módulo para la autenticación de los clientes.

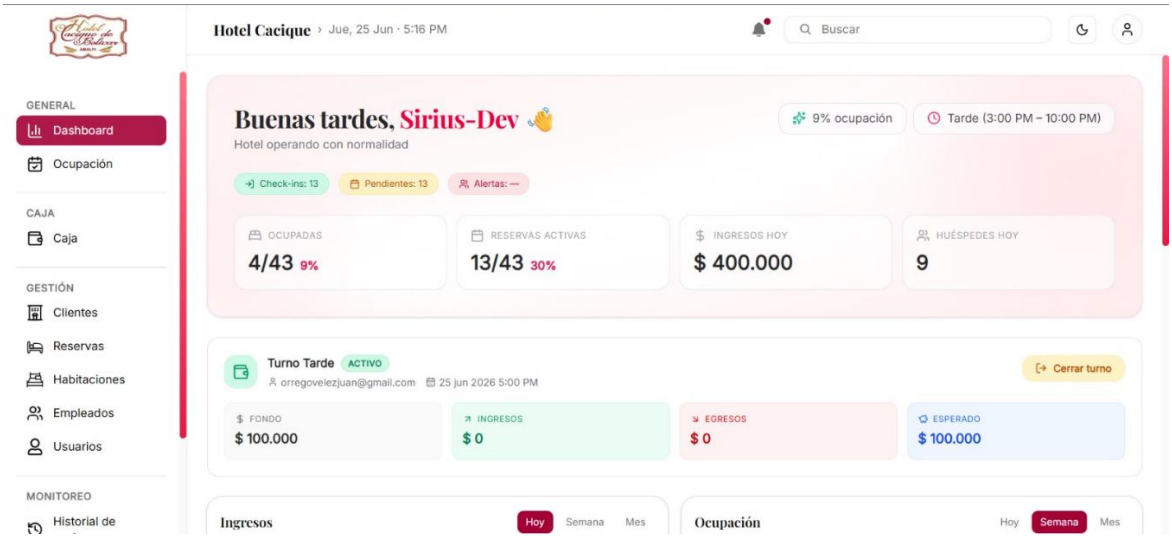



*Figura 15 Módulo de administrador*

***Nota.*** Módulo donde el administrador gestiona todos los datos del hotel.

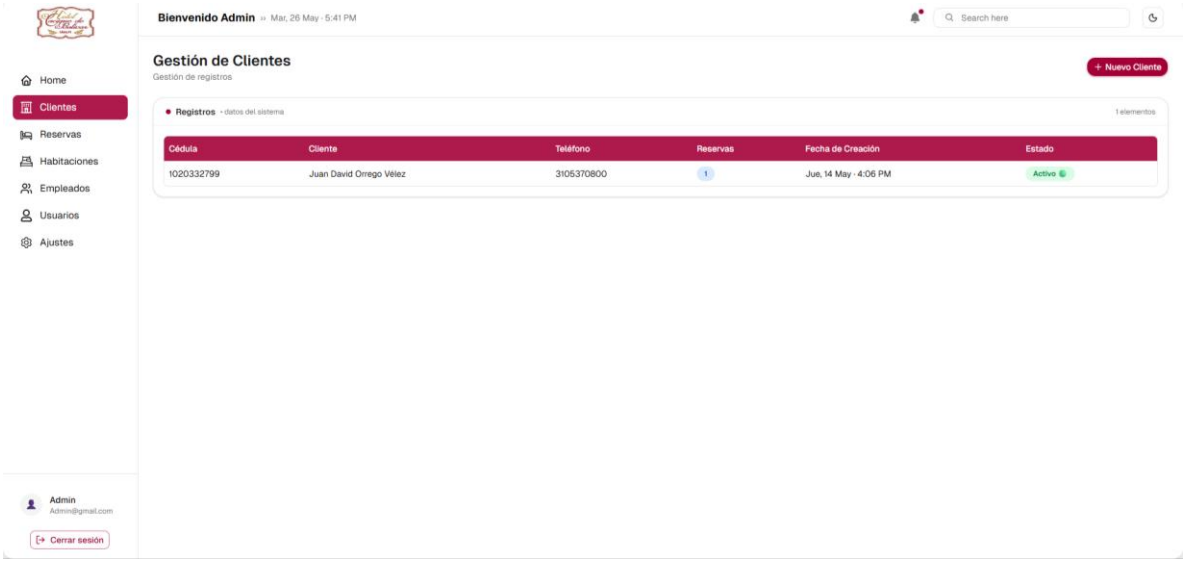



*Figura 16 Módulo de clientes*

***Nota.*** Pantalla utilizada por los clientes.

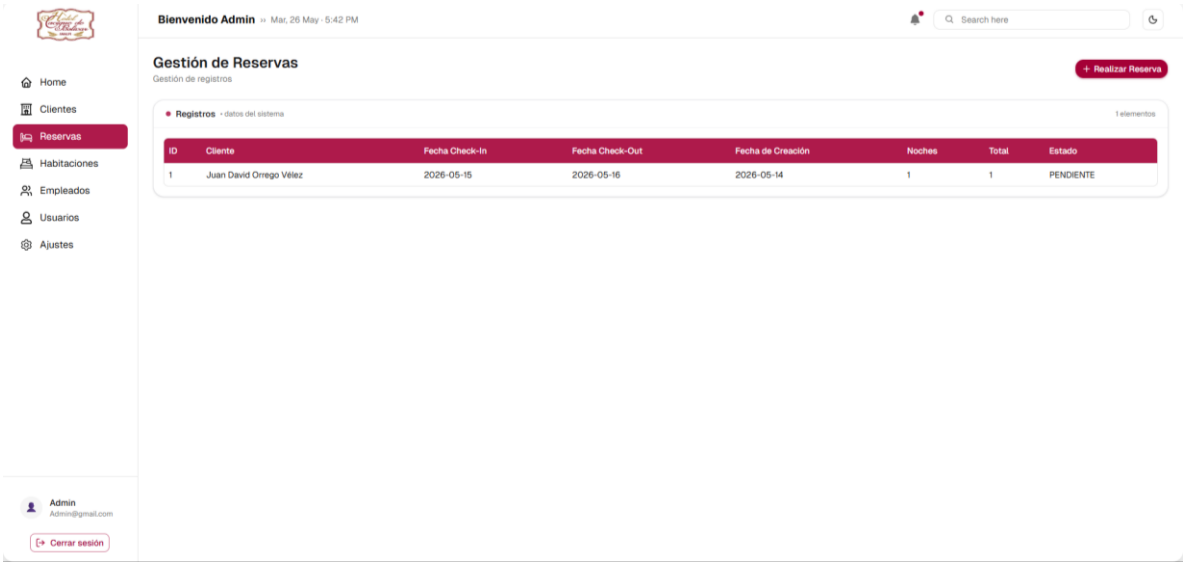



*Figura 17 Módulo de gestión de reservas*

***Nota.*** Pantalla utilizada para las gestiones de reserva que hacen los clientes.

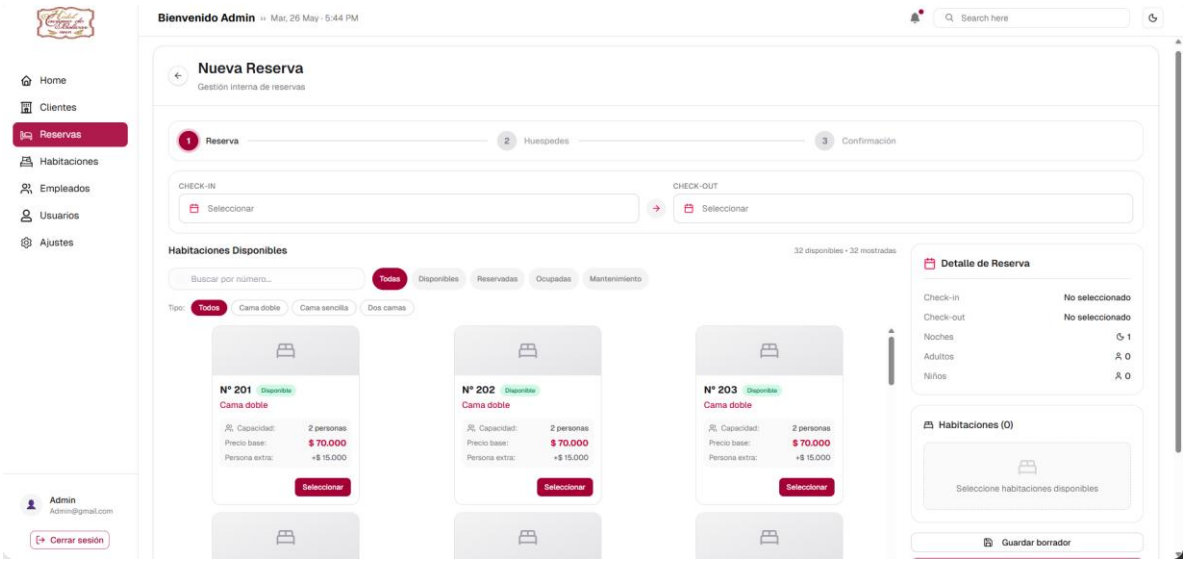



*Figura 18 Módulos de crear reservas*

**Nota.** En este módulo para crear una nueva reserva.

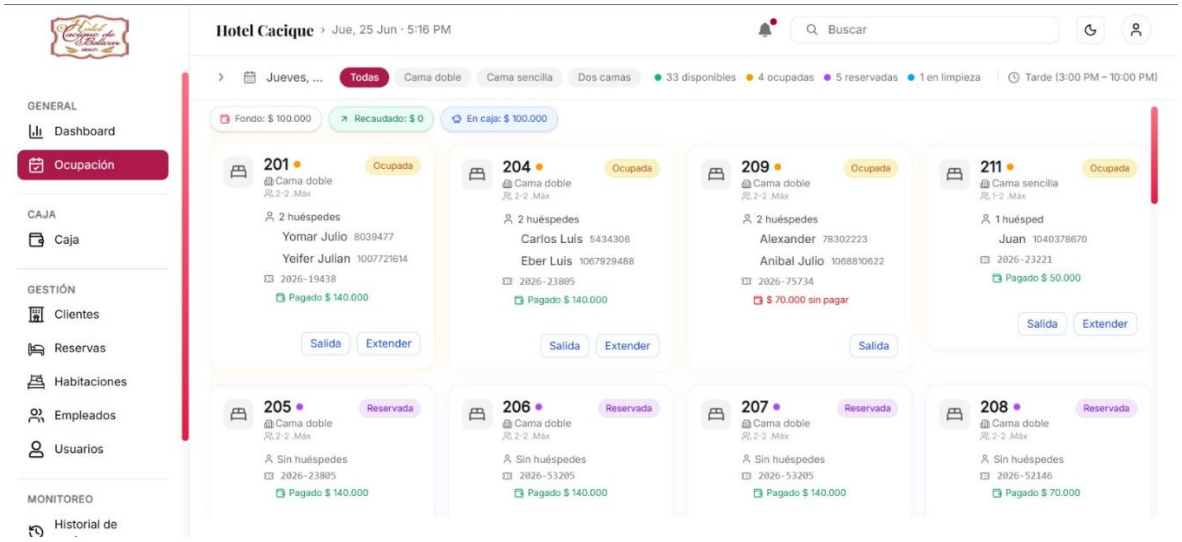



*Figura 19 Modulo de ocupación de las habitaciones*

***Nota.*** En este módulo es donde se puede ver las habitaciones ocupadas en el hotel.

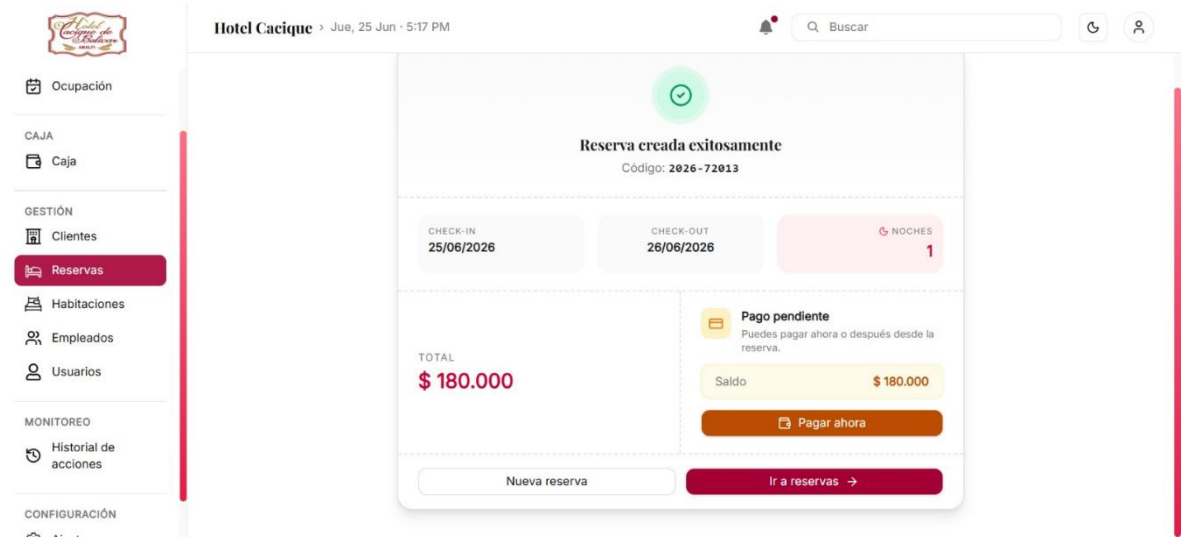



*Figura 20 Módulo de reserva exitosa*

***Nota***. Módulo de la reserva de la habitacion fue exitosa.

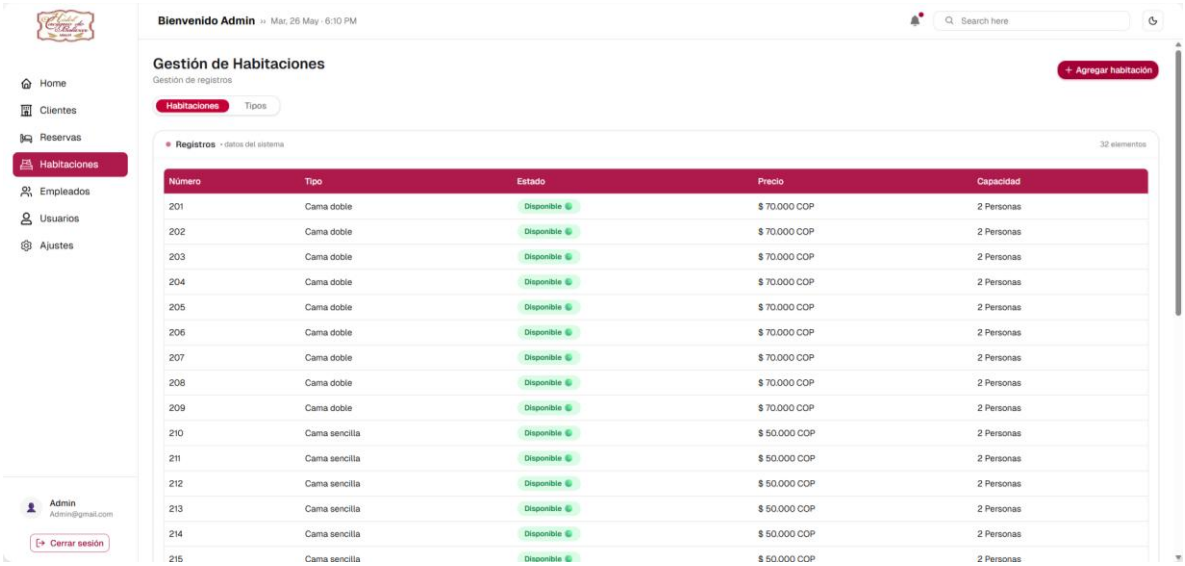



*Figura 21 Módulo de habitaciones*

***Nota.*** Pantalla utilizada para las gestiones de habitaciones que tiene el hotel.

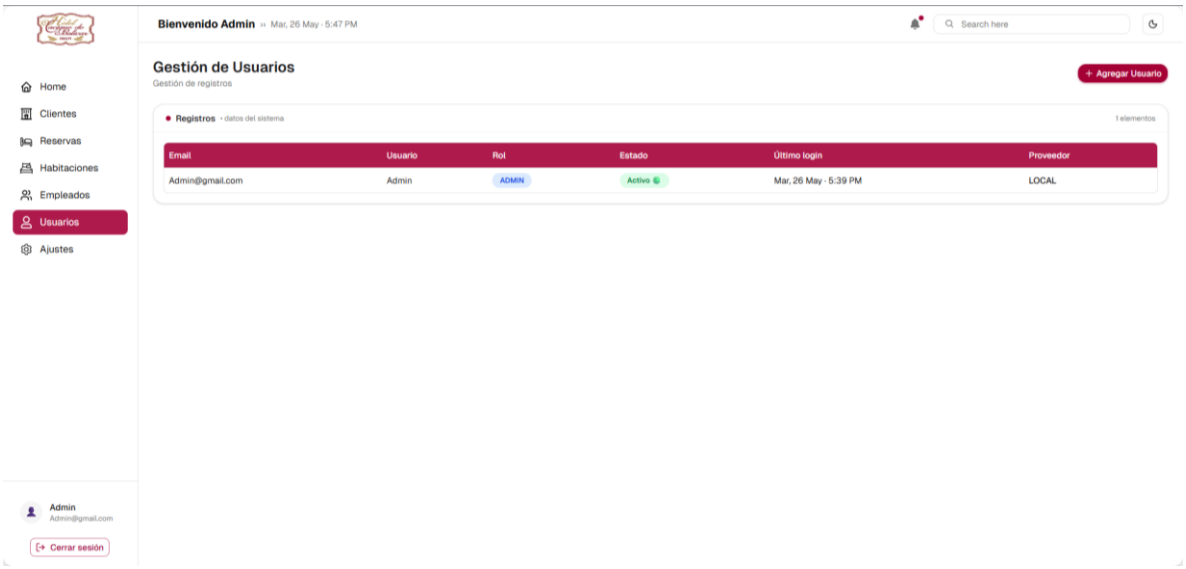



*Figura 22 Módulo de usuarios*

***Nota.*** Pantalla utilizada para las gestiones de usuarios que tiene el hotel en su momento.

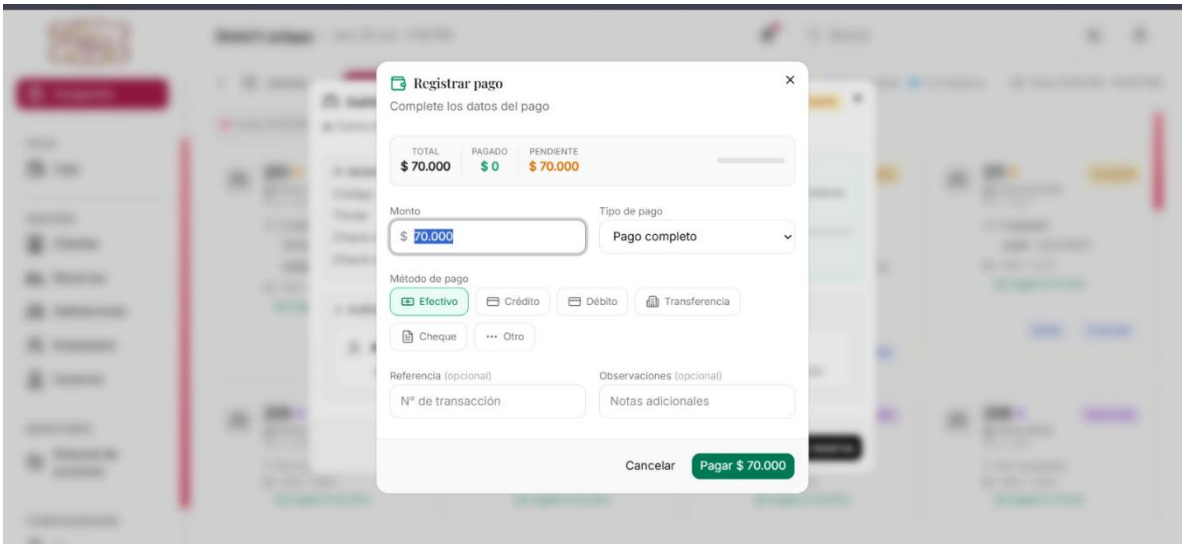



*Figura 23 Registro de pago*

***Nota.*** Módulo que registra los pagos de las reservas.

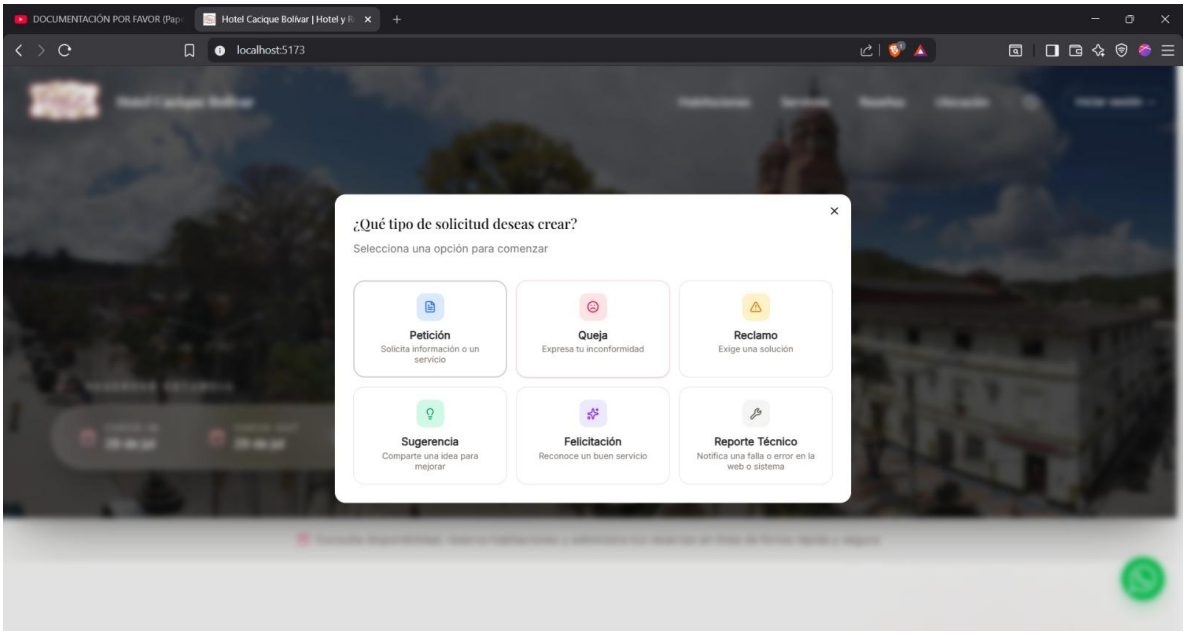



*Figura 24 Módulo de PQRS+*

***Nota.*** Pantalla utilizada para el registro y gestión de solicitudes PQRS+.

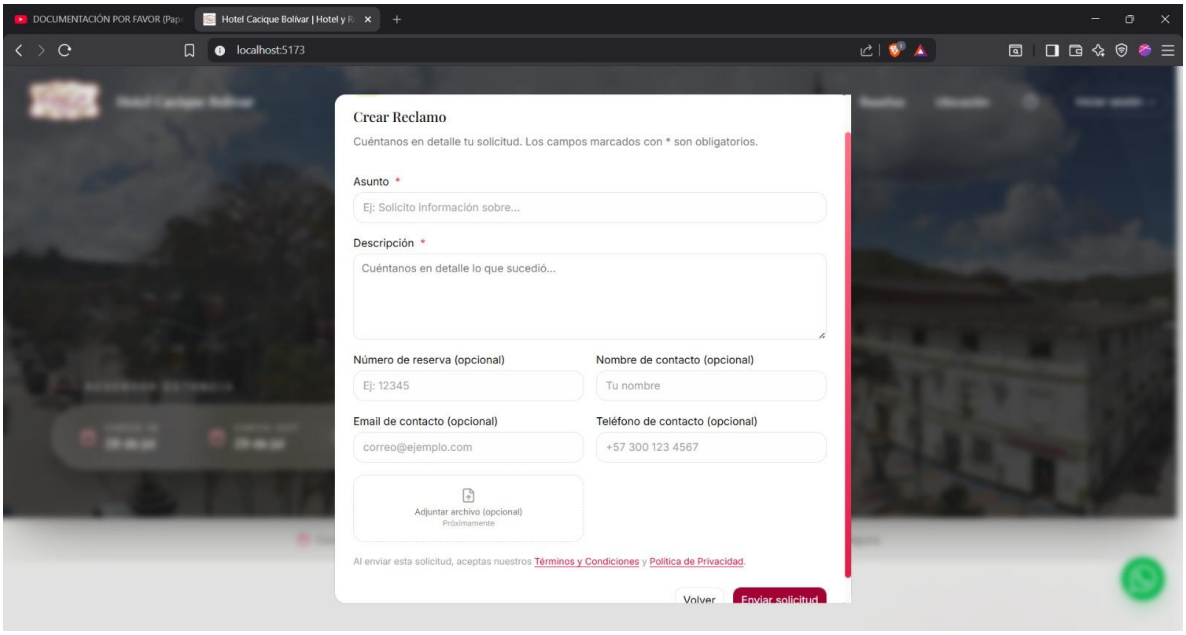



*Figura 25 Módulo de Crear PQRS+*

***Nota***. Pantalla utilizada para la creación de PQRS+.

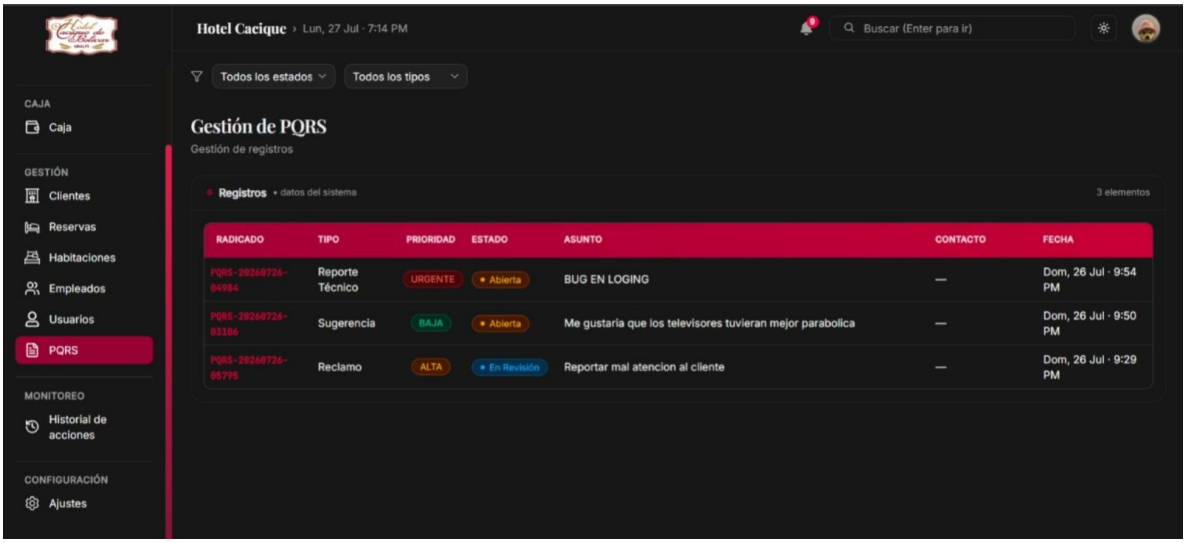



*Figura 26 Módulo de Gestión de PQRS+*

***Nota***. Pantalla utilizada para visualizar PQRS+ creados por los usuarios.

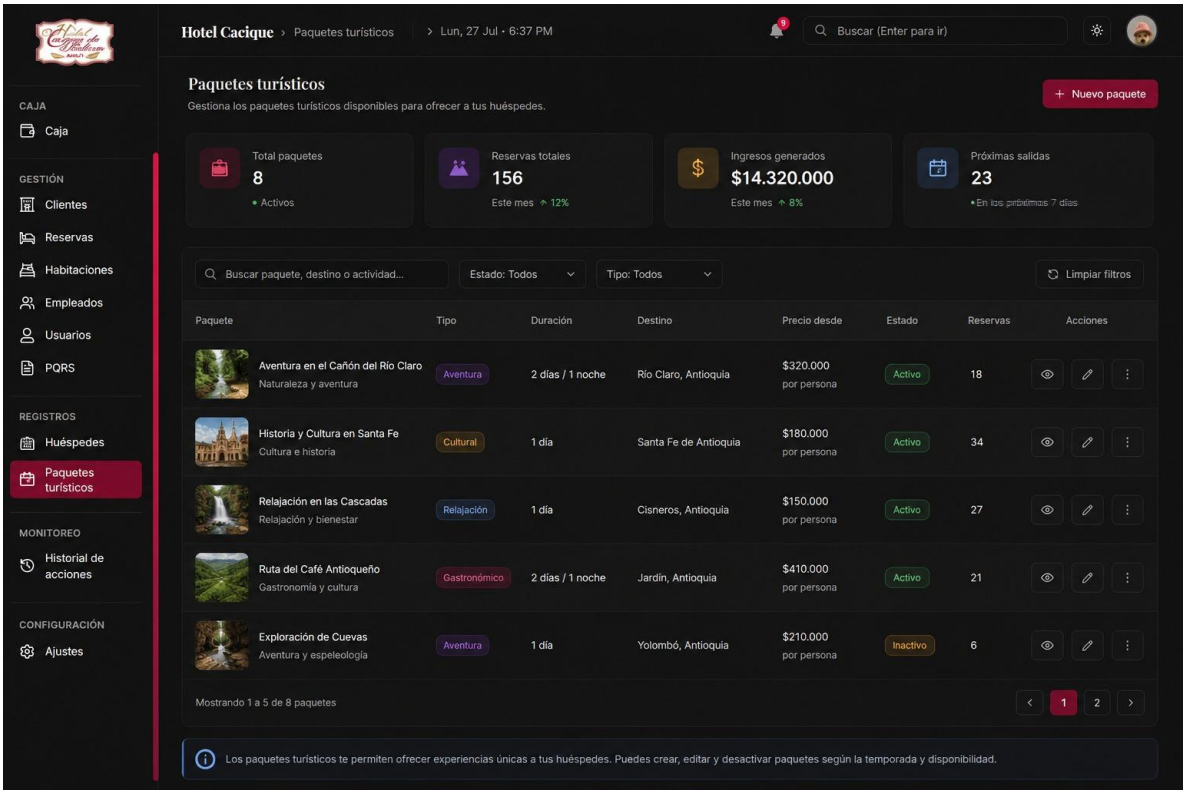



*Figura 27 Módulo de Paquetes Turístico*

***Nota***. Pantalla utilizada para la consulta y administración de paquetes turísticos.

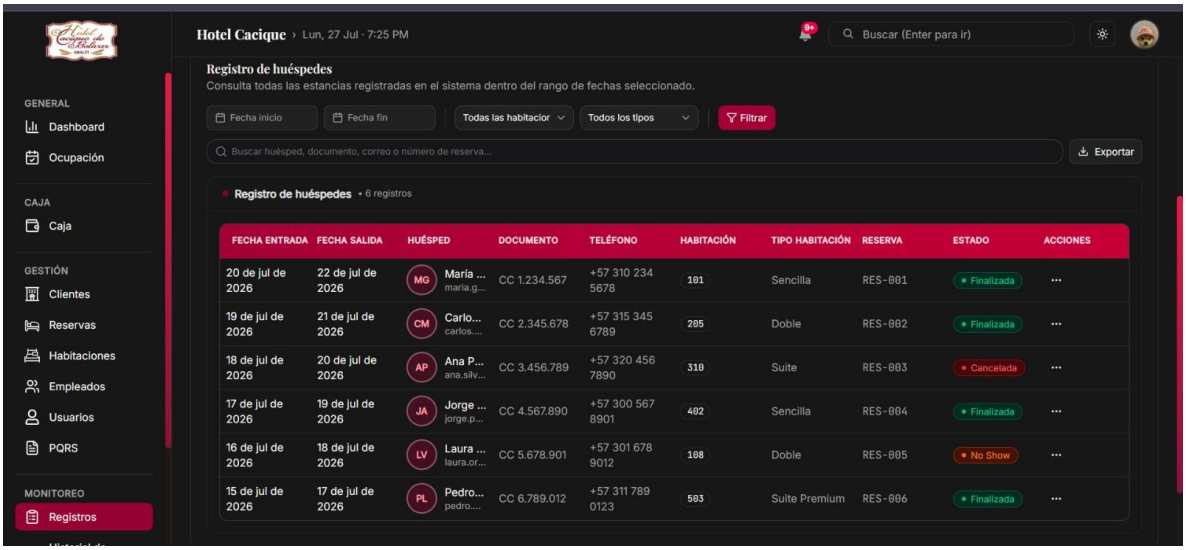



*Figura 28 Módulo de Informes*

***Nota***. Pantalla utilizada para la generación de informes del sistema.

**Navegación del sistema 7.2.4.** La navegación de un sistema se traduce en la organización y estructuración de las distintas secciones, módulos, opciones y funcionalidades que permiten al usuario ir automáticamente de un punto a otro de la aplicación para acceder a la información o acciones que necesita. Una navegación bien estructurada ayuda a interactuar con el sistema, disminuye la complejidad que percibe el usuario y favorece que las tareas se puedan realizar de una manera más eficiente.

La navegación es un factor que hace referencia a un aspecto relacionado con la usabilidad de una interfaz, dado que puede condicionar la capacidad para que los usuarios puedan buscar la información y llevar a cabo las actividades planteadas. La literatura relativa a la evaluación de sitios web apoya una fácil navegación como uno de los factores

que pueden incidir en la usabilidad de la interfaz y de la experiencia de los usuarios (Singla & Aggarwal, 2021). La forma en que se plantea, puede incidir sobre la carga cognitiva, la utilidad que se le da y la satisfacción que experimenta durante la interacción con el sistema (Zhang et al., 2021).

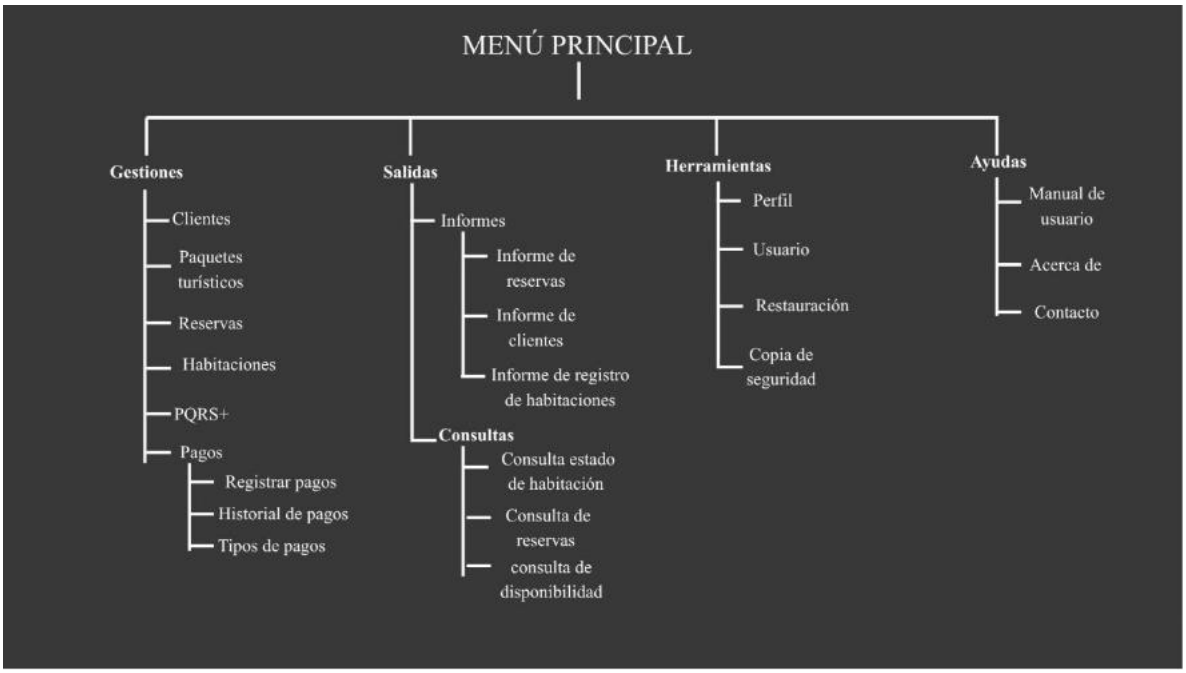



*Figura 29 Mapa de navegación*

***Nota***. Mapa de navegación del sistema hotelero

**Modelado UML 7.2.5.**

- **Tabla casos de usos**

  En esta tabla se muestra la lista de casos de uso, que son las descripciones de las interacciones entre los actores y el sistema para lograr un objetivo específico. Los

casos de uso son esenciales para capturar los requisitos funcionales, garantizar que todas las funcionalidades importantes estén cubiertas y proporcionar una base para diseñar las pruebas del sistema. Esto facilita el desarrollo de un sistema que satisfaga las necesidades y expectativas de los usuarios.

*Tabla 20 Casos de usos*

| Gestión | Funcionalidad | Cliente | Recepcionista | Gerente | Administrador |
|---|---|---|---|---|---|
| **Gestión de Clientes** | Registrar cliente | X | X | | X |
| | Consultar clientes | X | X | X | X |
| | Actualizar información del cliente | X | X | | X |
| | Inhabilitar cliente | | | | X |
| **Gestión de Reservas** | Registrar reserva | X | X | | X |
| | Consultar reservas | X | X | X | X |

| | | | | | |
|---|---|---|---|---|---|
| | Modificar reserva | X | X | | X |
| | Cancelar reserva | X | X | X | X |
| | Inhabilitar reserva | | X | | X |
| **Gestión de Tipos de Reserva** | Registrar tipo de reserva | | X | | X |
| | Consultar tipos de reserva | X | X | X | X |
| | Actualizar tipo de reserva | | X | | X |
| | Inhabilitar tipo de reserva | | X | | X |
| **Gestión de Habitaciones** | Registrar habitación | | X | X | X |

| | | | | | |
|---|---|---|---|---|---|
| | Consultar habitaciones | | X | X | X |
| | Actualizar habitación | | X | X | X |
| | Inhabilitar habitación | | X | X | X |
| **Gestión de Tipos de Habitación** | Registrar tipo de habitación | | X | X | X |
| | Consultar tipos de habitación | X | X | X | X |
| | Actualizar tipo de habitación | | X | X | X |
| | Inhabilitar tipo de habitación | | | X | X |
| **Gestión de Paquetes Turísticos** | Registrar paquete turístico | | | X | X |

| | | | | | |
|---|---|---|---|---|---|
| | Consultar paquetes turísticos | X | X | X | X |
| | Actualizar paquete turístico | | | X | X |
| | Inhabilitar paquete turístico | | | X | X |
| **Gestión de Tipos de Paquetes Turístico** | Registrar tipo de paquete | | | X | X |
| | Consultar tipos de paquete | X | X | X | X |
| | Actualizar tipo de paquete | | | X | X |

| | | | | | |
|---|---|---|---|---|---|
| | Inhabilitar tipo de paquete | | | X | X |
| **Gestión de Pagos** | Registrar pago | | X | X | X |
| | Consultar pagos | X | X | X | X |
| | Actualizar pago | | | | X |
| | Inhabilitar pago | | | | X |
| **Gestión de Tipos de Pago** | Registrar tipo de pago | | X | X | X |
| | Consultar tipos de pago | X | X | X | X |
| | Actualizar tipo de pago | | | | X |
| | Inhabilitar tipo de pago | | | | X |

| | | | | | |
|---|---|---|---|---|---|
| **Gestión de PQRS+** | Registrar PQRS+ | X | | | X |
| | Consultar PQRS+ | X | X | X | X |
| | Responder PQRS+ | | X | X | X |
| | Actualizar estado de PQRS+ | | X | X | X |
| **Gestión de Tipos de PQRS+** | Registrar tipo de PQRS+ | | X | X | X |
| | Consultar tipos de PQRS+ | X | X | X | X |
| | Actualizar tipo de PQRS+ | | X | X | X |
| | Inhabilitar tipo de PQRS+ | | | | X |

| | | | | | |
|---|---|---|---|---|---|
| **Gestión de Informes** | Generar informe de clientes | | | X | X |
| | Generar informe de reservas | | | X | X |
| | Generar informe de pagos | | | X | X |
| | Generar informe de PQRS+ | | X | X | X |
| **Gestión de Consultas** | Consultar huéspedes | | X | X | X |
| | Consultar disponibilidad | X | X | X | X |

| | | | | | |
|---|---|---|---|---|---|
| | Consultar estado de reservas | | X | X | X |
| | Consultar reclamos (PQRS+) | X | X | X | X |
| **Administración de Usuarios** | Registrar usuario | | | | X |
| | Consultar usuarios | | | | X |
| | Actualizar usuarios | | | | X |
| | Inhabilitar usuarios | | | | X |
| **Administración de Perfiles** | Registrar perfil | | | | X |
| | Consultar perfiles | | | | X |

| | | | | | |
|---|---|---|---|---|---|
| | Actualizar perfiles | | | | X |
| | Inhabilitar perfiles | | | | X |

***Nota***. Tabla de casos de uso del sistema hotelero.

- **Diagramas de casos de usos**

  Un diagrama de casos de uso es una representación gráfica que ilustra cómo interactúan los usuarios (llamados actores) con un sistema software y qué funcionalidades ofrece el sistema en respuesta a dichas interacciones. Los actores se muestran externos al sistema, mientras que los casos de uso representan las acciones o servicios que este proporciona. Este tipo de diagrama es una herramienta esencial en el análisis de requisitos, ya que ayuda a visualizar y comprender las necesidades y expectativas de los usuarios, así como las funcionalidades que el sistema debe implementar para satisfacer esas necesidades.

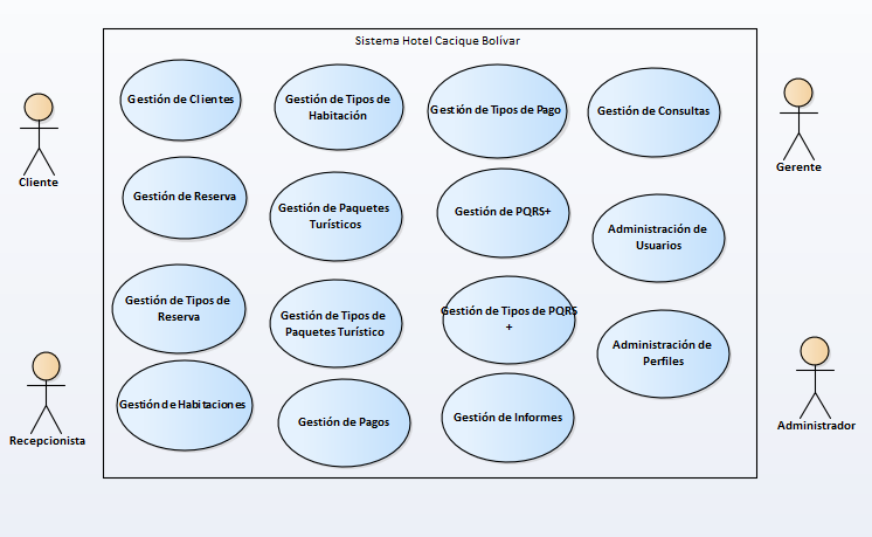



*Figura 30 Diagrama de casos de uso*

***Nota***. En la imagen se muestra las acciones que tiene los usuarios en gestión de clientes.

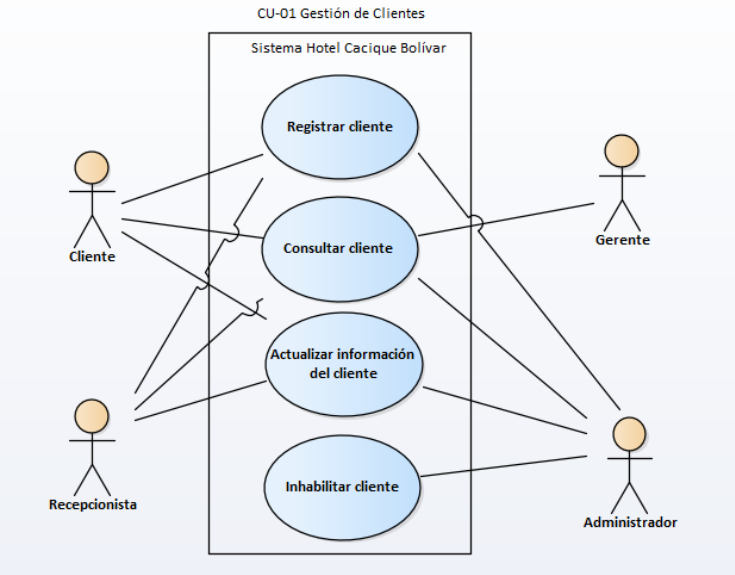



*Figura 31 Diagrama de Casos de Uso de Gestión de Clientes*

***Nota***. En la imagen se muestra las acciones que puede hacer cada usuario en gestión clientes.

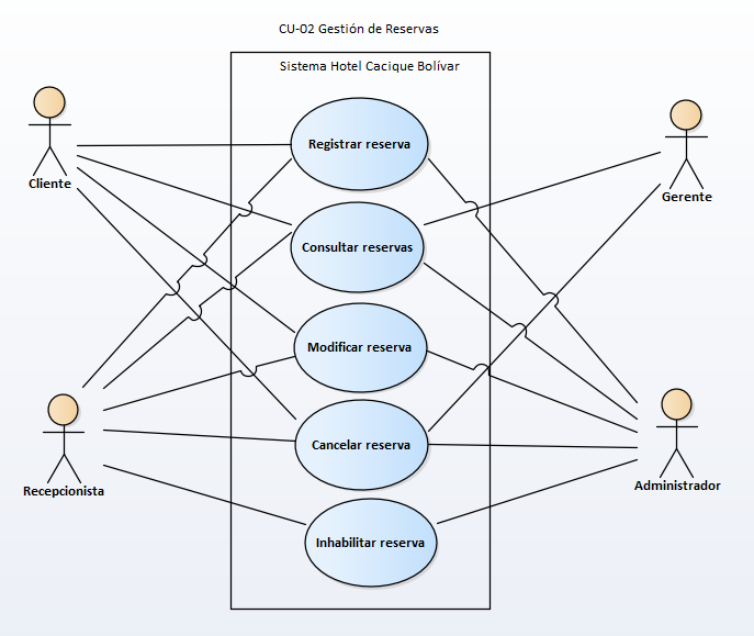



*Figura 32 Diagrama de Casos de Uso: Detalle del Módulo Gestión de Reserva*

***Nota***. En la imagen se muestra las acciones que puede hacer cada usuario en gestión reservas.

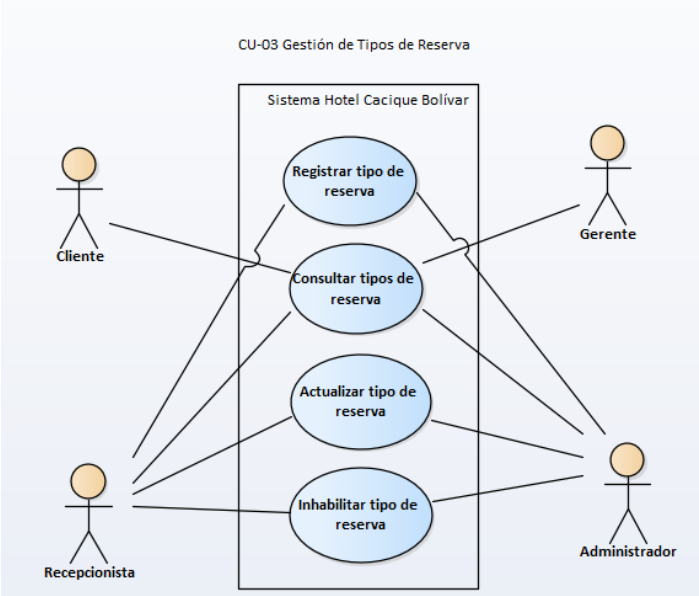



*Figura 33 Diagrama de Casos de Uso: Detalle del Módulo Gestión de Tipo de Reserva*

***Nota***. En la imagen se muestra las acciones que puede hacer cada usuario en gestión tipo de reserva.

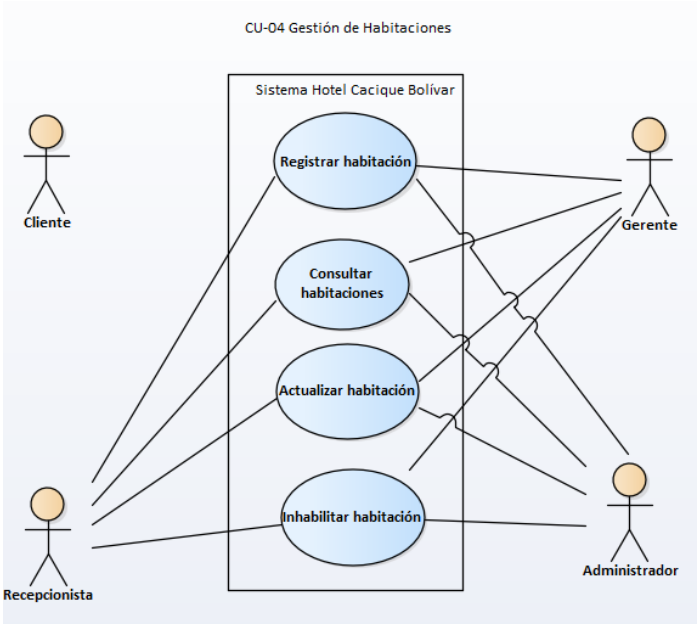



*Figura 34 Diagrama de Casos de Uso: Detalle del Módulo Gestión de Habitación*

***Nota***. En la imagen se muestra las acciones que puede hacer cada usuario en gestión de habitación.

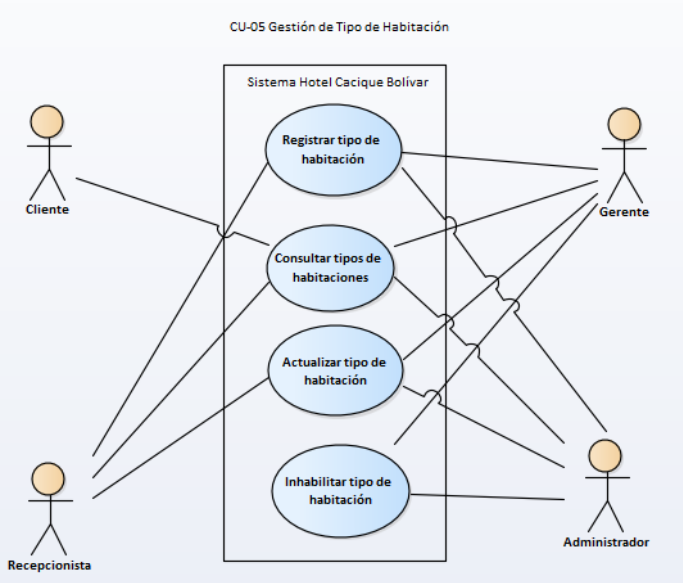



*Figura 35 Diagrama de Casos de Uso: Detalle del Módulo Gestión de tipo de Habitación*

***Nota***. En la imagen se muestra las acciones que puede hacer cada usuario en gestión tipo de habitación.

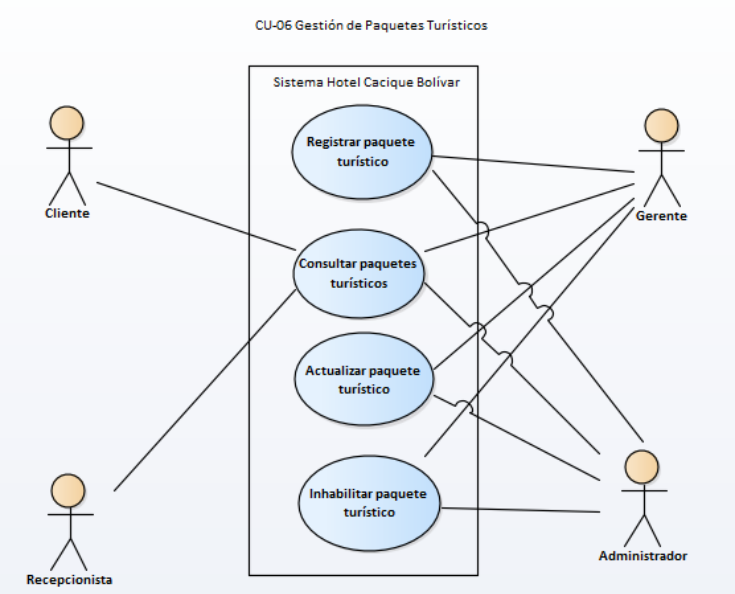



*Figura 36 Diagrama de Casos de Uso: Detalle del Módulo Gestión Paquetes Turísticos*

***Nota***. En la imagen se muestra las acciones que puede hacer cada usuario en gestión de paquetes turísticos.

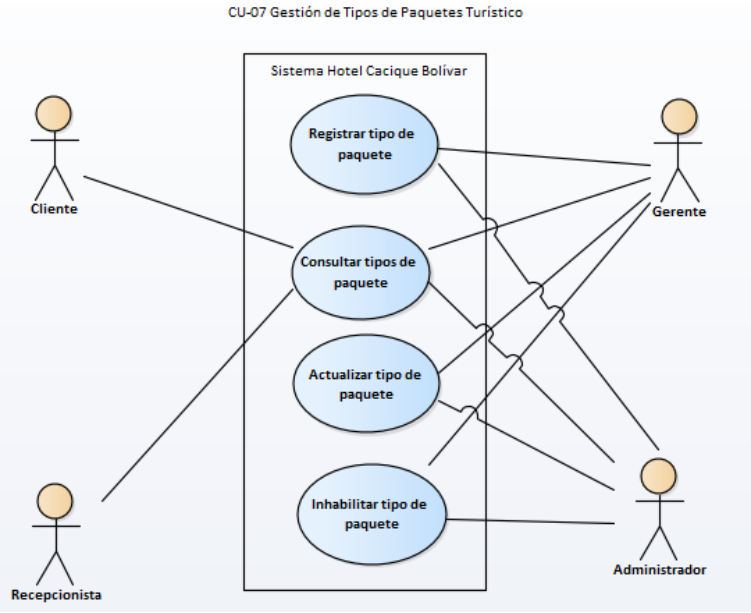



*Figura 37 Diagrama de Casos de Uso: Detalle del Módulo Gestión de Tipo de Paquetes Turístico*

***Nota.*** En la imagen se muestra las acciones que puede hacer cada usuario en gestión tipo de habitación.

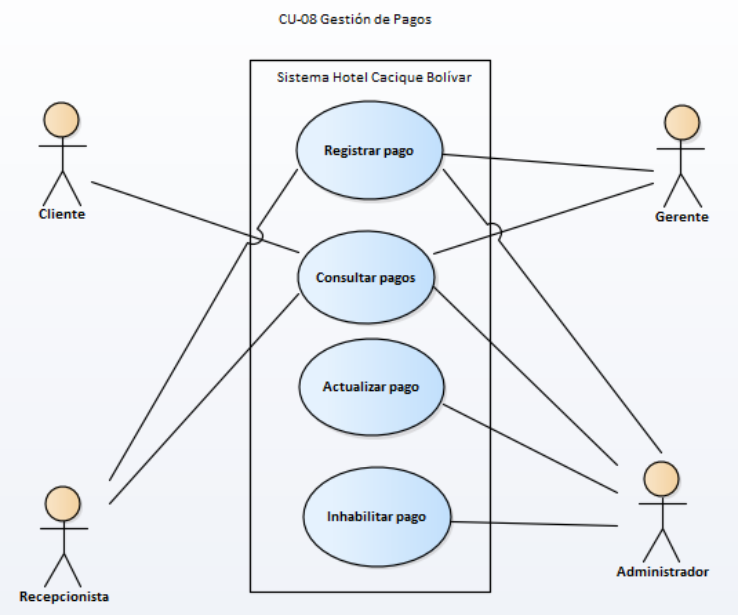



*Figura 38 Diagrama de Casos de Uso: Detalle del Módulo Gestión de pago*

***Nota***. En la imagen se muestra las acciones que puede hacer cada usuario en gestión de pago.

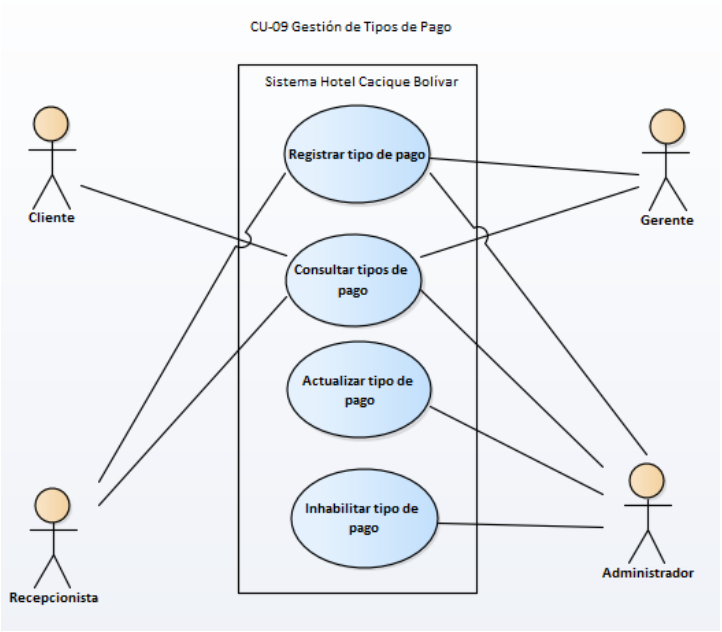



*Figura 39 Diagrama de Casos de Uso: Detalle del Módulo Gestión tipo de pago*

***Nota***. En la imagen se muestra las acciones que puede hacer cada usuario en gestión tipo de pago.

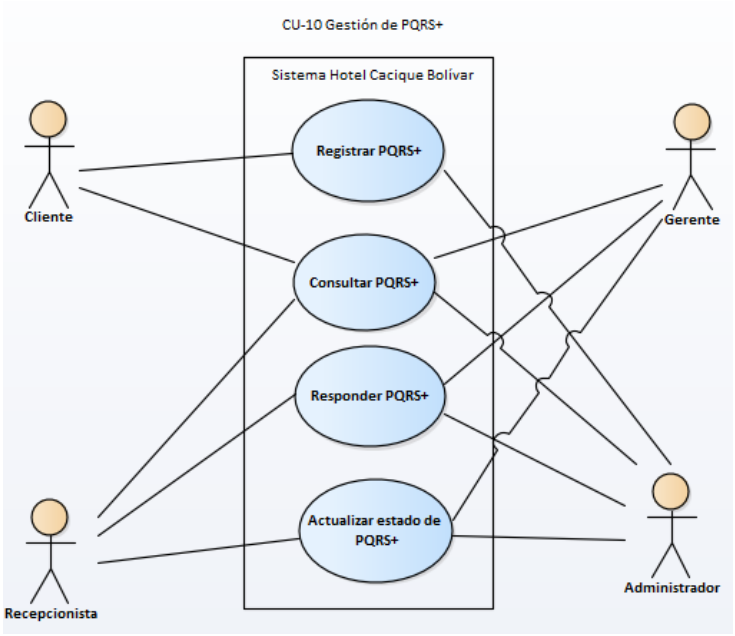



*Figura 40 Diagrama de Casos de Uso: Detalle del Módulo Gestión de PQRS+*

***Nota***. En la imagen se muestra las acciones que puede hacer cada usuario en gestión de PQRS+.

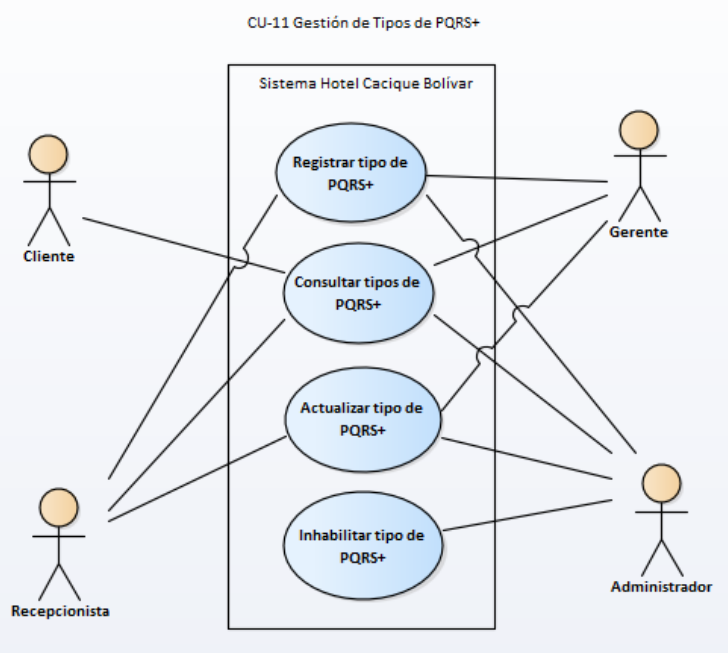



*Figura 41 Diagrama de Casos de Uso: Detalle del Módulo Gestión de tipo de PQRS+*

***Nota***. En la imagen se muestra las acciones que puede hacer cada usuario en gestión de tipo de PQRS+.

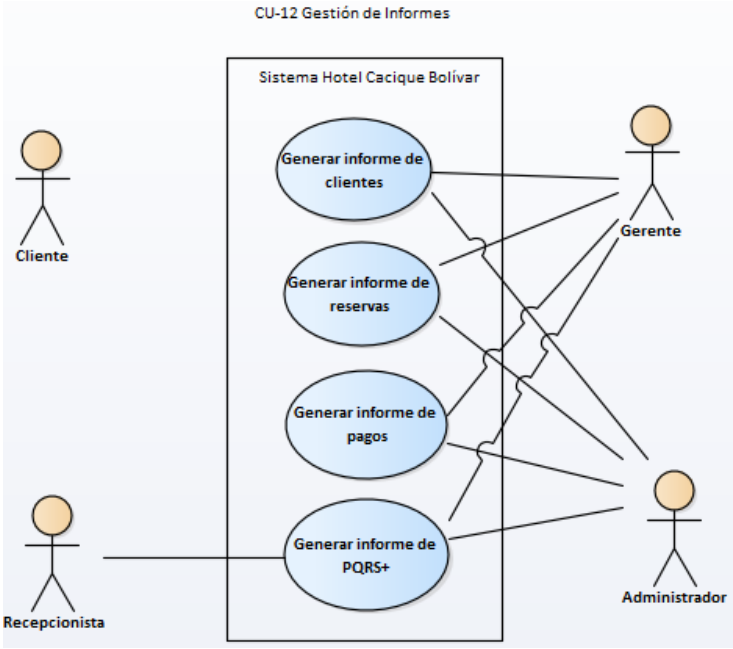



*Figura 42 Diagrama de Casos de Uso: Detalle de Gestión de informes*

***Nota***. En la imagen se muestra las acciones que puede hacer cada usuario en gestión de informes.

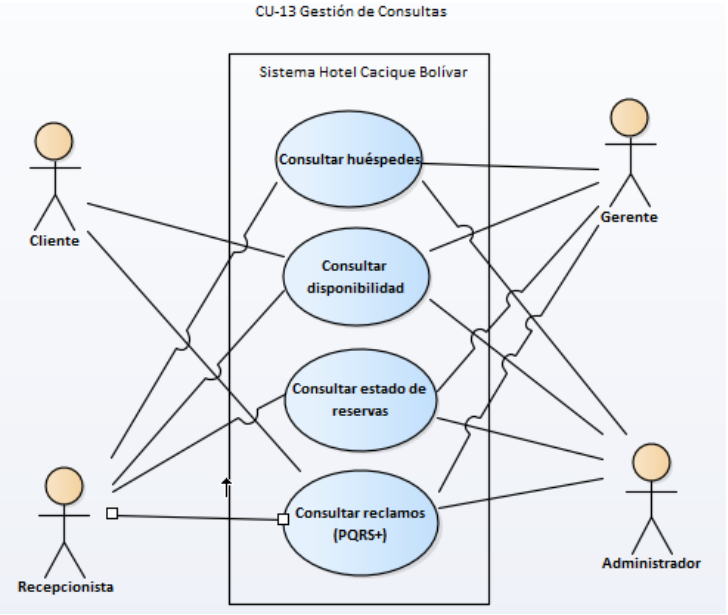



*Figura 43 Diagrama de Casos de Uso: Detalle del Módulo Gestión de consultas*

***Nota***. En la imagen se muestra las acciones que puede hacer cada usuario en gestión de consultas.

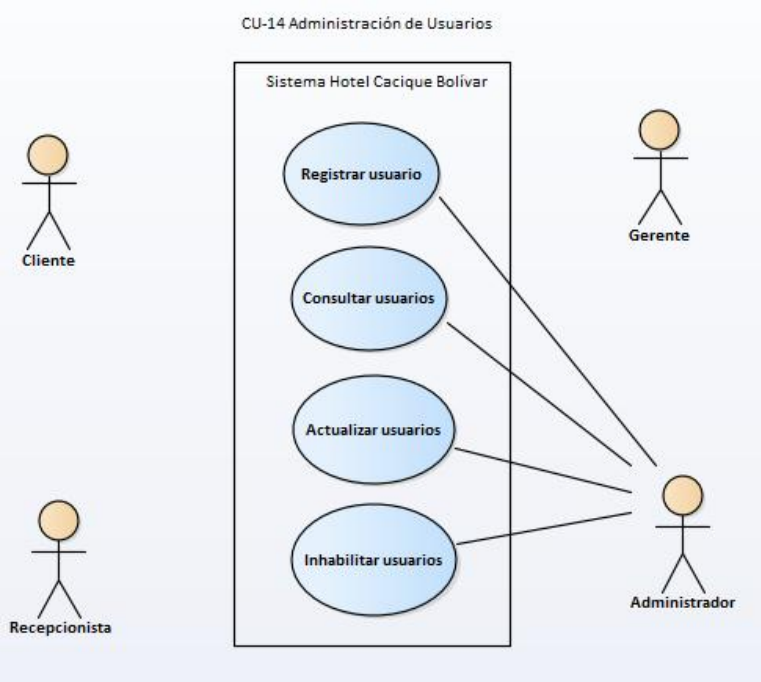



*Figura 44 Diagrama de Casos de Uso: Detalle del Módulo de Administración de usuarios*

***Nota***. En la imagen se muestra las acciones que puede hacer cada usuario de la administración de usuarios.

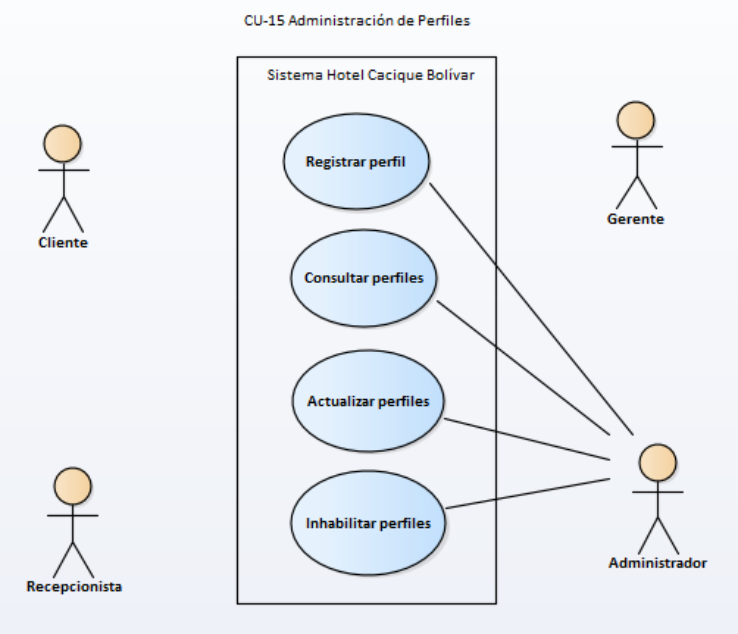



*Figura 45 Diagrama de Casos de Uso: Detalle del Módulo de Administración de perfiles*

***Nota***. En la imagen se muestra las acciones que puede hacer cada usuario de la administración de perfiles.

- **Escenarios de los casos de usos**

  En esta tabla se muestra la lista de escenarios de los casos de uso, los cuales son instancias específicas de interacciones entre los actores y el sistema para lograr un objetivo determinado. Cada escenario describe una situación particular en la que un actor utiliza una funcionalidad del sistema, detallando los pasos exactos que se siguen, las entradas proporcionadas, las acciones realizadas por el sistema y las salidas generadas. Los escenarios de los casos de uso son útiles para comprender cómo se comportará el sistema en situaciones concretas, identificar posibles

problemas o necesidades adicionales, y validar la corrección y la completitud de los casos de uso. Esto contribuye a garantizar que el sistema satisfaga de manera efectiva las necesidades de los usuarios y funcione correctamente en la práctica.

*Tabla 21 Escenario de Caso de Uso – Gestión de Clientes*

| **Nombre** | Gestión de Clientes | |
|---|---|---|
| **Caso de Uso** | CU-01 | |
| **Descripción** | Permite registrar, consultar, actualizar e inhabilitar la información de los clientes del Hotel Cacique Bolívar, garantizando un adecuado control de los datos personales necesarios para la gestión de reservas y demás servicios ofrecidos por el hotel. | |
| **Actor** | Cliente, Recepcionista, Gerente y Administrador (de acuerdo con los permisos asignados). | |
| **Precondiciones** | El usuario debe haber iniciado sesión en el sistema y contar con los permisos necesarios para acceder al módulo de Gestión de Clientes. | |
| **Flujo básico** | | |
| **Paso** | **Actor** | **Sistema** |
| 1 | El usuario accede al módulo **Gestión de Clientes**. | El sistema muestra las funcionalidades |

| | | |
|---|---|---|
| | | disponibles según el rol del usuario. |
| 2 | El usuario selecciona la operación que desea realizar (Registrar, Consultar, Actualizar o Inhabilitar cliente). | El sistema presenta el formulario correspondiente a la operación seleccionada. |
| 3 | El usuario ingresa o consulta la información requerida del cliente. | El sistema valida que la información ingresada cumpla con los requisitos establecidos. |
| 4 | El usuario confirma la operación seleccionada. | El sistema procesa la solicitud y verifica la existencia del cliente cuando sea necesario. |
| 5 | | El sistema registra, consulta, actualiza o inhabilita la información del cliente según la operación realizada. |

| | | |
|---|---|---|
| 6 | | El sistema muestra un mensaje indicando que la operación fue realizada correctamente. |
| **Flujo Alternativo** | | |
| **Paso** | **Actor** | **Sistema** |
| 1 | | El sistema detecta que la información ingresada es incorrecta o que el cliente no existe cuando corresponde. |
| 2 | | El sistema informa el motivo del error y solicita corregir la información. |
| 3 | El usuario corrige la información o realiza una nueva búsqueda. | El sistema valida nuevamente los datos ingresados. |
| 4 | | Si la información es válida, el sistema ejecuta |

| | | |
|---|---|---|
| | | correctamente la operación solicitada. |
| **Postcondición** | La información del cliente queda registrada, consultada, actualizada o inhabilitada correctamente en el sistema, según la operación realizada. | |
| **Requisito Funcional** | RF-001 al RF-006 | |

***Nota*.** Esta tabla describe el escenario del caso de uso Gestión de Clientes, integrando las funcionalidades de registrar, consultar, actualizar e inhabilitar clientes, de acuerdo con los permisos definidos para cada tipo de usuario del sistema.

*Tabla 22 Escenario del Caso de Uso – Gestión de Reservas*

| **Nombre** | Gestión de Reservas | |
|---|---|---|
| **Caso de Uso** | CU-02 | |
| **Descripción** | Permite registrar, consultar, actualizar, cancelar e inhabilitar las reservas de habitaciones del Hotel Cacique Bolívar, garantizando el control de la disponibilidad, la información del huésped y el estado de cada reserva. | |
| **Actor** | Cliente, Recepcionista y Administrador (de acuerdo con los permisos asignados). | |
| **Precondiciones** | El usuario debe haber iniciado sesión en el sistema. Debe existir al menos un cliente registrado y una habitación disponible para realizar una reserva. | |
| **Flujo básico** | | |
| **Paso** | **Actor** | **Sistema** |
| **1** | El usuario accede al módulo **Gestión de Reservas**. | El sistema muestra las funcionalidades |

| | | |
|---|---|---|
| | | disponibles según el rol del usuario. |
| **2** | El usuario selecciona la operación que desea realizar (Registrar, Consultar, Actualizar, Cancelar o Inhabilitar reserva). | El sistema presenta el formulario correspondiente a la operación seleccionada. |
| **3** | El usuario ingresa o consulta la información de la reserva (cliente, habitación, fechas de ingreso y salida, tipo de reserva y demás datos requeridos). | El sistema valida que la información sea correcta y verifica la disponibilidad de la habitación seleccionada. |
| **4** | El usuario confirma la operación. | El sistema procesa la solicitud y verifica las reglas de negocio correspondientes. |
| **5** | | El sistema registra, consulta, actualiza, cancela o inhabilita la reserva según la operación seleccionada. |

| | | |
|---|---|---|
| **6** | | El sistema actualiza la disponibilidad de la habitación cuando corresponda y muestra un mensaje indicando que la operación fue realizada correctamente. |
| **Flujo Alternativo** | | |
| **Paso** | **Actor** | **Sistema** |
| **1** | | El sistema detecta información incompleta, fechas inválidas o que la habitación seleccionada no se encuentra disponible. |
| **2** | | El sistema informa el motivo del error y solicita corregir la información. |

| | | |
|---|---|---|
| **3** | El usuario modifica la información o selecciona otra habitación disponible. | El sistema valida nuevamente los datos ingresados. |
| **4** | | Si la información es válida, el sistema ejecuta correctamente la operación seleccionada y actualiza el estado de la reserva. |
| **Postcondición** | La reserva queda registrada, consultada, actualizada, cancelada o inhabilitada correctamente en el sistema, manteniendo actualizada la disponibilidad de las habitaciones y la información asociada al cliente. | |
| **Requisito Funcional** | RF-007 al RF-012 | |

***Nota*.** Esta tabla describe el escenario del caso de uso Gestión de Reservas, integrando las funcionalidades de registrar, consultar, actualizar, cancelar e inhabilitar reservas, garantizando el control de la disponibilidad de habitaciones y la correcta administración de las reservas del Hotel Cacique Bolívar.

*Tabla 23 Escenario del Caso de Uso – Gestión de Tipos de Reserva*

| **Nombre** | Gestión de Tipos de Reserva | |
|---|---|---|
| **Caso de Uso** | CU-03 | |
| **Descripción** | Permite registrar, consultar, actualizar e inhabilitar los diferentes tipos de reserva del Hotel Cacique Bolívar, garantizando una adecuada clasificación de las modalidades de reserva disponibles para los clientes. | |
| **Actor** | Recepcionista y Administrador (de acuerdo con los permisos asignados). | |
| **Precondiciones** | El usuario debe haber iniciado sesión en el sistema y contar con permisos para acceder al módulo de Gestión de Tipos de Reserva. | |
| **Flujo básico** | | |
| **Paso** | **Actor** | **Sistema** |
| **1** | El usuario accede al módulo **Gestión de Tipos de Reserva**. | El sistema muestra las funcionalidades disponibles según el rol del usuario. |

| | | |
|---|---|---|
| **2** | El usuario selecciona la operación que desea realizar (Registrar, Consultar, Actualizar o Inhabilitar tipo de reserva). | El sistema presenta el formulario correspondiente a la operación seleccionada. |
| **3** | El usuario registra, consulta o modifica la información del tipo de reserva (nombre, descripción y demás características definidas por el hotel). | El sistema valida que la información ingresada sea correcta y verifica que no existan registros duplicados. |
| **4** | El usuario confirma la operación. | El sistema procesa la solicitud y verifica las reglas de negocio correspondientes. |
| **5** | | El sistema registra, consulta, actualiza o inhabilita el tipo de reserva según la operación seleccionada. |
| **6** | | El sistema muestra un mensaje indicando que la |

| | | |
|---|---|---|
| | | operación fue realizada correctamente. |
| **Flujo Alternativo** | | |
| **Paso** | **Actor** | **Sistema** |
| **1** | | El sistema detecta información incompleta, datos inválidos o un tipo de reserva previamente registrado. |
| **2** | | El sistema informa el motivo del error y solicita corregir la información. |
| **3** | El usuario corrige la información ingresada. | El sistema valida nuevamente los datos registrados. |
| **4** | | Si la información es válida, el sistema ejecuta correctamente la operación seleccionada. |

| | |
|---|---|
| **Postcondición** | El tipo de reserva queda registrado, consultado, actualizado o inhabilitado correctamente en el sistema, permitiendo una adecuada administración de las modalidades de reserva ofrecidas por el Hotel Cacique Bolívar. |
| **Requisito Funcional** | RF-013 al RF-018 |

***Nota.*** Esta tabla describe el escenario del caso de uso **Gestión de Tipos de Reserva**, integrando las funcionalidades de registrar, consultar, actualizar e inhabilitar los tipos de reserva, garantizando una adecuada clasificación y administración de las modalidades de reserva disponibles en el Hotel Cacique Bolívar.

*Tabla 24 Escenario del Caso de Uso – Gestión de Habitaciones*

| **Nombre** | Gestión de Habitaciones |
|---|---|
| **Caso de Uso** | CU-04 |
| **Descripción** | Permite registrar, consultar, actualizar e inhabilitar las habitaciones del Hotel Cacique Bolívar, garantizando una adecuada administración de su disponibilidad, características y estado para |

| | |
|---|---|
| | la correcta prestación del servicio de hospedaje. |
| **Actor** | Gerente y Administrador (de acuerdo con los permisos asignados). |
| **Precondiciones** | El usuario debe haber iniciado sesión en el sistema y contar con permisos para acceder al módulo de Gestión de Habitaciones. |

| **Flujo básico** | | |
|---|---|---|
| **Paso** | **Actor** | **Sistema** |
| **1** | El usuario accede al módulo **Gestión de Habitaciones**. | El sistema muestra las funcionalidades disponibles según el rol del usuario. |
| **2** | El usuario selecciona la operación que desea realizar (Registrar, Consultar, Actualizar o Inhabilitar habitación). | El sistema presenta el formulario correspondiente a la operación seleccionada. |
| **3** | El usuario registra, consulta o modifica la información de la habitación (número, | El sistema valida la información ingresada y verifica que el número de |

| | | |
|---|---|---|
| | tipo de habitación, capacidad, estado, precio y demás características). | habitación no se encuentre registrado previamente. |
| **4** | El usuario confirma la operación. | El sistema procesa la solicitud y verifica las reglas de negocio correspondientes. |
| **5** | | El sistema registra, consulta, actualiza o inhabilita la habitación según la operación seleccionada. |
| **6** | | El sistema muestra un mensaje indicando que la operación fue realizada correctamente. |
| **Flujo Alternativo** | | |
| **Paso** | **Actor** | **Sistema** |
| **1** | | El sistema detecta información incompleta, |

| | | |
|---|---|---|
| | | datos inválidos o un número de habitación previamente registrado. |
| **2** | | El sistema informa el motivo del error y solicita corregir la información. |
| **3** | El usuario corrige la información ingresada. | El sistema valida nuevamente los datos registrados. |
| **4** | | Si la información es válida, el sistema ejecuta correctamente la operación seleccionada. |
| **Postcondición** | La habitación queda registrada, consultada, actualizada o inhabilitada correctamente en el sistema, permitiendo una adecuada administración de la disponibilidad y del inventario de habitaciones del Hotel Cacique Bolívar. | |
| **Requisito Funcional** | RF-043 al RF-048 | |

***Nota*.** Esta tabla describe el escenario del caso de uso **Gestión de Habitaciones**, integrando las funcionalidades de registrar, consultar, actualizar e inhabilitar habitaciones, garantizando el control del inventario, la disponibilidad y las características de las habitaciones del Hotel Cacique Bolívar.

*Tabla 25 Escenario del Caso de Uso – Gestión de Tipos de Habitación*

| **Nombre** | Gestión de Tipos de Habitación | |
|---|---|---|
| **Caso de Uso** | CU-05 | |
| **Descripción** | Permite registrar, consultar, actualizar e inhabilitar los diferentes tipos de habitación del Hotel Cacique Bolívar, definiendo sus características para facilitar la administración y clasificación de las habitaciones disponibles. | |
| **Actor** | Gerente y Administrador (de acuerdo con los permisos asignados). | |
| **Precondiciones** | El usuario debe haber iniciado sesión en el sistema y contar con permisos para acceder al módulo Gestión de Tipos de Habitación. | |
| **Flujo básico** | | |
| **Paso** | **Actor** | **Sistema** |

| | | |
|---|---|---|
| 1 | El usuario accede al módulo Gestión de Tipos de Habitación. | El sistema muestra las funcionalidades disponibles según el rol del usuario. |
| 2 | El usuario selecciona la operación que desea realizar (Registrar, Consultar, Actualizar o Inhabilitar tipo de habitación). | El sistema presenta el formulario correspondiente. |
| 3 | El usuario registra, consulta o modifica la información del tipo de habitación (nombre, descripción, capacidad y características). | El sistema valida que la información ingresada sea correcta y verifica que no existan registros duplicados. |
| 4 | El usuario confirma la operación. | El sistema procesa la solicitud y verifica las reglas de negocio. |
| 5 | | El sistema registra, consulta, actualiza o inhabilita el tipo de |

| | | |
|---|---|---|
| | | habitación según la operación seleccionada. |
| 6 | | El sistema muestra un mensaje indicando que la operación fue realizada correctamente. |
| **Flujo Alternativo** | | |
| **Paso** | **Actor** | **Sistema** |
| 1 | | El sistema detecta información incompleta, datos inválidos o un tipo de habitación previamente registrado. |
| 2 | | El sistema informa el motivo del error y solicita corregir la información. |
| 3 | El usuario corrige la información. | El sistema valida nuevamente los datos. |

| | | |
|---|---|---|
| 4 | | Si la información es válida, el sistema ejecuta correctamente la operación seleccionada. |
| **Postcondición** | El tipo de habitación queda registrado, consultado, actualizado o inhabilitado correctamente en el sistema. | |
| **Requisito Funcional** | RF-049 al RF-054 | |

***Nota.*** Esta tabla describe el escenario del caso de uso Gestión de Tipos de Habitación, integrando las funcionalidades de registrar, consultar, actualizar e inhabilitar los tipos de habitación, garantizando una adecuada clasificación de las habitaciones del Hotel Cacique Bolívar.

*Tabla 26 Escenario del Caso de Uso – Gestión de Paquetes Turísticos*

| **Nombre** | Gestión de Paquetes Turísticos |
|---|---|
| **Caso de Uso** | CU-06 |
| **Descripción** | Permite registrar, consultar, actualizar e inhabilitar los paquetes turísticos ofrecidos por el Hotel Cacique Bolívar, facilitando la |

| | |
|---|---|
| | administración de los servicios complementarios disponibles para los clientes. |
| **Actor** | Gerente y Administrador (de acuerdo con los permisos asignados). |
| **Precondiciones** | El usuario debe haber iniciado sesión en el sistema y contar con permisos para acceder al módulo Gestión de Paquetes Turísticos. |

| **Flujo básico** | | |
|---|---|---|
| **Paso** | **Actor** | **Sistema** |
| 1 | El usuario accede al módulo Gestión de Paquetes Turísticos. | El sistema muestra las funcionalidades disponibles según el rol del usuario. |
| 2 | El usuario selecciona la operación que desea realizar (Registrar, Consultar, Actualizar o Inhabilitar paquete turístico). | El sistema presenta el formulario correspondiente. |

| | | |
|---|---|---|
| 3 | El usuario registra, consulta o modifica la información del paquete turístico (nombre, destino, costo, duración, descripción y demás características). | El sistema valida la información y verifica que el paquete no exista previamente. |
| 4 | El usuario confirma la operación. | El sistema procesa la solicitud y verifica las reglas de negocio. |
| 5 | | El sistema registra, consulta, actualiza o inhabilita el paquete turístico. |
| 6 | | El sistema muestra un mensaje indicando que la operación fue realizada correctamente. |
| **Flujo Alternativo** | | |
| **Paso** | **Actor** | **Sistema** |
| 1 | | El sistema detecta información incompleta, datos inválidos o un |

| | | |
|---|---|---|
| | | paquete previamente registrado. |
| 2 | | El sistema informa el motivo del error y solicita corregir la información. |
| 3 | El usuario corrige la información. | El sistema valida nuevamente los datos ingresados. |
| 4 | | Si la información es válida, el sistema ejecuta correctamente la operación seleccionada. |
| **Postcondición** | El paquete turístico queda registrado, consultado, actualizado o inhabilitado correctamente en el sistema. | |
| **Requisito Funcional** | RF-019 al RF-024 | |

***Nota.*** Esta tabla describe el escenario del caso de uso Gestión de Paquetes Turísticos, integrando las funcionalidades de registrar, consultar, actualizar e inhabilitar los paquetes turísticos ofrecidos por el Hotel Cacique Bolívar.

*Tabla 27 Escenario del Caso de Uso – Gestión de Tipos de Paquetes Turísticos*

| **Nombre** | Gestión de Tipos de Paquetes Turísticos | |
|---|---|---|
| **Caso de Uso** | CU-07 | |
| **Descripción** | Permite registrar, consultar, actualizar e inhabilitar los diferentes tipos de paquetes turísticos ofrecidos por el Hotel Cacique Bolívar, facilitando la clasificación de los servicios turísticos disponibles. | |
| **Actor** | Gerente y Administrador (de acuerdo con los permisos asignados). | |
| **Precondiciones** | El usuario debe haber iniciado sesión en el sistema y contar con permisos para acceder al módulo Gestión de Tipos de Paquetes Turísticos. | |
| **Flujo básico** | | |
| **Paso** | **Actor** | **Sistema** |
| 1 | El usuario accede al módulo Gestión de Tipos de Paquetes Turísticos. | El sistema muestra las funcionalidades |

| | | |
|---|---|---|
| | | disponibles según el rol del usuario. |
| 2 | El usuario selecciona la operación que desea realizar (Registrar, Consultar, Actualizar o Inhabilitar tipo de paquete). | El sistema presenta el formulario correspondiente. |
| 3 | El usuario registra, consulta o modifica la información del tipo de paquete (nombre, descripción y características). | El sistema valida la información y verifica que el tipo de paquete no exista previamente. |
| 4 | El usuario confirma la operación. | El sistema procesa la solicitud y verifica las reglas de negocio. |
| 5 | | El sistema registra, consulta, actualiza o inhabilita el tipo de paquete turístico. |
| 6 | | El sistema muestra un mensaje indicando que la operación fue realizada correctamente. |

| Flujo Alternativo | | |
|---|---|---|
| **Paso** | **Actor** | **Sistema** |
| 1 | | El sistema detecta información incompleta, datos inválidos o un tipo de paquete previamente registrado. |
| 2 | | El sistema informa el motivo del error y solicita corregir la información. |
| 3 | El usuario corrige la información. | El sistema valida nuevamente los datos registrados. |
| 4 | | Si la información es válida, el sistema ejecuta correctamente la operación seleccionada. |
| **Postcondición** | El tipo de paquete turístico queda registrado, consultado, actualizado o inhabilitado correctamente en el sistema. | |

| | |
|---|---|
| **Requisito Funcional** | RF-025 al RF-030 |

***Nota.*** Esta tabla describe el escenario del caso de uso Gestión de Tipos de Paquetes Turísticos, integrando las funcionalidades de registrar, consultar, actualizar e inhabilitar los tipos de paquetes turísticos, garantizando una adecuada clasificación de los servicios complementarios ofrecidos por el Hotel Cacique Bolívar.

*Tabla 28 Especificación de Mensajes de Respuesta para Escenarios de Uso*

| **Nombre** | Gestión de Pagos | |
|---|---|---|
| **Caso de Uso** | CU-08 | |
| **Descripción** | Permite registrar, consultar, actualizar e inhabilitar los pagos asociados a las reservas del Hotel Cacique Bolívar, garantizando el control de las transacciones realizadas por los clientes. | |
| **Actor** | Gerente y Administrador (de acuerdo con los permisos asignados). | |
| **Precondiciones** | El usuario debe haber iniciado sesión en el sistema y debe existir una reserva registrada para asociar el pago. | |
| **Flujo básico** | | |
| **Paso** | **Actor** | **Sistema** |
| 1 | El usuario accede al módulo **Gestión de Pagos**. | El sistema muestra las funcionalidades disponibles según el rol del usuario. |

| | | |
|---|---|---|
| 2 | El usuario selecciona la operación que desea realizar (Registrar, Consultar, Actualizar o Inhabilitar pago). | El sistema presenta el formulario correspondiente. |
| 3 | El usuario registra, consulta o modifica la información del pago (reserva, cliente, tipo de pago, valor, fecha y estado). | El sistema valida la información y verifica que la reserva exista. |
| 4 | El usuario confirma la operación. | El sistema procesa la solicitud y verifica las reglas de negocio. |
| 5 | | El sistema registra, consulta, actualiza o inhabilita el pago según la operación seleccionada. |
| 6 | | El sistema muestra un mensaje indicando que la operación fue realizada correctamente. |
| **Flujo Alternativo** | | |

| Paso | Actor | Sistema |
|---|---|---|
| 1 | | El sistema detecta información incompleta, datos inválidos o una reserva inexistente. |
| 2 | | El sistema informa el motivo del error y solicita corregir la información. |
| 3 | El usuario corrige la información ingresada. | El sistema valida nuevamente los datos registrados. |
| 4 | | Si la información es válida, el sistema ejecuta correctamente la operación seleccionada. |
| **Postcondición** | El pago queda registrado, consultado, actualizado o inhabilitado correctamente en el sistema. | |
| **Requisito Funcional** | RF-031 al RF-036 | |

***Nota.*** Esta tabla describe el escenario del caso de uso **Gestión de Pagos**, integrando las funcionalidades de registrar, consultar, actualizar e inhabilitar pagos, garantizando el control de las transacciones económicas del Hotel Cacique Bolívar.

*Tabla 29 Escenario del Caso de Uso – Gestión de Tipos de Pago*

| **Nombre** | Gestión de Tipos de Pago | |
|---|---|---|
| **Caso de Uso** | CU-09 | |
| **Descripción** | Permite registrar, consultar, actualizar e inhabilitar los diferentes tipos de pago utilizados por el Hotel Cacique Bolívar, facilitando la administración de los medios de pago disponibles. | |
| **Actor** | Gerente y Administrador (de acuerdo con los permisos asignados). | |
| **Precondiciones** | El usuario debe haber iniciado sesión en el sistema y contar con permisos para acceder al módulo Gestión de Tipos de Pago. | |
| **Flujo básico** | | |
| **Paso** | **Actor** | **Sistema** |
| 1 | El usuario accede al módulo **Gestión de Tipos de Pago**. | El sistema muestra las funcionalidades |

| | | |
|---|---|---|
| | | disponibles según el rol del usuario. |
| 2 | El usuario selecciona la operación que desea realizar (Registrar, Consultar, Actualizar o Inhabilitar tipo de pago). | El sistema presenta el formulario correspondiente. |
| 3 | El usuario registra, consulta o modifica la información del tipo de pago (nombre, descripción y características). | El sistema valida la información y verifica que no exista un tipo de pago previamente registrado. |
| 4 | El usuario confirma la operación. | El sistema procesa la solicitud y verifica las reglas de negocio. |
| 5 | | El sistema registra, consulta, actualiza o inhabilita el tipo de pago. |
| 6 | | El sistema muestra un mensaje indicando que la operación fue realizada correctamente. |

| Flujo Alternativo | | |
|---|---|---|
| **Paso** | **Actor** | **Sistema** |
| 1 | | El sistema detecta información incompleta, datos inválidos o un tipo de pago previamente registrado. |
| 2 | | El sistema informa el motivo del error y solicita corregir la información. |
| 3 | El usuario corrige la información ingresada. | El sistema valida nuevamente los datos registrados. |
| 4 | | Si la información es válida, el sistema ejecuta correctamente la operación seleccionada. |
| **Postcondición** | El tipo de pago queda registrado, consultado, actualizado o inhabilitado correctamente en el sistema. | |

| | |
|---|---|
| **Requisito Funcional** | RF-037 al RF-042 |

***Nota.*** Esta tabla describe el escenario del caso de uso **Gestión de Tipos de Pago**, integrando las funcionalidades de registrar, consultar, actualizar e inhabilitar los diferentes medios de pago disponibles en el Hotel Cacique Bolívar.

*Tabla 30 Escenario del Caso de Uso – Gestión de PQRS+*

| **Nombre** | Gestión de PQRS+ | |
|---|---|---|
| **Caso de Uso** | CU-010 | |
| **Descripción** | Permite registrar, consultar, responder y actualizar el estado de las PQRS+ (Peticiones, Quejas, Reclamos, Solicitudes, Sugerencias y Felicitaciones) realizadas por los clientes, garantizando el seguimiento y la mejora continua del servicio prestado por el Hotel Cacique Bolívar. | |
| **Actor** | Cliente, Recepcionista, Gerente y Administrador (de acuerdo con los permisos asignados). | |
| **Precondiciones** | El usuario debe haber iniciado sesión en el sistema. Para registrar una PQRS+ debe existir un cliente registrado. | |
| **Flujo básico** | | |
| **Paso** | **Actor** | **Sistema** |

| | | |
|---|---|---|
| 1 | El usuario accede al módulo **Gestión de PQRS+**. | El sistema muestra las funcionalidades disponibles según el rol del usuario. |
| 2 | El usuario selecciona la operación que desea realizar (Registrar, Consultar, Responder o Actualizar estado). | El sistema presenta el formulario correspondiente. |
| 3 | El usuario registra o consulta la información de la PQRS+ (cliente, tipo, descripción, fecha y estado). | El sistema valida la información y verifica que el registro exista cuando corresponda. |
| 4 | El usuario confirma la operación. | El sistema procesa la solicitud y verifica las reglas de negocio. |
| 5 | | El sistema registra, consulta, responde o actualiza el estado de la PQRS+ según la operación seleccionada. |

| | | |
|---|---|---|
| 6 | | El sistema muestra un mensaje indicando que la operación fue realizada correctamente. |
| **Flujo Alternativo** | | |
| **Paso** | **Actor** | **Sistema** |
| 1 | | El sistema detecta información incompleta, datos inválidos o una PQRS+ inexistente. |
| 2 | | El sistema informa el motivo del error y solicita corregir la información. |
| 3 | El usuario corrige la información ingresada. | El sistema valida nuevamente los datos registrados. |
| 4 | | Si la información es válida, el sistema ejecuta |

| | | |
|---|---|---|
| | | correctamente la operación seleccionada. |
| **Postcondición** | La PQRS+ queda registrada, consultada, respondida o con su estado actualizado correctamente, permitiendo el seguimiento de las solicitudes realizadas por los clientes. | |
| **Requisito Funcional** | RF-055 al RF-060 | |

***Nota*.** Esta tabla describe el escenario del caso de uso Gestión de PQRS+, integrando las funcionalidades de registrar, consultar, responder y actualizar el estado de las solicitudes, fortaleciendo la comunicación entre los clientes y el Hotel Cacique Bolívar y facilitando el seguimiento de cada caso.

*Tabla 31 Escenario del Caso de Uso – Gestión de Tipos de PQRS+*

| **Nombre** | Gestión de Tipos de PQRS+ | |
|---|---|---|
| **Caso de Uso** | CU-011 | |
| **Descripción** | Permite registrar, consultar, actualizar e inhabilitar los diferentes tipos de PQRS+ (Petición, Queja, Reclamo, Solicitud, Sugerencia y Felicitación), facilitando la correcta clasificación de las solicitudes registradas por los clientes. | |
| **Actor** | Recepcionista, Gerente y Administrador (de acuerdo con los permisos asignados). | |
| **Precondiciones** | El usuario debe haber iniciado sesión en el sistema y contar con permisos para acceder al módulo Gestión de Tipos de PQRS+. | |
| **Flujo básico** | | |
| **Paso** | **Actor** | **Sistema** |
| 1 | El usuario accede al módulo **Gestión de Tipos de PQRS+**. | El sistema muestra las funcionalidades |

| | | |
|---|---|---|
| | | disponibles según el rol del usuario. |
| 2 | El usuario selecciona la operación que desea realizar (Registrar, Consultar, Actualizar o Inhabilitar tipo de PQRS+). | El sistema presenta el formulario correspondiente. |
| 3 | El usuario registra, consulta o modifica la información del tipo de PQRS+. | El sistema valida la información y verifica que no exista previamente un registro con el mismo nombre. |
| 4 | El usuario confirma la operación. | El sistema procesa la solicitud y verifica las reglas de negocio. |
| 5 | | El sistema registra, consulta, actualiza o inhabilita el tipo de PQRS+. |
| 6 | | El sistema muestra un mensaje indicando que la |

| | | |
|---|---|---|
| | | operación fue realizada correctamente. |
| **Flujo Alternativo** | | |
| **Paso** | **Actor** | **Sistema** |
| 1 | | El sistema detecta información incompleta, datos inválidos o un tipo de PQRS+ previamente registrado. |
| 2 | | El sistema informa el motivo del error y solicita corregir la información. |
| 3 | El usuario corrige la información ingresada. | El sistema valida nuevamente los datos registrados. |
| 4 | | Si la información es válida, el sistema ejecuta correctamente la operación seleccionada. |

| | |
|---|---|
| **Postcondición** | El tipo de PQRS+ queda registrado, consultado, actualizado o inhabilitado correctamente en el sistema. |
| **Requisito Funcional** | RF-061 al RF-066 |

***Nota*.** Esta tabla describe el escenario del caso de uso **Gestión de Tipos de PQRS+**, integrando las funcionalidades de registrar, consultar, actualizar e inhabilitar los diferentes tipos de solicitudes gestionadas por el Hotel Cacique Bolívar.

*Tabla 32 Escenario del Caso de Uso – Gestión de Informes*

| **Nombre** | Gestión de Informes | |
|---|---|---|
| **Caso de Uso** | CU-012 | |
| **Descripción** | Permite generar y exportar informes relacionados con clientes, reservas, pagos y PQRS+, proporcionando información estratégica para apoyar la toma de decisiones administrativas del Hotel Cacique Bolívar. | |
| **Actor** | Gerente y Administrador. | |
| **Precondiciones** | El usuario debe haber iniciado sesión y contar con permisos para acceder al módulo de informes. Debe existir información registrada en el sistema. | |
| **Flujo básico** | | |
| **Paso** | **Actor** | **Sistema** |
| 1 | El usuario accede al módulo **Gestión de Informes**. | El sistema muestra las opciones de informes disponibles. |

| | | |
|---|---|---|
| 2 | El usuario selecciona el tipo de informe que desea generar. | El sistema solicita los criterios de búsqueda (fechas, clientes, reservas, etc.). |
| 3 | El usuario ingresa los filtros requeridos. | El sistema consulta la información almacenada en la base de datos. |
| 4 | El usuario confirma la generación del informe. | El sistema genera el informe solicitado. |
| 5 | | El sistema presenta el informe en pantalla. |
| 6 | El usuario selecciona la opción Exportar. | El sistema exporta el informe en el formato seleccionado y confirma la operación. |
| **Flujo Alternativo** | | |
| **Paso** | **Actor** | **Sistema** |
| 1 | | El sistema detecta que no existen registros para los criterios seleccionados. |

| | | |
|---|---|---|
| 2 | | El sistema informa que no se encontraron resultados. |
| 3 | El usuario modifica los filtros. | El sistema realiza nuevamente la consulta. |
| 4 | | Si existen registros, el sistema genera el informe correctamente. |
| **Postcondición** | El informe queda generado y puede ser visualizado o exportado para su utilización administrativa. | |
| **Requisito Funcional** | RF-075 al RF-082 | |

***Nota*.** Esta tabla describe el escenario del caso de uso **Gestión de Informes**, permitiendo generar información consolidada sobre clientes, reservas, pagos y PQRS+, apoyando la gestión administrativa del Hotel Cacique Bolívar.

*Tabla 33 Escenario del Caso de Uso – Gestión de Consultas*

| **Nombre** | Gestión de Consultas |
|---|---|
| **Caso de Uso** | CU-013 |

| | |
|---|---|
| **Descripción** | Permite realizar consultas sobre huéspedes, disponibilidad de habitaciones, estado de reservas y reclamos registrados mediante PQRS+, facilitando el acceso rápido a la información del hotel. |
| **Actor** | Cliente, Recepcionista, Gerente y Administrador. |
| **Precondiciones** | El usuario debe haber iniciado sesión y existir información registrada en el sistema. |

| **Flujo básico** | | |
|---|---|---|
| **Paso** | **Actor** | **Sistema** |
| 1 | El usuario accede al módulo **Gestión de Consultas**. | El sistema presenta las consultas disponibles. |
| 2 | El usuario selecciona el tipo de consulta. | El sistema solicita los criterios de búsqueda. |
| 3 | El usuario ingresa la información requerida. | El sistema realiza la consulta correspondiente. |

| | | |
|---|---|---|
| 4 | El usuario confirma la búsqueda. | El sistema procesa la información almacenada. |
| 5 | | El sistema presenta los resultados encontrados. |
| 6 | | El sistema permite aplicar filtros adicionales o finalizar la consulta. |
| **Flujo Alternativo** | | |
| **Paso** | **Actor** | **Sistema** |
| 1 | | El sistema no encuentra información relacionada con la consulta. |
| 2 | | El sistema informa que no existen registros. |
| 3 | El usuario modifica los criterios de búsqueda. | El sistema ejecuta nuevamente la consulta. |
| 4 | | El sistema presenta los resultados correspondientes. |
| **Postcondición** | | |

| | |
|---|---|
| | El usuario obtiene la información solicitada de manera rápida y confiable.El informe queda generado y puede ser visualizado o exportado para su utilización administrativa. |
| **Requisito Funcional** | RF-083 al RF-086 |

***Nota*.** Esta tabla describe el escenario del caso de uso **Gestión de Consultas**, permitiendo consultar la información relacionada con huéspedes, disponibilidad, reservas y PQRS+, facilitando la toma de decisiones y la atención de los clientes.

*Tabla 34 Escenario del Caso de Uso – Administración de Usuarios*

| **Nombre** | Administración de Usuarios |
|---|---|
| **Caso de Uso** | CU-014 |
| **Descripción** | Permite registrar, consultar, actualizar e inhabilitar los usuarios del sistema, asignando los roles correspondientes para controlar el acceso a las diferentes funcionalidades del Hotel Cacique Bolívar |
| **Actor** | Administrador |
| **Precondiciones** | |

| | | |
|---|---|---|
| | El administrador debe haber iniciado sesión en el sistema y contar con permisos para administrar usuarios. | |
| **Flujo básico** | | |
| **Paso** | **Actor** | **Sistema** |
| 1 | El administrador accede al módulo Administración de Usuarios. | El sistema muestra las opciones Registrar, Consultar, Actualizar e Inhabilitar usuarios. |
| 2 | El administrador selecciona la operación que desea realizar. | El sistema presenta el formulario correspondiente. |
| 3 | El administrador registra, consulta o modifica la información del usuario (nombre, documento, correo, usuario, contraseña y rol). | El sistema valida la información ingresada y verifica que no existan registros duplicados. |
| 4 | El administrador confirma la operación. | El sistema procesa la solicitud correspondiente. |
| 5 | | El sistema registra, consulta, actualiza o |

| | | |
|---|---|---|
| | | inhabilita el usuario según la operación seleccionada. |
| 6 | | El sistema muestra un mensaje indicando que la operación fue realizada correctamente. |
| **Flujo Alternativo** | | |
| **Paso** | **Actor** | **Sistema** |
| 1 | | El sistema detecta información incompleta o inconsistente. |
| 2 | | El sistema informa el motivo del error y solicita corregir los datos. |
| 3 | El administrador corrige la información. | El sistema valida nuevamente los datos ingresados. |
| 4 | | Si la información es correcta, el sistema |

| | | |
|---|---|---|
| | | ejecuta la operación seleccionada. |
| **Postcondición** | Los usuarios quedan registrados, consultados, actualizados o inhabilitados correctamente, manteniendo el control de acceso al sistema. | |
| **Requisito Funcional** | RF-067 al RF-070 | |

***Nota.*** Esta tabla describe el escenario del caso de uso **Administración de Usuarios**, permitiendo controlar el registro, consulta, actualización e inhabilitación de los usuarios del sistema, garantizando la seguridad y administración de los accesos.

*Tabla 35 Escenario del Caso de Uso – Administración de Perfiles*

| **Nombre** | Administración de Perfiles | |
|---|---|---|
| **Caso de Uso** | CU-015 | |
| **Descripción** | Permite registrar, consultar, actualizar e inhabilitar los perfiles del sistema, definiendo los permisos y privilegios asociados a cada rol del Hotel Cacique Bolívar. | |
| **Actor** | Administrador. | |
| **Precondiciones** | El administrador debe haber iniciado sesión y contar con permisos para administrar perfiles. | |
| **Flujo básico** | | |
| **Paso** | **Actor** | **Sistema** |
| 1 | El administrador accede al módulo Administración de Perfiles. | El sistema muestra las opciones Registrar, Consultar, Actualizar e Inhabilitar perfiles. |

| | | |
|---|---|---|
| 2 | El administrador selecciona la operación que desea realizar. | El sistema presenta el formulario correspondiente. |
| 3 | El administrador registra, consulta o modifica la información del perfil y los permisos asociados. | El sistema valida la información ingresada y verifica que el perfil no exista previamente cuando corresponda. |
| 4 | El administrador confirma la operación. | El sistema procesa la solicitud correspondiente. |
| 5 | | El sistema registra, consulta, actualiza o inhabilita el perfil seleccionado. |
| 6 | | El sistema muestra un mensaje indicando que la operación fue realizada correctamente. |
| **Flujo Alternativo** | | |
| **Paso** | **Actor** | **Sistema** |

| | | |
|---|---|---|
| 1 | | El sistema detecta información incompleta o inconsistente. |
| 2 | | El sistema informa el motivo del error y solicita corregir la información. |
| 3 | El administrador corrige los datos. | El sistema valida nuevamente la información ingresada. |
| 4 | | Si la información es correcta, el sistema ejecuta la operación seleccionada. |
| **Postcondición** | Los perfiles quedan registrados, consultados, actualizados o inhabilitados correctamente, manteniendo la administración de los permisos del sistema. | |
| **Requisito Funcional** | RF-071 al RF-074 | |

***Nota*.** Esta tabla describe el escenario del caso de uso **Administración de Perfiles**, permitiendo administrar los perfiles y permisos del sistema para garantizar un adecuado control de acceso de acuerdo con los roles definidos.

**Diseño de base de datos 7.2.6.** En esta parte se halla el diseño de la base de datos de la aplicación, que tiene como fin ordenar y almacenar la información de forma estructurada, eficiente y consistente. El diseño sirve para identificar las entidades que están en el sistema, sus atributos, las claves que las identifican y las relaciones que hay entre ellas.

Para representar la estructura de la información se concretó a utilizar el modelo entidad-relación (ER), que permite identificar las entidades, los atributos y las relaciones necesarias para que, a continuación, construir el modelo relacional. Bagui y Earp (2022), afirman que los diagramas entidad-relación son una herramienta clave para el modelado de datos y sirven para realizar la transformación del diseño conceptual a la base de datos relacional.

- **Modelo de datos**

  Define qué datos se almacenan, cómo se relacionan entre sí, y cómo interactúan con el sistema. Para un proyecto, como el de gestión de reservas en un hotel, el modelo de datos es crucial para garantizar que la información fluya de manera eficiente y esté bien organizada.

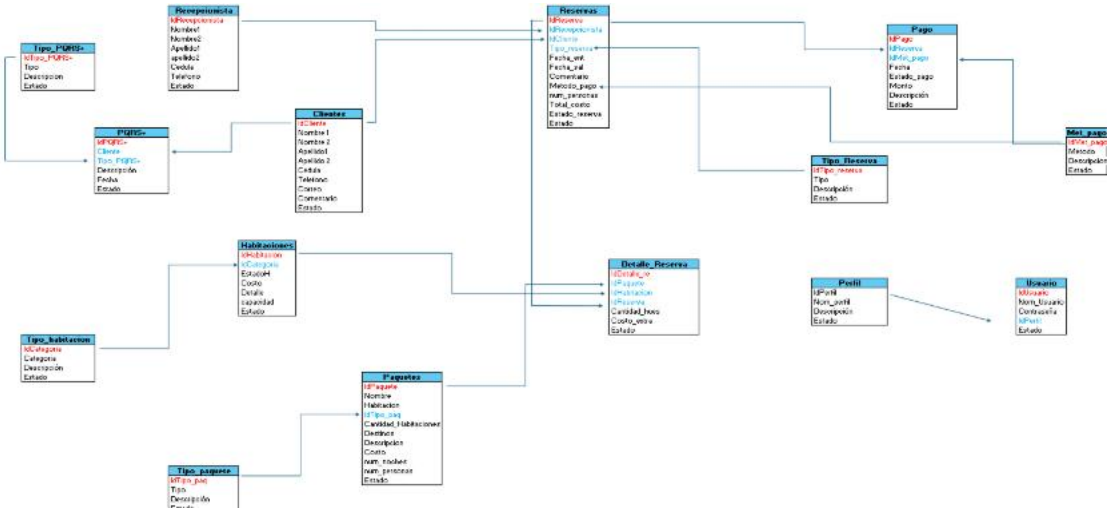


*Figura 46 Modelo de Datos*

***Nota***. En esta imagen se demuestra las tablas de la aplicación con sus respetivas claves principales, foráneas y sus respetivas uniones.

- **Diccionario de datos**

  Un diccionario de datos es un recurso que describe de manera detallada la estructura de los datos dentro de un sistema o base de datos. Proporciona una descripción clara y organizada de los elementos de datos, incluyendo sus nombres, tipos de datos, formatos, restricciones, relaciones y el propósito de cada campo en el sistema. El diccionario de datos actúa como una guía o referencia que facilita el entendimiento y manejo de la base de datos tanto para los desarrolladores como para los usuarios técnicos.

*Tabla 36 Diccionario Técnico de Clases y Atributos (Cliente)*

| **Tabla: Cliente** | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Descripción:** | **En esta tabla se guardará todos los clientes.** | | | | | | |
| **Campo** | **Tipo de Datos** | **Tamaño** | **Nulo** | **Incrementar** | **Tipo de Campo** | **Tabla Relacionada** | **Descripción** |
| IdCliente | Varchar | 20 | No | | Clave Primaria | | En este campo se guardará el id de los clientes. |
| Cedula | Varchar | 20 | No | | | | Aquí se guardará la cedula de los clientes. |
| Nombre 1 | Varchar | 60 | No | | | | Aquí se guardará el primer |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | | | | | | | nombre de los clientes |
| Nombre 2 | Varchar | 60 | No | | | | Aquí se guardará el segundo nombre de los clientes |
| Apellido 1 | Varchar | 60 | No | | | | Aquí se guardará el primer apellido de los clientes |
| Apellido 2 | Varchar | 60 | No | | | | Aquí se guardará el segundo apellido de los clientes |
| Teléfono | Varchar | 10 | No | | | | Aquí se guardará el |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | | | | | | | teléfono de los clientes |
| Correo | Varchar | 60 | No | | | | Aquí se guardará el correo de los clientes |
| Comentario | Varchar | 100 | Si | | | | Aquí se guardará algunas observaciones de los clientes |

***Nota***. En esta tabla se muestra los datos que está compuesto la tabla cliente y sus respectivos elementos.

*Tabla 37 Diccionario Técnico de Clases y Atributos (Recepcionista)*

| **Tabla: Recepcionista** | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Descripción:** | **En esta tabla se guardará las recepcionistas.** | | | | | | |
| **Campo** | **Tipo de Datos** | **Tamaño** | **Nulo** | **Incrementar** | **Tipo de Campo** | **Tabla Relacionada** | **Descripción** |
| IdRecepcionista | Varchar | 20 | No | | Clave Primaria | | En este campo se guardará el id de las recepcionistas |
| Cedula | Varchar | 20 | No | | | | Aquí se guardará la cedula de las recepcionistas |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Nombre 1 | Varchar | 60 | No | | | | Aquí se guardará el primer nombre de las recepcionistas |
| Nombre 2 | Varchar | 60 | No | | | | Aquí se guardará el segundo nombre de las recepcionistas |
| Apellido 1 | Varchar | 60 | No | | | | Aquí se guardará el primer apellido de las |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | | | | | | | recepcionistas |
| Apellido 2 | Varchar | 60 | No | | | | Aquí se guardará el segundo apellido las recepcionistas |
| Teléfono | Varchar | 10 | No | | | | Aquí se guardará el teléfono de las recepcionistas |

***Nota*.** En esta tabla se muestra los datos que está compuesto la tabla recepcionistas y sus respectivos elementos.

*Tabla 38 Diccionario Técnico de Clases y Atributos (Habitación)*

| **Tabla: Habitaciones** | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Descripción:** | **En esta tabla se guardará las habitaciones.** | | | | | | |
| **Campo** | **Tipo de Datos** | **Tamaño** | **Nulo** | **Incrementar** | **Tipo de Campo** | **Tabla Relacionada** | **Descripción** |
| IdHabitaciones | Varchar | 20 | No | | Clave Primaria | | En este campo se guardará el id de las habitaciones |
| IdCategoria | Varchar | 20 | No | | FK | Tipo_habitaciones | Aquí se guardará la categoría de las |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | | | | | | | habitaciones |
| Estado_habitaciones | Varchar | 60 | No | | | | Aquí se guardará el estados de las habitaciones |
| Costo | Float | 60 | No | | | | Aquí se guardará el costo de las habitaciones |
| Detalle | Varchar | 60 | Si | | | | Aquí se guardará los detalles de las |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | | | | | | | habitaciones |
| Capacidad | Varchar | 60 | No | | | | Aquí se guardará las capacidades de las habitaciones |

***Nota*.** En esta tabla se muestra los datos que está compuesto la tabla habitaciones y sus respectivos elementos.

*Tabla 39Diccionario Técnico de Clases y Atributos (Tipo de Habitación)*

| **Tabla: Tipo_habitaciones** | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Descripción:** | **En esta tabla se guardará el tipo de habitaciones.** | | | | | | |
| **Campo** | **Tipo de Datos** | **Tamaño** | **Nulo** | **Incrementar** | **Tipo de Campo** | **Tabla Relacionada** | **Descripción** |
| IdCategoria | Varchar | 20 | No | | Clave Primaria | | En este campo se guardará el id de las habitaciones |
| Categoría | Varchar | 20 | No | | | | Aquí se guardará la categoría de las habitaciones |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Descripción | Varchar | 60 | No | | | | Aquí se guardará la descripción de las habitaciones |

***Nota*.** En esta tabla se muestra los datos que está compuesto la tabla de tipo de habitaciones y sus respectivos elementos.

*Tabla 40 Diccionario Técnico de Clases y Atributos (Paquetes)*

| **Tabla: Paquetes** | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Descripción:** | **En esta tabla se guardará los paquetes** | | | | | | |
| **Campo** | **Tipo de Datos** | **Tamaño** | **Nulo** | **Incrementar** | **Tipo de Campo** | **Tabla Relacionada** | **Descripción** |
| IdPaquetes | Varchar | 20 | No | | Clave Primaria | | En este campo se guardará el id de paquetes |
| Nombre | Varchar | 20 | No | | | | Aquí se guardará el nombre del paquete |
| Habitación | Varchar | 60 | No | | | | Aquí se guardará |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | | | | | | | la habitación |
| Cantidad | Int | 10 | No | | | | Aquí se guardará la cantidad de habitaciones |
| idTipo_paq | Varchar | 20 | Si | | FK | Tipo_paquete | Aquí se guardará el tipo de paquete |
| Destinos | Varchar | 20 | No | | | | Aquí se guardará el destino del paquete |
| Descripción | Varchar | 60 | Si | | | | Aquí se guardará la |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | | | | | | | descripción del paquete |
| Costo | Float | 60 | No | | | | Aquí se guardará el costo del paquete |
| Num_noches | Int | 10 | No | | | | Aquí se guardará el número de noches |
| Num_personas | Int | 10 | No | | | | Aquí se guardará el número de personas |

***Nota*.** En esta tabla se muestra los datos que está compuesto la tabla de paquetes y sus respectivos elementos.

*Tabla 41 Diccionario Técnico de Clases y Atributos (Tipo Paquetes)*

| **Tabla: Tipo_paquete** | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Descripción:** | **En esta tabla se guardará el tipo de paquetes** | | | | | | |
| **Campo** | **Tipo de Datos** | **Tamaño** | **Nulo** | **Incrementar** | **Tipo de Campo** | **Tabla Relacionada** | **Descripción** |
| IdTipo_paq | Varchar | 20 | No | | Clave Primaria | | En este campo se guardará el id de tipo de paquetes |
| Tipo | Varchar | 20 | No | | | | Aquí se guardará el tipo de paquetes |
| Descripción | Varchar | 60 | Si | | | | Aquí se guardará la |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | | | | | | | descripción de tipo de paquetes |

***Nota.*** En esta tabla se muestra los datos que está compuesto la tabla de tipo de paquetes y sus respectivos elementos.

*Tabla 42 Diccionario Técnico de Clases y Atributos (Usuario)*

| **Tabla: Usuario** | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Descripción:** | **En esta tabla se guardará los usuarios** | | | | | | |
| **Campo** | **Tipo de Datos** | **Tamaño** | **Nulo** | **Incrementar** | **Tipo de Campo** | **Tabla Relacionada** | **Descripción** |
| IdUsuarios | Varchar | 20 | No | | Clave Primaria | | En este campo se guardará el id de usuario |
| Nombre_usuario | Varchar | 20 | No | | | | Aquí se guardará el nombre del usuario |
| Contraseña | Varchar | 60 | No | | | | Aquí se guardará la |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | | | | | | | contraseña del usuario |
| | Varchar | 20 | No | | FK | Perfil | Aquí se guardará el id del perfil del usuario |
| Descripción | **Varchar** | **60** | **Si** | | | | Aquí se guardará la descripción del usuario |

***Nota.*** En esta tabla se muestra los datos que está compuesto la tabla de usuarios y sus respectivos elementos.

*Tabla 43 Diccionario Técnico de Clases y Atributos (Perfil)*

| **Tabla: Perfil** |
|---|

| **Descripción:** | **En esta tabla se guardará los perfil** | | | | | | |
|---|---|---|---|---|---|---|---|
| **Campo** | **Tipo de Datos** | **Tamaño** | **Nulo** | **Incrementar** | **Tipo de Campo** | **Tabla Relacionada** | **Descripción** |
| IdPerfil | Varchar | 20 | No | | Clave Primaria | | En este campo se guardará el id del perfil |
| Nombre_perfil | Varchar | 20 | No | | | | Aquí se guardará el nombre del perfil |
| Descripción | Varchar | 60 | Si | | | | Aquí se guardará la descripción del perfil |

***Nota*.** En esta tabla se muestra los datos que está compuesto la tabla de perfil y sus respectivos elementos.

*Tabla 44 Diccionario Técnico de Clases y Atributos (Tipo Reserva)*

| **Tabla: Tipo_reserva** | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Descripción:** | **En esta tabla se guardará los tipos de reserva** | | | | | | |
| **Campo** | **Tipo de Datos** | **Tamaño** | **Nulo** | **Incrementar** | **Tipo de Campo** | **Tabla Relacionada** | **Descripción** |
| IdTipo_reserva | Varchar | 20 | No | | Clave Primaria | | En este campo se guardará el id del tipo de reserva |
| Tipo | Varchar | 20 | No | | | | Aquí se guardará el tipo de reserva |
| Descripción | Varchar | 60 | Si | | | | Aquí se guardará |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | | | | | | | la descripción del tipo de reserva |

***Nota*.** En esta tabla se muestra los datos que está compuesto la tabla del tipo de reserva y sus respectivos elementos.

*Tabla 45 Diccionario Técnico de Clases y Atributos (Reservas)*

| **Tabla: Reserva** | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Descripción:** | **En esta tabla se guardará las reservas** | | | | | | |
| **Campo** | **Tipo de Datos** | **Tamaño** | **Nulo** | **Incrementar** | **Tipo de Campo** | **Tabla Relacionada** | **Descripción** |
| IdReserva | Varchar | 20 | No | | Clave Primaria | | En este campo se guardará el id de reserva |
| IdRecepcionista | Varchar | 20 | No | | FK | Recepcionista | Aquí se guardará el id de las recepcionistas |
| IdCliente | Varchar | 20 | Si | | FK | Cliente | Aquí se guardará el |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | | | | | | | id de los clientes |
| IdTipo_reserva | Varchar | 20 | No | | FK | Tipo reserva | Aquí se guardará el id del tipo de reservas |
| Fecha_ent | Date | | Si | | | | Aquí se guardará la fecha de entrada |
| Fecha_sal | Date | | No | | | | Aquí se guardará la fecha de salida |
| IdMet_pago | Varchar | 20 | Si | | FK | Met_pago | Aquí se guardará el id del método de pago |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Num_personas | Int | 10 | No | | | | Aquí se guardará el número de personas |
| Total_Costo | Float | 60 | No | | | | Aquí se guardará el total de costo |
| Estado_reserva | Varchar | 20 | No | | | | Aquí se guardará el estado de la reserva |
| Comentario | Varchar | 60 | Si | | | | Aquí se guardará el comentario de las reservas |

***Nota.*** En esta tabla se muestra los datos que está compuesto la tabla de reserva y sus respectivos elementos.

*Tabla 46 Diccionario Técnico de Clases y Atributos (PQRS+)*

| **Tabla: PQRS+** | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Descripción:** | **En esta tabla se guardará las PQRS+ (Preguntas, Quejas, Reclamos y Solicitudes)** | | | | | | |
| **Campo** | **Tipo de Datos** | **Tamaño** | **Nulo** | **Incrementar** | **Tipo de Campo** | **Tabla Relacionada** | **Descripción** |
| IdPQRS+ | Varchar | 20 | No | | Clave Primaria | | En este campo se guardará el id de las PQRS+ |
| IdCliente | Varchar | 20 | No | | FK | Cliente | Aquí se guardará el id de los clientes |
| IdTipo_PQRS+ | Varchar | 20 | Si | | FK | Tipo_PQRS+ | Aquí se guardará el id de los |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | | | | | | | tipos de PQRS+ |
| Descripción | Varchar | 100 | No | | | | Aquí se guardará la descripción de las PQRS+ |
| Fecha | Date | | No | | | | Aquí se guardará la fecha |

***Nota*.** En esta tabla se muestra los datos que está compuesto la tabla de PQRS+ y sus respectivos elementos.

*Tabla 47 Diccionario Técnico de Clases y Atributos (Tipo PQRS+)*

| **Tabla: Tipo_PQRS+** | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Descripción:** | **En esta tabla se guardará los tipos de las PQRS+ (Preguntas, Quejas, Reclamos y Solicitudes)** | | | | | | |
| **Campo** | **Tipo de Datos** | **Tamaño** | **Nulo** | **Incrementar** | **Tipo de Campo** | **Tabla Relacionada** | **Descripción** |
| IdTipo_PQRS+ | Varchar | 20 | No | | Clave Primaria | | En este campo se guardará el id de los tipos de PQRS+ |
| Tipo | Varchar | 20 | No | | | | Aquí se guardará el tipo de PQRS+ |
| Descripción | Varchar | 100 | No | | | | Aquí se guardará |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | | | | | | | la descripción de tipo de las PQRS+ |

***Nota*.** En esta tabla se muestra los datos que está compuesto la tabla de los tipos de PQRS+ y sus respectivos elementos.

- **Diccionario de clases**

*Tabla 48 Diccionario de clase Paquetes*

| **Clase: Paquetes** | | | |
|---|---|---|---|
| **Atributo** | **Tipo** | **Visibilidad** | **Descripción** |
| IdPaquetes | String | Public | Identificador único del paquete turístico. |
| Nombre | String | Public | Nombre asignado al paquete turístico. |
| Habitación | String | Public | Información relacionada con la habitación incluida en el paquete. |
| Cantidad | Integer | Public | Cantidad de habitaciones incluidas en el paquete. |
| idTipo_paq | String | Public | Identificador del tipo de paquete al que pertenece. |
| Destinos | String | Public | Destinos incluidos en el paquete turístico. |
| Descripción | String | Public | Descripción de los servicios o características del paquete. |
| Costo | Float | Public | Costo establecido para el paquete turístico. |
| Num_noches | Integer | Public | Número de noches contempladas en el paquete. |
| Num_personas | Integer | Public | Número de personas contempladas en el paquete. |

| Método | Visibilidad | Parámetros de entrada | | Valores que retorna | | Descripción |
|---|---|---|---|---|---|---|
| | | Tipo | Descripción | Tipo | Descripción | |
| GuardarPaquete() | Public | string, string, string, int, string, string, string, float, int, int, | IdPaquetes, Nombre, Habitación, Cantidad, idTipo_paq, Destinos, Descripción, Costo, Num_noches , Num_personas | int | | Devuelve 1 si se registra correctamente el paquete turístico, 0 de lo contrario. |

| | | | | | | |
|---|---|---|---|---|---|---|
| ConsultarPaquete() | Public | String | IdPaquetes | | | Devuelve la información del paquete turístico cuando existe; de lo contrario, retorna null. |
| | | | | | | |
| ModificarPaquete() | Public | string, string, string, int, string, string, string, float, int, int | IdPaquetes Nombre Habitación Cantidad idTipo_paq Destinos Descripción Costo | Boolean | … | Retorna true si se realiza correctamente la actualización del paquete turístico y false de lo contrario. |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | Num_noches<br>Num_personas | | | |
| InhabilitarPaquete() | Public | String | IdPaquetes | Boolean | … | Retorna true si se realiza correctamente la inhabilitación del paquete turístico y false de lo contrario. |

***Nota*.** La tabla presenta el diccionario de clase paquetes con su respetiva información.

*Tabla 49 Diccionario de clase Usuario*

| **Clase: Usuario** | | | | | | |
|---|---|---|---|---|---|---|
| **Atributo** | **Tipo** | **Visibilidad** | **Descripción** | | | |
| IdUsuarios | String | Public | Identificador único del usuario. | | | |
| Nombre_usuario | String | Public | Nombre utilizado por el usuario para ingresar al sistema. | | | |
| Contraseña | String | Public | Credencial utilizada para la autenticación del usuario. | | | |
| idPerfil | String | Public | Identificador del perfil asignado al usuario. | | | |
| Descripción | String | Public | Información descriptiva asociada al usuario. | | | |
| **Método** | **Visibilidad** | **Parámetros de entrada** | | **Valores que retorna** | | **Descripción** |
| | | **Tipo** | **Descripción** | **Tipo** | **Descripción** | |
| GuardarUsuario() | Public | string, string, string, string, | IdUsuarios Nombre_usuario Contraseña | int | | Devuelve 1 si se registra correctame |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | string | idPerfil<br>Descripción | | | nte el usuario, 0 de lo contrario. |
| ConsultarUsuario() | Public | String | IdUsuarios | | | Devuelve la información del usuario cuando existe; de lo contrario, retorna null. |
| | | | | | | |
| ModificarUsuario() | Public | string, string, string, string, string | IdUsuarios<br>Nombre_usuario<br>Contraseña<br>idPerfil<br>Descripción | Boolean | … | Retorna true si se realiza correctamente la actualización del usuario y |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | | | false de lo contrario. |
| InhabilitarUsuario() | Public | String | IdUsuarios | Boolean | … | Retorna true si se realiza correctamente la inactivación del usuario mediante borrado lógico y false de lo contrario. |

***Nota*.** La tabla presenta el diccionario de clase usuarios con su respetiva información.

*Tabla 50 Diccionario de clase Perfil*

| **Clase: Perfil** | | | | | | |
|---|---|---|---|---|---|---|
| **Atributo** | **Tipo** | **Visibilidad** | **Descripción** | | | |
| IdPerfil | String | Public | Identificador único del perfil. | | | |
| Nombre_perfil | String | Public | Nombre del perfil asignado dentro del sistema. | | | |
| Descripción | String | Public | Descripción de las características y permisos asociados al perfil. | | | |
| **Método** | **Visibilidad** | **Parámetros de entrada** | | **Valores que retorna** | | **Descripción** |
| | | **Tipo** | **Descripción** | **Tipo** | **Descripción** | |
| GuardarPerfil() | Public | string, string, string | IdPerfil Nombre_perfil Descripción | Int | | Devuelve 1 si crea el perfil, 0 de lo contrario |

| | | | | | | |
|---|---|---|---|---|---|---|
| ConsultarPerfil() | Public | String | IdPerfil | | | Devuelve la información del perfil en caso de que exista; de lo contrario, retorna null. |
| | | | | | | |
| ModificarPerfil() | Public | string, string, string | IdPerfil Nombre_perfil Descripción | Boolean | … | Retorna true si se realiza correctamente la actualización del perfil y false de lo contrario. |
| InhabilitarPerfil() | Public | string | IdPerfil | Boolean | … | Retorna true si se realiza correctamente la |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | | | inhabilitación del perfil y false de lo contrario. |

***Nota*.** La tabla presenta el diccionario de clase perfil con su respetiva información.

*Tabla 51 Diccionario de clase PQRS+*

| **Clase: PQRS+** | | | | | |
|---|---|---|---|---|---|
| **Atributo** | **Tipo** | **Visibilidad** | **Descripción** | | |
| IdPQRS+ | String | Public | Identificador único de la solicitud (Pregunta, Queja, Reclamo, Solicitud) en el sistema. | | |
| Tipo_PQRS+ | String | Public | Tipo de la solicitud, que puede ser una Pregunta, Queja, Reclamo o Solicitud, para clasificar el tipo de PQRS+. | | |
| Cliente | String | Public | Información del cliente que realiza el PQRS+, que puede incluir datos como nombre, correo electrónico, o teléfono. | | |
| Descripcion | String | Public | Detalle o explicación sobre la solicitud presentada por el cliente, describiendo el motivo o naturaleza del PQRS+. | | |
| Fecha | String | Public | Fecha en la que se registró la solicitud (PQRS+) en el sistema. | | |
| Estado | Boolean | Public | Estado de PQRS+ | | |
| **Método** | **Visibilidad** | **Parámetros de entrada** | | **Valores que retorna** | **Descripción** |

| | | Tipo | Descripción | Tipo | Descripción | |
|---|---|---|---|---|---|---|
| GuardarPQRS+() | Public | string, string, string, string, string, boolean | IdPQRS+ Tipo_PQRS+ Cliente Descripcion Fecha Estado | int | | Devuelve 1 si crea el usuario, 0 de lo contrario |
| ConsultarPQRS+() | Public | String | IdPQRS+ | | | Devuelve la información de la PQRS+ en caso de que exista; de lo contrario, retorna null. |
| | | | | | | |

| | | | | | | |
|---|---|---|---|---|---|---|
| ModificarPQRS+() | Public | string, string, string, string, string, boolean | IdPQRS+ Tipo_PQRS+ Cliente Descripcion Fecha Estado | Boolean | … | Retorna true si se realiza correctamente la actualización de la pqrs+ y false de lo contrario. |
| InhabilitarPQRS+() | Public | string, string, string, string, string, boolean | IdPQRS+ Tipo_PQRS+ Cliente Descripcion Fecha Estado | Boolean | … | Retorna true si se realiza la inactivación del registro (borrado lógico), false de lo contrario |

***Nota.*** La tabla presenta el diccionario de clase PQRS+ con su respetiva información.

*Tabla 52 Diccionarios de clase Tipo PQRS+*

<table>
<tr><th colspan="7">Clase: Tipo_PQRS+</th></tr>
<tr><th>Atributo</th><th>Tipo</th><th>Visibilidad</th><th colspan="4">Descripción</th></tr>
<tr><td>IdTipo_PQRS+</td><td>String</td><td>Public</td><td colspan="4">Identificador único del tipo de PQRS+ en el sistema, usado para clasificar la solicitud como Pregunta, Queja, Reclamo o Solicitud.</td></tr>
<tr><td>Tipo</td><td>String</td><td>Public</td><td colspan="4">Tipo de la solicitud PQRS+, como Pregunta, Queja, Reclamo o Solicitud, que ayuda a categorizar la naturaleza del PQRS+.</td></tr>
<tr><td>Descripcion</td><td>String</td><td>Public</td><td colspan="4">Detalle o explicación sobre el motivo de la solicitud, proporcionando información más específica sobre el problema, consulta o requerimiento del cliente.</td></tr>
<tr><td>Estado</td><td>Boolean</td><td>Public</td><td colspan="4">Estado del tipo de PQRS+</td></tr>
<tr><th rowspan="2">Método</th><th rowspan="2">Visibilidad</th><th colspan="2">Parámetros de entrada</th><th colspan="2">Valores que retorna</th><th rowspan="2">Descripción</th></tr>
<tr><th>Tipo</th><th>Descripción</th><th>Tipo</th><th>Descripción</th></tr>
</table>

| | | | | | | |
|---|---|---|---|---|---|---|
| GuardarTipo_PQRS+ | Public | string, string, string, boolean | IdTipo_PQRS+ Tipo Descripcion Estado | int | | Devuelve 1 si crea el usuario, 0 de lo contrario |
| Consultar Tipo_PQRS+ | Public | String | IdTipo_PQRS+ | | | Devuelve la carrera en caso de que exista de lo contrario retorna null |
| | | | | | | |
| Modificar Tipo_PQRS+ | Public | string, string, string, boolean | IdTipo_PQRS+ Tipo Descripcion Estado | Boolean | … | Retorna true si se realiza la actualización, false de |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | | | los contrario |
| Inhabilitar Tipo_PQRS+ | Public | string, string, string, boolean | IdTipo_PQRS+ Tipo Descripcion Estado | Boolean | … | Retorna true si se realiza la inactivación del registro (borrado lógico), false de lo contrario |

***Nota*.** La tabla presenta el diccionario de clase de tipo de PQRS+ con su respetiva información.

*Tabla 53 Diccionarios de clase Habitaciones*

| **Clase: Habitaciones** | | | |
|---|---|---|---|
| **Atributo** | **Tipo** | **Visibilidad** | **Descripción** |

| | | | |
|---|---|---|---|
| IdHabitaciones | String | Public | Identificador único de cada habitación en el sistema, utilizado para hacer referencia a una habitación específica. |
| IdCategoria | String | Public | Identificador de la categoría de la habitación, que puede ser utilizada para clasificarla según características como el tipo. |
| Capacidad | String | Public | Número máximo de personas que pueden alojarse en la habitación, basado en su tamaño y configuración. |
| EstadoH | String | Public | Estado actual de la habitación, que puede ser "disponible", "ocupada", "en mantenimiento", etc. |
| Costo | String | Public | Precio por noche o por estancia de la habitación |
| Estado | Boolean | Public | Estado de habitaciones |

| **Método** | **Visibilidad** | **Parámetros de entrada** | | **Valores que retorna** | | **Descripción** |
|---|---|---|---|---|---|---|
| | | **Tipo** | **Descripción** | **Tipo** | **Descripción** | |
| Guardar Habitaciones | Public | string, string, | IdHabitaciones IdCategoria | int | | Devuelve 1 si crea el usuario, 0 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | string, string, string, string, boolean | CapacidadEstadoHCostoEstado | | | de lo contrario |
| Consultar Habitaciones | Public | String | IdHabitaciones | | | Devuelve la carrera en caso de que exista de lo contrario retorna null |
| | | | | | | |
| Modificar Habitaciones | Public | string, string, string, string, string, | IdHabitacionesIdCategoriaCapacidadEstadoHCostoEstado | Boolean | … | Retorna true si se realiza la actualización, false de los contrario |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | string, boolean | | | | |
| Inhabilitar Habitaciones | Public | string, string, string, string, string, string, boolean | IdHabitaciones IdCategoria Capacidad EstadoH Costo Estado | Boolean | … | Retorna true si se realiza la inactivación del registro (borrado lógico), false de lo contrario |

***Nota*.** La tabla presenta el diccionario de clase de habitación con su respetiva información.

*Tabla 54 Diccionarios de clase Tipo habitación*

| **Clase: Tipo_Habitaciones** | | | |
|---|---|---|---|
| **Atributo** | **Tipo** | **Visibilidad** | **Descripción** |
| IdCategoria | String | Public | Identificador único de la categoría. |

| | | | | | | |
|---|---|---|---|---|---|---|
| Categoria | String | Public | Nombre de la categoría | | | |
| Descripcion | String | Public | Detalles o características de la categoría. | | | |
| Estado | Boolean | Public | Estado del tipo de habitación | | | |
| **Método** | **Visibilidad** | **Parámetros de entrada** | | **Valores que retorna** | | **Descripción** |
| | | **Tipo** | **Descripción** | **Tipo** | **Descripción** | |
| Guardar Tipo_Habitación | Public | string, string, string, boolean | IdCategoria Categoria Descripcion Estado | int | | Devuelve 1 si crea el usuario, 0 de lo contrario |
| Consultar Tipo_Habitación | Public | String | IdTipo_habitacion | | | Devuelve la carrera en caso de que exista de lo contrario retorna null |
| | | | | | | |

| | | | | | | |
|---|---|---|---|---|---|---|
| Modificar Tipo_Habitación | Public | string, string, string, boolean | IdCategoria Categoria Descripcion Estado | Boolean | … | Retorna true si se realiza la actualización, false de los contrario |
| Inhabilitar Tipo_Habitación | Public | string, string, string, boolean | IdCategoria Categoria Descripcion Estado | Boolean | … | Retorna true si se realiza la inactivación del registro (borrado lógico), false de lo contrario |

***Nota*.** La tabla presenta el diccionario de clase de tipo de habitación con su respetiva información.

*Tabla 55 Diccionarios de clase Tipo paquete*

| **Clase: Tipo_Paquete** | | | | | | |
|---|---|---|---|---|---|---|
| **Atributo** | **Tipo** | **Visibilidad** | **Descripción** | | | |
| IdTipo_paq | String | Public | Identificador único del tipo de paquete. | | | |
| Tipo | String | Public | Categoría del paquete | | | |
| Descripcion | String | Public | Detalles sobre el tipo de paquete. | | | |
| Estado | Boolean | Public | Estado del tipo de paquete | | | |
| **Método** | **Visibilidad** | **Parámetros de entrada** | | **Valores que retorna** | | **Descripción** |
| | | **Tipo** | **Descripción** | **Tipo** | **Descripción** | |
| Guardar Tipo_paquete | Public | string, string, string, boolean | IdTipo_paq Categoria Descripcion Estado | Int | | Devuelve 1 si crea el usuario, 0 de lo contrario |

| | | | | | | |
|---|---|---|---|---|---|---|
| Consultar Tipo_paquete | Public | String | IdTipo_paq | | | Devuelve la carrera en caso de que exista de lo contrario retorna null |
| | | | | | | |
| Modificar Tipo_paquete | Public | string, string, string, boolean | IdTipo_paq Categoria Descripcion Estado | Boolean | … | Retorna true si se realiza la actualización, false de los contrario |
| Inhabilitar Tipo_paquete | Public | string, string, string, boolean | IdTipo_paq Categoria Descripcion Estado | Boolean | … | Retorna true si se realiza la inactivación del registro (borrado lógico), false |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | | | de lo contrario |

***Nota.*** La tabla presenta el diccionario de clase de tipo de paquete con su respetiva información.

*Tabla 56 Diccionarios de clase Clientes*

| **Clase: Clientes** | | | | | | |
|---|---|---|---|---|---|---|
| **Atributo** | **Tipo** | **Visibilidad** | **Descripción** | | | |
| IdClientes | String | Public | Identificador único del cliente. | | | |
| Nombre1 | String | Public | Primer nombre del cliente. | | | |
| Nombre2 | String | Public | Segundo nombre del cliente | | | |
| Apellido1 | String | Public | Primer apellido del cliente. | | | |
| Apellido2 | String | Public | Segundo apellido del cliente. | | | |
| Cedula | String | Public | Número de cédula o identificación del cliente. | | | |
| Correo | String | Public | Dirección de correo electrónico del cliente. | | | |
| Teléfono | String | Public | Número de teléfono del cliente. | | | |
| Comentario | String | Public | Notas o comentarios adicionales sobre el cliente. | | | |
| Estado | Boolean | Public | Estado de clientes | | | |
| **Método** | **Visibilidad** | **Parámetros de entrada** | | **Valores que retorna** | | **Descripción** |
| | | **Tipo** | **Descripción** | **Tipo** | **Descripción** | |

| | | | | | | |
|---|---|---|---|---|---|---|
| Guardar Clientes | Public | String<br>String<br>String<br>String<br>String<br>String<br>String<br>String<br>boolean | IdClientes<br>Nombre1<br>Nombre2<br>Apellido1<br>Apellido2<br>Cedula<br>Correo<br>Teléfono<br>Comentario<br>Estado | int | | Devuelve 1 si crea el usuario, 0 de lo contrario |
| Consultar Clientes | Public | String | IdCliente | | | Devuelve la carrera en caso de que exista de lo contrario retorna null |
| | | | | | | |

| | | | | | | |
|---|---|---|---|---|---|---|
| Modificar Clientes | Public | String<br>String<br>String<br>String<br>String<br>String<br>String<br>String<br>boolean | IdClientes<br>Nombre1<br>Nombre2<br>Apellido1<br>Apellido2<br>Cedula<br>Correo<br>Teléfono<br>Comentario<br>Estado | Boolean | … | Retorna true si se realiza la actualización, false de los contrario |
| Inhabilitar Clientes | Public | String<br>String<br>String<br>String<br>String<br>String<br>String<br>String<br>String | IdClientes<br>Nombre1<br>Nombre2<br>Apellido1<br>Apellido2<br>Cedula<br>Correo<br>Teléfono | Boolean | … | Retorna true si se realiza la inactivación del registro (borrado lógico), false de lo contrario |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | String<br>boolean | Comentario<br>Estado | | | |

***Nota*.** La tabla presenta el diccionario de clase de cliente con su respetiva información.

*Tabla 57 Diccionarios de clase Recepcionista*

| **Clase: Recepcionista** | | | | | | |
|---|---|---|---|---|---|---|
| **Atributo** | **Tipo** | **Visibilidad** | **Descripción** | | | |
| IdRecepcionista | String | Public | Identificador único del recepcionista. | | | |
| Nombre1 | String | Public | Primer nombre del recepcionista. | | | |
| Nombre2 | String | Public | Segundo nombre del recepcionista | | | |
| Apellido1 | String | Public | Primer apellido del recepcionista. | | | |
| Apellido2 | String | Public | Segundo apellido del recepcionista. | | | |
| Cedula | String | Public | Número de cédula o identificación del recepcionista. | | | |
| Teléfono | String | Public | Número de teléfono del recepcionista. | | | |
| Estado | Boolean | Public | Estado de recepcionista | | | |
| Correo | String | Privada | Almacena la dirección de correo electrónico del usuario para fines de contacto. | | | |
| **Método** | **Visibilidad** | **Parámetros de entrada** | | **Valores que retorna** | | **Descripción** |
| | | **Tipo** | **Descripción** | **Tipo** | **Descripción** | |

| | | | | | | |
|---|---|---|---|---|---|---|
| Guardar Recepcionista | Public | String<br>String<br>String<br>String<br>String<br>String<br>boolean | IdRecepcionista<br>Nombre1<br>Nombre2<br>Apellido1<br>Apellido2<br>Cedula<br>Teléfono<br>Estado | int | | Devuelve 1 si crea el usuario, 0 de lo contrario |
| Consultar Recepcionista | Public | String | IdRecepcionista | | | Devuelve la carrera en caso de que exista de lo contrario retorna null |
| | | | | | | |
| Modificar Recepcionista | Public | String | IdRecepcionista | Boolean | … | Retorna true si se realiza la |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | String<br>String<br>String<br>String<br>String<br>String<br>boolean | Nombre1<br>Nombre2<br>Apellido1<br>Apellido2<br>Cedula<br>Teléfono<br>Estado | | | actualización, false de los contrario |
| Inhabilitar Recepcionista | Public | String<br>String<br>String<br>String<br>String<br>String<br>String<br>boolean | IdRecepcionista<br>Nombre1<br>Nombre2<br>Apellido1<br>Apellido2<br>Cedula<br>Teléfono<br>Estado | Boolean | … | Retorna true si se realiza la inactivación del registro (borrado lógico), false de lo contrario |

***Nota.*** La tabla presenta el diccionario de clase de recepcionista con su respetiva información.

*Tabla 58 Diccionarios de clase Pago*

| **Clase: Pago** | | | | | | |
|---|---|---|---|---|---|---|
| **Atributo** | **Tipo** | **Visibilidad** | **Descripción** | | | |
| IdPago | String | Public | Identificador único del pago. | | | |
| IdReserva | String | Public | Identificador de la reserva asociada al pago. | | | |
| IdMet_pago | String | Public | Identificador del método de pago (ej. tarjeta, efectivo). | | | |
| Estado_pago | String | Public | Estado del pago(pendiente, completado, cancelado). | | | |
| Fecha | String | Public | Fecha en que se realizó el pago. | | | |
| Monto | Double | Public | Número de cédula o identificación del recepcionista. | | | |
| Descripcion | String | Public | Cantidad de dinero pagada. | | | |
| Estado | Boolean | Public | Estado de pago | | | |
| **Método** | **Visibilidad** | **Parámetros de entrada** | | **Valores que retorna** | | **Descripción** |
| | | **Tipo** | **Descripción** | **Tipo** | **Descripción** | |

| | | | | | | |
|---|---|---|---|---|---|---|
| Guardar pago | Public | String<br>String<br>String<br>String<br>String<br>Double<br>String<br>boolean | IdPago<br>IdReserva<br>IdMet_pago<br>Estado_pago<br>Fecha<br>Monto<br>Descripcion<br>Estado | int | | Devuelve 1 si crea el usuario, 0 de lo contrario |
| Consultar pago | Public | String | IdPago | | | Devuelve la carrera en caso de que exista de lo contrario retorna null |
| | | | | | | |
| Modificar pago | Public | String<br>String | IdPago<br>IdReserva | Boolean | … | Retorna true si se realiza la |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | String<br>String<br>String<br>Double<br>String<br>boolean | IdMet_pago<br>Estado_pago<br>Fecha<br>Monto<br>Descripcion<br>Estado | | | actualización, false de los contrario |
| Inhabilitar pago | Public | String<br>String<br>String<br>String<br>String<br>Double<br>String<br>boolean | IdPago<br>IdReserva<br>IdMet_pago<br>Estado_pago<br>Fecha<br>Monto<br>Descripcion<br>Estado | Boolean | … | Retorna true si se realiza la inactivación del registro (borrado lógico), false de lo contrario |

***Nota*.** La tabla presenta el diccionario de clase de pago con su respetiva información.

*Tabla 59 Diccionarios de clase Met_pago*

| **Clase: Met_pago** | | | | | | |
|---|---|---|---|---|---|---|
| **Atributo** | **Tipo** | **Visibilidad** | **Descripción** | | | |
| IdMet_pago | String | Public | Identificador único del método de pago. | | | |
| metodo | String | Public | Tipo de método de pago (ej. tarjeta de crédito, efectivo). | | | |
| Descripcion | String | Public | Detalles sobre el método de pago. | | | |
| Estado | Boolean | Public | Estado del met_pago | | | |
| **Método** | **Visibilidad** | **Parámetros de entrada** | | **Valores que retorna** | | **Descripción** |
| | | **Tipo** | **Descripción** | **Tipo** | **Descripción** | |
| Guardar met_pago | Public | string, string, string, boolean | IdMet_pago Método Descripcion Estado | int | | Devuelve 1 si crea el usuario, 0 de lo contrario |

| | | | | | | |
|---|---|---|---|---|---|---|
| Consultar met_pago | Public | String | IdTipo_paq | | | Devuelve la carrera en caso de que exista de lo contrario retorna null |
| | | | | | | |
| Modificar met_pago | Public | string, string, string, boolean | IdMet_pago Método Descripcion Estado | Boolean | … | Retorna true si se realiza la actualización, false de los contrario |
| Inhabilitar met_pago | Public | string, string, string, boolean | IdMet_pago Método Descripcion Estado | Boolean | … | Retorna true si se realiza la inactivación del registro (borrado lógico), false |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | | | de lo contrario |

***Nota*.** La tabla presenta el diccionario de clase de método de pago con su respetiva información.

*Tabla 60 Diccionarios de clase Reservas*

| **Clase: Reservas** | | | |
|---|---|---|---|
| **Atributo** | **Tipo** | **Visibilidad** | **Descripción** |
| IdReserva | String | Public | Identificador único de la reserva. |
| IdCliente | String | Public | Identificador del cliente que realiza la reserva. |
| IdRecepcionista | String | Public | Identificador del recepcionista que gestiona la reserva. |
| Tipo_reserva | String | Public | Tipo de reserva (ej. estándar, paquete). |
| Método_pago | String | Public | Método de pago utilizado para la reserva (ej. tarjeta, efectivo). |
| Estado_reserva | String | Public | Estado de la reserva (confirmada, cancelada, pendiente). |
| Fecha_ent | String | Public | Fecha de entrada del cliente en el hotel. |

| | | | | | | |
|---|---|---|---|---|---|---|
| Fecha_sal | String | Public | Fecha de salida del cliente del hotel. | | | |
| Total_costo | Double | Public | Costo total de la reserva. | | | |
| Comentario | String | Public | Notas adicionales sobre la reserva. | | | |
| Estado | String | Public | Estado de Reservas | | | |
| **Método** | **Visibilidad** | **Parámetros de entrada** | | **Valores que retorna** | | **Descripción** |
| | | **Tipo** | **Descripción** | **Tipo** | **Descripción** | |
| Guardar Reservas | Public | String<br>String<br>String<br>String<br>String<br>String<br>String<br>String<br>Double | IdReserva<br>IdCliente<br>IdRecepcionista<br>Tipo_reserva<br>Método_pago<br>Estado_reserva<br>Fecha_ent<br>Fecha_sal<br>Total_costo | int | | Devuelve 1 si crea el usuario, 0 de lo contrario |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | String<br>String | Comentario<br>Estado | | | |
| Consultar<br>Reservas | Public | String | IdReservas | | | Devuelve la carrera en caso de que exista de lo contrario retorna null |
| | | | | | | |
| Modificar<br>Reservas | Public | String<br>String<br>String<br>String<br>String<br>String<br>String | IdReserva<br>IdCliente<br>IdRecepcionista<br>Tipo_reserva<br>Método_pago<br>Estado_reserva<br>Fecha_ent | Boolean | … | Retorna true si se realiza la actualización, false de los contrario |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | String<br>Double<br>String<br>String | Fecha_sal<br>Total_costo<br>Comentario<br>Estado | | | |
| Inhabilitar<br>Reservas | Public | String<br>String<br>String<br>String<br>String<br>String<br>String<br>String<br>Double<br>String<br>String | IdReserva<br>IdCliente<br>IdRecepcionista<br>Tipo_reserva<br>Método_pago<br>Estado_reserva<br>Fecha_ent<br>Fecha_sal<br>Total_costo<br>Comentario<br>Estado | Boolean | … | Retorna true si se realiza la inactivación del registro (borrado lógico), false de lo contrario |

***Nota*.** La tabla presenta el diccionario de clase de reservas con su respetiva información.

*Tabla 61 Diccionarios de clase Tipo reserva*

| **Clase: Tipo_reserva** | | | | | | |
|---|---|---|---|---|---|---|
| **Atributo** | **Tipo** | **Visibilidad** | **Descripción** | | | |
| IdTipo_reserva | String | Public | Identificador único del tipo de reserva. | | | |
| Tipo | String | Public | Tipo de reserva (ej. estándar, grupal, paquete). | | | |
| Descripcion | String | Public | Detalles sobre el tipo de reserva. | | | |
| Estado | Boolean | Public | Estado de Tipo_reserva | | | |
| **Método** | **Visibilidad** | **Parámetros de entrada** | | **Valores que retorna** | | **Descripción** |
| | | **Tipo** | **Descripción** | **Tipo** | **Descripción** | |
| Guardar tipo_reserva | Public | string,<br>string,<br>string,<br>boolean | IdTipo_reserva<br>Tipo<br>Descripcion<br>Estado | int | | Devuelve 1 si crea el usuario, 0 de lo contrario |

| | | | | | | |
|---|---|---|---|---|---|---|
| Consultar tipo_reserva | Public | String | IdTipo_reserva | | | Devuelve la carrera en caso de que exista de lo contrario retorna null |
| | | | | | | |
| Modificar tipo_reserva | Public | string,<br>string,<br>string,<br>boolean | IdTipo_reserva<br>Tipo<br>Descripcion<br>Estado | Boolean | … | Retorna true si se realiza la actualización, false de los contrario |
| Inhabilitar tipo_reserva | Public | string,<br>string,<br>string,<br>boolean | IdTipo_reserva<br>Tipo<br>Descripcion<br>Estado | Boolean | … | Retorna true si se realiza la inactivación del registro (borrado |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | | | lógico), false de lo contrario |

***Nota*.** La tabla presenta el diccionario de clase de tipo de reservas con su respetiva información.

*Tabla 62 Diccionarios de clase Detalle reserva*

| **Clase: Detalle_reserva** | | | |
|---|---|---|---|
| **Atributo** | **Tipo** | **Visibilidad** | **Descripción** |
| IdDetalle_re | String | Public | Identificador único del detalle de la reserva. |
| IdHabitacion | String | Public | Identificador de la habitación asociada al detalle. |
| IdPaquete | String | Public | Identificador del paquete asociado al detalle. |
| IdReserva | String | Public | Identificador de la reserva relacionada con el detalle. |
| Costo_Extra | Double | Public | Costo adicional por servicios o extras en la reserva. |
| Cantidad_hues | Int | Public | Número de huéspedes adicionales en la reserva. |
| Estado | Boolean | Public | Estado de Detalle_reserva |

| Método | Visibilidad | Parámetros de entrada | | Valores que retorna | | Descripción |
|---|---|---|---|---|---|---|
| | | Tipo | Descripción | Tipo | Descripción | |
| Guardar Detalle_reserva | Public | String<br>String<br>String<br>String<br>double<br>int<br>boolean | IdDetalle_re<br>IdHabitacion<br>IdPaquete<br>IdReserva<br>Costo_Extra<br>Cantidad_hues<br>Estado | int | | Devuelve 1 si crea el usuario, 0 de lo contrario |
| Consultar Detalle_reserva | Public | String | IdDetalle_re | | | Devuelve la carrera en caso de que exista de lo contrario retorna null |
| | | | | | | |

| | | | | | | |
|---|---|---|---|---|---|---|
| Modificar Detalle_reserva | Public | String<br>String<br>String<br>String<br>double<br>int<br>boolean | IdDetalle_re<br>IdHabitacion<br>IdPaquete<br>IdReserva<br>Costo_Extra<br>Cantidad_hu<br>es<br>Estado | Boolea<br>n | … | Retorna true si se realiza la actualizació<br>n, false de los contrario |
| InhabilitarDiagr<br>ad Detalle_reserva | Public | String<br>String<br>String<br>String<br>double<br>int<br>boolean | IdDetalle_re<br>IdHabitacion<br>IdPaquete<br>IdReserva<br>Costo_Extra<br>Cantidad_hu<br>es<br>Estado | Boolea<br>n | … | Retorna true si se realiza la inactivación del registro (borrado lógico), false de lo contrario |

***Nota*.** La tabla presenta el diccionario de clase de detales de las reservas con su respetiva información.

**Diagrama de clases 7.2.7.** Diagrama de Clase es una herramienta esencial dentro del modelado orientado a objetos que se utiliza para representar la estructura estática del sistema, mostrando las clases, sus atributos, sus métodos y las relaciones entre ellas. Es fundamental para visualizar cómo las diferentes entidades del sistema interactúan y se relacionan entre sí.

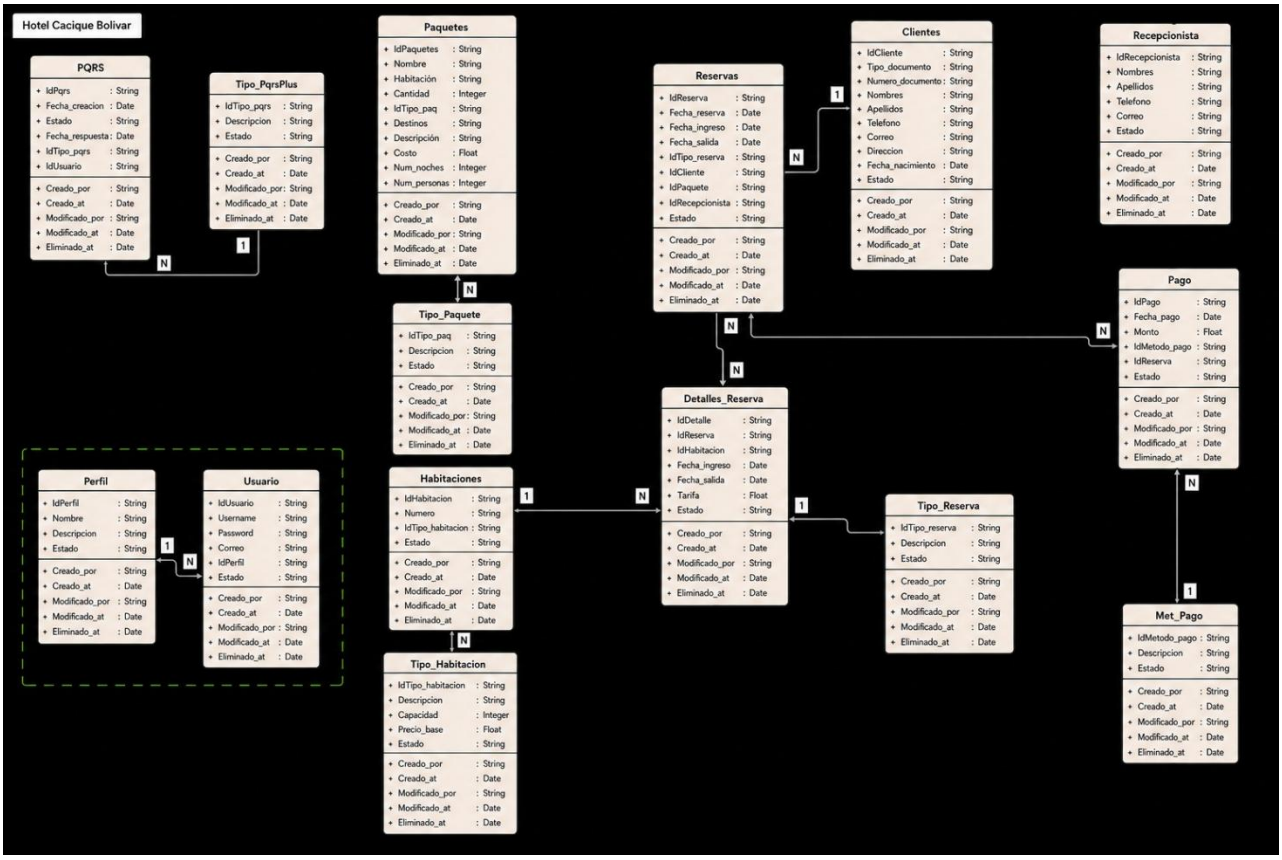



*Figura 47 Diagrama de clases*

***Nota.*** En la imagen se presenta el diagrama de clases diseñada para el proyecto de hotel.

**Análisis de riesgos 7.2.8.** En esta tabla se presenta el análisis de riesgo, el cual identifica y evalúa los posibles riesgos que pueden afectar el éxito del proyecto. Cada riesgo se describe en términos de su probabilidad de ocurrencia y su impacto potencial en el proyecto. Además, se proporciona una evaluación del nivel de riesgo, que combina la probabilidad y el impacto para determinar la importancia relativa de cada riesgo. Este análisis es fundamental para anticipar y mitigar los riesgos antes de que se conviertan en problemas reales, lo que ayuda a garantizar que el proyecto se complete dentro del tiempo, el presupuesto y los objetivos establecidos.

*Tabla 63 Protocolo de Mensajería para la Mitigación de Errores*

| **Análisis de riesgos** | | | |
|---|---|---|---|
| **Descripción del riesgo** | **Tipo de Riesgo** | **Probabilidad** | **Plan de acción** |
| No contar con un servidor que procese y almacene la información del sistema. | Tecnológico<br>Tecnológico<br>Tecnológico<br>Tecnológico | Media | Adquirir un servidor con la capacidad suficiente para garantizar el almacenamiento, procesamiento y disponibilidad de la información. |
| El sistema operativo del equipo no sea compatible con la aplicación. | | Media | Desarrollar el sistema utilizando tecnologías compatibles con diferentes versiones de los sistemas operativos soportados. |

| | | | |
|---|---|---|---|
| El acceso a Internet sea inestable o insuficiente. | | Media | Contratar un servicio de Internet con mayor ancho de banda y establecer mecanismos de respaldo de la conexión. |
| El sistema presente problemas de conexión con la base de datos. | | Media | Implementar pruebas de conectividad, respaldos periódicos y mecanismos de recuperación ante fallos de la base de datos. |
| El cliente no tenga conocimientos suficientes sobre el uso del sistema. | Cliente<br>Cliente<br>Cliente<br>Cliente | Baja | Capacitar a los usuarios mediante manuales, tutoriales y sesiones de entrenamiento antes |

| | | | |
|---|---|---|---|
| | | | de la puesta en marcha. |
| El cliente no defina claramente los requerimientos del sistema. | | Baja | Realizar reuniones periódicas para validar requisitos, documentar acuerdos y obtener aprobación antes de iniciar cada fase del desarrollo. Establecer reuniones programadas y canales oficiales de comunicación para resolver dudas y validar avances del proyecto. Programar revisiones funcionales al |
| El cliente no mantenga una comunicación constante con el equipo de desarrollo. | | Baja | |
| El cliente no participe en las revisiones y validaciones del sistema. | | Baja | |

| | | | |
|---|---|---|---|
| | | | finalizar cada módulo y solicitar la aprobación formal del cliente. |
| No realizar revisiones técnicas periódicas de requisitos, diseño y código. | Proceso<br>Proceso<br>Proceso<br>Proceso | Baja | Ejecutar revisiones técnicas durante todo el ciclo de desarrollo y documentar los hallazgos encontrados.<br>Elaborar y ejecutar casos de prueba para verificar el correcto funcionamiento de cada módulo antes de su implementación.<br>Registrar todas las incidencias, mejoras y resultados de las |
| No ejecutar pruebas funcionales y de calidad del sistema. | | Baja | |
| No documentar los resultados de las revisiones técnicas y pruebas realizadas. | | Baja | |

| | | | |
|---|---|---|---|
| | | | pruebas para mantener la trazabilidad del proyecto. |
| No aplicar estándares de calidad durante el desarrollo del software. | | Baja | Implementar estándares de ingeniería de software y buenas prácticas para garantizar la calidad, seguridad e integridad del sistema. |
| Información registrada incorrectamente por los usuarios. | Producto<br>Producto<br>Producto<br>Producto | Media | Implementar validaciones de campos, restricciones de integridad y controles para minimizar errores en |

| | | | |
|---|---|---|---|
| | | | el ingreso de información. |
| Funcionalidades insuficientes para cubrir las necesidades del hotel. | | Baja | Recopilar retroalimentación de los usuarios y planificar mejoras continuas del sistema. |
| Interfaz poco intuitiva que dificulte el uso del sistema. | | Media | Realizar pruebas de usabilidad y aplicar principios de diseño centrado en el usuario para mejorar la experiencia de uso. |
| Dependencia total del sistema para la operación del hotel. | | Baja | Establecer procedimientos manuales de contingencia y realizar copias de |

| | | | |
|---|---|---|---|
| | | | seguridad periódicas para garantizar la continuidad del servicio ante fallos del sistema. |

***Nota.*** Tabla donde se tiene los protocolos de mensajería para la mitigación de errores

## Desarrollo e implementación del Sistema 7.3.

**Herramientas de desarrollo 7.3.1**. Durante el desarrollo del sistema de gestión hotelera para el Hotel Cacique Bolívar se emplearon diferentes herramientas de software que facilitaron el proceso de programación, pruebas, control de versiones y administración de la base de datos.

*Tabla 64 Herramientas de desarrollo utilizadas*

| Herramienta | Función |
|---|---|
| IntelliJ IDEA | Desarrollo y depuración del backend en Java con Spring Boot. |
| Visual Studio Code | Desarrollo del frontend utilizando React. |
| HeidiSQL | Administración y gestión de la base de datos MySQL. |
| Postman | Pruebas de los servicios REST implementados en el backend. |
| Git | Control de versiones del código fuente durante el desarrollo. |
| GitHub | Almacenamiento y administración del repositorio del proyecto. |
| Maven | Gestión de dependencias y automatización de la compilación del proyecto. |

***Nota.*** La tabla presenta las principales herramientas utilizadas durante el proceso de desarrollo del sistema.

**Tecnologías utilizadas 7.3.2.** El desarrollo del sistema de gestión hotelera para el Hotel Cacique Bolívar se realizó utilizando tecnologías modernas que garantizan un alto rendimiento, escalabilidad, seguridad y facilidad de mantenimiento.

*Tabla 65 Stack tecnológico del sistema*

| Categoría | Tecnología | Versión | Función en el proyecto |
|---|---|---|---|
| Lenguaje de programación | Java | 21 | Desarrollo de la lógica de negocio del backend. |
| Framework Backend | Spring Boot | 4.0.5 | Desarrollo de servicios REST y gestión de la aplicación. |
| Persistencia (ORM) | Spring Data JPA / Hibernate | 6.x | Acceso y manipulación de la base de datos mediante entidades. |
| Base de datos | MySQL | 8.x | Almacenamiento de la información del sistema. |
| Seguridad | Spring Security | 6.x | Implementación de autenticación y autorización basada en roles. |
| Frontend | React | 18.x | Desarrollo de la interfaz gráfica del usuario y consumo de los servicios REST. |
| API | REST | HTTP | Comunicación entre el frontend y el backend. |

| | | | |
|---|---|---|---|
| Gestión de dependencias | Apache Maven | 3.9 | Administración de librerías y automatización de la construcción del proyecto. |
| Servidor de aplicaciones | Apache Tomcat (Embebido) | 10.x | Ejecución del backend desarrollado en Spring Boot. |
| Control de versiones | Git y GitHub | Git 2.x | Gestión del código fuente y control de versiones del proyecto. |

***Nota*.** La tabla presenta las principales tecnologías utilizadas durante el desarrollo del sistema, indicando la función que desempeña cada una dentro de la arquitectura implementada.

Esta integración permite una aplicación sólida y escalable, donde el backend en Spring Boot gestiona la lógica de negocio y los procesos del hotel, mientras que el frontend en React provee una interfaz dinámica. Ambas capas se comunican mediante servicios REST, asegurando la separación de responsabilidades y un fácil mantenimiento.

**Implementación de la arquitectura 7.3.3.** La arquitectura que se definió en la fase del diseño se implantó siguiendo el patrón cliente-servidor y una arquitectura en capas. Se desarrolló el backend a través de Spring Boot, organizando el código en diferentes paquetes especializados que separan las distintas lógicas de negocio, la lógica de acceso a datos, la lógica de seguridad y la lógica de cliente.

En la otra parte, su frontend fue desarrollado a través de React, organizando la aplicación en reutilización de componentes, módulos funcionales y de servicio de consumo de API REST. La comunicación entre ambas aplicaciones se realizó mediante peticiones HTTP, logrando así una separación de los lógicos del servidor y el de interfaz de usuario.

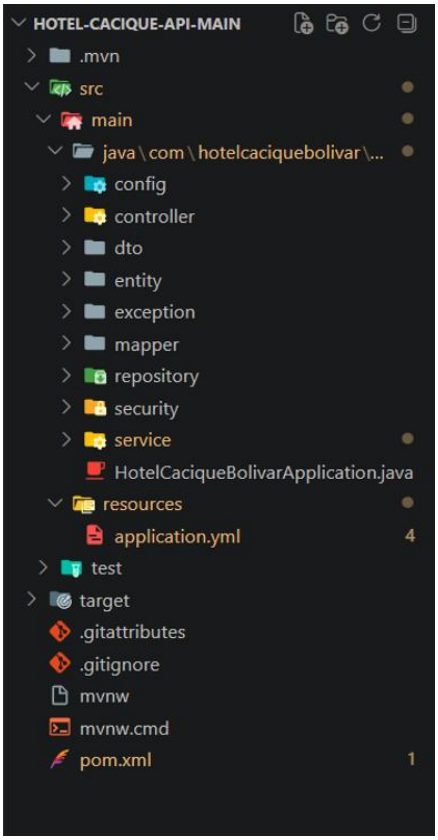



*Figura 48 Arquitectura implementada del backend.*

***Nota.*** La imagen muestra la implementación de la arquitectura en capas del backend, organizada mediante paquetes que separan las responsabilidades de cada componente del sistema.

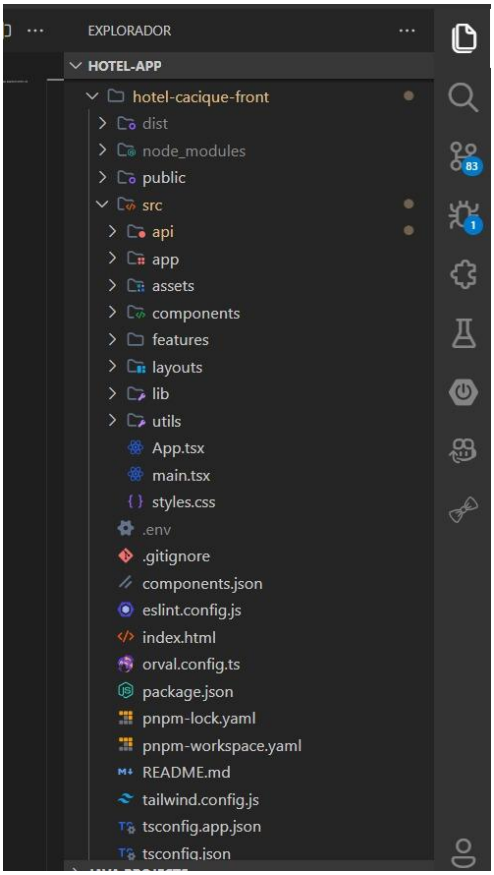



*Figura 49 Arquitectura implementada del frontend.*

***Nota.*** La imagen presenta la estructura del frontend desarrollada en React, organizada mediante componentes y módulos que permiten una mejor reutilización del código y facilitan el mantenimiento de la aplicación.

**Estructura del proyecto 7.3.4.** El proyecto fue dividido en dos aplicaciones independientes: un backend desarrollado en Spring Boot y un frontend desarrollado en

React. Esta organización permite mantener una arquitectura desacoplada, facilitando el desarrollo, mantenimiento y escalabilidad del sistema.

- **Backend (Spring Boot)**

  El backend se encuentra organizado en paquetes que cumplen funciones específicas:

  - **config:** configuración general del sistema.
  - **controller:** recepción de las solicitudes HTTP.
  - **dto:** objetos de transferencia de datos.
  - **entity:** representación de las entidades de la base de datos.
  - **exception:** manejo de excepciones.

    **mapper:** conversión entre entidades y DTO.
  - **repository:** acceso a la base de datos.
  - **security:** autenticación y autorización.
  - **service:** implementación de la lógica de negocio.

- **Frontend (React)**

  El frontend está organizado de la siguiente manera:

  - **api:** consumo de los servicios REST.
  - **app:** configuración principal de la aplicación.
  - **assets:** recursos gráficos.

- **components:** componentes reutilizables.
- **features:** módulos funcionales del sistema.
- **layouts:** estructura visual de las páginas.
- **lib:** bibliotecas auxiliares.
- **utils:** funciones de apoyo.

Esta organización favorece la reutilización de componentes y facilita el crecimiento del proyecto conforme se agregan nuevas funcionalidades.

**Evidencias del desarrollo 7.3.5.**

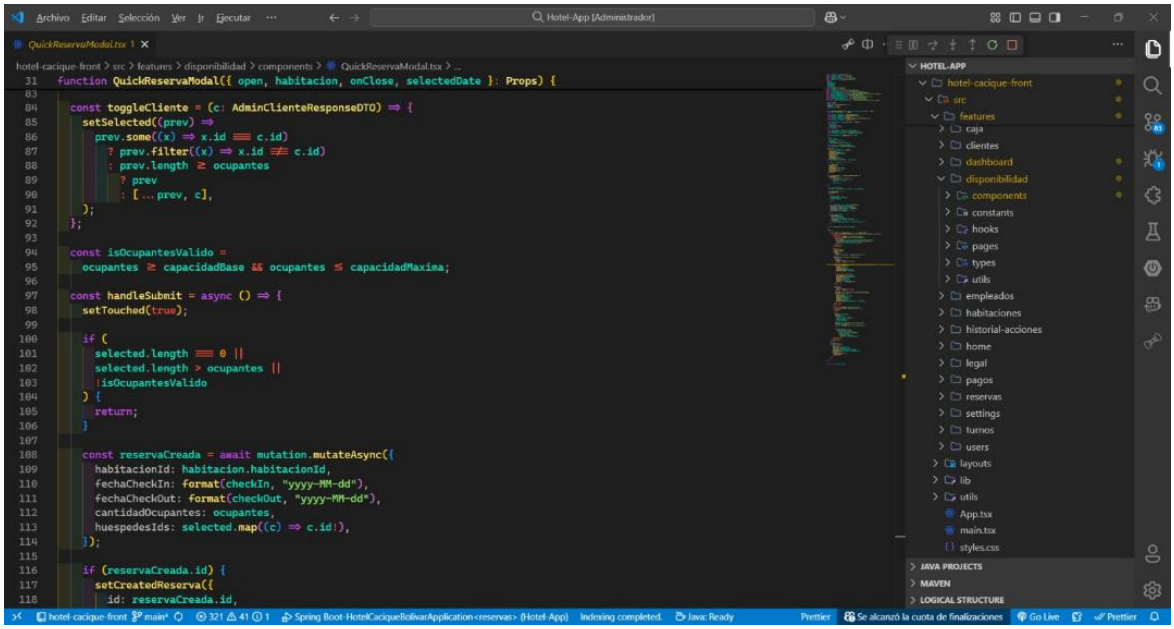



*Figura 50 Proyecto React en VS Code.*

***Nota***. En esta imagen se puede ver el proyecto de react abierto en VS Code.

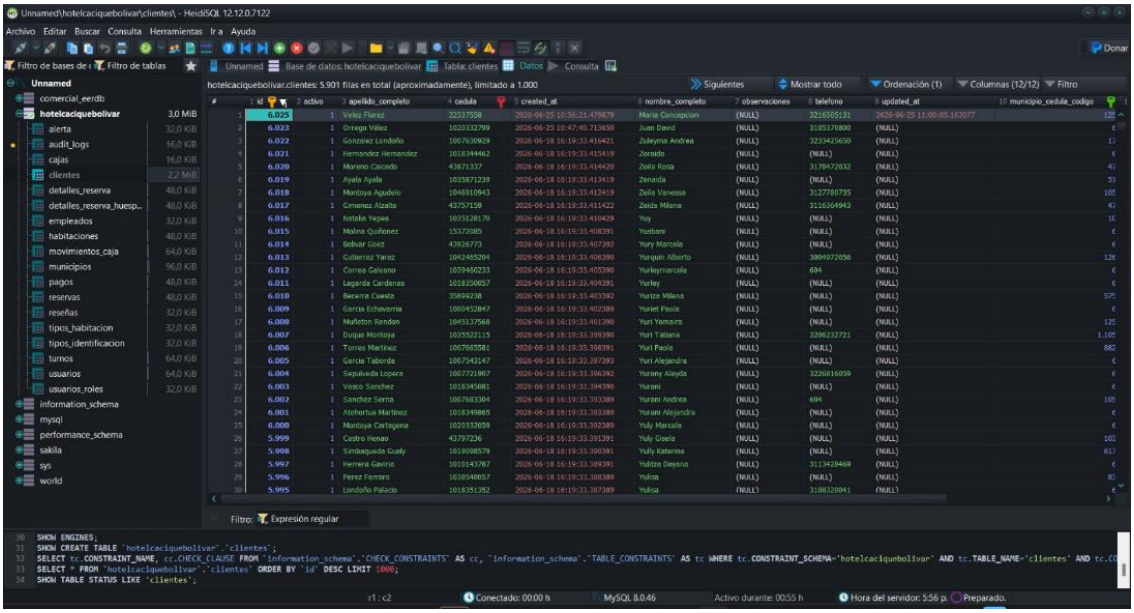


*Figura 51 Base de datos en HeidiSQL*

***Nota.*** En la imagen se puede ver las tablas de bases de datos utilizadas en el proyecto.

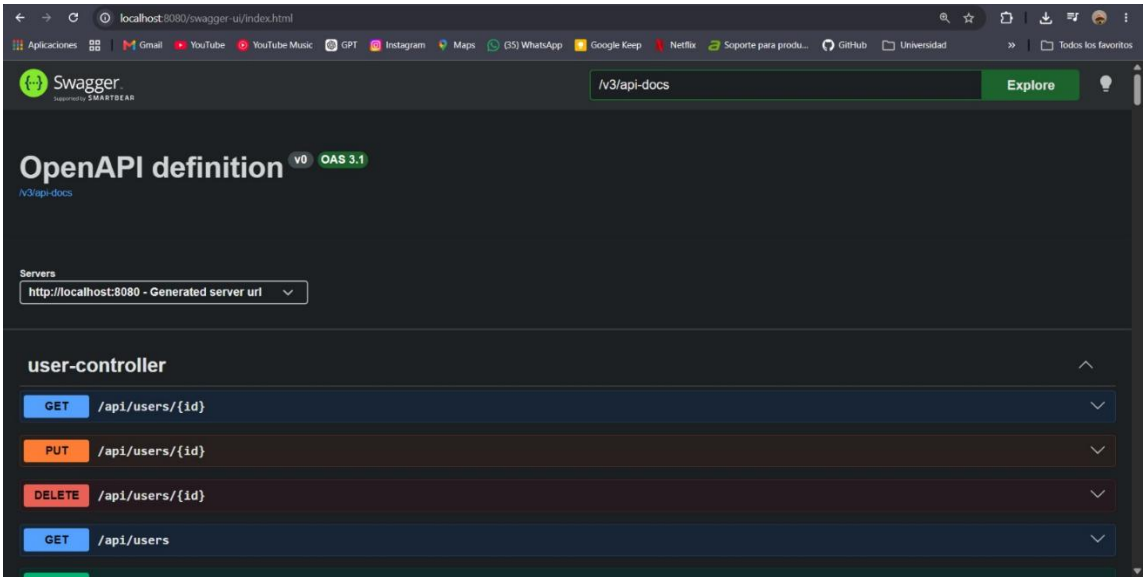



*Figura 52 Swagger*

***Nota.*** En la imagen se puede ver la documentación de rest api con swagger.

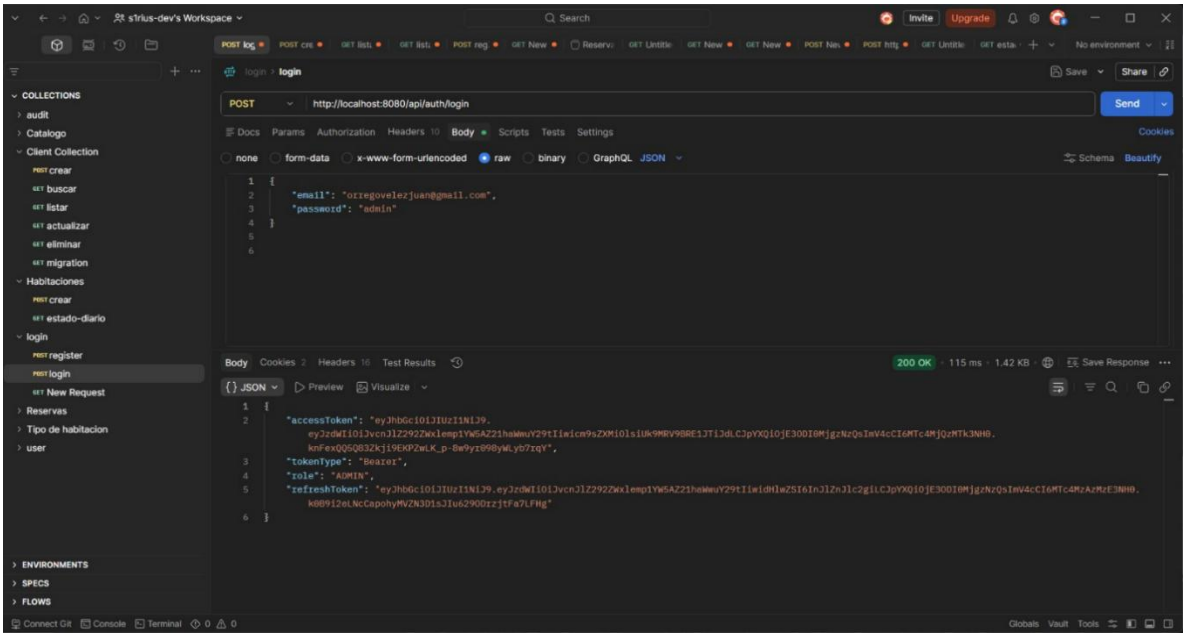



*Figura 53 Prueba de Postman con el endpoint http:/localhost:8080/api/auth/login*

***Nota.*** En la imagen se puede demostrar que el endpoint está funcionando correctamente.

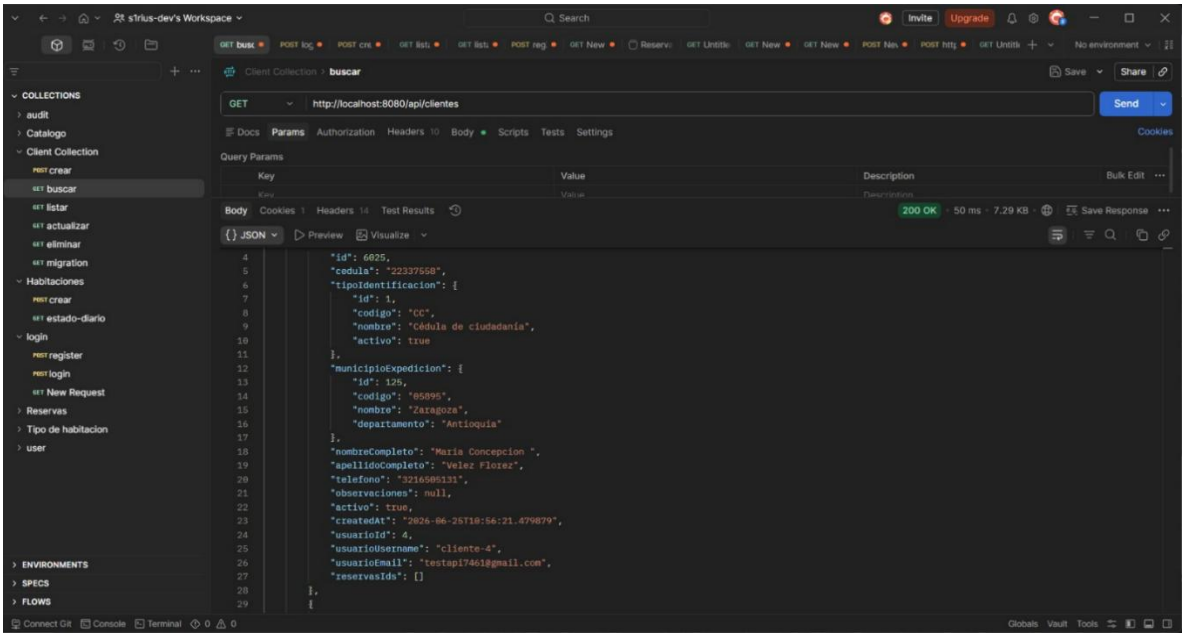



*Figura 54 Prueba de Postman con el endpoint http:/localhost:8080/api/clientes*

***Nota.*** En la imagen se puede demostrar que el endpoint está funcionando correctamente.

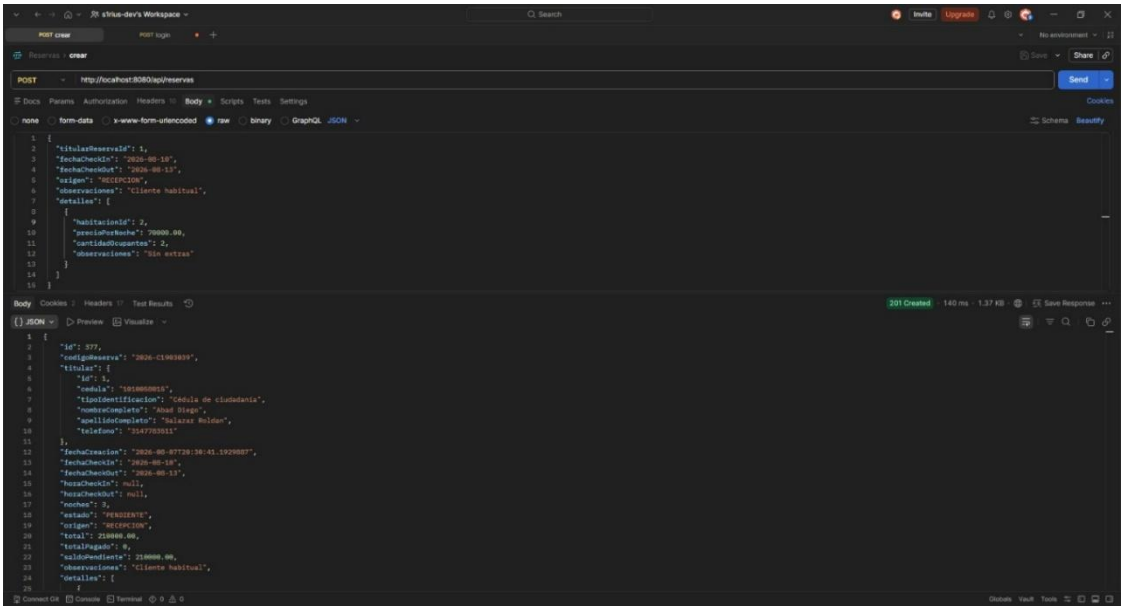



*Figura 55 Prueba de Postman con el endpoint http:/localhost:8080/api/reservas*

***Nota.*** En la imagen se puede demostrar que el endpoint está funcionando correctamente.

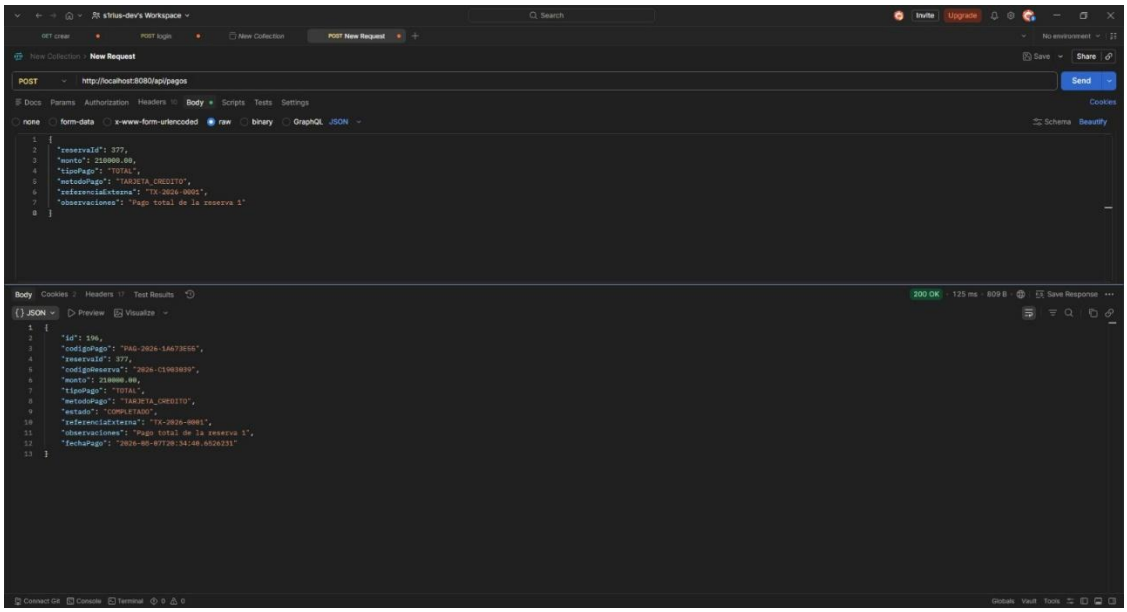



*Figura 56 Prueba de Postman con el endpoint http:/localhost:8080/api/pagos*

***Nota.*** *En la imagen se puede demostrar que el endpoint está funcionando correctamente.*

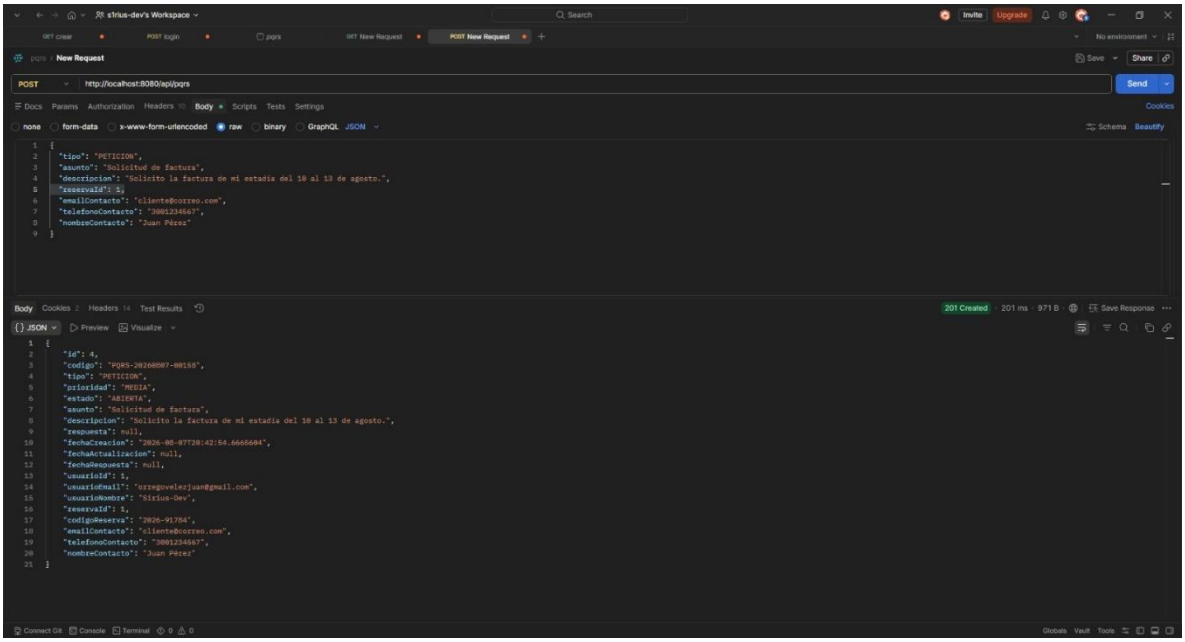



*Figura 57 Prueba de Postman con el endpoint http:/localhost:8080/api/pqrs*

***Nota.*** *En la imagen se puede demostrar que el endpoint está funcionando correctamente.*

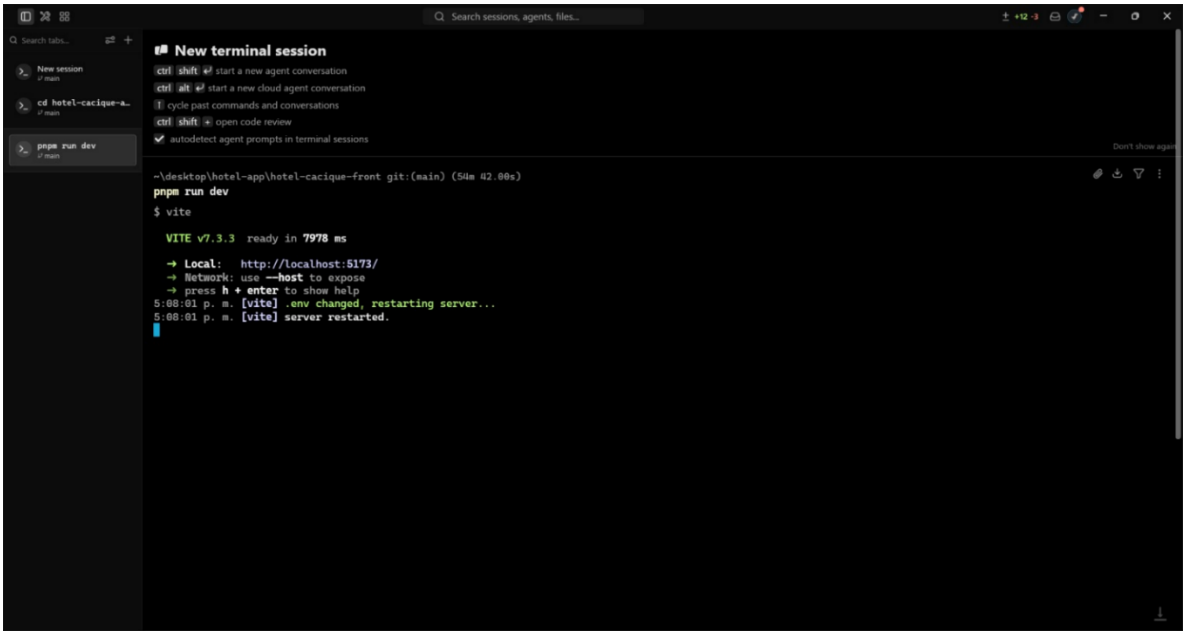



*Figura 58 Consola ejecutando npm run dev*

***Nota.*** En la imagen se puede demostrar que el proyecto de react se está ejecutando correctamente.

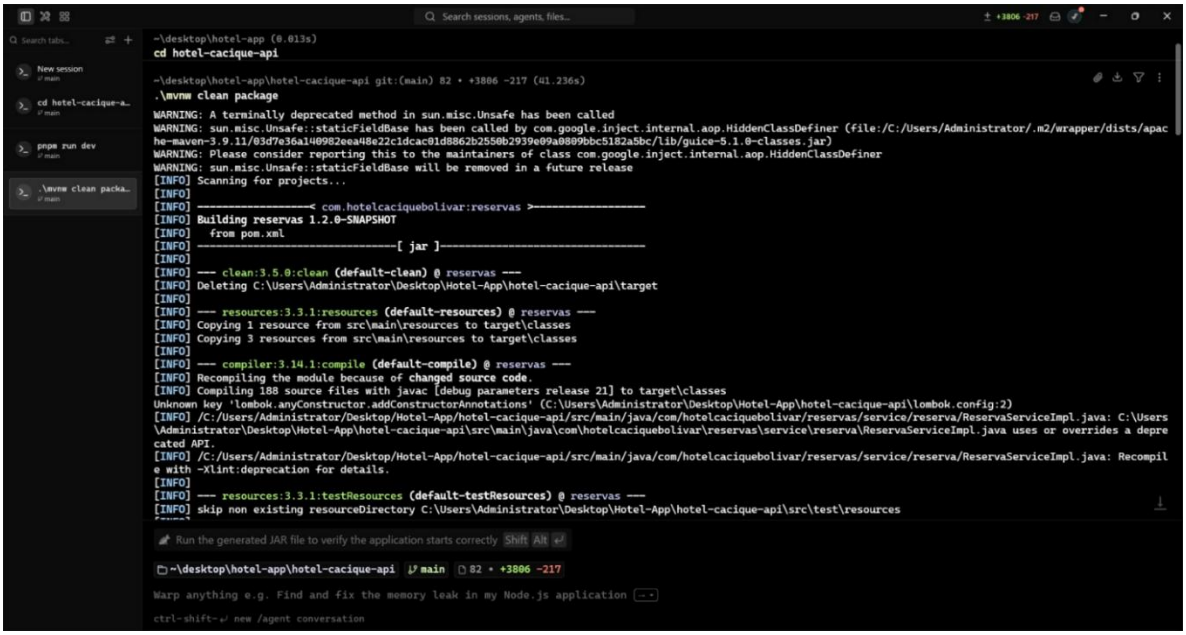



*Figura 59 Consola ejecutando mvn spring-boot:run*

***Nota.*** En la imagen se puede demostrar que el proyecto del backend se está ejecutando correctamente.

## Validación y pruebas del Sistema 7.4.

**Plan de pruebas 7.4.1.** Con el propósito de asegurar que el sistema de gestión hotelera funcionaba correctamente, se elaboró un plan de pruebas orientado a validar las funcionalidades básicas que se habían implementado durante el transcurso del desarrollo.

Las pruebas se llevaban a cabo sobre módulos críticos del sistema, donde se verificaba tanto el funcionamiento del backend como la interacción en la interfaz implementada en React.

- **Los módulos que se probaban eran los siguientes:**

- Inicio de sesión.
- Gestión de usuarios.
- Gestión de clientes.
- Gestión de habitaciones.
- Gestión de reservas.
- Gestión de pagos.
- Gestión de paquetes turísticos.
- Consulta de reportes.

Las pruebas se llevaban a cabo utilizando el sistema en una instalación local, mediante el uso de Postman para la validación de los servicios REST y a través de la interfaz web para comprobar el comportamiento desde la óptica de usuario.

**Casos de prueba 7.4.2.**

*Tabla 66 Casos de prueba del sistema*

| Caso | Módulo | Acción | Resultado esperado |
|---|---|---|---|
| **CP-001** | Inicio de session | Ingresar un usuario y contraseña válidos. | El sistema permite el acceso. |
| **CP-002** | Inicio de session | Ingresar un usuario inexistente. | El sistema muestra un mensaje indicando que el usuario no existe. |
| **CP-003** | Inicio de session | Ingresar una contraseña incorrecta. | El sistema informa que las credenciales son inválidas. |
| **CP-004** | Clientes | Buscar un cliente registrado. | El sistema muestra la información del cliente. |
| **CP-005** | Clientes | Buscar un cliente inexistente. | El sistema informa que el cliente no existe. |
| **CP-006** | Habitaciones | Consultar una habitación registrada. | El sistema muestra la información de la habitación. |
| **CP-007** | Habitaciones | Consultar una habitación inexistente. | El sistema informa que la habitación no existe. |

| | | | |
|---|---|---|---|
| **CP-008** | Reservas | Buscar una reserva registrada. | El sistema muestra la información de la reserva. |
| **CP-009** | Reservas | Buscar una reserva inexistente. | El sistema informa que la reserva no existe. |
| **CP-010** | Usuarios | Buscar un usuario registrado. | El sistema muestra la información del usuario. |
| **CP-011** | Usuarios | Buscar un usuario inexistente. | El sistema informa que el usuario no existe. |
| **CP-012** | API REST | Realizar una petición a un endpoint válido. | El sistema responde correctamente con los datos solicitados. |
| **CP-013** | Paquetes Turísticos | Crear un paquete turístico | El sistema registra correctamente el paquete. |
| **CP-014** | Paquetes Turísticos | Consultar un paquete existente | El sistema muestra la información del paquete. |
| **CP-015** | Paquetes Turísticos | Modificar un paquete | El sistema actualiza la información correctamente. |
| **CP-016** | PQRS+ | Registrar una PQRS+ | El sistema almacena correctamente la solicitud. |

| | | | |
|---|---|---|---|
| **CP-017** | PQRS+ | Consultar una PQRS+ | El sistema muestra la información registrada. |
| **CP-018** | Pagos | Registrar un pago | El sistema registra el pago exitosamente. |
| **CP-019** | Informes | Generar informe de reservas | El sistema genera el informe solicitado. |
| **CP-020** | Consultas | Consultar disponibilidad | El sistema muestra las habitaciones disponibles. |
| **CP-021** | Inicio de session | Intentar iniciar sesión dejando usuario y contraseña vacíos. | El sistema solicita completar los campos obligatorios y no permite el acceso. |
| **CP-022** | Inicio de session | Intentar iniciar sesión utilizando credenciales inválidas. | El sistema rechaza el acceso e informa que las credenciales no son válidas. |
| **CP-023** | Clientes | Intentar registrar un cliente dejando campos obligatorios vacíos. | El sistema muestra mensajes de validación y no permite registrar el cliente. |
| **CP-024** | Clientes | Ingresar un número de documento con formato inválido. | El sistema muestra un mensaje de validación y no permite guardar la información incorrecta. |

| | | | |
|---|---|---|---|
| **CP-025** | Habitaciones | Intentar registrar una habitación dejando campos obligatorios vacíos. | El sistema solicita completar los campos requeridos y no permite guardar el registro. |
| **CP-026** | Habitaciones | Ingresar información inválida en los datos de la habitación. | El sistema muestra un mensaje de validación y no permite registrar la información incorrecta. |
| **CP-027** | Reservas | Intentar realizar una reserva sin seleccionar las fechas requeridas. | El sistema solicita ingresar las fechas necesarias y no permite continuar con la reserva. |
| **CP-028** | Reservas | Ingresar una fecha de salida anterior a la fecha de ingreso. | El sistema informa que las fechas de la reserva no son válidas y no permite continuar. |
| **CP-029** | Reservas | Intentar reservar una habitación que no se encuentre disponible. | El sistema informa que la habitación no está disponible para las fechas seleccionadas. |
| **CP-030** | Paquetes Turísticos | Intentar registrar un paquete turístico dejando campos obligatorios vacíos. | El sistema solicita completar los campos requeridos y no permite guardar el paquete. |

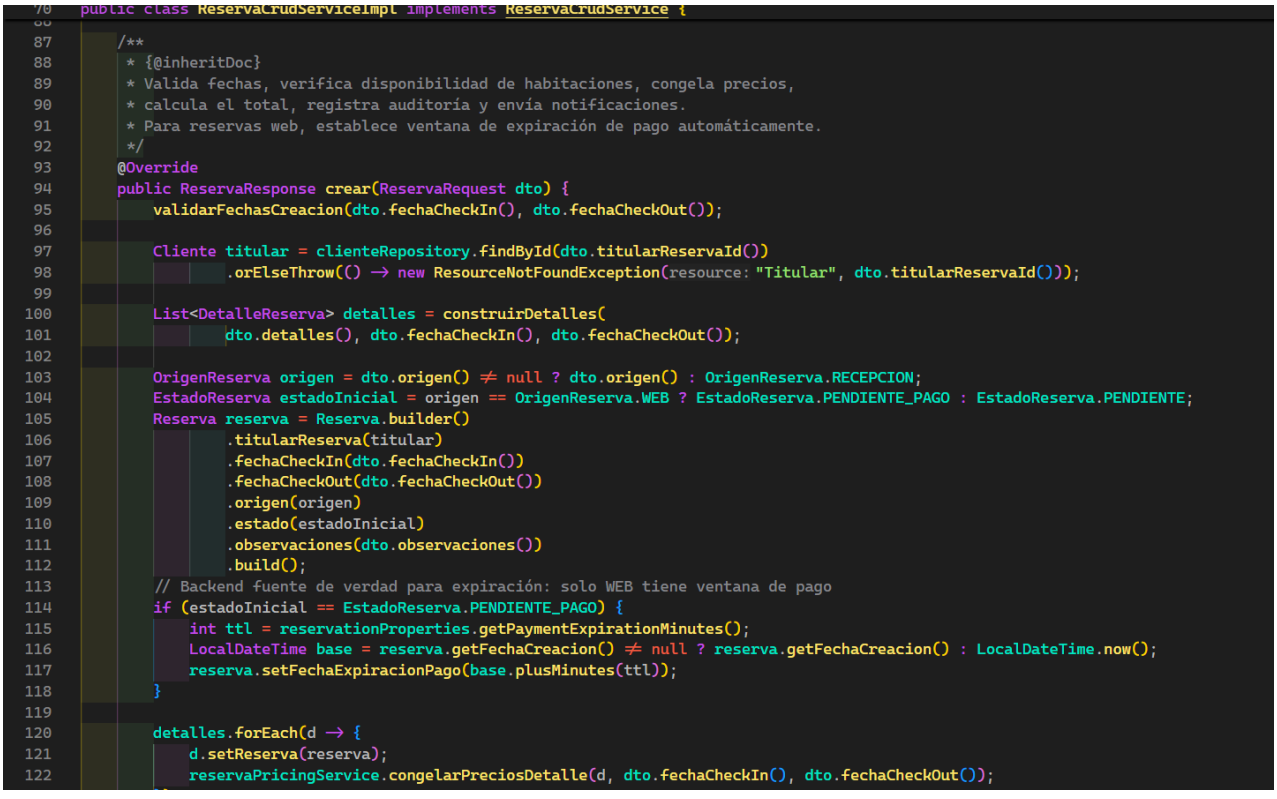
| | | | |
|---|---|---|---|
| **CP-031** | Paquetes Turísticos | Ingresar un costo inválido para un paquete turístico. | El sistema muestra un mensaje de validación y no permite registrar el costo incorrecto. |
| **CP-032** | PQRS+ | Intentar registrar una PQRS+ sin ingresar la descripción de la solicitud. | El sistema solicita ingresar la descripción y no permite registrar la PQRS+. |
| **CP-033** | PQRS+ | Intentar registrar una PQRS+ sin seleccionar el tipo de solicitud. | El sistema solicita seleccionar el tipo de PQRS+ y no permite registrar la solicitud. |
| **CP-034** | Pagos | Intentar registrar un pago sin ingresar el monto. | El sistema solicita ingresar el monto correspondiente y no permite guardar el pago. |
| **CP-035** | Pagos | Ingresar un monto de pago inválido. | El sistema muestra un mensaje de validación y no permite registrar el pago. |
| **CP-036** | Informes | Intentar generar un informe sin seleccionar el rango de fechas requerido. | El sistema solicita seleccionar las fechas necesarias y no genera el informe. |

| | | | |
|---|---|---|---|
| **CP-037** | Consultas | Realizar una consulta de disponibilidad sin ingresar los datos requeridos. | El sistema solicita completar los datos necesarios antes de realizar la consulta. |
| **CP-038** | Usuarios | Intentar registrar un usuario sin ingresar la contraseña. | El sistema solicita ingresar la contraseña y no permite registrar el usuario. |
| **CP-039** | Usuarios | Intentar registrar un usuario dejando campos obligatorios vacíos. | El sistema muestra los mensajes de validación correspondientes y no permite guardar el usuario. |
| **CP-040** | Perfil | Intentar registrar un perfil sin ingresar el nombre del perfil. | El sistema solicita completar el nombre del perfil y no permite registrar el perfil. |

***Nota*.** Los casos de prueba fueron diseñados para validar el correcto funcionamiento de las funcionalidades principales del sistema de gestión hotelera, verificando el cumplimiento de los requisitos funcionales establecidos durante la fase de análisis y diseño.

**Pruebas de caja blanca 7.4.3.**

- **Método crear()**
    - **Objetivo:** Verificar mediante la técnica de prueba de caja blanca el flujo lógico del método crear(), identificando las decisiones presentes, los caminos independientes y validando el correcto funcionamiento del proceso de creación de reservas.
    - **Código analizado**



```
public class ReservaCrudServiceImpl implements ReservaCrudService {

    /**
     * {@inheritDoc}
     * Valida fechas, verifica disponibilidad de habitaciones, congela precios,
     * calcula el total, registra auditoría y envía notificaciones.
     * Para reservas web, establece ventana de expiración de pago automáticamente.
     */
    @Override
    public ReservaResponse crear(ReservaRequest dto) {
        validarFechasCreacion(dto.fechaCheckIn(), dto.fechaCheckOut());

        Cliente titular = clienteRepository.findById(dto.titularReservaId())
                .orElseThrow(() -> new ResourceNotFoundException(resource: "Titular", dto.titularReservaId()));

        List<DetalleReserva> detalles = construirDetalles(
                dto.detalles(), dto.fechaCheckIn(), dto.fechaCheckOut());

        OrigenReserva origen = dto.origen() != null ? dto.origen() : OrigenReserva.RECEPCION;
        EstadoReserva estadoInicial = origen == OrigenReserva.WEB ? EstadoReserva.PENDIENTE_PAGO : EstadoReserva.PENDIENTE;
        Reserva reserva = Reserva.builder()
                .titularReserva(titular)
                .fechaCheckIn(dto.fechaCheckIn())
                .fechaCheckOut(dto.fechaCheckOut())
                .origen(origen)
                .estado(estadoInicial)
                .observaciones(dto.observaciones())
                .build();
        // Backend fuente de verdad para expiración: solo WEB tiene ventana de pago
        if (estadoInicial == EstadoReserva.PENDIENTE_PAGO) {
            int ttl = reservationProperties.getPaymentExpirationMinutes();
            LocalDateTime base = reserva.getFechaCreacion() != null ? reserva.getFechaCreacion() : LocalDateTime.now();
            reserva.setFechaExpiracionPago(base.plusMinutes(ttl));
        }

        detalles.forEach(d -> {
            d.setReserva(reserva);
            reservaPricingService.congelarPreciosDetalle(d, dto.fechaCheckIn(), dto.fechaCheckOut());
```

*Figura 60 Parte 1 de código analizado para la prueba*



```
        detalles.forEach(d -> {
            d.setReserva(reserva);
            reservaPricingService.congelarPreciosDetalle(d, dto.fechaCheckIn(), dto.fechaCheckOut());
        });
        reserva.setDetalles(detalles);
        BigDecimal fullTotal = reservaPricingService.calcularTotalReserva(reserva);
        reserva.setTotal(fullTotal);
        // Si fechaExpiracionPago se seteó con base estimada antes del PrePersist, recalibrar tras fechaCreacion definitiva
        if (estadoInicial == EstadoReserva.PENDIENTE_PAGO && reserva.getFechaCreacion() == null) {
            // fechaCreacion se asigna en PrePersist al guardar; ajustaremos tras save si hace falta
        }

        detalles.forEach(d -> d.getHabitacion().setEstado(EstadoHabitacion.RESERVADA));

        Reserva guardada = reservaRepository.save(reserva);
        // Asegurar expiración alineada a fechaCreacion real (PrePersist)
        if (guardada.getEstado() == EstadoReserva.PENDIENTE_PAGO) {
            int ttl = reservationProperties.getPaymentExpirationMinutes();
            LocalDateTime expiracionEsperada = guardada.getFechaCreacion().plusMinutes(ttl);
            if (guardada.getFechaExpiracionPago() == null || !guardada.getFechaExpiracionPago().equals(expiracionEsperada)) {
                guardada.setFechaExpiracionPago(expiracionEsperada);
                guardada = reservaRepository.save(guardada);
            }
        }

        auditService.log(AuditAction.CREATE, entity: "Reserva", guardada.getId(),
                String.format(format: "Reserva creada — %s\nHabitación: %s\nTitular: %s %s\nEstadía: %s → %s",
                        guardada.getCodigoReserva(),
                        detalles.stream().map(d -> d.getHabitacion().getNumero()).reduce((a, b) -> a + ", " + b).orElse(other: "N/A"),
                        titular.getNombreCompleto(), titular.getApellidoCompleto(),
                        guardada.getFechaCheckIn().format(DATE_FMT), guardada.getFechaCheckOut().format(DATE_FMT)));

        ActorInfo actor = notificacionMessageBuilder.obtenerActor();
```

*Figura 61 Parte 2 de código analizado para la prueba*



```
        String titularNombre = titular.getNombreCompleto() + " " + titular.getApellidoCompleto();
        Map<String, Object> notifDetalles = new HashMap<>();
        notifDetalles.put(key: "clienteNombre", titularNombre);
        notifDetalles.put(key: "codigoReserva", guardada.getCodigoReserva());
        notifDetalles.put(key: "fechaCheckIn", guardada.getFechaCheckIn().toString());
        notifDetalles.put(key: "fechaCheckOut", guardada.getFechaCheckOut().toString());
        notifDetalles.put(key: "total", guardada.getTotal());
        String origenNotif = (guardada.getOrigen() == OrigenReserva.WEB) ? "WEB" : actor.origen();
        String origenNombre = (guardada.getOrigen() == OrigenReserva.WEB) ? null : actor.origenNombre();

        String msg;
        if (guardada.getOrigen() == OrigenReserva.WEB) {
            msg = actor.prefijo() + " " + titularNombre
                    + " realizó una nueva reserva desde el sitio web. Código: " + guardada.getCodigoReserva()
                    + ". Check-in: " + guardada.getFechaCheckIn().format(DATE_FMT) + ".";
        } else {
            msg = actor.prefijo() + " " + actor.nombre()
                    + " registró una reserva para " + titularNombre
                    + ". Código: " + guardada.getCodigoReserva()
                    + ". Check-in: " + guardada.getFechaCheckIn().format(DATE_FMT) + ".";
        }
        notificacionService.notificarATodos(TipoNotificacion.NUEVA_RESERVA,
                msg,
                origenNotif, origenNombre,
                notifDetalles,
                referenciaTipo: "RESERVA", guardada.getId(),
                List.of(Rol.ADMIN, Rol.RECEPCIONISTA),
                actor.email());

        return reservaMapper.toResponse(guardada);
    }
```

*Figura 62 Parte final de código analizado para la prueba*

**Nota.** Escogimos esta parte de código no se puede implementar mejor las pruebas y dar un mejor resultado.

- **Grafo de flujo**

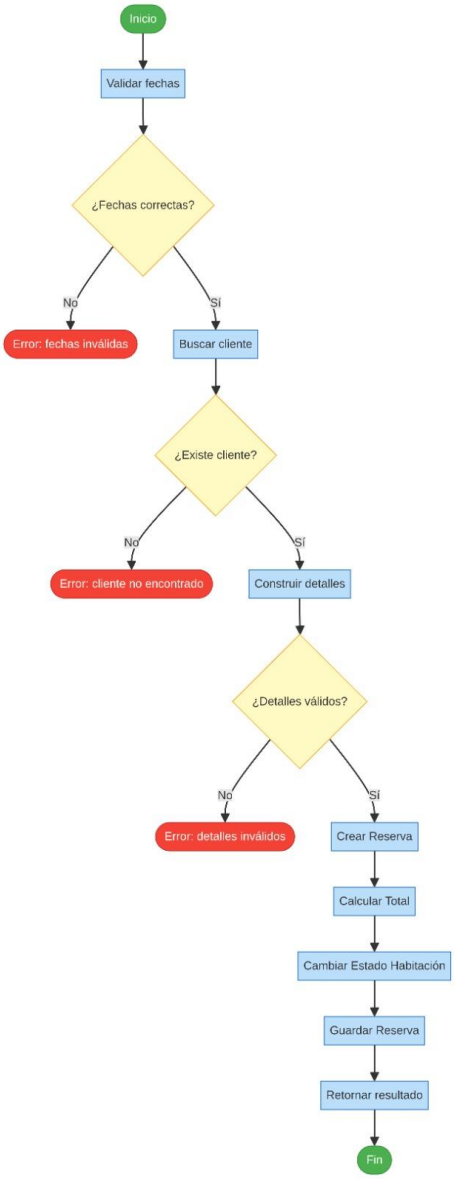



*Figura 63 Grafo de flujo*

**Nota.** Grafo de flujo utilizado en la prueba de caja blanca

- **Complejidad ciclomática**

Para calcular la complejidad ciclomática se utiliza la fórmula:

V(G) = Número de decisiones + 1

En el método se identifican tres puntos de decisión que pueden alterar el flujo de ejecución:

- ValidarFechasCreacion()
- clienteRepository.findById(...).orElseThrow(...)
- construirDetalles(...)

Total de decisiones = 3

Entonces:

V(G) = 3 + 1 = 4

Complejidad ciclomática = 4

Esto significa que se requieren 4 caminos independientes para cubrir completamente el flujo lógico del método.

- **Caminos independientes**

Durante el análisis del método crear(), se identificaron tres puntos donde el flujo de ejecución puede cambiar:

- **Camino independiente CL-1**

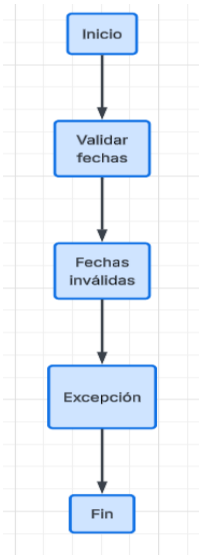



*Figura 64 Primer Camino Independiente*

**Nota.** Representa el escenario en el que el método detecta fechas inválidas y genera la excepción correspondiente

- **Camino independiente CL-2**




*Figura 65 Segundo Camino Independiente*

**Nota.** Representa el escenario en el que el cliente no existe y el sistema genera la excepción correspondiente.

- **Camino independiente CL-3**

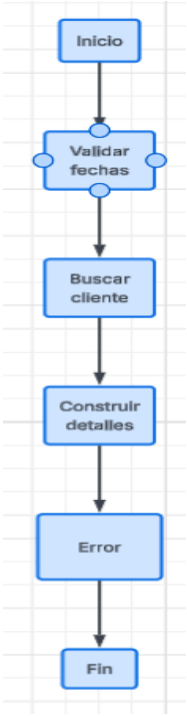



*Figura 66 Tercer Camino Independiente*

**Nota.** Representa el escenario en el que la habitación o el detalle de la reserva es inválido, impidiendo la creación de la reserva.

- **Camino independiente CL-4**

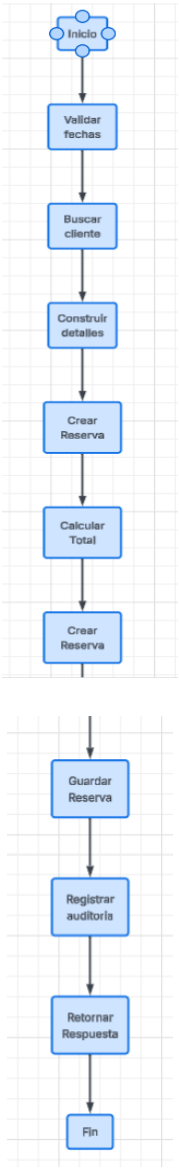



*Figura 67 Cuarto Camino Independiente*

**Nota.** Representa el flujo exitoso en el que la reserva se crea correctamente.

- **Tabla de resultado**

*Tabla 67 Tabla de Resultado*

| Caso | Camino | Entrada | Resultado esperado |
|---|---|---|---|
| CP-01 | CI-1 | Fechas inválidas | Se lanza ValidationException. |
| CP-02 | CI-2 | Cliente inexistente | Se lanza ResourceNotFoundException. |
| CP-03 | CI-3 | Habitación ocupada o detalle inválido | Se lanza la excepción correspondiente. |
| CP-04 | CI-4 | Datos válidos | Se crea la reserva correctamente. |

**Nota.** Cada caso de prueba corresponde a un camino independiente identificado en el grafo de flujo

- **Casos de prueba**

*Tabla 68 Tabla de Casos de Prueba*

| Caso | Datos de entrada | Camino | Resultado esperado |
|---|---|---|---|
| **VF-01** | checkIn = null, checkOut = null | P1 | Se lanza ValidationException indicando que las fechas son obligatorias. |
| **VF-02** | checkIn = 15/07/2026, checkOut = 10/07/2026 | P2 | Se lanza ValidationException indicando que la fecha de salida debe ser posterior o igual a la fecha de entrada. |
| **VF-03** | checkIn = ayer, checkOut = mañana | P3 | Se lanza ValidationException indicando que la fecha de ingreso no puede ser una fecha pasada. |
| **VF-04** | checkIn = hoy, checkOut = hoy + MAX_DIAS_ESTADIA + 1 | P4 | Se lanza ValidationException indicando que la estadía supera el máximo permitido. |
| **VF-05** | checkIn = hoy, checkOut = hoy + 3 días | P5 | La validación finaliza correctamente y el método permite continuar con la creación de la reserva. |

| | | | |
|---|---|---|---|
| **VF-06** | Intentar cubrir noches < 0 | P6 | No es posible ejecutar este camino porque el método ya lanzó una excepción cuando checkOut < checkIn. |

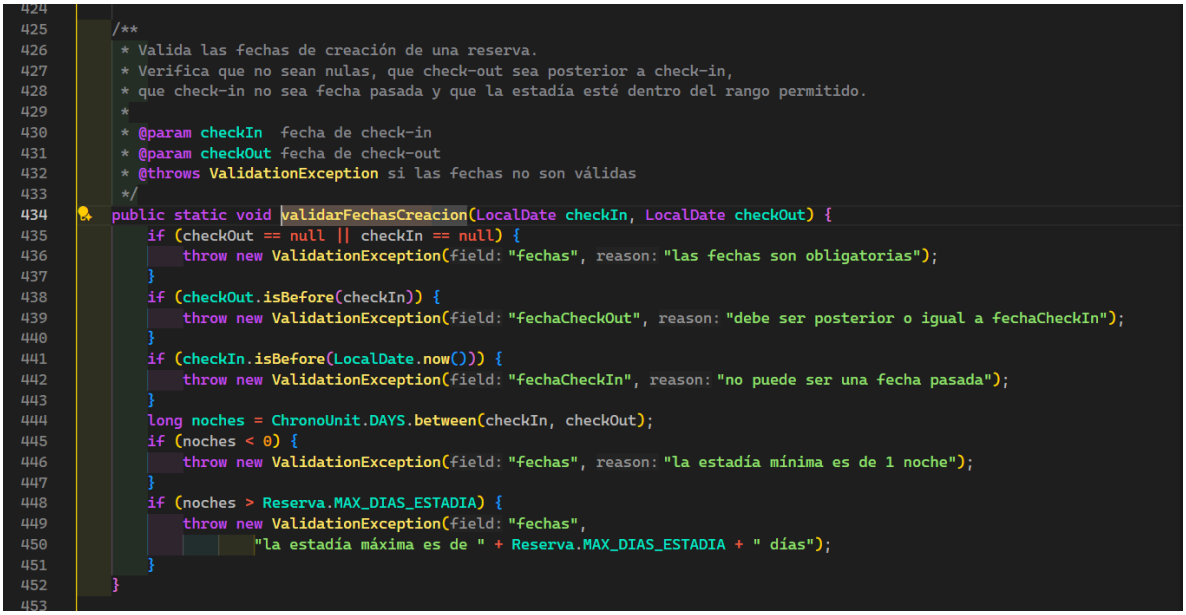
**Nota.** Los datos de entrada representan escenarios válidos e inválidos para verificar el comportamiento del sistema.

- **Método validarFechasCreacion()**

  - **Justificación**

    El método crear() está relacionado con el método validarFechasCreacion() que permite comprobar que las fechas que introduce el usuario cumplen con las reglas de negocio antes de poder realizar la creación de una reserva. Debido al hecho de que este método tiene la mayor cantidad de estructuras de decisión (if), se realizó un análisis de forma independiente y usando la técnica de caja blanca.

  - **Código analizado**



```
424
425     /**
426      * Valida las fechas de creación de una reserva.
427      * Verifica que no sean nulas, que check-out sea posterior a check-in,
428      * que check-in no sea fecha pasada y que la estadía esté dentro del rango permitido.
429      *
430      * @param checkIn  fecha de check-in
431      * @param checkOut fecha de check-out
432      * @throws ValidationException si las fechas no son válidas
433      */
434     public static void validarFechasCreacion(LocalDate checkIn, LocalDate checkOut) {
435         if (checkOut == null || checkIn == null) {
436             throw new ValidationException(field: "fechas", reason: "las fechas son obligatorias");
437         }
438         if (checkOut.isBefore(checkIn)) {
439             throw new ValidationException(field: "fechaCheckOut", reason: "debe ser posterior o igual a fechaCheckIn");
440         }
441         if (checkIn.isBefore(LocalDate.now())) {
442             throw new ValidationException(field: "fechaCheckIn", reason: "no puede ser una fecha pasada");
443         }
444         long noches = ChronoUnit.DAYS.between(checkIn, checkOut);
445         if (noches < 0) {
446             throw new ValidationException(field: "fechas", reason: "la estadía mínima es de 1 noche");
447         }
448         if (noches > Reserva.MAX_DIAS_ESTADIA) {
449             throw new ValidationException(field: "fechas",
450                     "la estadía máxima es de " + Reserva.MAX_DIAS_ESTADIA + " días");
451         }
452     }
453
```

*Figura 68 Codigo de Prueba (Validar Fecha de Creación)*

**Nota.** El código muestra cómo se implementa las validaciones necesarias para garantizar la integridad de las fechas de la reserva.

- **Grafo de flujo**

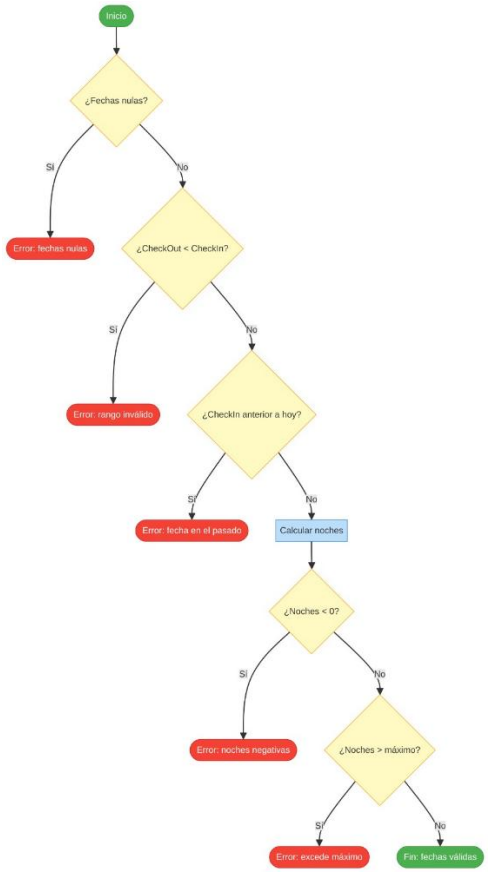



*Figura 69 Grafo de Flujo (Validación de Fecha de Creación)*

**Nota.** Grafo de flujo utilizado en la prueba de caja blanca de validación de fecha de creación.

- **Complejidad ciclomática**

Para calcular la complejidad ciclomática se utiliza la fórmula:

*V(G)=Nu´mero de decisiones+1*

En el método existen las siguientes decisiones:

1. checkOut == null || checkIn == null
2. checkOut.isBefore(checkIn)
3. checkIn.isBefore(LocalDate.now())
4. noches < 0
5. noches > Reserva.MAX_DIAS_ESTADIA

Total de decisiones = 5

Entonces:

Complejidad ciclomática = 6

Esto significa que se requieren 6 caminos independientes para cubrir completamente el flujo lógico del método.

- **Caminos independientes**
    - **Camino P1 – Fechas nulas**

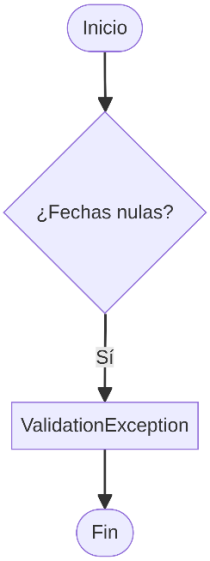



*Figura 70 Primer Camino Independiente (Validación Fecha de Creación)*

**Nota.** Representa el escenario en el que las fechas de entrada son nulas.

- **Camino P2 – Fecha de salida menor que la fecha de entrada**

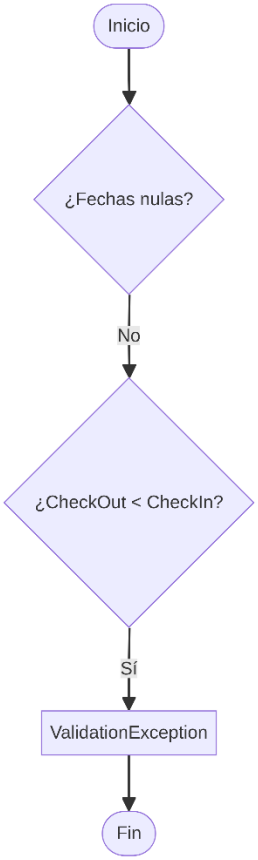



*Figura 71 Segundo Camino Independiente (Validación Fecha de Creación)*

**Nota.** Representa el escenario en el que la fecha de salida es anterior a la fecha de ingreso

- **Camino P3 – Fecha de ingreso en el pasado**

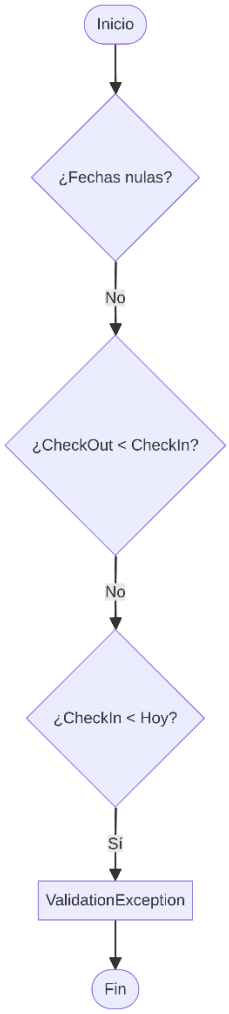



*Figura 72 Tercer Camino Independiente (Validación Fecha de Creación)*

**Nota.** Representa el escenario en el que la fecha de ingreso corresponde a una fecha pasada

- **Camino P4 – Estadía mayor al máximo permitido**

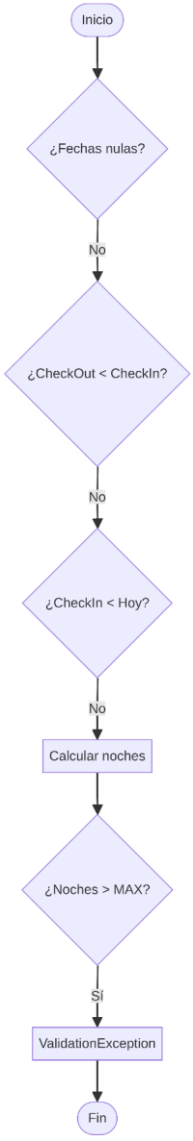



*Figura 73 Cuarto Camino Independiente (Validación Fecha de Creación)*

**Nota.** Representa el escenario en el que la duración de la estancia supera el máximo permitido.

- **Camino 5 (Fechas válidas):**

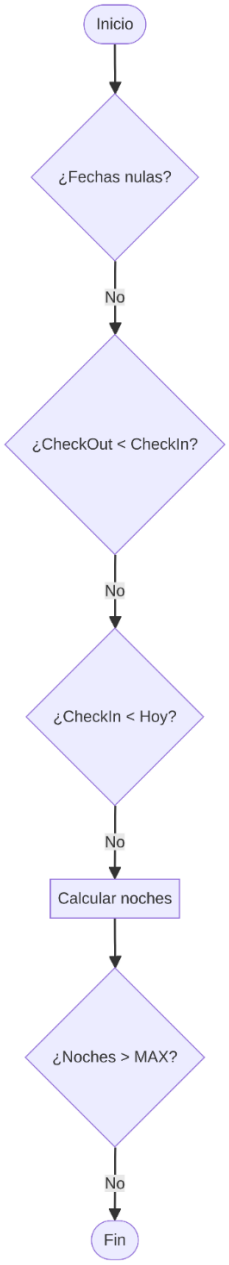



*Figura 74 Quinto Camino Independiente (Validación Fecha de Creación)*

**Nota.** Representa el flujo válido en el que todas las validaciones son superadas y el método finaliza correctamente.

- **Camino P6 – Condición noches < 0**

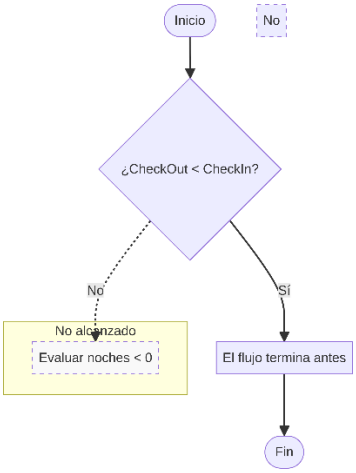



*Figura 75 Sexto Camino Independiente (Validación Fecha de Creación)*

**Nota.** Corresponde a un camino inalcanzable, ya que las validaciones previas impiden que el número de noches sea menor que cero.

- **Casos de prueba**

*Tabla 69 Casos de Prueba (Validación Fecha de Creación)*

| Caso | Datos de entrada | Camino | Resultado esperado |
|---|---|---|---|
| VF-01 | checkIn = null, checkOut = null | P1 | Se lanza ValidationException indicando que las fechas son obligatorias. |
| VF-02 | checkIn = 15/07/2026, checkOut = 10/07/2026 | P2 | Se lanza ValidationException indicando que la fecha de salida debe ser posterior o igual a la fecha de entrada. |
| VF-03 | checkIn = ayer, checkOut = mañana | P3 | Se lanza ValidationException indicando que la fecha de ingreso no puede ser una fecha pasada. |
| VF-04 | checkIn = hoy, checkOut = hoy + MAX_DIAS_ESTADIA + 1 | P4 | Se lanza ValidationException indicando que la estadía supera el máximo permitido. |
| VF-05 | checkIn = hoy, checkOut = hoy + 3 días | P5 | La validación finaliza correctamente y el método permite continuar con la creación de la reserva. |

| | | | |
|---|---|---|---|
| VF-06 | Intentar cubrir noches < 0 | P6 | No es posible ejecutar este camino porque el método ya lanzó una excepción cuando checkOut < checkIn. |

**Nota.** Los casos de prueba fueron definidos a partir de los caminos independientes identificados en el análisis de caja blanca.

- **Observación**

Durante el análisis de caja blanca se identificó que la condición:

if (noches < 0)

corresponde a un camino inalcanzable.

Esto se debe a que previamente el método valida:

if (checkOut.isBefore(checkIn))

Cuando esta condición se evalúa como true, el método inmediatamente lanza una ValidationException y termina con la ejecución. Es decir, nunca se puede llegar a calcular el número de noches con un valor negativo.

Así las cosas, la condición noches < 0 aparece como una validación redundante, dado que la misma situación se encuentra controlado por una verificación anterior. En una revisión de código, esta condición podría ser simplificada o incluso eliminada del método manteniendo el mismo comportamiento del mismo.

**Pruebas de caja negra 7.4.4.**

- **Gestión de Reservas**

  El módulo de Gestión de Reservas permite administrar el proceso de reserva de habitaciones del Hotel Cacique Bolívar. Recibe la información proporcionada por el usuario o el recepcionista, verifica la disponibilidad de las habitaciones y genera una respuesta indicando si la reserva fue realizada correctamente o si existe alguna condición que impide completar el proceso.

- **Diagrama**

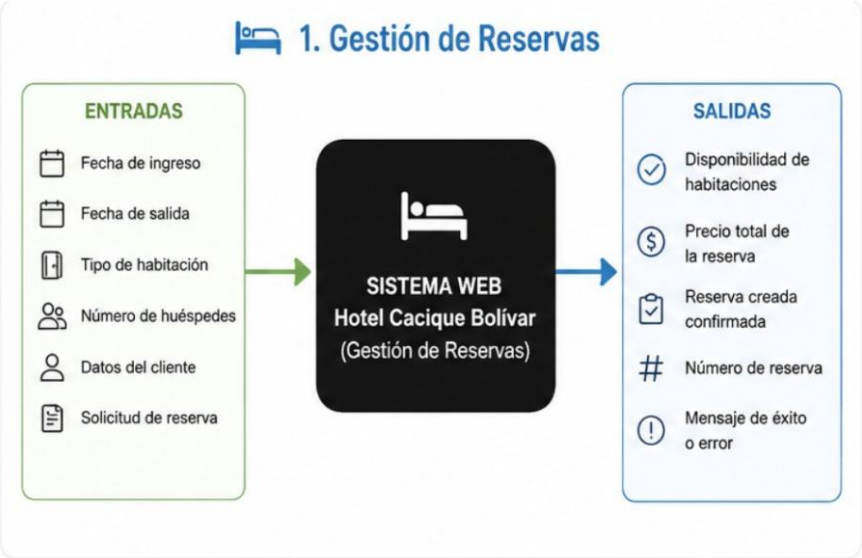



*Figura 76 Diagrama de Gestión de Reserva en Caja Negra*

**Nota.** El diagrama muestra las principales entradas y salidas del módulo de Gestión de Reservas durante la prueba de caja negra.

- **Descripción**
    - **Entradas**

*Tabla 70 Tabla de Entrada de Gestión de Reserva para la prueba de Caja Negra*

| Entrada | Descripción |
|---|---|
| Fecha de ingreso | Fecha en la que inicia la estadía del huésped. |
| Fecha de salida | Fecha en la que finaliza la estadía. |
| Tipo de habitación | Categoría de habitación seleccionada por el cliente. |
| Número de huéspedes | Cantidad de personas que ocuparán la habitación. |
| Datos del cliente | Información del cliente necesaria para registrar la reserva. |
| Solicitud de reserva | Confirmación de que el usuario desea realizar la reserva. |

**Nota.** Las entradas corresponden a los datos requeridos por el sistema para procesar una reserva

- **Salidas**

*Tabla 71 Tabla de Salida de Gestión de Reserva para la prueba de Caja Negra*

| Salida | Descripción |
|---|---|
| Disponibilidad de habitaciones | Lista de habitaciones disponibles según los criterios ingresados. |
| Precio total de la reserva | Valor calculado de acuerdo con las fechas y el tipo de habitación. |
| Reserva creada | Confirmación de que la reserva fue registrada exitosamente. |
| Número de reserva | Identificador único asignado a la reserva. |
| Mensaje de éxito o error | Información sobre el resultado del proceso. |

**Nota.** Las salidas representan los resultados esperados tras procesar correctamente la solicitud de reserva.

- **Validaciones**
    - La fecha de ingreso no puede ser anterior a la fecha actual.
    - La fecha de salida debe ser posterior a la fecha de ingreso.
    - Debe existir disponibilidad para el tipo de habitación solicitado.

- Los datos obligatorios del cliente deben estar completos.
- El número de huéspedes debe ser válido para la habitación seleccionada.

- **Gestión de Clientes:** El módulo de Gestión de Clientes permite registrar, consultar y actualizar la información de los clientes del hotel. El sistema procesa la información ingresada y almacena o recupera los datos necesarios para apoyar los diferentes procesos administrativos y de reservas.

- **Diagrama**

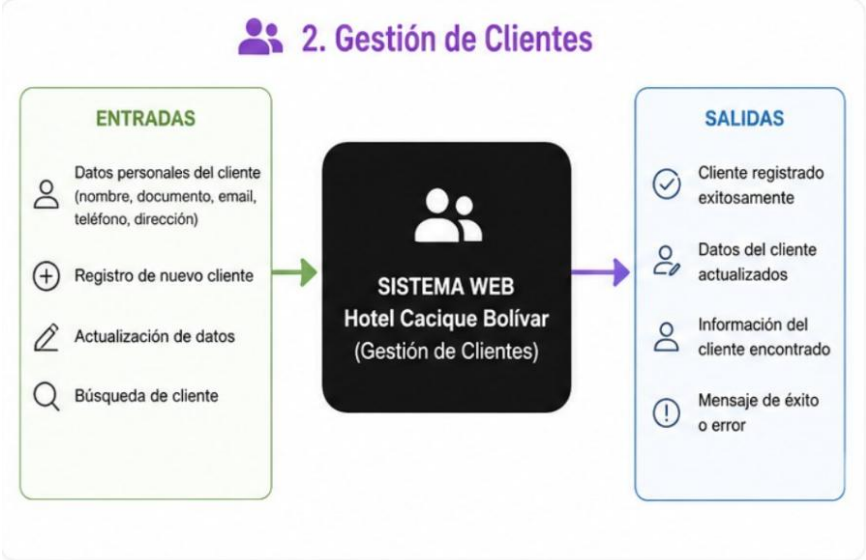



*Figura 77 Diagrama de Gestión de cliente en Caja Negra*

**Nota.** El diagrama muestra las principales entradas y salidas del módulo de Gestión de Clientes durante la prueba de caja negra.

- **Descripción**
    - **Entradas**

*Tabla 72 Tabla de Entrada de Gestión de Cliente para la prueba de Caja Negra*

| Entrada | Descripción |
|---|---|
| Datos personales del cliente | Nombre, documento, correo electrónico, teléfono y dirección. |
| Registro de nuevo cliente | Solicitud para crear un nuevo registro. |
| Actualización de datos | Información modificada de un cliente existente. |
| Búsqueda de cliente | Documento, nombre u otro criterio de búsqueda. |

**Nota.** Las entradas corresponden a los datos requeridos para registrar, actualizar y consultar la información de los clientes.

- **Salidas**

*Tabla 73 Tabla de Salidas de Gestión de Cliente para la prueba de Caja Negra*

| Salida | Descripción |
|---|---|
| Cliente registrado | Confirmación del registro exitoso del cliente. |
| Datos actualizados | Confirmación de la actualización de la información. |
| Información del cliente | Datos encontrados durante la consulta. |
| Mensaje de éxito o error | Resultado del proceso realizado. |

**Nota.** Las salidas representan los resultados esperados tras ejecutar las operaciones del módulo de Gestión de Clientes.

- **Validaciones**
    - El documento de identidad no debe estar registrado previamente al crear un cliente.
    - Los campos obligatorios deben estar diligenciados.
    - El formato del correo electrónico debe ser válido.
    - El número de documento debe cumplir el formato establecido.
    - Solo se pueden actualizar clientes existentes.

- **Gestión de Pagos:** El módulo de Gestión de Clientes permite registrar, consultar y actualizar la información de los clientes del hotel. El sistema procesa la información ingresada y almacena o recupera los datos necesarios para apoyar los diferentes procesos administrativos y de reservas.

- **Diagrama**

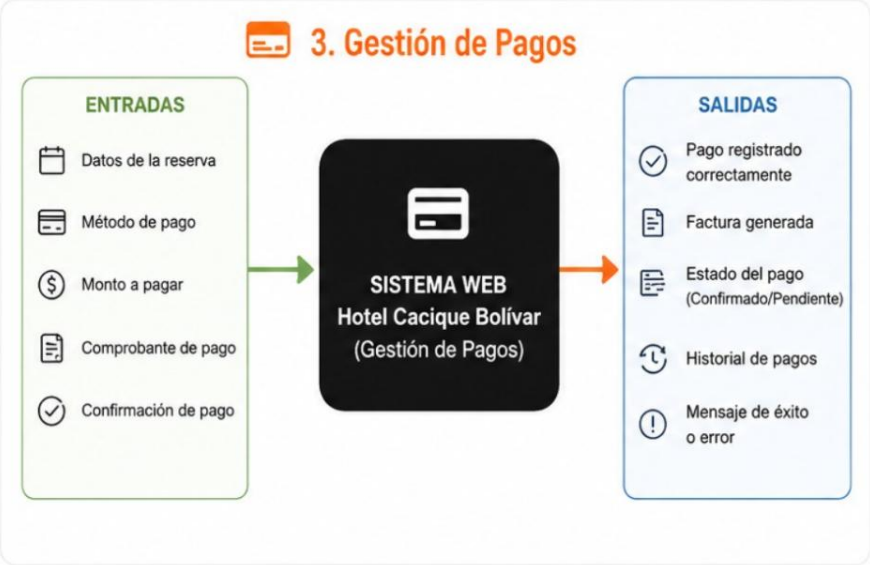



*Figura 78 Diagrama de Gestión de Pagos en Caja Negra*

**Nota.** El diagrama muestra las principales entradas y salidas del módulo de Gestión de Pagos durante la prueba de caja negra.

- **Descripción**
    - **Entradas**

*Tabla 74 Tabla de Entrada de Gestión de Pago para la prueba de Caja Negra*

| Entrada | Descripción |
|---|---|
| Datos de la reserva | Información de la reserva a la que corresponde el pago. |
| Método de pago | Medio seleccionado para realizar el pago. |
| Monto a pagar | Valor que será registrado en la transacción. |
| Comprobante de pago | Evidencia del pago cuando sea requerida. |

**Nota.** Las entradas corresponden a los datos necesarios para registrar y procesar el pago de una reserva.

    - **Salidas**

*Tabla 75 Tabla de Salidas de Gestión de Pago para la prueba de Caja Negra*

| Salida | Descripción |
|---|---|
| Pago registrado | Confirmación del registro exitoso del pago. |
| Factura generada | Documento correspondiente a la transacción realizada. |

| | |
|---|---|
| Estado del pago | Estado actualizado de la transacción. |
| Historial de pagos | Registro de los pagos asociados a la reserva. |
| Mensaje de éxito o error | Resultado del proceso. |

**Nota.** Las salidas representan los resultados esperados tras procesar correctamente el pago de una reserva.

- **Validaciones**
    - La reserva debe existir.
    - El valor del pago debe ser mayor que cero.
    - El método de pago debe ser válido.
    - El pago no debe superar el saldo pendiente.
    - La información obligatoria debe estar completa.

- **Gestión de PQRS+:** El módulo de Gestión de PQRS+ permite registrar, administrar y dar seguimiento a las peticiones, quejas, reclamos y sugerencias realizadas por los clientes. El sistema recibe la información suministrada por el usuario, registra el caso y permite realizar el seguimiento hasta su resolución.

- **Diagrama**

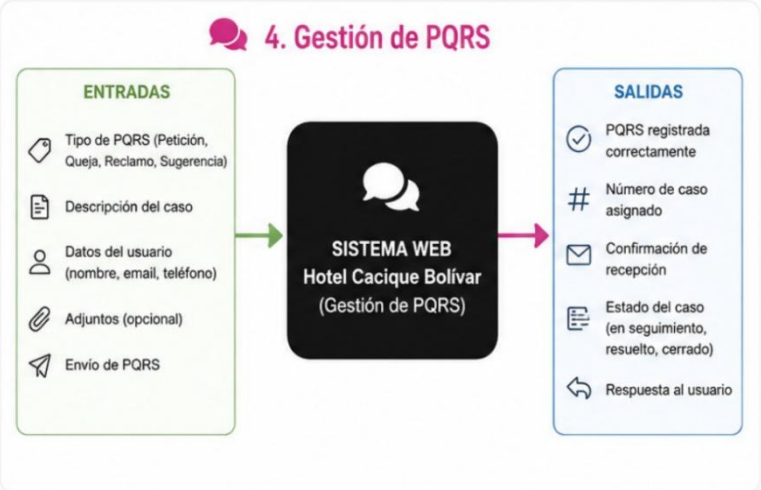



*Figura 79 Diagrama de Gestión de PQRS+ en Caja Negra*

**Nota.** El diagrama muestra las principales entradas y salidas del módulo de Gestión de PQRS+ durante la prueba de caja negra.

- **Descripción**
    - **Entradas**

*Tabla 76 Tabla de Entrada de Gestión de PQRS+ para la prueba de Caja Negra*

| Entrada | Descripción |
|---|---|
| Tipo de PQRS+ | Clasificación del caso como petición, queja, reclamo o sugerencia. |
| Descripción del caso | Información detallada del motivo de la solicitud. |
| Datos del Usuario | Información de contacto del solicitante. |
| Archivos adjuntos | Documentos o evidencias que respaldan la solicitud. |
| Envío de PQRS+ | Confirmación del registro del caso. |

**Nota.** Las entradas corresponden a la información necesaria para registrar y gestionar una solicitud de PQRS+.

- **Salidas**

*Tabla 77 Tabla de Salidas de Gestión de PQRS+ para la prueba de Caja Negra*

| Salida | Descripción |
|---|---|
| PQRS+ registrada | Confirmación de que la solicitud fue almacenada correctamente. |
| Número de caso | Identificador único asignado a la PQRS+. |
| Confirmación de recepción | Mensaje que informa que el caso fue recibido. |
| Estado del caso | Estado actualizado del proceso (registrado, en proceso, resuelto o cerrado). |
| Respuesta al usuario | Respuesta emitida por el personal encargado. |

**Nota.** Las salidas representan los resultados esperados tras registrar y gestionar una solicitud de PQRS+.

- **Validaciones**
    - Debe seleccionarse un tipo de PQRS+.
    - La descripción del caso es obligatoria.
    - Los datos de contacto deben estar completos.
    - Los archivos adjuntos deben cumplir los formatos permitidos.
    - El caso debe quedar asociado a un número único para su seguimiento.

**Resultados generales 7.4.5.** Una vez ejecutados los casos de prueba desarrollados en el plan de pruebas se corroboró que las funcionalidades principales del sistema de gestión hotelera funcionaban correctamente. Los resultados obtenidos evidencian que los módulos desarrollados conforman las especificaciones funcionales que se confeccionaron durante la fase de análisis-diseño.

*Tabla 78 Resultados de los casos de prueba*

| Caso de Prueba | Módulo | Resultado Esperado | Resultado Obtenido | Estado |
|---|---|---|---|---|
| **CP-001** | Inicio de session | Permitir el acceso con credenciales válidas. | El sistema permitió el acceso correctamente. | Aprobado |
| **CP-002** | Inicio de session | Mostrar mensaje de usuario inexistente. | El sistema notificó que el usuario no existe. | Aprobado |
| **CP-003** | Inicio de session | Mostrar mensaje de contraseña incorrecta. | El sistema informó que las credenciales son inválidas. | Aprobado |
| **CP-004** | Clientes | Mostrar la información del cliente registrado. | El sistema presentó correctamente los datos del cliente. | Aprobado |

| | | | | |
|---|---|---|---|---|
| **CP-005** | Clientes | Informar que el cliente no existe. | El sistema mostró el mensaje correspondiente. | Aprobado |
| **CP-006** | Habitaciones | Mostrar la información de la habitación registrada. | El sistema presentó correctamente la información de la habitación. | Aprobado |
| **CP-007** | Habitaciones | Informar que la habitación no existe. | El sistema mostró el mensaje correspondiente. | Aprobado |
| **CP-008** | Reservas | Mostrar la información de la reserva registrada. | El sistema presentó correctamente la información de la reserva. | Aprobado |
| **CP-009** | Reservas | Informar que la reserva no existe. | El sistema mostró el mensaje correspondiente. | Aprobado |
| **CP-010** | Usuarios | Mostrar la información del | El sistema presentó correctamente la | Aprobado |

| | | | | |
|---|---|---|---|---|
| | | usuario registrado. | información del usuario. | |
| **CP-011** | Usuarios | Informar que el usuario no existe. | El sistema mostró el mensaje correspondiente. | Aprobado |
| **CP-012** | API REST | Responder correctamente a un endpoint válido. | La API respondió exitosamente con la información solicitada. | Aprobado |
| **CP-013** | Paquetes Turísticos | Registrar correctamente un paquete turístico. | El sistema registró el paquete exitosamente. | Aprobado |
| **CP-014** | Paquetes Turísticos | Consultar un paquete existente. | El sistema mostró correctamente la información del paquete. | Aprobado |
| **CP-015** | Paquetes Turísticos | Modificar un paquete turístico. | El sistema actualizó correctamente la | Aprobado |

| | | | | |
|---|---|---|---|---|
| | | | información del paquete. | |
| **CP-016** | PQRS+ | Registrar una PQRS+. | El sistema almacenó correctamente la solicitud. | Aprobado |
| **CP-017** | PQRS+ | Consultar una PQRS+ registrada. | El sistema presentó correctamente la información de la PQRS+. | Aprobado |
| **CP-018** | Pagos | Registrar un pago. | El sistema registró correctamente el pago. | Aprobado |
| **CP-019** | Informes | Generar un informe de reservas. | El sistema generó correctamente el informe solicitado. | Aprobado |
| **CP-020** | Consultas | Consultar disponibilidad de habitaciones. | El sistema mostró correctamente las habitaciones disponibles. | Aprobado |

| | | | | |
|---|---|---|---|---|
| **CP-021** | Inicio de session | Mostrar un mensaje solicitando completar los campos obligatorios cuando usuario y contraseña estén vacíos. | El sistema muestra un mensaje diciendo que estos campos son obligatorios. | Aprobado |
| **CP-022** | Inicio de session | Impedir el acceso cuando se ingresen credenciales con datos inválidos. | El sistema muestra un mensaje diciendo que correo o contraseña incorrectos. | Aprobado |
| **CP-023** | Clientes | Impedir el registro de un cliente cuando se dejen campos obligatorios vacíos. | El sistema muestra un mensaje diciendo que estos campos son obligatorios. | Aprobado |

| | | | | |
|---|---|---|---|---|
| **CP-024** | Clientes | Mostrar un mensaje de validación cuando se ingrese un número de documento con formato inválido. | El sistema muestra un mensaje de validación cuando la cedula contiene letras o es menor a 6 caracteres | Aprobado |
| **CP-025** | Habitaciones | Impedir el registro de una habitación cuando se deje un campo obligatorio vacío. | El sistema muestra un mensaje diciendo que el numero debe tener al menos un carácter. | Aprobado |
| **CP-026** | Habitaciones | Mostrar un mensaje de validación cuando se ingrese una capacidad inválida. | El sistema muestra un mensaje diciendo que la capacidad mínima es de 1 | Aprobado |

| | | | | |
|---|---|---|---|---|
| **CP-027** | Reservas | Impedir el registro de una reserva cuando no se ingresen las fechas requeridas. | No se puede hacer, ya que no permite escoger fechas de los días anteriores al día actual. | Aprobado |
| **CP-028** | Reservas | Mostrar un mensaje de validación cuando la fecha de salida sea anterior a la fecha de ingreso. | No se puede hacer ya que el sistema no permite seleccionar fechas de salida que sean anteriores a la de ingreso. | Aprobado |
| **CP-029** | Reservas | Impedir la reserva de una habitación que no se encuentre disponible para las fechas seleccionadas. | El sistema rechazó la reserva y mostró el mensaje de habitación no disponible. | Aprobado |

| | | | | |
|---|---|---|---|---|
| **CP-030** | Paquetes Turísticos | Impedir el registro de un paquete turístico cuando se dejen campos obligatorios vacíos. | El sistema muestra un mensaje diciendo que estos campos son obligatorios. | Aprobado |
| **CP-031** | Paquetes Turísticos | Mostrar un mensaje de validación cuando se ingrese un costo inválido. | El sistema muestra un mensaje diciendo que el costo no puede tener un valor negativo | Aprobado |
| **CP-032** | PQRS+ | Impedir el registro de una PQRS+ cuando la descripción se encuentre vacía. | El sistema muestra un mensaje diciendo que la descripción debe tener al menos 20 caracteres | Aprobado |
| **CP-033** | PQRS+ | Impedir el registro de una PQRS+ cuando | El sistema muestra un mensaje diciendo que es obligatorio | Aprobado |

| | | | | |
|---|---|---|---|---|
| | | no se seleccione el tipo de solicitud. | seleccionar el tipo de solicitud. | |
| **CP-034** | Pagos | Impedir el registro de un pago cuando no se ingrese el monto correspondiente. | El sistema muestra un mensaje que dice que debe ingresar un monto para registrar el pago | Aprobado |
| **CP-035** | Pagos | Mostrar un mensaje de validación cuando se ingrese un monto de pago inválido. | El sistema muestra un mensaje diciendo que el monto debe ser mayor a cero. | Aprobado |
| **CP-036** | Informes | Solicitar el rango de fechas cuando se intente generar un informe sin | El sistema muestra un mensaje diciendo que debe ingresar un rango de fechas para poder generar el informe | Aprobado |

| | | | | |
|---|---|---|---|---|
| | | seleccionar las fechas requeridas. | | |
| **CP-037** | Consultas | Solicitar los datos requeridos cuando se realice una consulta de disponibilidad sin seleccionar una fecha. | El sistema muestra un mensaje diciendo que por favor ingrese una fecha para consultar la disponibilidad | Aprobado |
| **CP-038** | Usuarios | Impedir el registro de un usuario cuando se deje la contraseña vacía. | El sistema impidió el registro y mostrando la validación correspondiente. | Aprobado |
| **CP-039** | Usuarios | Impedir el registro de un usuario cuando se dejen campos obligatorios vacíos. | El sistema muestra que los campos son obligatorios | Aprobado |

| | | | | |
|---|---|---|---|---|
| **CP-040** | Perfil | Impedir el registro de un perfil cuando el nombre del perfil se encuentre vacío. | El sistema impide el registro y mostrando los campos que tiene que validar | Aprobado |

**Nota.** Los casos de prueba ejecutados, donde se contrasta el comportamiento real del sistema frente a los escenarios previstos para validar la efectividad de las funciones e interfaces implementadas.

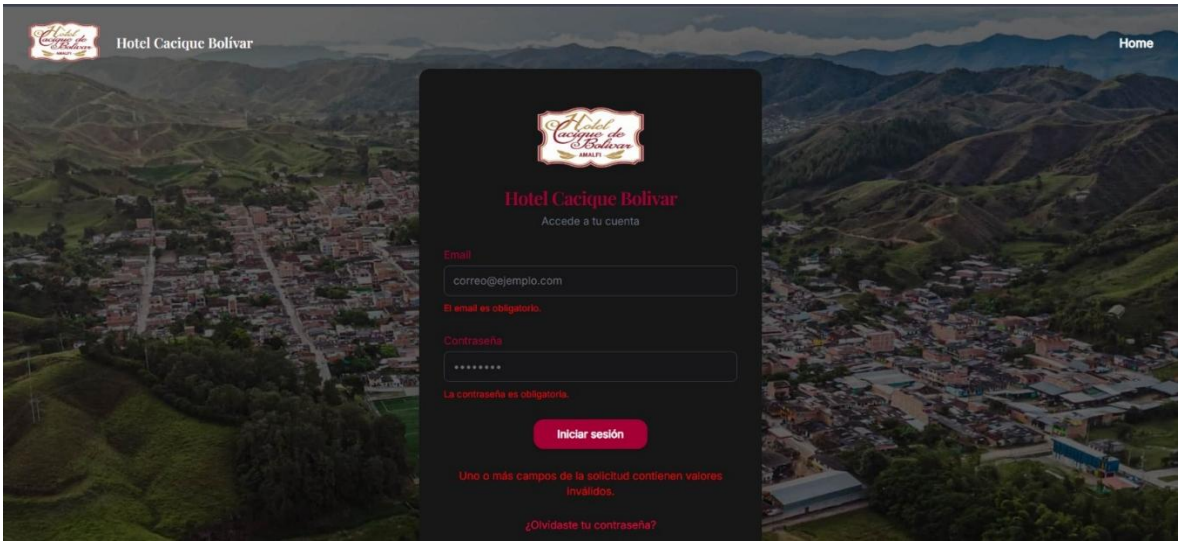



*Figura 80 CP-021 Caso de prueba de inicio de sesión*

***Nota.*** En la figura se puede ver el resultado del sistema donde muestra el mensaje que los capos son obligatorios.

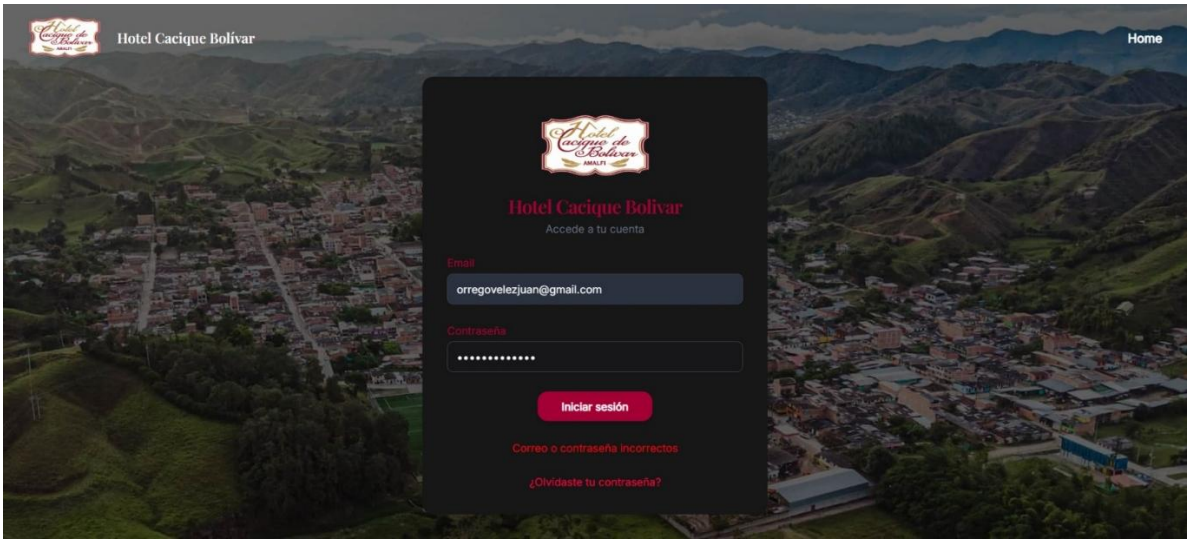



*Figura 81 CP-022 Caso de prueba de inicio de sesión*

***Nota.*** En la figura se puede ver el resultado del sistema donde muestra el mensaje que el correo o la contraseña son incorrectos.

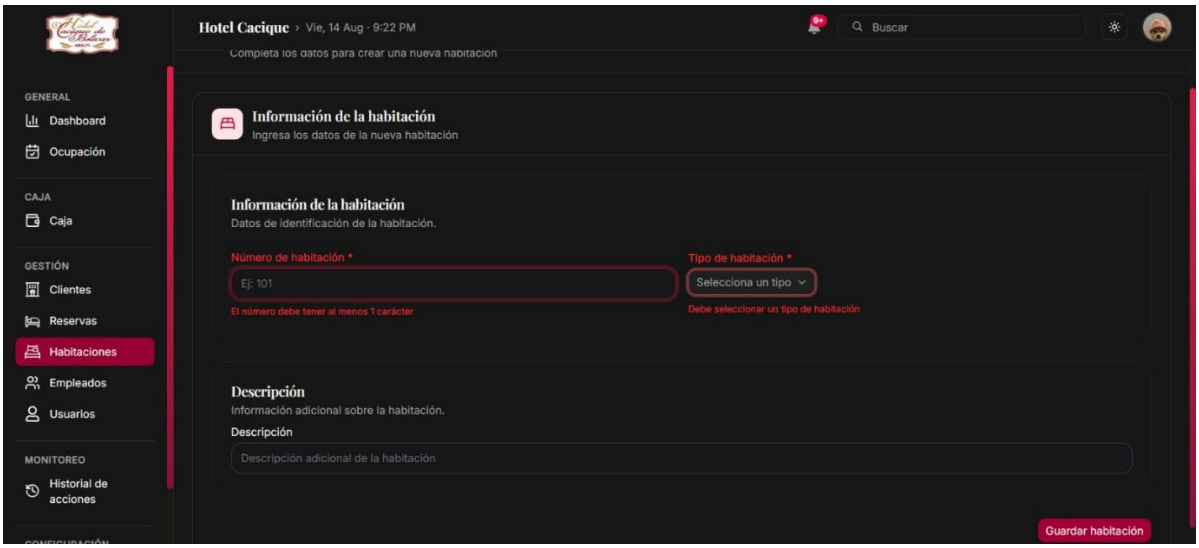



*Figura 82 CP – 025 Caso de prueba de Habitación*

***Nota.*** En la figura se puede ver el resultado del sistema donde muestra el mensaje que es obligatorio llenar los capos.

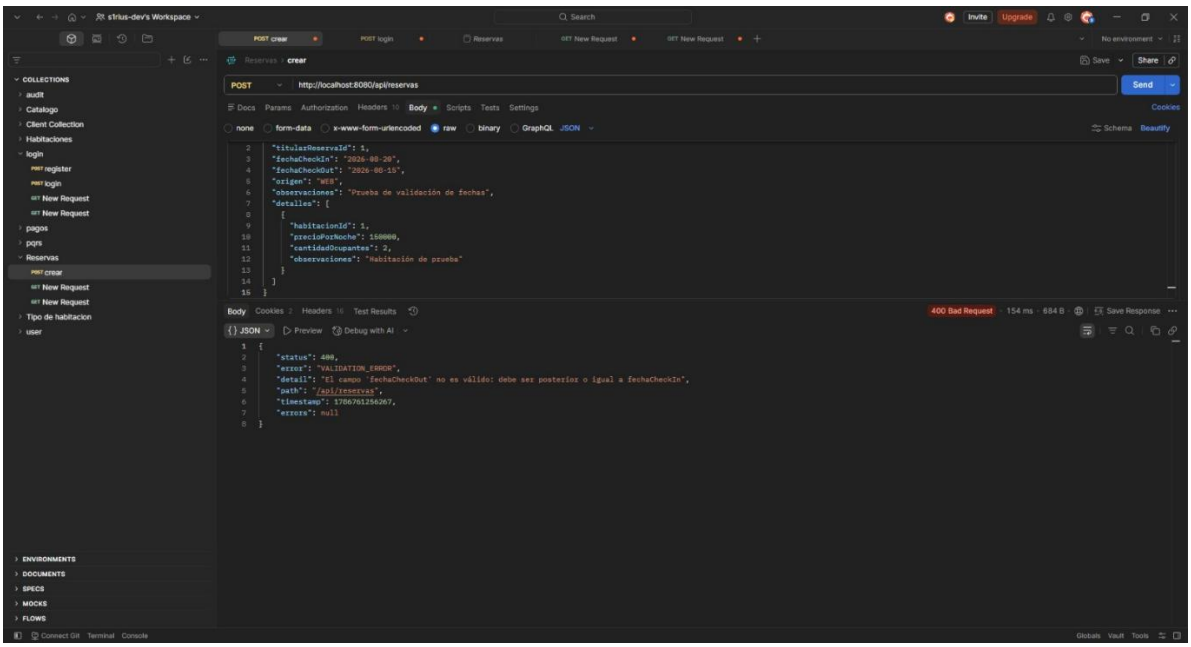



*Figura 83 CP-028 Caso de prueba de Reservas*

***Nota.*** En la figura se puede ver el resultado en postman donde muestra el mensaje que la fecha de salida no puedes ser posterior o igual a la fecha de ingreso.

- **Análisis de resultados:** De los doces casos de prueba realizados, el 100% obtuvo resultados satisfactorios, demostrando que las funcionalidades básicas del sistema funcionan de acuerdo con lo requerido. Las pruebas realizadas se demuestra una buena conexión entre el frontend y el backend

  Se llevó a cabo el cometido de la prueba de caja blanca sobre los métodos crear() y validarFechasCreacion(), obteniendo la cobertura de los caminos independientes identificados mediante la métrica de complejidad ciclomática. Las pruebas ejecutadas

permitieron verificar el correcto comportamiento del proceso de creación de las reservas. Además, se detectó una validación redundante en la condición noches < 0.

La condición que resulta inalcanzable debido a una validación previa a cualquier otra. Como conclusiones generales, los resultados obtenidos evidencian que la lógica implementada cumple con las reglas de negocio definidas para el sistema de gestión hotelera.

## Despliegue e instalación 7.5.

**Objetivo del despliegue e instalación 7.5.1.** Realizar el despliegue e instalación del sistema de gestión hotelera en un entorno de servidor, configurando los servicios necesarios para permitir su ejecución y acceso mediante la red local e Internet. Para ello, se integraron el frontend desarrollado en React, el backend desarrollado en Spring Boot y la base de datos MySQL, utilizando IIS como servidor web y Cloudflare Tunnel para establecer el acceso externo al sistema.

**Arquitectura del despliegue 7.5.2**. El sistema fue desplegado utilizando un equipo con Windows Server como servidor principal. La arquitectura implementada permite que los usuarios accedan al sistema desde la red local o desde Internet mediante el dominio configurado.

- **La estructura general del despliegue es la siguiente:**

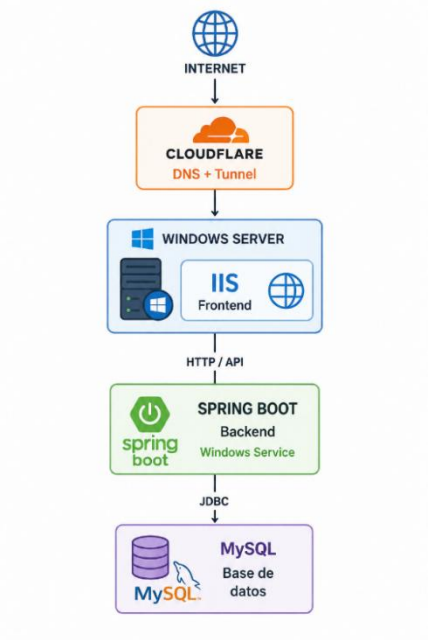



*Figura 84 Arquitectura de despliegue de la página del hotel*

**Nota.** En esta figura se puede explicar cómo se desplego la página en hotel para que las personas las puedan utilizar.

De esta manera, IIS se encarga de servir la aplicación frontend, mientras que Spring Boot proporciona los servicios del backend y realiza las operaciones necesarias sobre la base de datos.

**Configuración del servidor Windows 7.5.3**. Se utilizó un equipo con Windows Server como servidor de la aplicación. En este equipo se instalaron y configuraron los componentes necesarios para ejecutar el sistema.

- **Entre los principales componentes configurados se encuentran:**
  - Internet Information Services (IIS).
  - Java/JDK requerido por el backend.

- MySQL.
- Spring Boot.
- Aplicación frontend compilada.
- Servicio de Windows para la ejecución permanente del backend.
- Cloudflare Tunnel.
- Configuración de red y puertos necesarios para la comunicación entre los componentes.

Esta configuración permite que el sistema continúe disponible sin depender de que un usuario tenga que iniciar manualmente la aplicación desde una terminal.

**Despliegue del frontend mediante IIS 7.5.4.** El frontend del sistema fue desarrollado utilizando React. Para su despliegue se generó la versión de producción de la aplicación y posteriormente los archivos resultantes fueron ubicados en el directorio configurado para IIS.

IIS fue configurado para servir los archivos correspondientes al frontend y permitir que los usuarios accedan a la aplicación mediante el dominio establecido.

A diferencia del entorno de desarrollo, donde React se ejecuta mediante un servidor de desarrollo, en el entorno de producción se utiliza la versión compilada de la aplicación, permitiendo que IIS se encargue de entregar los recursos al navegador.

**Configuración del backend como servicio de Windows 7.5.5.** El backend fue desarrollado utilizando Spring Boot y Java. Para evitar que el servidor dependiera de una

consola abierta manualmente, la aplicación fue configurada para ejecutarse como un servicio de Windows.

Para esto se utilizó una herramienta/librería disponible en GitHub que permite registrar la aplicación Java como un servicio del sistema operativo.

- **Esta configuración permite que:**
  - Spring Boot se inicie automáticamente junto con el servidor.
  - El backend permanezca ejecutándose en segundo plano.
  - No sea necesario mantener una terminal abierta.
  - El servicio pueda iniciarse, detenerse o reiniciarse desde Windows.
  - La aplicación pueda continuar funcionando después de reiniciar el servidor.

La aplicación backend se encuentra configurada para atender las solicitudes provenientes del frontend y procesarlas mediante la API REST desarrollada en Spring Boot.

**Configuración de la base de datos 7.5.6.** La aplicación utiliza MySQL como sistema gestor de base de datos. Durante el despliegue se configuró la conexión entre Spring Boot y MySQL mediante las propiedades correspondientes del proyecto. El backend utiliza esta conexión para realizar las operaciones de consulta, creación, modificación y eliminación de información relacionada con los diferentes módulos del sistema, entre ellos clientes, habitaciones, reservas, pagos y demás entidades utilizadas por la aplicación.

**Configuración de IIS y comunicación entre frontend y backend 7.5.7.** Una vez desplegado el frontend, se configuró IIS para permitir que la aplicación pudiera comunicarse con el backend mediante las rutas de la API. El frontend realiza las solicitudes HTTP correspondientes hacia el backend desarrollado en Spring Boot, mientras que este procesa las peticiones y accede a la base de datos cuando es necesario.

- **De esta manera se mantiene separada la responsabilidad de cada componente:**
  - React: interfaz gráfica y experiencia del usuario.
  - IIS: publicación del frontend.
  - Spring Boot: lógica de negocio y API REST.
  - MySQL: almacenamiento de información.

**Configuración del acceso mediante Cloudflare 7.5.8.** Para permitir el acceso al sistema desde Internet se configuró Cloudflare junto con un túnel hacia el servidor. Cloudflare Tunnel permite establecer la comunicación entre el dominio público y el servidor local sin necesidad de exponer directamente el servidor mediante una dirección IP pública. El dominio utilizado para el sistema fue configurado para dirigir las solicitudes hacia los servicios correspondientes del servidor.

- **La comunicación puede representarse de la siguiente manera:**

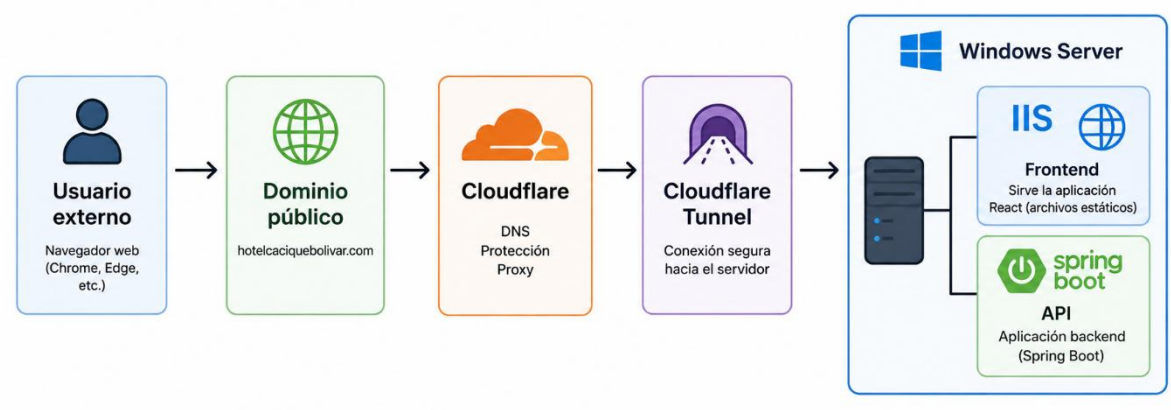



*Figura 85 Configuración del acceso mediante Cloudflare*

Esta configuración permitió publicar el sistema manteniendo el servidor dentro de la infraestructura local.

**Acceso desde la red local 7.5.9.** Además del acceso mediante Internet, el sistema fue configurado para poder ser utilizado desde dispositivos conectados a la misma red local del servidor. Los equipos de la red pueden acceder al sistema utilizando la configuración de red correspondiente, permitiendo realizar pruebas y utilizar la aplicación sin depender exclusivamente del acceso externo. Esto permitió comprobar el funcionamiento del sistema tanto dentro de la infraestructura local como mediante el dominio publicado.

**Verificación del despliegue 7.5.10.** Después de realizar la instalación y configuración de los diferentes componentes se realizaron pruebas para verificar que todos los servicios funcionaran correctamente.

- **Se verificaron principalmente los siguientes aspectos:**

    1. Inicio correcto del servidor Windows.

2. Ejecución automática del backend como servicio.
3. Funcionamiento de IIS.
4. Carga correcta del frontend.
5. Comunicación entre React y Spring Boot.
6. Comunicación entre Spring Boot y MySQL.
7. Acceso al sistema desde la red local.
8. Acceso al sistema mediante Internet.
9. Resolución correcta del dominio.
10. Funcionamiento de las principales funcionalidades del sistema.

Como parte de la verificación se realizaron operaciones sobre el sistema, incluyendo el inicio de sesión, consulta de información, gestión de clientes, habitaciones y reservas, comprobando que las operaciones realizadas desde la interfaz fueran procesadas correctamente por el backend y almacenadas en la base de datos.

**Resultado del despliegue 7.5.11.** El despliegue permitió llevar el sistema de gestión hotelera desde un entorno de desarrollo hacia un entorno de servidor funcional.

La implementación realizada permitió integrar el frontend desarrollado en React, el backend desarrollado en Spring Boot y la base de datos MySQL dentro de un servidor Windows, utilizando IIS para la publicación de la aplicación y un servicio de Windows para mantener el backend en ejecución.

Adicionalmente, mediante Cloudflare Tunnel se habilitó el acceso al sistema desde Internet a través del dominio configurado, manteniendo el servidor dentro de la red local. Con esto se comprobó que la aplicación puede ejecutarse de manera continua y ser utilizada tanto desde dispositivos conectados a la red local como desde usuarios externos mediante Internet.

## Reflexiones

**Reflexión de la Fase I – Análisis 8.1.** Se identificaron las necesidades y requerimientos del Hotel Cacique Bolívar a partir del levantamiento de información y el análisis de los procesos administrativos de la empresa en la fase de análisis de negocio. Esta etapa permitió definir las historias de usuario, los requisitos funcionales y no funcionales, así como el alcance del sistema, garantizando con ello que la respuesta a elaborar un producto cumpliese las expectativas necesarias del hotel. En este sentido, esta etapa fue fundamental en el proceso de desarrollo de un sistema de información puesto que nos ayudó a evitar generar inconsistencias, así como a hacer crecer una base para las próximas fases.

**Reflexión de la Fase II – Diseño 8.2.** Durante la etapa de diseño se modeló la arquitectura del sistema mediante la construcción de diagramas UML, el modelo entidad-relación y los prototipos de las interfaces gráficas de usuario. Esta etapa nos permitía observar cómo quedaría estructurado el sistema antes de la fase de implementación, ayudando a identificar las relaciones de las distintas partes del sistema, garantizando la coherencia entre los requisitos funcionales y la estructura de la solución de diseño y reduciendo posibles errores en las etapas posteriores al desarrollo.

**Reflexión de la Fase III – Desarrollo 8.3.** El sistema web fue implementado con Java, Spring Boot, React y MySQL en la etapa de desarrollo, en el que integran los módulos de usuarios, clientes, habitaciones, reservaciones, pagos, paquetes turísticos y PQRS+. En este stage, se ponen en práctica los conocimientos adquiridos durante la formación, y se

enfrentan los retos sobre la integración de las partes del sistema y el trabajo con el cliente, aportando al desarrollo de las competencias técnicas del equipo y sacando como output una solución funcional para la gestión hotelera.

**Reflexión de la Fase IV – Pruebas 8.4.** En la fase de la prueba se verificó el correcto funcionamiento de los distintos módulos aplicando las pruebas de tipo caja negra y de tipo caja blanca. Los resultados obtenidos nos permitieron verificar el cumplimiento de los requisitos de tipo funcional y el correcto funcionamiento del sistema ante entradas válidas y ante entradas de tipo error. Esta fase de la prueba también nos ayudó a corregir las inconsistencias detectadas en el curso del desarrollo, así como también a garantizar la calidad del software antes de la última entrega.

**Reflexión de la Fase V — Despliegue 8.5.** Durante la fase de puesta en servicio el sistema se trasladó del entorno de desarrollo a un servidor de producción, configurando también Windows, IIS y Cloudflare Tunnel para acceder a él desde Internet, haciendo así posible comprobar que la solución no solo sirve en un entorno controlado, sino que es capaz de funcionar ininterrumpidamente y ser usada por el Hotel Cacique Bolívar, en condiciones reales, por última vez cerrando así el ciclo de desarrollo del proyecto.

**Reflexión Final del Proyecto 8.6.** La creación del sistema de administración hotelera del Hotel Cacique Bolívar permitió integrar tanto los conocimientos inherentes al analizar, diseñar, programar e implementar software de un proyecto real. Así, la solución de un hotel contribuye a optimizar los procedimientos administrativos, gracias a la automatización de la gestión de clientes, habitaciones, reservas, pagos, paquetes turísticos o el módulo PQRS+,

facilitando, por tanto, la organización de la información y la atención al cliente. Pero, además, el proyecto permitió fortalecer las capacidades referidas al manejo de la técnica y la administración de grupos y evidenció la relevancia que tiene la ingeniería del software para la creación de soluciones tecnológicas en el ámbito de una necesidad real.

# Conclusiones y recomendaciones

**Conclusiones 9.1.** La implementación del sistema de gestión hotelera para el Hotel Cacique Bolívar generó el ejercicio práctico de los saberes adquiridos durante el proceso de formación académica con la finalidad de responder a las necesidades que se pudieron identificar en la gestión de la aplicación de los procesos administrativos del hotel. Gracias al levantamiento de información y al análisis de necesidades, se fueron descubriendo sus necesidades relacionadas con la administración de clientes, habitaciones, reservas, pagos, paquetes turísticos, PQRS+, y se pudo transformarlas en un conjunto de funcionalidades que se agruparon dentro de una solución tecnológica.

La solución fue diseñada a partir de una arquitectura cliente-servidor, donde Spring Boot se ocupó de la construcción del backend, React de la interfaz gráfica y el gestor de la base de datos fue MySQL. Gracias a la tratativa de los componentes anteriores se pudo construir una aplicación web estructurada donde por una parte se establece una separación de la interfaz de usuario con la lógica de negocio y la gestión de datos, lo que permite conservar el mantenimiento del sistema y poder realizar ampliaciones introduciendo nuevas características.

La utilización de la metodología de trabajo Scrum permitió organizar y realizar un seguimiento de las actividades correspondientes al desarrollo del proyecto mediante una planificación progresiva. Esta metodología permite realizar la distribución del trabajo, revisar

los avances y presentar los ajustes según las necesidades planteadas durante la actividad de desarrollo del sistema.

Se diseñaron diversos artefactos de ingeniería de software como diagramas UML, el diseño de la base de datos, la arquitectura del sistema, los casos de uso y la definición de requisitos funcionales y no funcionales, durante el diseño del sistema. Estos artefactos permitieron moldear un preámbulo sobre la implementación del sistema y atar los requerimientos identificados en el análisis y las funcionalidades que se implementaron.

La fase de pruebas permitió comprobar el funcionamiento de los distintos módulos implementados mediante distintos tipos de pruebas: prueba de caja negra y prueba de caja blanca. En particular, las pruebas sobre los métodos crear() y validarFechasCreacion() permitieron indagar sobre la lógica interna del sistema y comprobar su comportamiento sobre distintas condiciones de entrada. Durante este proceso se detectó una validación redundante relacionada con la condición de noches negativas, la cual podría simplificarse en versiones futuras para mejorar la estructura del código.

Los resultados obtenidos durante la validación de las funcionalidades desarrolladas permitieron verificar el correcto funcionamiento de las funcionalidades desarrolladas y comprobar que dichas funcionalidades cumplían con los requisitos establecidos. Del mismo modo, también el proceso de despliegue permitió trasladar la herramienta informática desde el entorno de desarrollo hacia un servidor Windows con IIS para el despliegue del frontend, con Spring Boot para la ejecución del backend, MySQL para la gestión de la información y Cloudflare Tunnel para tener acceso al sistema desde la Web.

En líneas generales, el proyecto se consideró que había cumplido los objetivos planteados al proporcionar al Hotel Cacique Bolívar una herramienta informática con el objetivo de apoyar la gestión de los principales procesos administrativos. La centralización de la información y la digitalización de procesos de clientes, habitaciones, reservas, pagos, paquetes turísticos y PQRS+ apoyan en la organización de la información y gestión de la actividad del establecimiento.

Por último, el sistema se desarrolló contemplando la posibilidad de añadir nuevas funcionalidades en futuras versiones. Entre las nuevas funcionalidades se encuentran la posibilidad de añadir módulos adicionales para la facturación, la generación de informes estadísticos, la integración con pasarelas de pago y el envío de notificaciones automáticas, en función de las necesidades que pueda tener el Hotel Cacique Bolívar en el futuro.

**Recomendaciones 9.2.** Para reducir la probabilidad de perder o corromper la base de los datos que el sistema utiliza, se recomienda hacer copias de seguridad de la base de datos de forma periódica y tener el procedimiento de recuperación de la información.

Para tener siempre actualizadas las partes tecnológicas que utilizan el sistema, se recomienda actualizar el servidor Java, Spring Boot, React, MySQL, IIS, y el resto de las herramientas que intervienen en su funcionamiento según sean las actualizaciones de seguridad compatibles a partir de la versión de toda la cadena de gruesor y de vanguardia de una pieza como el sistema informa como un mismo proceso, no sólo el más seguro, sino también el más organizado y con unos tiempos de reacción menos rabiosos hacia la manera

de informarse que la que nos ofrece la seguridad que atiene actualmente la implementación y la seguridad.

Para aumentar la durabilidad de la estructura de seguridad del sistema, se recomienda que dicho sistema adquiera progresivamente su naturaleza de seguridad, en temas que tengan que ver no sólo con la gestión de usuarios y contraseñas, sino con la gestión de permisos de acceso y la protección de la información almacenada en la base de datos

Para un trabajo futuro se sugiere examinar la posibilidad de incluir en la solución una pasarela de pagos que permita hacer pagos electrónicos directamente desde el sistema, ampliando las opciones de pago de los usuarios y reforzando así el proceso de gestión de reservas.

Igualmente, se recomienda implementar funcionalidades que hagan referencia a la posibilidad de generar reportes y estadísticas, de forma que la información almacenada sirva como soporte del análisis de la operación del hotel, así como del proceso de toma de decisiones administrativas.

Finalmente, se recomienda que se lleven a cabo capacitaciones para aquellos empleados cuya actividad esté relacionada con la utilización del sistema, así como la elaboración de una documentación que actualice tanto la operación del sistema como sus funcionalidades y procedimientos de operación y de mantenimiento, a fin de facilitar su uso y garantizar la continuidad del sistema en el tiempo.

## Referencias

Codd, E. F. (1970). A relational model of data for large shared data banks. *Communications of the ACM*, https://doi.org/10.1145/362384.362685.

Dalgiç,A.,Güler,O.,& Birdir,K. (2018). Promotion in the hospitality industry. In *The Routledge Handbook of Hospitality Marketing* (pp. (pp. 168–179)). Routledge.

Dieste, O., Juristo, N., & others. (2021). Agile requirements engineering: From user stories to software architectures. *2021 IEEE 29th International Requirements Engineering Conference* (p. https://doi.org/10.1109/RE51729.2021.00076). IEEE.

Elmasri, R., & Navathe, S. B. (2015). *Fundamentals of Database Systems (7.ª ed.).* Pearson.

Fielding, R. T. (2000). *Architectural styles and the design of network-based software architectures.* Irvine: University of California, Irvine. Retrieved from https://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm

Irani, Z. (2002). Information systems evaluation: Navigating through the problem domain. *Information & Management*. Retrieved from https://www.sciencedirect.com/...

Kotler, P., Bowen, J., & Makens, J. (2017). *Marketing para turismo.* Pearson Educación.

Laudon, K. C., & Laudon, J. P. (2016). *Sistemas de información gerencial (14.ª ed.).* Pearson.

Malan, R., Broy, M., & others. (2020). Towards the formalization of non-functional requirements in conceptual design. *Research in Engineering Design*. Retrieved from https://doi.org/10.1007/s00163-020-00345-6

Marcotte, E. (2010). Responsive Web Design. *A List Apart*. Retrieved from https://alistapart.com/article/responsive-web-design/

Montgomery, L., Fucci, D., Bouraffa, A., Scholz, L., & Maalej, W. (2022). Empirical research on requirements quality: A systematic mapping study. *Requirements Engineering*. Retrieved from https://doi.org/10.1007/s00766-021-00367-z

*n.* (n.d.).

O'Donnell, E., Smith, J., & David, J. (2000). How information systems influence user decisions: A research framework and literature review. *International Journal of Accounting Information Systems*. Retrieved from https://www.sciencedirect.com/...

Olsson, T., Sentilles, S., & Papatheocharous, E. (2022). A systematic literature review of empirical research on quality requirements. *Requirements Engineering*. Retrieved from https://doi.org/10.1007/s00766-022-00373-9

Oracle. (2024). *Java Documentation.* (Oracle, Ed.) Retrieved from https://docs.oracle.com/en/java/

Oracle. (2024). *MySQL Documentation.* (Oracle, Ed.) Retrieved from https://dev.mysql.com/doc/

Pressman, R. S., & Maxim, B. R. (2020). *Ingeniería del software: Un enfoque práctico (9.ª ed.).* McGraw-Hill.

Radford, K. J. (1974). Information systems and managerial decision making. Omega. Retrieved from https://www.sciencedirect.com/...

Rahy, S. (2021). Managing non-functional requirements in agile software development. *IET Software*. Retrieved from https://doi.org/10.1049/sfw2.12037

React. (2024). *React Documentation.* Retrieved from https://react.dev/

Schwaber, K., & Sutherland, J. (2020). *The Scrum Guide.* (Scrum.org, Ed.) Retrieved from https://scrumguides.org/

Singla, B. S., & Aggarwal, H. (2021). Effect of information architecture on the usability of a university website: A comparative study of selected websites of Punjab (India). *International Journal of Distance Education Technologies*. Retrieved from https://doi.org/10.4018/IJDET.2021010103

Sommerville, I. (2016). *Ingeniería de software (10.ª ed.).* Pearson.

Spring. (2024). *Spring Security Reference.* Retrieved from https://docs.spring.io/spring-security/reference/

Standardization, I. O. (2019). *ISO 9241-210:2019 Ergonomics of human-system interaction — Part 210: Human-centred design for interactive systems.* ISO. Retrieved from https://www.iso.org/standard/77520.html

Standardization, I. O. (2020). *NEN-EN-ISO 25065:2020: Systems and software engineering—Software product quality requirements and evaluation (SQuaRE)—Common Industry Format (CIF) for usability: User requirements specification.* ISO.

Turismo, O. M. (2023). *Transformación digital en el turismo.* Organización Mundial del Turismo. Retrieved from https://www.unwto.org/es

VMware. (2024). *Spring Boot Reference Documentation.* (VMware, Ed.) Retrieved from https://docs.spring.io/spring-boot/docs/current/reference/html/

Walls, C. (2022). *Spring in Action (6.ª ed.).* Manning Publications. Retrieved from https://books.google.com.co/books?id=2zVbEAAAQBAJ&redir_esc=y

Zhang, Y., Chen, Y., & others. (2021). How information presentation formats influence usage behaviour of course management systems: Flow diagram navigation versus menu navigation. *Behaviour & Information Technology*. Retrieved from https://doi.org/10.1080/0144929X.2020.1813331