

**UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS**

**FACULTAD DE TECNOLOGÍAS EDUCATIVAS**

**Título**: Sistema de Gestión Hotelera

**Trabajo de Curso de Ingeniería de Software I**

Autores:

David Avila de la Rosa

Frank Ernesto Cortiñas Peña

Mabel María Castillo Moreno

Maydelín del Carmen Ibarra Núñez

Oniel José Aguilar Milán

Tutor:

Faure Arturo Fiallo Zequeira

**La Habana, septiembre 2024**

**“Año 66 de la Revolución”**

# Resumen

En los últimos años, la industria hotelera ha adoptado tecnologías avanzadas que transforman su funcionamiento, optimizando procesos y mejorando la experiencia del cliente. Los sistemas de gestión hotelera (SGH) han evolucionado hacia soluciones basadas en la nube, análisis de datos y metodologías ágiles, adaptándose a un entorno competitivo y cambiante. Este trabajo propone el desarrollo de un SGH utilizando un modelo prescriptivo y la metodología RUP, con Django para el backend y Angular para el frontend. Se analiza el estado del arte, identificando herramientas similares y los conceptos clave que guían esta evolución hacia sistemas ágiles, escalables y centrados en el usuario. El proyecto busca alinear la digitalización en la industria hotelera con un enfoque estructurado y flexible, ofreciendo una solución moderna y eficiente para la gestión operativa. Se utilizarán herramientas como Visual Studio Code, para el desarrollo del frontend y el backend y PgAdmin para la gestión eficiente de bases de datos. Este conjunto de herramientas y tecnologías no solo optimiza el desarrollo, sino que también garantiza la alta calidad, seguridad y escalabilidad del sistema a lo largo del tiempo. Además, la integración fluida entre backend, frontend y base de datos permitirá una implementación eficiente y sin interrupciones, asegurando que el sistema de gestión hotelera sea robusto, flexible y capaz de adaptarse a las necesidades del negocio conforme este crezca. La elección de tecnologías de código abierto también ofrece una ventaja en términos de costos y soporte continuo.

**Palabras clave:**

1. Gestión
2. Desarrollo
3. Tecnologías
4. Escalabilidad
5. Integración

**Abstract**

In recent years, the hospitality industry has adopted advanced technologies that are transforming its operations, optimizing processes, and enhancing the customer experience. Hotel management systems (HMS) have evolved into cloud-based solutions, data analysis tools, and agile methodologies, adapting to a competitive and ever-changing environment. This work proposes the development of an HMS using a prescriptive model and the RUP methodology, with Django for the backend and Angular for the frontend. The state of the art is analyzed, identifying similar tools and key concepts that guide this evolution toward agile, scalable, and user-centered systems. The project seeks to align digitalization in the hospitality industry with a structured and flexible approach, offering a modern and efficient solution for operational management. Tools such as Visual Studio Code will be used for frontend and backend development, and PgAdmin for efficient database management. This set of tools and technologies not only optimizes development but also ensures high quality, security, and scalability of the system over time. Furthermore, seamless integration between backend, frontend, and database will enable efficient and uninterrupted implementation, ensuring the hotel management system is robust, flexible, and capable of adapting to business needs as it grows. The choice of open-source technologies also provides an advantage in terms of costs and continuous support.

**Keywords:**

1. Management

2.Development

3.Technologies

4. Scalability

5.Integration

# Índice

Resumen-----------------------------------------------------------------------------------------2

Palabras claves---------------------------------------------------------------------------------2

Introducción--------------------------------------------------------------------------------------4

# Capítulo I. Estudio del Estado del Arte----------------------------------------7

# Capítulo II. Modelado del Contexto----------------------------------------21

# Capítulo III. Modelado del Sistema ----------------------------------------21

# Capítulo IV. Modelado de la Estructura y Comportamiento------21

**Introducción**

En la actualidad, la industria hotelera enfrenta un entorno cada vez más competitivo y dinámico, impulsado por el crecimiento del turismo, las plataformas de reserva en línea y las altas expectativas de los clientes. En este contexto, las empresas del sector deben optimizar sus procesos internos, mejorar la experiencia del huésped y responder con agilidad a las demandas del mercado. Sin embargo, muchas organizaciones hoteleras aún enfrentan desafíos significativos debido a la dependencia de métodos tradicionales o sistemas fragmentados que dificultan la coordinación eficiente de sus operaciones.

Entre las principales problemáticas que se identifican se encuentran:

1. **Gestión ineficiente de reservas y ocupación:** Los procesos manuales o sistemas desconectados provocan errores en las reservas, sobreventa de habitaciones y pérdida de ingresos.
2. **Control deficiente de inventarios y servicios:** La falta de integración entre áreas como recepción, mantenimiento y alimentos y bebidas dificulta la gestión de recursos, generando desperdicio o desabasto.
3. **Experiencia inconsistente para los clientes:** La ausencia de datos centralizados limita la personalización y agilidad en el servicio al huésped.
4. **Limitada capacidad de análisis y toma de decisiones:** Sin herramientas adecuadas para analizar datos, los gerentes tienen dificultades para identificar tendencias y oportunidades de mejora.

### ****Problema a Resolver:****

¿Cómo desarrollar un Sistema de Gestión Hotelera que satisfaga las necesidades del huésped?

### ****Objetivos****

**General:**  
Diseñar un sistema de gestión hotelera que integre las funciones críticas del hotel, desde la reserva y check-in hasta la facturación y control de inventarios, con enfoque en la eficiencia y la satisfacción del cliente.

**Específicos:**

1. Centralizar la gestión de reservas, ocupación y servicios adicionales en una única plataforma.
2. Automatizar tareas repetitivas para reducir errores operativos y mejorar la productividad del personal.
3. Proporcionar herramientas de análisis y reportes para apoyar la toma de decisiones estratégicas.
4. Ofrecer funcionalidades que faciliten la personalización del servicio al cliente.

### ****Tareas o Acciones a Investigar****

1. Análisis de las necesidades específicas de un hotel mediano y los flujos de trabajo actuales.
2. Estudio comparativo de las soluciones de software existentes en el mercado.
3. Diseño de la arquitectura del sistema y definición de módulos clave (reservas, check-in/check-out, inventarios, etc.).
4. Pruebas piloto del sistema para evaluar su rendimiento y efectividad.
5. Recopilación de retroalimentación del personal y clientes para mejorar iterativamente el sistema.

### ****Métodos****

**Teóricos:**

* Revisión bibliográfica sobre gestión hotelera, sistemas de información y tendencias tecnológicas en el sector.
* Modelos de mejora continua y optimización de procesos.

**Empíricos:**

* Entrevistas a personal y huéspedes para entender las necesidades y expectativas.
* Encuestas a personal y huéspedes
* Pruebas de usabilidad y desempeño del sistema en entornos reales.

**Herramientas Propias de la Ingeniería de Software:**

* Desarrollo ágil para iteraciones rápidas y centradas en el usuario.
* Lenguajes y frameworks para desarrollo de software (por ejemplo, Python, Java, o Node.js).
* Bases de datos relacionales o no relacionales para almacenar información crítica.
* Integración de APIs para funcionalidades como pagos en línea o conexión con plataformas de terceros.

Este enfoque permitirá abordar de manera integral las necesidades del hotel, garantizando una solución tecnológica eficiente y ajustada a las demandas actuales del mercado

# Capítulo I. Estudio del Estado del Arte

**Introducción**

En los últimos años, la industria hotelera ha experimentado una transformación significativa debido a la incorporación de tecnologías avanzadas y herramientas de gestión que optimizan los procesos internos y mejoran la experiencia del cliente. Los sistemas de gestión hotelera (SGH) han evolucionado para integrarse con soluciones basadas en la nube, análisis de datos y técnicas de desarrollo ágil, como la metodología XP, adaptándose a un entorno altamente competitivo y dinámico.

Este trabajo presenta el desarrollo de un sistema de gestión hotelera utilizando un modelo prescriptivo como guía estructural, la metodología RUP con un enfoque de desarrollo tradicional, Django para el backend y Angular para el frontend. A continuación, se exploran los conceptos fundamentales y el estado del arte en los últimos años, destacando herramientas similares y su clasificación, los puntos coincidentes entre los conceptos relacionados y, finalmente, una definición clara del fenómeno estudiado. El fenómeno analizado puede definirse como:

"La evolución de los sistemas de gestión hotelera hacia soluciones ágiles, escalables y centradas en la experiencia del usuario, integrando tecnologías avanzadas y metodologías tradicionales para satisfacer las demandas de la industria moderna." Este proyecto se alinea con las tendencias de digitalización en la industria hotelera, combinando un enfoque estructurado y flexible para el desarrollo. Utilizando herramientas modernas como Django y Angular, el sistema propuesto promete ofrecer una solución innovadora y efectiva para gestionar las operaciones hoteleras con mayor eficiencia y adaptabilidad.

## 1.1 Conceptos asociados al tema

Es importante identificar los conceptos clave relacionados con este tema. A continuación, se presenta un esquema de cómo podría estructurarse esta sección, incluyendo conceptos fundamentales y ejemplos de referencias utilizando el formato APA.

· **Sistema de Gestión**  
Es un conjunto de procedimientos, herramientas y recursos diseñados para planificar, controlar y optimizar las operaciones de una organización. Un sistema de gestión eficiente permite mejorar la productividad y la calidad del servicio.  
Referencia: ISO 9001:2015. Sistemas de gestión de la calidad. (Organización Internacional de Normalización, 2015).

· **Hotelera**  
Hace referencia al conjunto de actividades relacionadas con la operación y administración de servicios de alojamiento, incluidas la recepción, la limpieza, la restauración y la atención al cliente.  
Referencia: Organización Mundial del Turismo (2019). Compendio de estadísticas de turismo. Madrid: OMT.

· **Software de Gestión Hotelera (PMS)**  
Un sistema de gestión de propiedades (Property Management System, PMS) es una herramienta tecnológica utilizada en la industria hotelera para administrar reservas, registros de entrada y salida, inventario de habitaciones y procesos financieros.  
Referencia: Ivanov, S., & Zhechev, V. (2012). Hotel revenue management systems: A review of applications and practices. Tourism Review, 67(3), 51-63. https://doi.org/10.1108/TR-02-2012-0015

· **Business Intelligence (BI) en Hotelería**  
Se refiere al uso de tecnologías y estrategias de análisis de datos para mejorar la toma de decisiones en la gestión hotelera, como el análisis de ocupación y la optimización de tarifas.  
Referencia: Chen, H., Chiang, R. H., & Storey, V. C. (2012). Business Intelligence and analytics: From Big Data to Big Impact. MIS Quarterly, 36(4), 1165-1188.

· **Customer Relationship Management (CRM)**  
Es una estrategia de gestión empresarial centrada en mejorar las relaciones con los clientes a través de la recopilación y análisis de datos sobre sus preferencias y comportamientos. En hotelería, esto incluye la personalización de servicios.  
Referencia: Kotler, P., Bowen, J., & Makens, J. (2017). Marketing for Hospitality and Tourism. 7th edition. Pearson Education.

· **Normativas y Seguridad de Datos**  
En sistemas de gestión hotelera, es esencial garantizar la protección de los datos sensibles de los clientes y cumplir con normativas como el Reglamento General de Protección de Datos (RGPD).  
Referencia: Unión Europea (2016). Reglamento General de Protección de Datos (RGPD). Diario Oficial de la Unión Europea, L119.

## 1.2 Análisis de mercado

Implica evaluar soluciones existentes que ofrecen funcionalidades similares y comprender sus características, tecnologías y enfoques. Esto ayuda a determinar la factibilidad de la solución a desarrollar y a establecer las bases para el diseño del sistema.

### ****1. Caracterización de sistemas similares****

#### ****1.1. Ejemplos de sistemas existentes:****

**Opera PMS (Property Management System)**

* 1. **Dominio:** Gestión hotelera y hospitalidad.
  2. **Tipo de software:** Sistema de gestión integral para hoteles y resorts.
  3. **Plataformas:** Basado en la nube y con soporte para instalación local.
  4. **Tecnologías:** Utiliza bases de datos SQL, API REST y tecnologías web modernas.
  5. **Principales operaciones:**
     1. Gestión de reservas.
     2. Facturación y pagos.
     3. Check-in/check-out.
     4. Gestión de habitaciones y disponibilidad.
     5. Integración con sistemas de terceros (como channel managers y sistemas de puntos de venta).

**Cloudbeds**

* 1. **Dominio:** Hotelería, hostales, alquileres vacacionales.
  2. **Tipo de software:** Software como servicio (SaaS).
  3. **Plataformas:** Totalmente basado en la nube, accesible desde dispositivos móviles y PC.
  4. **Tecnologías:** Frameworks de desarrollo web como React y Node.js, almacenamiento en la nube (AWS o GCP).
  5. **Principales operaciones:**
     1. Gestión de reservas directas y a través de OTA (Online Travel Agencies).
     2. Sincronización en tiempo real con canales de distribución.
     3. Informes financieros y analíticos.
     4. Gestión de tarifas dinámicas.

**RoomRaccoon**

* 1. **Dominio:** Hotelería de pequeña y mediana escala.
  2. **Tipo de software:** SaaS.
  3. **Plataformas:** Aplicación web y móvil.
  4. **Tecnologías:** Desarrollo con Python/Django, Angular y servicios en la nube.
  5. **Principales operaciones:**
     1. Generación de cotizaciones y facturas.
     2. Gestión automatizada de tarifas y disponibilidad.
     3. Comunicación directa con los huéspedes.
     4. Integración con sistemas de domótica.

### ****2. Dominio de aplicación****

El sistema de gestión hotelera se enfoca en:

* Automatizar procesos operativos de hoteles y alojamientos.
* Mejorar la experiencia del huésped.
* Facilitar la integración con otros servicios relacionados con la hospitalidad (OTA, puntos de venta, etc.).

### ****3. Herramientas y tecnologías utilizadas****

* **Lenguajes de programación:** Python, JavaScript (Node.js, React, Angular).
* **Bases de datos:** PostgreSQL
* **Infraestructura:** Nube (AWS, Azure, Google Cloud), servidores locales para soluciones híbridas.
* **Integraciones:** API REST o SOAP para conectarse con canales de distribución y sistemas financieros.
* **Seguridad:** Autenticación basada en OAuth2, cifrado SSL/TLS.

### ****4. Plataformas de ejecución****

* Aplicaciones basadas en navegador (web apps).
* Compatibilidad con dispositivos móviles (iOS, Android).
* Implementaciones híbridas (nube y local) para necesidades específicas.

### ****5. Principales operaciones a considerar****

Basado en los sistemas analizados, las funcionalidades esenciales incluyen:

1. **Gestión de reservas:**
   * Registro de reservas directas e indirectas (a través de OTA).
   * Verificación de disponibilidad en tiempo real.
2. **Gestión de huéspedes:**
   * Registro de check-in y check-out.
   * Historial y preferencias de los huéspedes.
3. **Gestión de habitaciones:**
   * Asignación de habitaciones.
   * Monitoreo de limpieza y mantenimiento.
4. **Facturación y pagos:**
   * Generación automática de facturas.
   * Gestión de métodos de pago (tarjetas, transferencias).
5. **Informes y análisis:**
   * Indicadores de ocupación.
   * Análisis de ingresos por habitación.
6. **Integración con terceros:**
   * Conexión con sistemas de distribución y herramientas de marketing.
   * Integración con sistemas de puntos de venta.

El análisis de homólogos revela que el desarrollo de un sistema de gestión hotelera es factible y necesario para satisfacer la creciente demanda de automatización y digitalización en el sector hotelero. Este estudio proporciona un marco claro para definir los requisitos funcionales y tecnológicos en las siguientes etapas del desarrollo.

## 1.3 Fundamentación del proceso de software a desarrollar

El propósito principal del proceso de desarrollo es **crear una solución informática robusta y adaptable** que permita a los hoteles gestionar sus operaciones diarias de manera eficiente, reduciendo costos operativos, mejorando la experiencia del cliente y facilitando el acceso a datos relevantes para la toma de decisiones estratégicas.

**Objetivos específicos:**

* 1. Desarrollar un sistema intuitivo que simplifique la interacción del personal con las herramientas de gestión.
  2. Garantizar la interoperabilidad mediante estándares tecnológicos abiertos y APIs.
  3. Asegurar la escalabilidad del software para que sea utilizable en hoteles de diferentes tamaños y complejidades.

### ****Enfoque del proceso:** El desarrollo se fundamentará en un enfoque tradicional basado en** RUP (Rational Unified Process)**, lo que permitirá:**

### ****Claridad en los Requisitos desde el Inicio :**** El modelo tradicional, como el **modelo en cascada**, enfatiza la necesidad de recopilar y documentar requisitos de manera completa al principio del proyecto. Esto es ideal para un sistema de gestión hotelera cuando:

* Los requisitos son bien conocidos y estables (como gestión de reservas, check-in/check-out, facturación, etc.).
* El cliente tiene claridad sobre lo que necesita y espera del sistema.

### ****Estructura y Orden en el Desarrollo****

El modelo tradicional sigue un flujo lineal y secuencial de fases: **análisis, diseño, implementación, pruebas y mantenimiento**. Esta organización:

* Proporciona un marco claro para los desarrolladores y las partes interesadas.
* Facilita la planificación y asignación de recursos.
* Reduce la posibilidad de confusiones durante las fases del proyecto.

### ****Documentación Exhaustiva****

En un modelo tradicional, cada fase genera documentación detallada que:

* Sirve como referencia para el desarrollo y mantenimiento del sistema.
* Facilita la transferencia de conocimiento entre equipos.

### ****Riesgos Controlados en Proyectos Definidos****

El enfoque tradicional es adecuado cuando:

* Los riesgos del proyecto son bajos y el entorno técnico es estable.
* Los objetivos y tecnologías están bien definidos.  
  En un sistema hotelero estándar, las funcionalidades principales suelen ser claras y no requieren cambios significativos durante el desarrollo.
* Ideal para Equipos con Experiencia Limitada en Metodologías Ágiles

### Enfoque de ingeniería de software

La estrella de Boehm y Turner permite seleccionar el enfoque y la metodología de desarrollo de software más adecuada para un proyecto específico. Basada en cinco criterios fundamentales, esta herramienta evalúa y equilibra diferentes aspectos del proyecto, como son:

*Tamaño*: Cantidad de personas en el grupo de desarrollo.

*Personal*: Incluye las habilidades y motivaciones del personal del proyecto (Sénior o Junior).

*Dinamismo*: Porcentaje de cambio de los requisitos.

*Cultura*: Capacidad y experiencia del equipo como grupo de desarrollo.

*Criticidad*: Impactos, posibles problemas y riesgos del proyecto.

**Aplicación del método de Boehm y Turner**

*Tamaño*

El grupo de desarrollo consta de cinco integrantes, lo cual corresponde a un equipo pequeño. Esto permite un mayor control y comunicación directa, pero requiere un enfoque estructurado para evitar errores.

*Personal*

Los cincointegrantes del equipo son Junior, con poca experiencia previa en desarrollo y sin haber trabajado como equipo en el pasado. Esto indica que necesitan una metodología que proporcione guías claras y fomente la organización para minimizar riesgos.

*Dinamismo*

Los requisitos del sistema de gestión hotelera están bien definidos desde el inicio y no se esperan cambios significativos durante el desarrollo. Esto sugiere que un enfoque más estructurado sería ideal para cumplir con los objetivos establecidos.

*Cultura*

El equipo no tiene experiencia previa trabajando junto, lo que implica que la madurez del grupo de desarrollo es baja. Este factor refuerza la necesidad de adoptar un enfoque que fomente procesos bien documentados y metodologías estandarizadas.

*Criticidad*

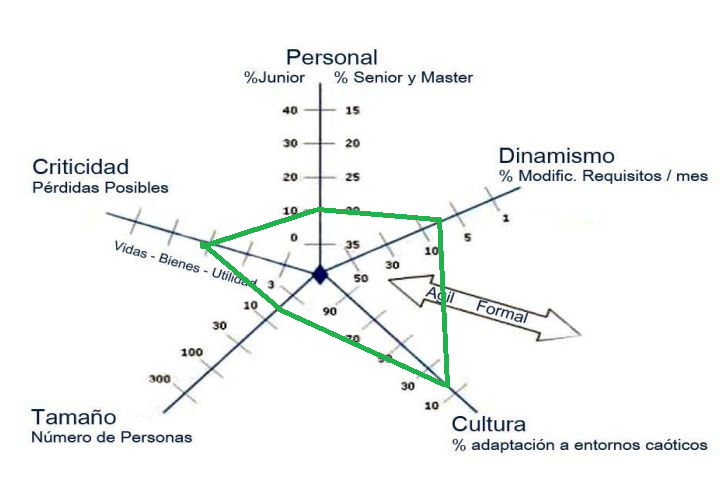
El sistema de gestión hotelera tiene una criticidad moderada. Sus riesgos e impactos potenciales incluyen:

Integridad de los datos: Manejo de información confidencial, como datos de clientes y reservas.

Seguridad: Garantizar que no haya accesos no autorizados a la información.

Cumplimiento legal: Asegurar la conformidad con normativas relacionadas con la protección de datos.

Selección del enfoque



Considerando las escasas experiencias con la que cuenta el equipo, el compromiso con la calidad del producto y la criticidad del proyecto se ha decidido que el proyecto se realizará bajo un enfoque tradicional.

### 1.3.2 Modelo(s) de proceso de software

El modelo utilizado es prescrictivo evolutivo ya que permiten que los ingenieros desarrollen versiones cada vez más completas del software. Ademas los requisitos finales no están bien definidos, aunque sí el esquema general de necesidades del cliente y es necesario satisfacer al cliente de forma rápida ante la presión del mercado.Por otra parte los modelos evolutivos son capaces de adaptarse a cambios en el entorno y las condiciones del mercado. En el sector hotelero, donde la demanda puede fluctuar estacionalmente o debido a eventos inesperados, esto es crucial

### 1.3.3 Método de ingeniería de software

### La selección de RUP (Rational Unified Process) como metodología para desarrollar el sistema de gestión hotelera, utilizando un enfoque tradicional y un modelo prescriptivo evolutivo, se fundamenta en su capacidad para proporcionar un marco estructurado y adaptable a proyectos complejos. RUP divide el desarrollo en cuatro fases: Inicio, Elaboración, Construcción y Transición, lo que facilita un progreso controlado y eficiente. Este enfoque permite definir claramente los objetivos y requisitos en la fase de Inicio, establecer una arquitectura sólida y mitigar riesgos en la Elaboración, desarrollar iterativamente los componentes del sistema en la Construcción y garantizar una implementación adecuada en la Transición. Además, su carácter iterativo y evolutivo asegura que el sistema se refine y mejore continuamente, adaptándose a los cambios en los requisitos y necesidades del negocio, mientras que el modelo prescriptivo asegura un control riguroso sobre el proceso, ideal para un sistema tan crítico como la gestión hotelera.

### 1.4.1 Herramienta CASE

Las herramientas CASE son fundamentales en el desarrollo de software, ya que facilitan la automatización de diversas actividades dentro del ciclo de vida del desarrollo. Para el sistema de gestión hotelera, se utilizarán herramientas CASE para mejorar la productividad, la calidad del software y la trazabilidad de los artefactos.

En el contexto de este proyecto, se utilizarán herramientas CASE en las siguientes áreas clave:

1. Modelado y Diseño

*Visual Paradigm*: Herramienta CASE que se puede considerar para el modelado y diseño. Permite crear diagramas UML, ERD (Diagrama de Entidad-Relación) y otros diagramas de alto nivel, y soporta la generación de código para diversas plataformas. Esta herramienta es útil para asegurar que el diseño del sistema esté alineado con los requisitos iniciales.

2. Desarrollo y Generación de Código

*WebRatio*: WebRatio es una herramienta CASE especializada en el desarrollo de aplicaciones web y móviles, que permite la creación de aplicaciones a partir de modelos. Utiliza un enfoque de desarrollo basado en modelos (MDA), donde los modelos de alto nivel se transforman automáticamente en código Java o HTML.

3.Pruebas y Calidad

*TestComplete*: TestComplete es una herramienta para la automatización de pruebas que también se puede usar para aplicaciones web. Permite probar tanto la interfaz de usuario como la funcionalidad interna de la página web, asegurando que el sistema cumpla con los requisitos.

4. Mantenimiento y Gestión de Versiones

*Git (junto con GitHub/GitLab/Bitbucket):* Git es una herramienta de control de versiones ampliamente utilizada para gestionar el código fuente de proyectos web. Las plataformas como GitHub, GitLab y Bitbucket permiten almacenar el código, colaborar en equipo y realizar un seguimiento de los cambios. Esto es crucial para proyectos web que involucran varias iteraciones o cambios frecuentes.

### 1.4.2 Lenguaje de modelado

**UML (Unified Modeling Language)** es un lenguaje de modelado estandarizado que se utiliza ampliamente en la ingeniería de software para especificar, visualizar, construir y documentar los artefactos de un sistema. Proporciona un conjunto de notaciones gráficas y semánticas para crear diversos diagramas que representan diferentes aspectos del software. Este facilita la comunicación entre los miembros del equipo y otros interesados, permite la creación de perfiles y extensiones que pueden adaptarse a necesidades específicas del proyecto, soporta varios tipos de diagramas que permiten representar diferentes vistas del sistema. Algunos diagramas comunes en UML son Diagramas de Clases, Diagramas de Secuencia,Diagramas de Casos de Uso. El uso de este lenguaje proporcionará un marco robusto y estandarizado para la creación de diagramas ingenieriles en el desarrollo del sistema de gestión hotelera

### 1.4.3 Marco de trabajo para el desarrollo de la solución informática

Un marco de trabajo, o framework, es una estructura de software que ofrece componentes modulares y adaptables para facilitar el desarrollo de aplicaciones. En esencia, un framework puede verse como una base genérica y funcional que permite personalizar y añadir elementos específicos para crear una solución particular. Sus principales propósitos incluyen agilizar el proceso de desarrollo, fomentar la reutilización de código preexistente y promover buenas prácticas de programación, como la implementación de patrones de diseño establecidos. Este enfoque optimiza los tiempos de desarrollo y mejora la calidad del software resultante.

### 1.4.4 Entorno de desarrollo integrado

Para el desarrollo del sistema de gestión hotelera con un backend en Python, se recomienda utilizar entornos de desarrollo integrados (IDE) que ofrezcan herramientas robustas para escribir, probar y depurar código de manera eficiente. Una opción destacada es **Visual Studio Code**, un editor de código fuente desarrollado por Microsoft, ampliamente reconocido por su versatilidad y facilidad de uso.

Visual Studio Code es compatible con Windows, macOS y Linux, lo que lo convierte en una solución accesible para equipos de desarrollo que trabajan en diferentes plataformas. Este editor es especialmente potente gracias a su extensibilidad, ya que permite personalizarlo mediante una amplia gama de extensiones, muchas de las cuales están diseñadas específicamente para Python, facilitando tareas como la gestión de entornos virtuales, el análisis de código y el soporte para frameworks populares como Django o Flask.

Entre sus funcionalidades clave se encuentra la integración nativa con Git, ideal para gestionar versiones y colaborar en equipo, algo crucial en un proyecto que requiere un desarrollo coordinado. También ofrece herramientas de depuración integradas que simplifican la identificación y resolución de errores en tiempo real, optimizando el flujo de trabajo.

Además, Visual Studio Code incluye una terminal integrada, que permite ejecutar comandos directamente desde el entorno, como la instalación de dependencias o la ejecución de pruebas unitarias. Su soporte avanzado para Python, combinado con características como resaltado de sintaxis y autocompletado inteligente, hacen de este IDE una opción ideal para desarrollar un sistema robusto y escalable, adaptado a las necesidades de la industria hotelera.

### 1.4.5 Lenguaje de programación

Un lenguaje de programación es una herramienta formal diseñada con reglas gramaticales precisas que permite a los programadores crear algoritmos para controlar el funcionamiento físico o lógico de un sistema informático. Estos lenguajes son fundamentales para desarrollar aplicaciones que procesen datos, ejecuten tareas específicas y resuelvan problemas de manera eficiente.

En el desarrollo del sistema de gestión hotelera, se ha optado por utilizar **Django** y **JavaScript** debido a su capacidad para cumplir con los requisitos del proyecto.

-**Django** es un framework de Python que facilita el desarrollo rápido y escalable de aplicaciones web, proporcionando una estructura robusta para la creación de backends eficientes y mantenibles. Este framework incluye herramientas y componentes predefinidos, lo que acelera el desarrollo y asegura buenas prácticas en la creación del sistema.

**-JavaScript** es un lenguaje de programación de alto nivel ampliamente utilizado en el desarrollo web para crear interactividad en las páginas. Es un lenguaje interpretado, lo que significa que el código se ejecuta línea por línea en tiempo real. Esto facilita la depuración y la iteración rápida durante el desarrollo, es un lenguaje orientado a objetos basado en prototipos, lo que permite una programación más flexible y dinámica.Es compatible con la mayoría de los navegadores web modernos en diferentes sistemas operativos.

### 1.4.6 Gestor de base de datos

Para el desarrollo del sistema de gestión hotelera, se ha seleccionado **PgAdmin**, una herramienta de administración y desarrollo de bases de datos especializada en **PostgreSQL**. PgAdmin proporciona una interfaz gráfica intuitiva que simplifica la gestión de bases de datos, permitiendo realizar tareas como la creación, modificación y administración de objetos de bases de datos, la ejecución de consultas SQL y la gestión de usuarios y permisos de forma eficiente.

La elección de PgAdmin se basa en su integración óptima con **PostgreSQL**, un sistema de gestión de bases de datos robusto y de código abierto, ampliamente utilizado por su escalabilidad y fiabilidad. Al ser una herramienta dedicada a PostgreSQL, PgAdmin maximiza las capacidades de este sistema, garantizando un entorno de desarrollo y administración de bases de datos eficiente y compatible.

Además, su diseño accesible permite a los desarrolladores trabajar de manera más ágil y productiva, lo que resulta clave en el desarrollo de un sistema complejo como el de gestión hotelera. La combinación de funcionalidad avanzada y facilidad de uso convierte a PgAdmin en la opción ideal para la implementación y administración de la base de datos del proyecto.

## Conclusiones parciales

Este capítulo ha proporcionado una visión integral de los conceptos clave relacionados con el tema de investigación, estableciendo las bases necesarias para el desarrollo de la solución informática propuesta. Se ha detallado el proceso de desarrollo del software, considerando la selección de un enfoque metodológico adecuado, que se fundamenta tanto en la experiencia del equipo como en la naturaleza crítica del proyecto. Se ha justificado la elección de un modelo prescriptivo tradicional, resaltando sus ventajas en términos de organización y control del proceso. Finalmente, se ha presentado el conjunto de herramientas y tecnologías elegidas para la construcción de la solución, subrayando su importancia en la creación de un sistema robusto, seguro y escalable, capaz de cumplir con los requisitos del proyecto de manera eficiente.

# Capítulo II. Modelado del Contexto

## Introducción

En el desarrollo de sistemas de software, entender profundamente el contexto del negocio es crucial para garantizar que el producto final cumpla con las necesidades y expectativas de los usuarios. Este capítulo se enfocará en el análisis del modelo conceptual y el diagrama de clases del sistema de gestión hotelera, herramientas esenciales para estructurar y visualizar las entidades y relaciones dentro del sistema. El modelo conceptual es una representación abstracta de los elementos clave del negocio hotelero, identificando las entidades fundamentales, sus atributos y las interacciones entre ellas. Junto con el diagrama de clases, que detalla las clases, sus atributos y métodos, y las relaciones entre ellas, se proporciona una base sólida para la implementación del sistema.

Además, se analizarán las reglas del negocio que guían procesos esenciales como las reservas y la gestión de habitaciones. Estas reglas son vitales para garantizar que el sistema opere de manera eficiente y conforme a las expectativas del negocio. Este capítulo tiene como objetivo proporcionar una comprensión clara del modelo conceptual, el diagrama de clases y las reglas del negocio, elementos clave para desarrollar un sistema hotelero robusto y alineado con las necesidades operativas.

## 2.1 Reglas del negocio

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Clasificación** | **Nombre** | **Descripción** |
| *1* | Inferencia | Asignación de Habitaciones | Regla que deduce la mejor habitación disponible para el huésped según sus preferencias. |
| *2* | Computacionales | Cálculo de Tarifas | Algoritmo que calcula las tarifas diarias basado en temporada, demanda y tipo de habitación. |
| *3* | Hechos | Registro de Huéspedes | Almacena datos de los huéspedes, como nombre, fecha de entrada y salida, y servicios solicitados. |
| *4* | Restricciones | Política de Cancelación | Establece las condiciones bajo las cuales una reserva puede ser cancelada sin penalización. |
| *5* | Facilitadores de Acción | Notificación de Disponibilidad | Sistema que envía alertas a los huéspedes sobre la disponibilidad de habitaciones o promociones. |
|  |  |  |  |

## 2.2 Modelo Conceptual

***Descripción de las clases del modelo del conceptual***

Este diagrama conceptual muestra las clases principales y sus relaciones en un sistema de gestión hotelera. A continuación, se describen las clases y su propósito:

**Huésped**:

Representa a los clientes que realizan reservaciones en el hotel.

Relación con Reserva: Los huéspedes pueden realizar una o más reservas.

**Recepcionista:**

Representa al empleado encargado de procesar las reservas y registrar a los huéspedes.

Relación con Reserva: Procesa una o más reservas.

**Reserva:**

Contiene la información relacionada con las reservas realizadas por los huéspedes.

Relación con Servicios: Incluye uno o más servicios.

Relación con Habitación: Genera asignaciones de habitaciones.

Relación con Pagos: Está vinculada a los pagos realizados.

**Servicios:**

Representa los servicios adicionales que el hotel ofrece (como spa, desayuno, etc.).

Relación con Reserva: Está involucrado en una o más reservas.

**Pagos:**

Representa los registros de pagos realizados por los huéspedes.

Relación con Reserva: Se generan pagos a partir de una o más reservas.

**Habitación:**

Representa las habitaciones del hotel que son asignadas a las reservas.

Relación con Reserva: Está relacionada con las reservas que las generan.

**Personal:**

Representa al equipo de trabajo que proporciona servicios a los huéspedes.

Relación con Servicios: Está encargado de proporcionar uno o más servicios.

Relación con Gerente: Es supervisado por un gerente.

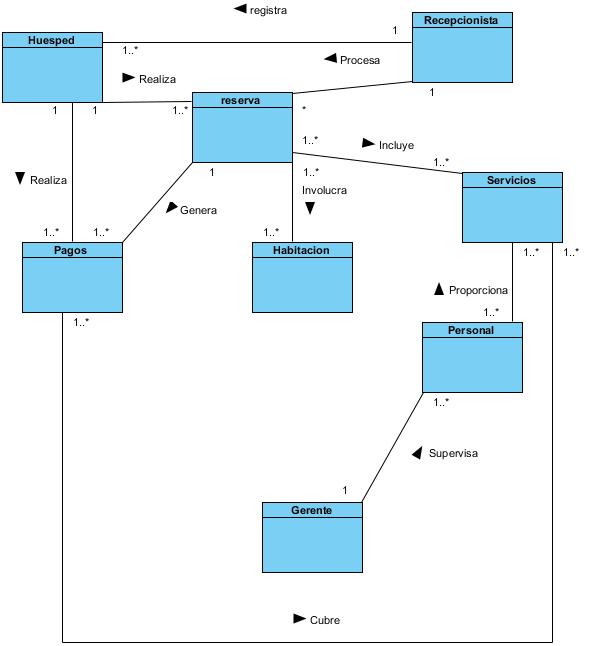
**Gerente:**

Representa al responsable de supervisar el personal y administrar las operaciones del hotel.

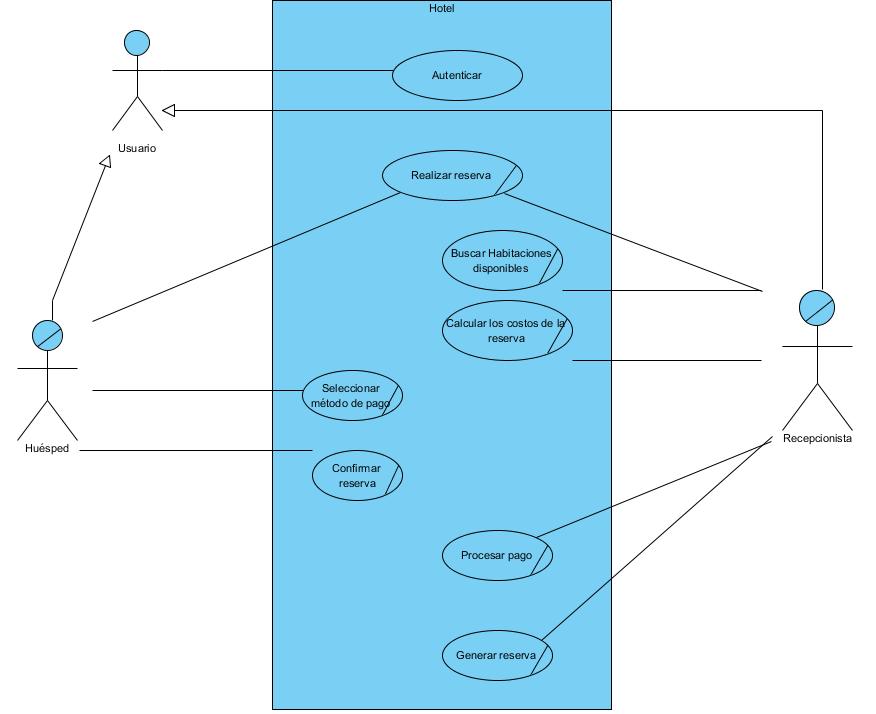
Relación con Personal: Supervisa al equipo de trabajo.

* En conjunto, estas clases y relaciones reflejan cómo interactúan los diferentes componentes en un sistema de gestión hotelera, desde la reserva de servicios hasta la supervisión del personal.

**Modelo Conceptual:**

****

### *Diagrama de Casos de Uso del Negocio*

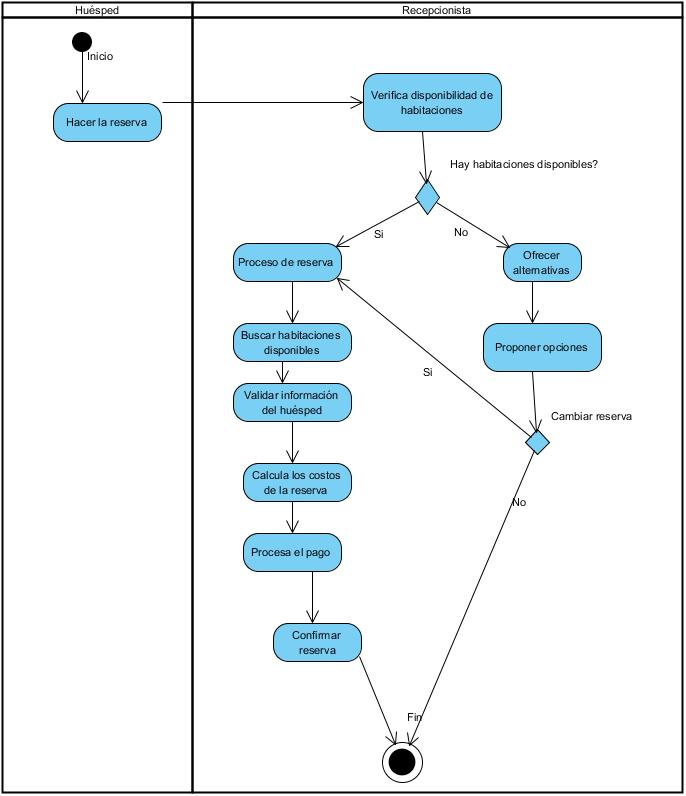
**

### *Descripción Textual de Casos de Uso del Negocio*

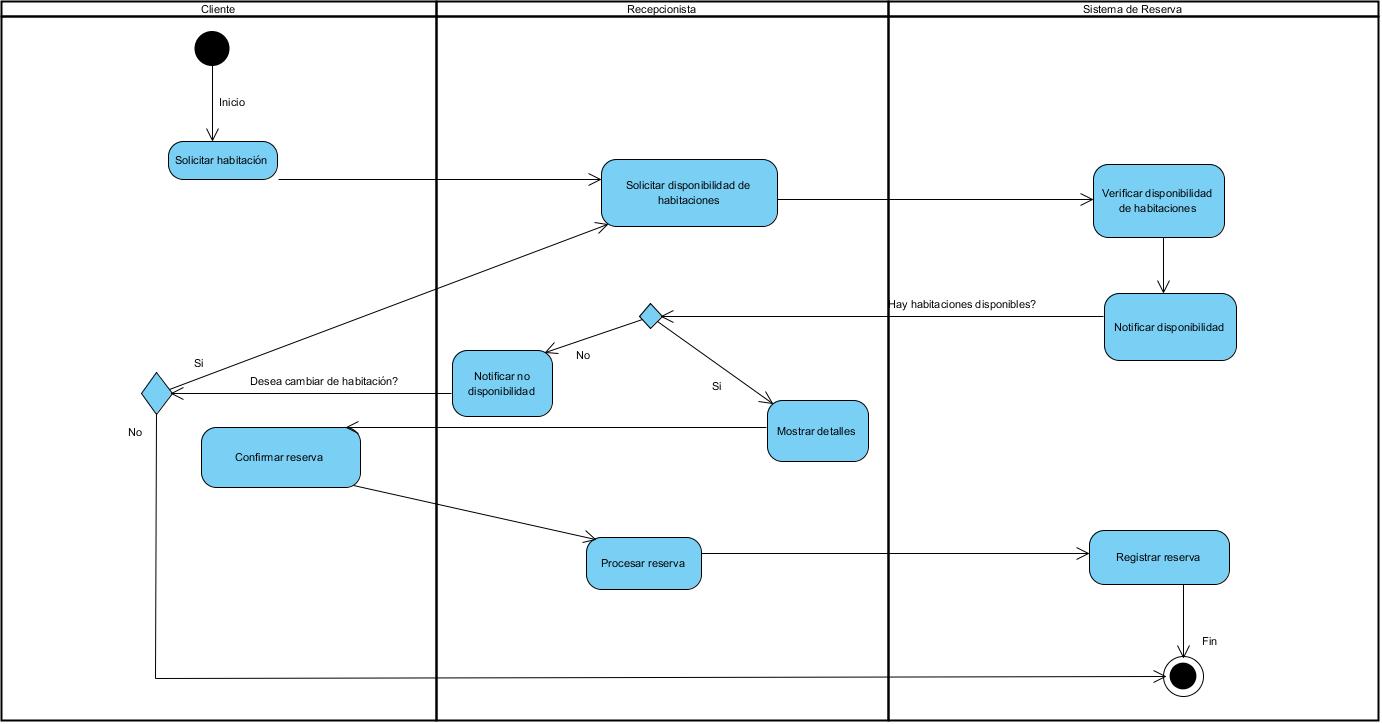
### Caso de Uso del Negocio <Reserva de Habitaciones>

|  |  |  |  |
| --- | --- | --- | --- |
| **Caso de Uso del Negocio** | *Reserva de Habitaciones* | | |
| **Actores** | *Huésped, Recepcionista, Gerente* | | |
| **Resumen** | El caso de uso comienza cuando un Huésped solicita reservar una habitación. El Huésped selecciona la opción "Reservar habitación". Se muestran las habitaciones disponibles. El Huésped selecciona una habitación, indica las fechas y confirma. Se registra la reserva y se genera una confirmación.  Resultado esperado:  Se genera y guarda una reserva en el sistema, y el Huésped recibe un número de confirmación. | | |
|  |  | | |
| **Acción del actor** | | **Respuesta del proceso de negocio** | |
| *actor (Huésped)*   * Seleccionar fechas: El Huésped indica las fechas de llegada y salida para la estancia. * Seleccionar habitación: El Huésped revisa las habitaciones disponibles y elige una que se ajuste a sus preferencias. * Proporcionar información personal: El Huésped ingresa sus datos personales (nombre, correo electrónico, número de teléfono, etc.). * Seleccionar método de pago: El Huésped elige el método de pago * Confirmar la reserva: El Huésped revisa los datos de la reserva (habitaciones, fechas, costos) y confirma la operación. * Finalizar el proceso: El Huésped recibe la confirmación de la reserva con un número de referencia. | | *Respuesta*   * Validar fechas ingresadas (Procesamiento del Sistema): verifica que las fechas proporcionadas por el Huésped sean válidas * Buscar habitaciones disponibles: consulta la base de datos para listar las habitaciones disponibles en las fechas seleccionadas. * Validar datos personales: valida que los datos proporcionados por el Huésped estén completos y sean correctos * Calcular costos de la reserva: calcula el costo total según el tipo de habitación seleccionada, la duración de la estancia y las tarifas aplicables. * Procesar el pago: valida los datos del método de pago y procesa la transacción. * Generar la reserva: Una vez procesado el pago, genera un registro de la reserva con un número único, fecha y detalles de la habitación. * Notificar la confirmación: envía una notificación al Huésped con los detalles de la reserva y el número de referencia. | |
| **Flujos alternativos** | | | |
| <No hay habitaciones disponibles>  **Condición que lo genera:**  El sistema no encuentra habitaciones disponibles para las fechas seleccionadas por el Huésped.  **Secuencia de Acciones:**  Actor (Huésped):  Ingresa las fechas y solicita una habitación.  Sistema de Gestión:  Verifica la disponibilidad y detecta que no hay habitaciones disponibles.  Notifica al Huésped que no hay habitaciones disponibles para las fechas seleccionadas.  Actor (Huésped):  Decide modificar las fechas o salir del proceso de reserva. | | | |
| **Acción del actor** | | | **Respuesta del proceso de negocio** |
| *Actor(Huésped)*   * Seleccionar fechas: El Huésped indica las fechas de llegada y salida para su estancia. * Solicitar disponibilidad: El Huésped consulta al sistema para ver qué habitaciones están disponibles en las fechas seleccionadas. * Recibir notificación: El Huésped recibe un mensaje indicando que no hay habitaciones disponibles para las fechas solicitadas. * Decidir una acción alternativa: * Puede intentar cambiar las fechas de su estancia. * Puede salir del sistema sin completar la reserva. | | | *Respuesta*   * Verifica que las fechas proporcionadas sean válidas * Consultar disponibilidad de habitaciones * Detectar falta de disponibilidad: Si no hay habitaciones disponibles, genera un mensaje de notificación.   **Ofrecer alternativas:**   * Puede sugerir fechas cercanas con disponibilidad o habitaciones en una categoría diferente * Informa al Huésped que no hay disponibilidad y, si corresponde, muestra opciones alternativas.   **Flujo con Alternativa:**   * Si el Huésped decide cambiar las fechas: Reinicia el proceso de búsqueda con las nuevas fechas seleccionadas. * Si el Huésped decide abandonar el proceso: Finaliza el proceso |
| **Mejoras propuestas** | **Automatización de la Disponibilidad de Habitaciones:** El sistema consultará automáticamente la disponibilidad en tiempo real, eliminando posibles errores humanos.  **Cálculo Automático de Costos**: Incorporar lógica de precios dinámica basada en la temporada, ocupación y promociones, permitiendo ajustes automáticos  **Notificaciones Automatizadas:** Generar notificaciones instantáneas por correo electrónico, mensajes de texto o aplicaciones móviles para confirmar la reserva.  **Manejo de Flujos Alternativos:** Cuando no haya disponibilidad, el sistema ofrecerá automáticamente opciones alternativas (fechas cercanas o habitaciones diferentes). | | |

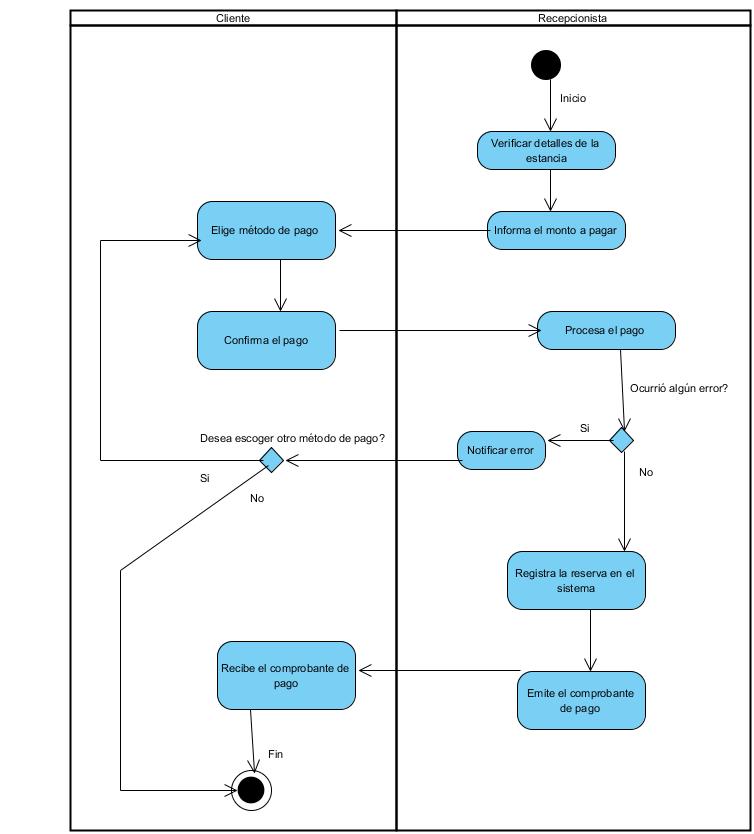
### Diagrama de Actividades del Caso de Uso del Negocio <Reserva de habitaciones>

****

### Diagrama de Actividades del Caso de Uso del Negocio <Búsqueda de habitaciones disponibles>

-

### Diagrama de Actividades del Caso de Uso del Negocio <Proceso de pago>



## Conclusiones parciales

En este capítulo, se ha realizado un paso crucial en el desarrollo del sistema de información: el modelado del contexto del negocio mediante un modelo conceptual. Este proceso ha permitido crear una representación detallada y visual de las principales entidades involucradas, sus relaciones y las reglas del negocio que rigen su funcionamiento. Además, se han identificado los requisitos clave del sistema, los cuales forman la base fundamental para la construcción de un proyecto de software exitoso. Este análisis no solo facilita la comprensión de la estructura y dinámica del negocio, sino que también proporciona una guía clara para la posterior fase de desarrollo del sistema.

# Capítulo III. Modelado del Sistema

## Introducción

En este capítulo, se detallarán los requisitos funcionales y no funcionales del sistema de gestión hotelera, que se utilizarán como base para la implementación y pruebas del sistema. Además, se describirán las características relacionadas con la usabilidad, confiabilidad, eficiencia, soporte y restricciones del diseño e implementación. Finalmente, se abordarán aspectos relacionados con la documentación de usuarios y la interfaz del sistema, asegurando que cumpla con los objetivos propuestos.

## 3.1 Requisitos funcionales

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Nombre** | **Descripción** | **Prioridad** | **Complejidad** |
| RF01 | Registro de clientes | Permitir el registro de nuevos clientes en el sistema. | Alta | Media |
| RF02 | Gestión de reservas | Gestionar reservas de habitaciones, incluyendo creación, modificación y cancelación. | Alta | Alta |
| RF03 | Disponibilidad de habitaciones | Verificar la disponibilidad de habitaciones en tiempo real | Alta | Alta |
| RF05 | Procesamiento de pagos | Procesar pagos de clientes y emitir recibos. | Alta | Media |
| RF06 | Generación de informes | Generar informes de ocupación, reservas y finanzas. | Media | Alta |
| RF07 | Confirmación de reservas. | Enviar confirmaciones de reserva por correo electrónico. | Alta | Baja |
| RF08 | Portal del cliente | Proporcionar un portal para que los clientes gestionen sus reservas en línea. | Alta | Alta |
| RF09 | Gestión de usuarios | Gestionar perfiles y permisos de usuarios del sistema. | Alta | Alta |
|  |  |  |  |  |

## 

## 3.2 Requisitos no funcionales

**Usabilidad:**

* La interfaz debe ser intuitiva y fácil de usar para usuarios de todas las edades (mayores de 18 años), permitiendo una navegación rápida y sin complicaciones.
* El sistema debe cumplir con los estándares de accesibilidad web(WCAG 2.1) ,para asegurar que personas con discapacidades puedan utilizarlos sin problemas.

**Eficiencia y desempeño:**

* El sistema debe ser capaz de manejar al menos 40 transacciones por Segundo durante los picos de demanda.
* La utilización de recurso(CPU, memoria ,disco )debe ser optimizada para garantizar un rendimiento eficiente

**Fiabilidad:**

* El sistema debe tener una disponibilidad del 99.9% para asegurar que este siempre accesible para los usuarios.
* Debe incluir mecanismos de respaldos y recuperación para prevenir la perdida de datos.

**Compatibilidad:**

* El sistema debe ser compatible con los navegadores web más utilizados como chrome, Firefox y safari

**Seguridad:**

* El sistema debe incluir autenticación de dos factores para mejorar la seguridad del acceso de los usuarios.

**Mantenibilidad:**

* El código del sistema debe estar bien documentado para facilitar el mantenimiento y futuras actualizaciones.

**Portabilidad:**

* El sistema debe ser fácilmente trasladable a diferentes entornos de servidores sin mayores ajustes.

##### 3.2.1 Usabilidad

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **No.** | **Sexo** | **Edad** | **Nivel de escolaridad** | **Ocupación** | **Experiencia profesional** | **Experiencia con la aplicación informática.** | **Tipo de discapa\_cidad** | **Otras** |
| 1 | Femenino | 18 a 65 años | Universitario | Recepcionista | 1 a 5 años | 1 a 2 años | Ninguna | N/A |
| 2 | Masculino | 25 a 60 años | Técnico Medio | Mantenimiento | 5 a 10 años | 1 a 3 años | Visual | N/A |
| 3 | Femenino | 30 a 55 años | Preuniversitario | Gerente | 10+ años | 5+ años | Ninguna | N/A |
| 4 | Masculino | 20 a 50 años | Secundario | Personal de limpieza | 2 a 5 años | 1 a 2 años | Ninguna | |  | | --- | | N/A | |

#### Tipo de Aplicación Informática

Aplicación WEB: La aplicación será accesible a través de navegadores web y estará optimizada para varios dispositivos, incluyendo computadoras de escritorio, portátiles y dispositivos móviles.

#### Finalidad

**Gestionar**: La finalidad principal de la aplicación es gestionar las reservas, pagos, y servicios del hotel de manera eficiente.

**Informar**: La aplicación debe proporcionar información clara y precisa sobre la disponibilidad de habitaciones, tarifas, y servicios adicionales.

**Entretener**: Ofrecer una interfaz amigable y atractiva que facilite el uso por parte de los clientes y personal del hotel.

#### Ambiente

**Hardware**: Se requiere un servidor con capacidades adecuadas para manejar un alto volumen de transacciones. Los dispositivos de los usuarios finales deben ser compatibles con navegadores modernos.

**Software**: El sistema debe ser compatible con navegadores web como Chrome, Firefox y Safari.

**Tiempo de Respuesta**: La aplicación debe ofrecer tiempos de respuesta rápidos, asegurando que las transacciones se procesen en menos de 2 segundos durante picos de demanda.

**Condiciones de Uso**: La aplicación se utilizará principalmente en un ambiente de oficina, pero también debe ser accesible desde dispositivos móviles para usuarios que necesitan acceso remoto.

##### 3.2.2 Confiabilidad

#### Requisitos relacionados con la confiabilidad

**Interrupción de Energía (Cliente)**:

**Requisito**: El sistema debe guardar automáticamente cualquier trabajo en curso al detectar una interrupción de energía en el cliente.

**Respuesta del Sistema**: Al reanudarse las operaciones, el sistema debe recuperar automáticamente la información guardada y permitir al usuario continuar desde el punto en que se interrumpió.

**Interrupción de Energía (Servidor)**:

**Requisito**: El sistema debe garantizar que toda la información crítica esté respaldada de manera segura en intervalos regulares.

**Respuesta del Sistema**: Al reanudarse las operaciones, el servidor debe restaurar automáticamente la información respaldada y reanudar el procesamiento desde el último estado guardado.

**Interrupción de Comunicaciones de Red (Cliente)**:

**Requisito**: El sistema debe tener un mecanismo de detección de interrupciones de red y guardar los datos pendientes localmente.

**Respuesta del Sistema**: Al reanudarse las operaciones, el sistema debe sincronizar automáticamente los datos locales con el servidor sin pérdida de información.

**Interrupción de Comunicaciones de Red (Servidor)**:

**Requisito**: El sistema debe mantener una cola de tareas pendientes para asegurar que ninguna información crítica se pierda.

**Respuesta del Sistema**: Al reanudarse las operaciones, el servidor debe procesar todas las tareas pendientes en la cola y actualizar las PC cliente.

**Desconexión de Dispositivos Periféricos**:

**Requisito**: El sistema debe detectar la desconexión de cualquier dispositivo periférico (impresora, escáner, capta huellas, etc.) y detener las operaciones relacionadas de manera segura.

**Respuesta del Sistema**: Al reanudarse las operaciones, el sistema debe permitir la reconexión del dispositivo y continuar la operación interrumpida sin pérdida de datos.

**Inactividad del Sistema**:

**Requisito**: El sistema debe cerrar sesión automáticamente después de 10 minutos de inactividad para asegurar la seguridad.

**Respuesta del Sistema**: Al cumplirse el tiempo definido, el sistema debe requerir autenticación del usuario para reanudar las operaciones.

**Manejo de Excepciones**:

**Requisito**: El sistema debe capturar y manejar todas las excepciones de manera segura y registrar los errores.

**Respuesta del Sistema**: En caso de una excepción, el sistema debe proporcionar un mensaje de error claro al usuario y asegurar que los datos se guarden de manera segura antes de intentar recuperar la operación.

**Cambio de Hora en la PC Cliente (Sistemas en Tiempo Real)**:

**Requisito**: El sistema debe detectar y ajustar automáticamente cualquier cambio en la hora del sistema para asegurar la precisión de los datos.

**Respuesta del Sistema**: Al cambiar la hora, el sistema debe sincronizarse automáticamente con el servidor y mantener la consistencia de la información.

**Cambio de Hora en el Servidor (Sistemas en Tiempo Real)**:

**Requisito**: El sistema debe mantener un reloj interno y sincronizarse automáticamente con servidores de tiempo confiables.

**Respuesta del Sistema**: Al cambiar la hora en el servidor, el sistema debe actualizar todas las PC cliente y asegurar que la información refleje el nuevo tiempo de manera consistente.

##### 3.2.3 Eficiencia

* **Tiempo de respuesta**: Las transacciones deben completarse en menos de 2 segundos en picos de demanda.
* **Escalabilidad**: Soportar hasta 100 usuarios concurrentes sin degradar el rendimiento.

##### 3.2.4 Software

* **Requerimientos de software**:
  + Sistema operativo: Windows, macOS, Linux.
  + Navegadores: Chrome, Firefox, Safari.
  + Servidor web: Nginx o Apache.

##### 3.2.5 Hardware

* **Servidor**:
  + Procesador: Quad-core o superior.
  + Memoria RAM: 16 GB.
  + Almacenamiento: 500 GB SSD.
* **Dispositivos del cliente**:
  + Procesador: Dual-core.
  + Memoria RAM: 4 GB.

##### 3.2.6 Soporte

* **Mantenimiento**: Documentación clara del código para facilitar futuras actualizaciones.
* **Herramientas**: Uso de Git para control de versiones.

##### 3.2.7 Restricciones de diseño e implementación

* **Lenguaje de programación**: Python para el backend y JavaScript para el frontend.
* **Base de datos**: PostgreSQL.
* **Framework**: Django.

##### 3.2.8 Documentación de usuarios en línea y ayuda del sistema

* **Guía de usuario**: Incluir tutoriales interactivos y FAQs.
* **Soporte en tiempo real**: Incorporar un sistema de chat para soporte.

##### 3.2.9 Apariencia o interfaz externa

* **Diseño responsivo**: Adaptable a diferentes tamaños de pantalla.
* **Esquema de colores**: Tonos cálidos y neutrales para un ambiente profesional.

##### 3.2.10 Requisitos de licencia

* **Software libre**: Utilizar licencias MIT o similares para componentes de software.

##### 3.2.11 Requisitos legales, de derecho de autor y otros

* **Protección de datos**: Cumplir con las regulaciones del RGPD.
* **Avisos legales**: Mostrar términos y condiciones claros para los usuarios.

# Conclusiones parciales

En este capítulo se han definido los requisitos necesarios para garantizar un desarrollo exitoso del sistema de gestión hotelera. Estos lineamientos establecen una base sólida para avanzar hacia las siguientes etapas del proyecto, asegurando que el sistema cumpla con las expectativas de los usuarios y los objetivos del negocio.

# 

# Capítulo IV. Modelado de la Estructura y Comportamiento

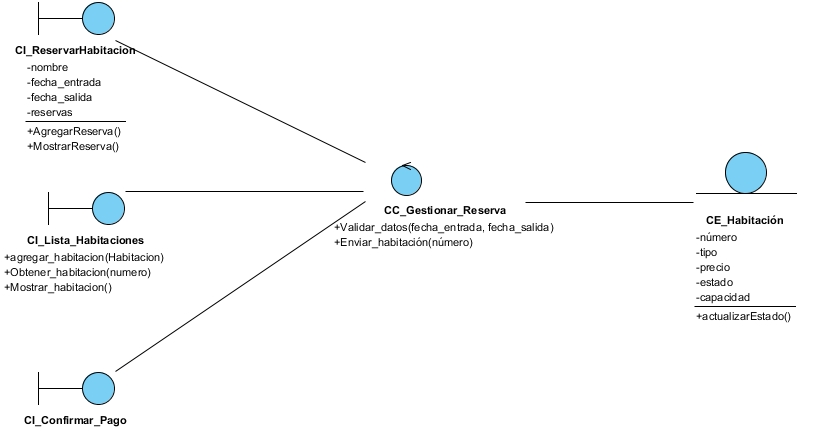
# Introducción

La estructura y comportamiento de un sistema puede definirse mediante sus componentes. Estos se reflejan en los diagramas de clases y de interacción, que permiten evaluar la estructura del sistema y su comportamiento. En este capítulo se utilizarán estos para tal fin.

## 

## 4.2 Diagramas de clases

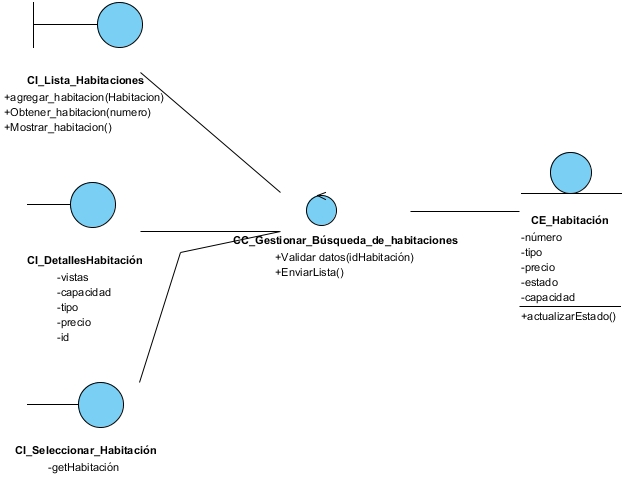
### 4.2.1 Diagrama de clases del Caso de uso <Reserva de Habitaciones>

****

**Descripción de clases del Caso de uso <Reserva de Habitaciones>**

|  |  |  |
| --- | --- | --- |
| **No.** | **Nombre de la Clase** | **Descripción** |
| 1 | Cliente | Cliente que realiza reservas; incluye datos personales y métodos para gestionar sus reservas. |
| 2 | Reservar habitación | Administra fechas, estado y habitaciones reservadas. |
| 3 | Habitación | Representa las habitaciones con número, tipo, precio y disponibilidad. |
| 4 | Generar reserva | Verifica disponibilidad y gestiona reservas y pagos. |
| 5 | Pago | Gestiona los pagos de reservas con detalles como monto y método. |
| 6 | Lista de habitaciones | Ofrece opciones de habitaciones |
| 7 | Eliminar reserva | Notifica la eliminación de la reserva |

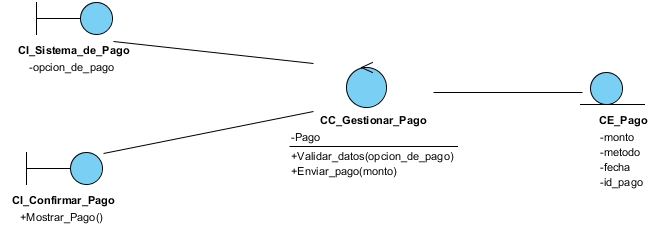
### 4.2.2 Diagrama de clases del Caso de uso <Búsqueda de habitaciones disponibles>



### Descripción de clases del Caso de uso <Búsqueda de habitaciones disponibles>

|  |  |  |
| --- | --- | --- |
| **No.** | **Nombre de la Clase** | **Descripción** |
| 1 | Habitación | Representa al usuario del sistema que solicita una habitación y confirma la reserva a través del recepcionista. |
| 2 | Gestionar | Es el intermediario entre el cliente y el sistema de reservas. Gestiona las solicitudes del cliente y utiliza el sistema para verificar disponibilidad, mostrar detalles, y procesar reservas. |
| 3 | Sistema de Reserva | Es el sistema encargado de manejar la lógica de negocio relacionada con la gestión de habitaciones, reservas y disponibilidad. |
| 4 | Reserva | Representa el registro de una habitación reservada, con información como fecha, habitación asociada y estado actual de la reserva. |

**4.2.3 Diagrama de clases del Caso de uso <Proceso de pago>**

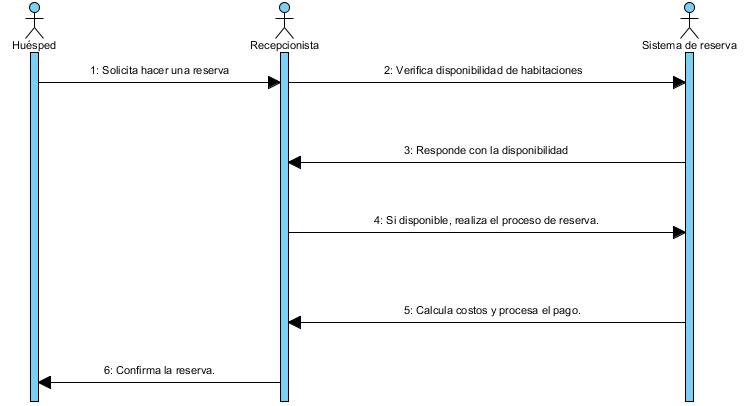
****

**Descripción de clases del Caso de uso <Proceso de pago>**

|  |  |  |
| --- | --- | --- |
| **No.** | **Nombre de la Clase** | **Descripción** |
| 1 | Cliente | Representa a la persona que realiza el pago. Este usuario interactúa con el recepcionista para elegir el método de pago, confirmar el pago y recibir el comprobante. |
| 2 | Recepcionista | Es el intermediario entre el cliente y el sistema. Verifica los detalles de la estancia, informa el monto a pagar, procesa el pago y notifica errores en caso de que algo falle. |
| 3 | Sistema de Pago | Es el sistema encargado de registrar las reservas y emitir comprobantes de pago una vez que la transacción ha sido procesada correctamente. |
| 4 | Pago | Representa la transacción realizada por el cliente, incluyendo detalles como el monto, el método de pago seleccionado y el estado actual de la operación (pendiente, completada o fallida). |

## 4.3 Diagramas de interacción

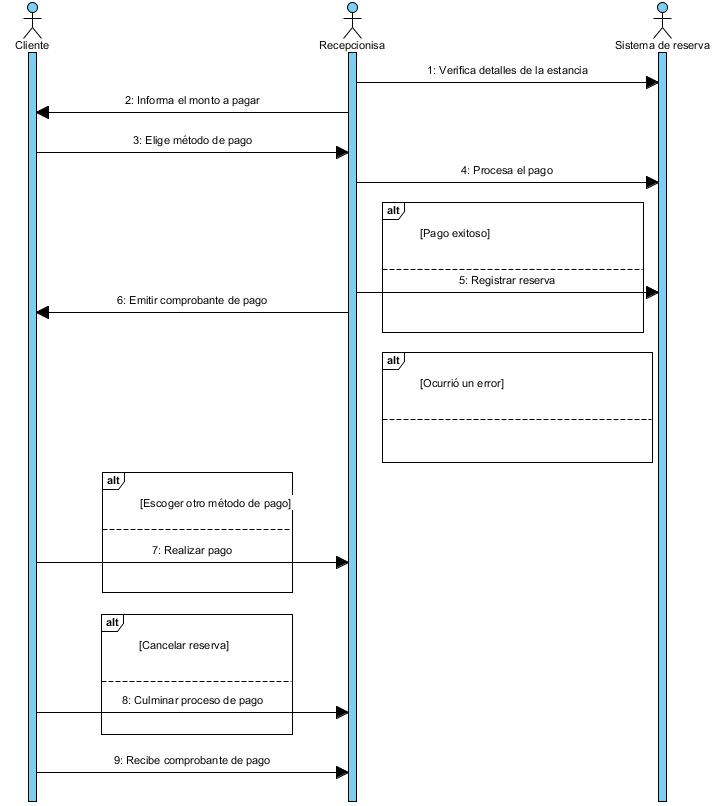
### 4.3.1 Diagrama de secuencia del Caso de uso Reserva de Habitaciones.



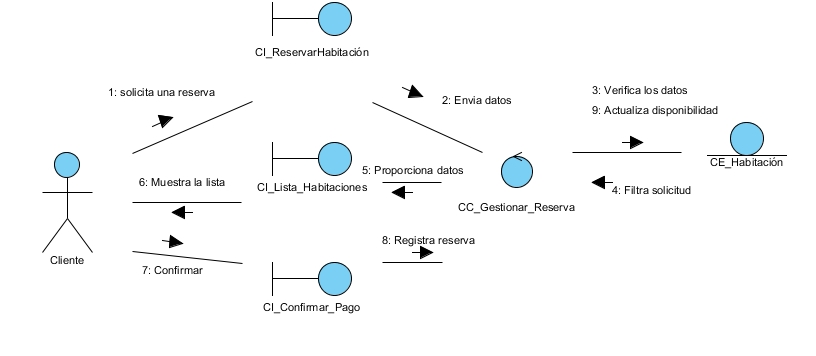
### 4.3.2 Diagrama de secuencia del Caso de uso Búsqueda de habitaciones.

### busqueda

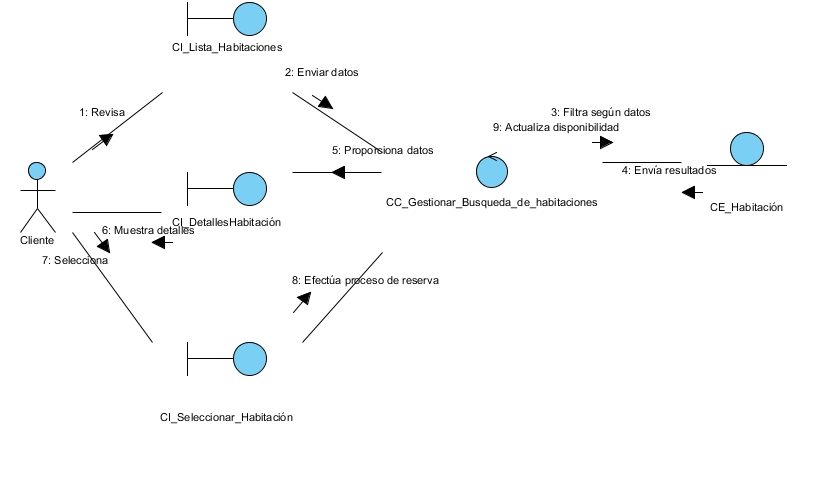
### 4.3.3 Diagrama de secuencia del Caso de uso Proceso de Pago.



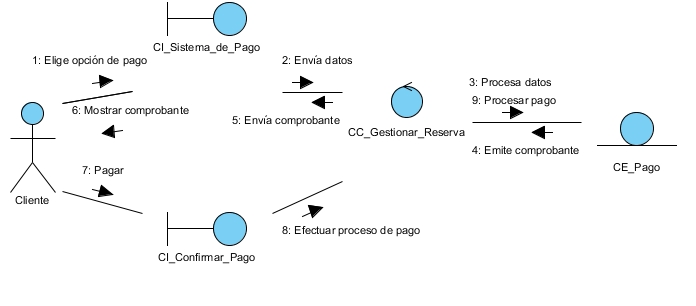
### 4.4.1 Diagrama de colaboración <Reserva de habitación>

****

### 4.4.2 Diagrama de colaboración <Buscar habitaciones disponibles>



### 4.4.3 Diagrama de colaboración <Proceso de pago>



## Conclusiones parciales

Se han desarrollado diagramas de clases e interacción para definir y manejar tanto la estructura como el comportamiento del sistema. El diagrama de clases es una herramienta de diseño esencial, ya que permite delinear la estructura interna de un programa con gran precisión. El diagrama de interacción es eficaz para gestionar el comportamiento del sistema en distintos escenarios. El diagrama de colaboración proporciona una visualización clara de las interacciones entre los actores y el sistema, facilitando el entendimiento de cómo se llevan a cabo los diferentes procesos.

`

# 

# Conclusiones generales

El desarrollo de este sistema de gestión hotelera para una cadena de hoteles ha demostrado ser un proyecto exitoso. La aplicación web resultante aborda de manera integral las necesidades de gestión del hotel, mejorando la eficiencia y la colaboración del personal.

El uso de un enfoque tradicional, específicamente la metodología RUP (Rational Unified Process), ha sido fundamental para el éxito del proyecto. Esta metodología ha permitido una estructuración clara y definida de todas las fases del desarrollo, promoviendo una colaboración continua entre los miembros del equipo de desarrollo.

Asimismo, la selección de herramientas y tecnologías adecuadas, como Django para el backend, JavaScript para el frontend y PostgreSQL como base de datos, ha contribuido a la solidez y escalabilidad de la solución. Estas tecnologías han permitido desarrollar una aplicación web robusta y eficiente, que cumple con los requisitos funcionales y no funcionales establecidos.

El modelado del sistema, a través de diagramas de casos de uso, clases, interacción y colaboración, ha sido fundamental para comprender la estructura y el comportamiento de la aplicación. Este enfoque ha facilitado la identificación y el abordaje de las necesidades específicas de la cadena hotelera, asegurando que la solución final sea adecuada y satisfaga las expectativas de los usuarios.

# Recomendaciones

**Optimizar la Experiencia del Usuario:** Asegúrate de que la interfaz de usuario sea intuitiva y fácil de navegar. La simplicidad y la claridad son clave para asegurar que tanto el personal del hotel como los huéspedes puedan utilizar el sistema eficientemente.

**Integración de Sistemas:** Implementa integraciones con otros sistemas clave como sistemas de pago, plataformas de reservas online, y sistemas de gestión de inventarios. Esto permitirá una gestión más unificada y eficiente.

**Seguridad y Protección de Datos:** Establece medidas robustas de seguridad para proteger la información confidencial de los huéspedes y del hotel. Cumple con todas las regulaciones de privacidad y protección de datos. **Escalabilidad y Flexibilidad:** Diseña el sistema de manera que pueda escalar y adaptarse a medida que crece el negocio. La capacidad de actualizar y expandir fácilmente el sistema es crucial para el crecimiento a largo plazo.

**Soporte y Mantenimiento:** Implementa un plan de soporte y mantenimiento continuo para asegurar que el sistema funcione sin problemas. Esto incluye actualizaciones regulares y la resolución rápida de cualquier problema técnico.

**Capacitación Continua:** Asegúrate de que el personal del hotel reciba capacitación continua sobre el uso del sistema. Esto no solo mejora la eficiencia, sino que también ayuda a los empleados a utilizar todas las funcionalidades del sistema de manera efectiva.

**Feedback de Usuarios:** Recolecta regularmente feedback de los usuarios (personal del hotel y huéspedes) para identificar áreas de mejora y adaptar el sistema a sus necesidades cambiantes.

# · Referencias Bibliográficas

*[Los párrafos se escribirán en Arial a 12 puntos y con espaciado 1,5 y con texto justificado y una línea en blanco como separador]*

*En las referencias solo se incluyen los trabajos citados explícitamente en el texto. Evite el uso de fuentes no confiables. Toda la bibliografía usada debe estar acotada en el cuerpo del documento siguiendo las pautas de las normas ISO 690 e ISO 690-2 de la forma (Apellido, año) ej. (Jiménez, 2010)*