|  |
| --- |
|  |
| Proyecto Cuarto con Vista    Rommies |
| Versión 1.0  Juan Jose Sanchez – 84874  Laura Tatiana Castaño Carlos Eduardo Rangel |
| Compañía de reservas de hotel |
|  |
| **Fecha 21/05/2025** |

**Tabla de Contenido**

[1. Descripción del Documento 5](#_Toc253781896)

[1.1. Propósito 5](#_Toc253781897)

[1.2. Audiencia 5](#_Toc253781898)

[1.3. Document Organization 5](#_Toc253781899)

[1.4. Terminologia y Definiciones 6](#_Toc253781900)

[1.5. Documentación del Proyecto Relacionada. 6](#_Toc253781901)

[1.6. Referencias Externas 6](#_Toc253781902)

[2. Visión General del Proyecto 6](#_Toc253781903)

[2.1. Contexto del negocio 6](#_Toc253781904)

[2.2. Contexto del Sistema 6](#_Toc253781905)

[2.3. Participantes Claves 7](#_Toc253781906)

[3. Drivers Arquitectonicos 7](#_Toc253781907)

[3.1. Restricciones de Negocio 7](#_Toc253781908)

[3.2. Restricciones Tecnicas. 7](#_Toc253781909)

[3.3. Requisitos Funcionales Claves. 7](#_Toc253781910)

[3.4. Requisitos de Atributos de Calidad 7](#_Toc253781911)

[3.4.1. Extensibilidad 7](#_Toc253781912)

[4. Tacticas de Arquitectura y Estilos Arquitectonicos. 8](#_Toc253781913)

[4.1. Tacticas 8](#_Toc253781914)

[4.2. Styles/Patterns 8](#_Toc253781915)

[5. Vistas de Arquitectura 8](#_Toc253781916)

[5.1. Vista Lógica 8](#_Toc253781917)

[5.2. Vista de Interacción 8](#_Toc253781918)

[5.3. Vista de Desarrollo 9](#_Toc253781919)

[5.4. Vista de Despliegue 9](#_Toc253781920)

[5.5. Diseño de Interfaces 9](#_Toc253781921)

[5.6. Plataforma Tecnológica 9](#_Toc253781922)

[6. Directorio 9](#_Toc253781923)

[6.1. Glosario 9](#_Toc253781924)

[6.2. Acronimos 9](#_Toc253781925)

[7. Apendice 9](#_Toc253781926)

[7.1. Lista de Pruebas de Concepto 9](#_Toc253781927)

[1. Descripción del Documento 2](#_Toc644912665)

[1.1. Propósito 2](#_Toc8216112)

[2. Visión General del Proyecto 3](#_Toc1341187842)

[2.1. Contexto del negocio 3](#_Toc1110609322)

[2.2. Contexto del Sistema 3](#_Toc2100055555)

[2.3. Interesados (Stakeholders) 3](#_Toc345268103)

[2.3.1. Stakeholders funcionales (usan el sistema directamente): 3](#_Toc1601221820)

[ Huéspedes: reservan y consultan cuartos desde diferentes canales (web, móvil, recepción). 3](#_Toc1268528584)

[ Recepcionistas: realizan reservas presenciales o telefónicas, gestionan disponibilidad. 3](#_Toc722500517)

[ Equipo de limpieza y mantenimiento: usan dispositivos para ver tareas, actualizar estados. 3](#_Toc1702542381)

[2.3.2. Stakeholders de negocio (influyen en requerimientos, estrategias o visión): 3](#_Toc165569651)

[2.3.3. Stakeholders técnicos: 3](#_Toc723653353)

[3. Requisitos y calidad 4](#_Toc1084337575)

[3.1. Requisitos Funcionales. 4](#_Toc182705469)

[3.2. Requisitos no funcionales 4](#_Toc1516210708)

[3.3. Atributos de Calidad. 5](#_Toc1166815942)

[3.4. Drivers Arquitectonicos 5](#_Toc196908112)

[4. Tácticas de Arquitectura y Estilos Arquitectónicos. 6](#_Toc1492546834)

[4.1. Tacticas de atributos de calidad 6](#_Toc84504411)

[2. Rendimiento 7](#_Toc1363844240)

[3. Interoperabilidad 7](#_Toc1494317552)

[4. Portabilidad 7](#_Toc1247690992)

[5. Usabilidad 8](#_Toc570495445)

[3. Rendimiento y Escalabilidad 8](#_Toc1834024999)

[5. Interoperabilidad 8](#_Toc46722590)

[4.2. Patrones Arquitectónicos 9](#_Toc330797489)

[4.3. Tácticas de patrones Arquitectónicos 9](#_Toc1382736813)

[3.Circuit Breaker 10](#_Toc1893755066)

[4. API Gateway 10](#_Toc1888333074)

[5. Vistas de Arquitectura 10](#_Toc829694896)

[5.1. Escenarios identificados 10](#_Toc1575805696)

[5.2. Vista Lógica 13](#_Toc213210974)

[5.3. Vista de Implementación 18](#_Toc651979116)

[5.4. Vista de Procesos 19](#_Toc296765777)

[5.5. Vista Física 21](#_Toc1247189637)

[6. Directorio 22](#_Toc1170342484)

[6.1. Glosario 22](#_Toc1107895074)

# Descripción del Documento

## Propósito

El propósito de este documento es plasmar de forma concisa la arquitectura propuesta para suplir con las necesidades y retos presentes para los interesados descritos más adelante. Para el desarrollo de esta arquitectura se identificarán propiedades y características esenciales como los requerimientos funcionales, requerimientos no funcionales, atributos de calidad, skateholders, matriz de ponderación de atributos de calidad, tácticas y estrategias para abordar el proyecto. Se diseña la arquitectura propuesta usando el modelo de 4 + 1 vistas en el entorno de Amazon Web Services (AWS) empleando las tácticas y estrategias seleccionadas.

# Visión General del Proyecto

## Contexto del negocio

La compañía de reservas quiere crear un sistema moderno que transforme cómo se hacen las reservas en resorts y spas exclusivos. No se trata solo de reservar una habitación, sino de permitirle al huésped elegir el cuarto exacto en el que quiere quedarse, viendo fotos y su ubicación dentro del hotel. Se desea tener listo el sistema antes de la temporada alta. Además, la compañía está invirtiendo en cerraduras inteligentes que se abren con el celular. Sin embargo, hay algunos retos: hay promesas que a veces no se cumplen y el equipo de ventas tiene mucho peso en las decisiones, lo que puede influir en las prioridades del proyecto

## Contexto del Sistema

El sistema que se va a construir será una plataforma web que permita hacer reservas desde varios canales: página web, celular, teléfono o directamente en la recepción del hotel. Los huéspedes podrán elegir una habitación por tipo o incluso un cuarto específico viendo fotos y su ubicación. Además, el sistema ayudará a los empleados a gestionar el estado de cada habitación (si está disponible, reservada o necesita limpieza) y permitirá organizar al personal de mantenimiento según las prioridades y próximas reservas. Este equipo usará dispositivos que estarán pegados a sus carros de limpieza, para recibir instrucciones en tiempo real. Todo esto se relacionará con el sistema actual de reservas, que seguirá manejando pagos y registros. Es un sistema que tiene que estar listo pronto, integrarse bien con lo que ya existe y estar preparado para incorporar nuevas tecnologías como cerraduras inteligentes

## Interesados (Stakeholders)

## **Stakeholders funcionales (usan el sistema directamente):**

## **Huéspedes:** reservan y consultan cuartos desde diferentes canales (web, móvil, recepción).

## **Recepcionistas:** realizan reservas presenciales o telefónicas, gestionan disponibilidad.

## **Equipo de limpieza y mantenimiento:** usan dispositivos para ver tareas, actualizar estados.

## **Stakeholders de negocio (influyen en requerimientos, estrategias o visión):**

* **Gerencia del hotel:** Toma decisiones importantes y se asegura que el sistema cumpla con las necesidades del negocio
* **Ventas y marketing:** definen qué cuartos promocionar

## **Stakeholders técnicos:**

* **Equipo Técnico:** Grupo responsable de diseñar, desarrollar, probar, desplegar y mantener el sistema, incluyendo software, infraestructura, integración y soporte tecnológico.

# Requisitos y calidad

## Requisitos Funcionales.

|  |  |  |
| --- | --- | --- |
| **ID** | **Descripción** | **Detalle** |
| **RF.001** | Múltiples canales de reserva | El sistema debe permitir que los huéspedes realicen reservas de habitaciones a través de plataforma web, aplicación móvil, llamada telefónica o presencialmente en recepción. |
| **RF.002** | Selección de habitaciones | El sistema debe permitir que los huéspedes seleccionen una habitación específica basándose en fotografías y ubicación dentro del hotel o elijan por tipo de habitación (estándar, de lujo, o suite). |
| **RF.003** | Actualización del estado de habitaciones | El sistema debe mantener actualizado el estado de todas las habitaciones en tiempo real (disponible, reservada, ocupada, en limpieza, o fuera de servicio). |
| **RF.004** | Generación automática de órdenes de limpieza | El sistema debe generar órdenes de servicio de limpieza automáticamente cuando una habitación cambie su estado a "lista para limpieza". |
| **RF.005** | Visualización de lista de limpieza en dispositivos móviles | El sistema debe mostrar la lista de habitaciones priorizadas para limpieza en los dispositivos instalados en los carros de limpieza del personal de mantenimiento. |
| **RF.006** | Priorización de limpieza | El sistema debe priorizar las órdenes de limpieza basándose en la proximidad de nuevas reservas y según el nivel de prioridad establecidos. |
| **RF.007** | Acceso del personal a información de habitaciones | Los empleados deben poder acceder al sistema y ver los estados, tareas y prioridades de las habitaciones. |
| **RF.008** | Gestión de estados y tareas de habitaciones | Los empleados deben poder cambiar y asignar el estado y tareas de las habitaciones. |

## Requisitos no funcionales

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID** | **Descripción** | **Detalle** | **Metricas** | **Interesados** |
| **RNF.001** | Accesibilidad para múltiples usuarios (escalabilidad) | Debe soportar cientos de huéspedes y menos de 20 empleados concurrentemente sin afectar el rendimiento. | Soporte mínimo de 500 usuarios concurrentes (480 huéspedes + 20 empleados) | Huéspedes, Equipo de limpieza, Equipo Técnico |
| **RNF.002** | Tiempo de respuesta (rendimiento) | El sistema debe responder de manera rapida y eficiente, cargando secciones como reserva de habitacion por tipo, seleccionar fecha, fotos de los cuartos y demas en menos de 2 segundos | > 2 segundos | Huéspedes, Recepcionistas |
| **RNF.003** | Integración con sistemas existentes | Debe conectarse con el sistema actual de pagos y registro. | Tiempo máximo de sincronización con sistema externo: 2 seg. Éxito en integración ≥ 99.5%. | Equipo Técnico, Recepcionistas |
| **RNF.004** | Multiplataforma | El sistema debe estar disponible a través de web, app móvil, recepción y teléfono | Disponibilidad de interfaz funcional en los 4 canales al 100%. Tiempo de respuesta < 3 seg en web y móvil. | Huéspedes, Recepcionistas |
| **RNF.005** | Interfaz intuitiva (Usabilidad) | La interfaz del sistema debe ser intuitiva y adaptada al rol del usuario (huésped, recepcionista, personal de limpieza), minimizando la necesidad de capacitación técnica. | El 95% de los usuarios debe poder completar su flujo principal (reserva o actualización de estado de limpieza) en menos de 3 pasos y en menos de 30 segundos tras 10 minutos de inducción. | Equipo de limpieza, Huéspedes |

## **Atributos de Calidad.**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **SkateHolders** | **Escalabilidad** | **Rendimiento** | **Interoperabilidad** | **Portabilidad** | **Usabilidad** | **Total** |
| **Huéspedes** | 0 | 35 | 10 | 15 | 40 | 100 |
| **Recepcionistas** | 0 | 40 | 30 | 0 | 30 | 100 |
| **Equipo de limpieza y mantenimiento** | 0 | 20 | 0 | 0 | 80 | 100 |
| **Gerencia del hotel** | 20 | 20 | 10 | 30 | 20 | 100 |
| **Ventas y marketing** | 10 | 25 | 10 | 30 | 25 | 100 |
| **Equipo Técnico** | 25 | 25 | 25 | 15 | 10 | 100 |
| **Promedio %** | 9.166666667 | 27.5 | 14.16666667 | 15 | 34.16666667 | 100 |

## **Drivers Arquitectonicos**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Atributo** | **Descripción** | **Subcategoria** | **Metrica** | **Impacto** | **Dificultad** | **Peso** | **Valor** |
| **Escalabilidad** | Capacidad del sistema para manejar un aumento de usuarios, datos o procesos sin perder rendimiento | Capacidad de carga concurrente | Soporta 500 usuarios concurrentes sin degradación del servicio. | 3 | 2 | 9.166666667 | 45.83333333 |
| **Usabilidad** | Facilidad con la que los usuarios (huéspedes, empleados, etc.) interactúan eficazmente con el sistema. | Facilidad de aprendizaje / Eficiencia de uso | 95% de usuarios completa su flujo principal en < 30 segundos tras 10 min de inducción. | 2 | 2 | 34.16666667 | 136.6666667 |
| **Portabilidad** | Capacidad del sistema para integrarse correctamente con otros sistemas existentes (pagos, registros, cerraduras). | Compatibilidad e integración externa | Integración con sistemas externos exitosa ≥ 99.5%, sincronización ≤ 2s. | 3 | 3 | 15 | 90 |

Organización de drivers arquitectónicos de mayor a menor con respecto a su valor final

|  |  |
| --- | --- |
| **Atributo** | **Valor** |
| **Usabilidad** | 136.6666667 |
| **Portabilidad** | 90 |
| **Escalabilidad** | 45.83333333 |

# Tácticas de Arquitectura y Estilos Arquitectónicos.

## Tacticas de atributos de calidad

**1. Escalabilidad**

El sistema debe poder atender picos de demanda, especialmente durante la temporada alta

* Balanceo de carga para distribuir tráfico entre servidores.
* Auto escalado horizontal en la nube.
* Caché para recursos frecuentes (imágenes, disponibilidad de cuartos).
* Colas asincrónicas para tareas no críticas.
* Separación de dominios como las reservas, limpieza de modo que se aislé la carga.

### **2. Rendimiento**

El sistema debe responder de manera rápida al cargar funcionalidades clave como la reserva de habitaciones o la visualización de fotos

**Tácticas:**

* Reducir las llamadas al backend por cada acción.
* Monitoreo de tiempos de respuesta con herramientas de monitoreo (APM).

### **3. Interoperabilidad**

El sistema debe integrarse sin fricciones con el sistema actual de pagos y registro, manteniendo la sincronización y evitando duplicidad de datos.

**Tácticas:**

* Uso de APIs bien definidas y documentadas.
* Adaptadores de integración para normalizar formatos de datos entre sistemas.
* Validación y transformación de datos en puntos de entrada/salida.
* Monitorización de los puntos de integración para detectar fallos rápidamente.

### **4. Portabilidad**

El sistema debe funcionar correctamente a través de múltiples canales (web, móvil, recepción y teléfono).

**Tácticas:**

* Desarrollo con frameworks multiplataforma como React para web o móvil.
* Diseño desacoplado del backend y frontend (API-first).
* Adaptación de interfaces a distintos dispositivos (responsive y mobile-first design).

### **5. Usabilidad**

El sistema debe ser intuitivo tanto para huéspedes como para empleados del hotel (que pueden tener bajo nivel técnico).

**Tácticas:**

* Pruebas de usabilidad con usuarios reales.
* Interfaces adaptadas según rol (huésped, limpieza, recepción).
* Consistencia visual y navegación simple (estándares UX/UI).
* Feedback inmediato al usuario (confirmaciones visuales y mensajes claros).

### **3. Rendimiento y Escalabilidad**

El sistema debe poder soportar muchos usuarios simultáneamente sin degradarse, especialmente en temporada alta.

**Tácticas:**

* Uso de cache (Redis, CDN para imágenes de habitaciones).
* Lazy loading en imágenes de cuartos y contenido pesado.
* Paginar listados y datos grandes.
* Escalado horizontal (contenedores y microservicios).
* Optimización de consultas a base de datos.

### **5. Interoperabilidad**

Integrarse con el sistema de reservas existente y, en el futuro, con hardware como cerraduras inteligentes.

**Tácticas:**

* Exposición de APIs RESTful bien documentadas.
* Adaptadores o gateways para interactuar con sistemas legados.
* Uso de formatos estándar (JSON, XML, OAuth).
* Arquitectura orientada a eventos para facilitar extensiones (pub/sub).

## Patrones Arquitectónicos

1. Microservicios

* Dividir la funcionalidad en servicios independientes como la gestión de habitaciones, reservas específicas, administración de limpieza, y notificaciones para el personal permite que cada componente pueda ser desarrollado, desplegado y escalado de forma autónoma, acelerando la entrega en una temporada alta crítica. Esta independencia facilita también la integración de tecnologías avanzadas, como las cerraduras inteligentes que se abren desde el celular

1. Patrón Bus de Evento (Event Bus)

* Este patrón es especialmente útil para coordinar en tiempo real las actualizaciones entre las reservas, la gestión de limpieza y los dispositivos IoT como las cerraduras inteligentes.

1. Circuit Breaker

* Protege las integraciones con sistemas externos (sistema de pagos existente, cerraduras inteligentes) de fallos en cascada. Si un servicio externo falla repetidamente, el circuit breaker se "abre" temporalmente y proporciona respuestas alternativas, ademas es fundamental para la interoperabilidad robusta y para mantener la disponibilidad del sistema durante la temporada alta.

1. Api Gateway

* Facilita la gestión de múltiples servicios desde un único punto de entrada. Es útil para manejar autenticación, login, enrutamiento, control de versiones y evitar exponer servicios internos directamente.

## Tácticas de patrones Arquitectónicos

## Microservicios

* **Descomposición por dominios**: Separar en servicios por funcionalidad (reservas, limpieza, notificaciones, gestión de habitaciones)
* **API contratos estables**: Definir interfaces claras entre servicios

**Bus de Eventos (Event Bus)**

* **Broker de mensajes centralizado**: Kafka como sistema de mensajería
* **Suscripción por interés**: Cada servicio se suscribe solo a los eventos relevantes

### **Circuit Breaker**

* **Respuestas fallback**: Proporcionar respuestas alternativas cuando los servicios externos no están disponibles
* **Reintentos inteligentes**: Implementar lógica de reintentos exponenciales para recuperación

### **4. API Gateway**

* **Punto de entrada único**: Centralizar el acceso a todos los microservicios
* **Enrutamiento inteligente**: Dirigir peticiones al servicio correspondiente

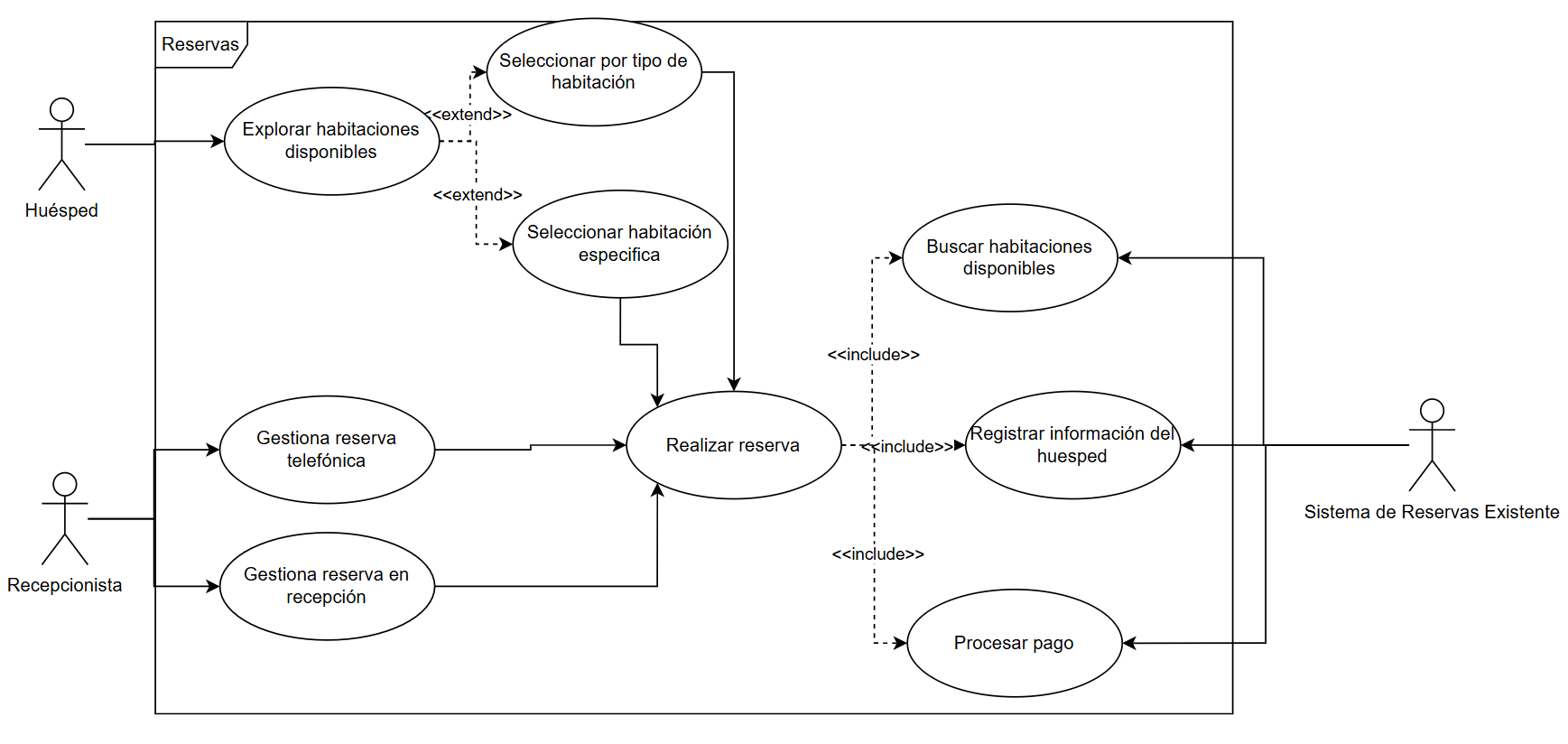
# Vistas de Arquitectura

## Escenarios identificados

**1 Escenario: Reservas**

Este escenario nos muestra el proceso de reservas de habitaciones en un hotel, tanto por parte del huésped como del recepcionista (presencial o telefónica). El flujo principal gira en torno al caso de uso "Realizar reserva". Se representan actores externos, casos de uso primarios, secundarios, relaciones de inclusión y extensión.

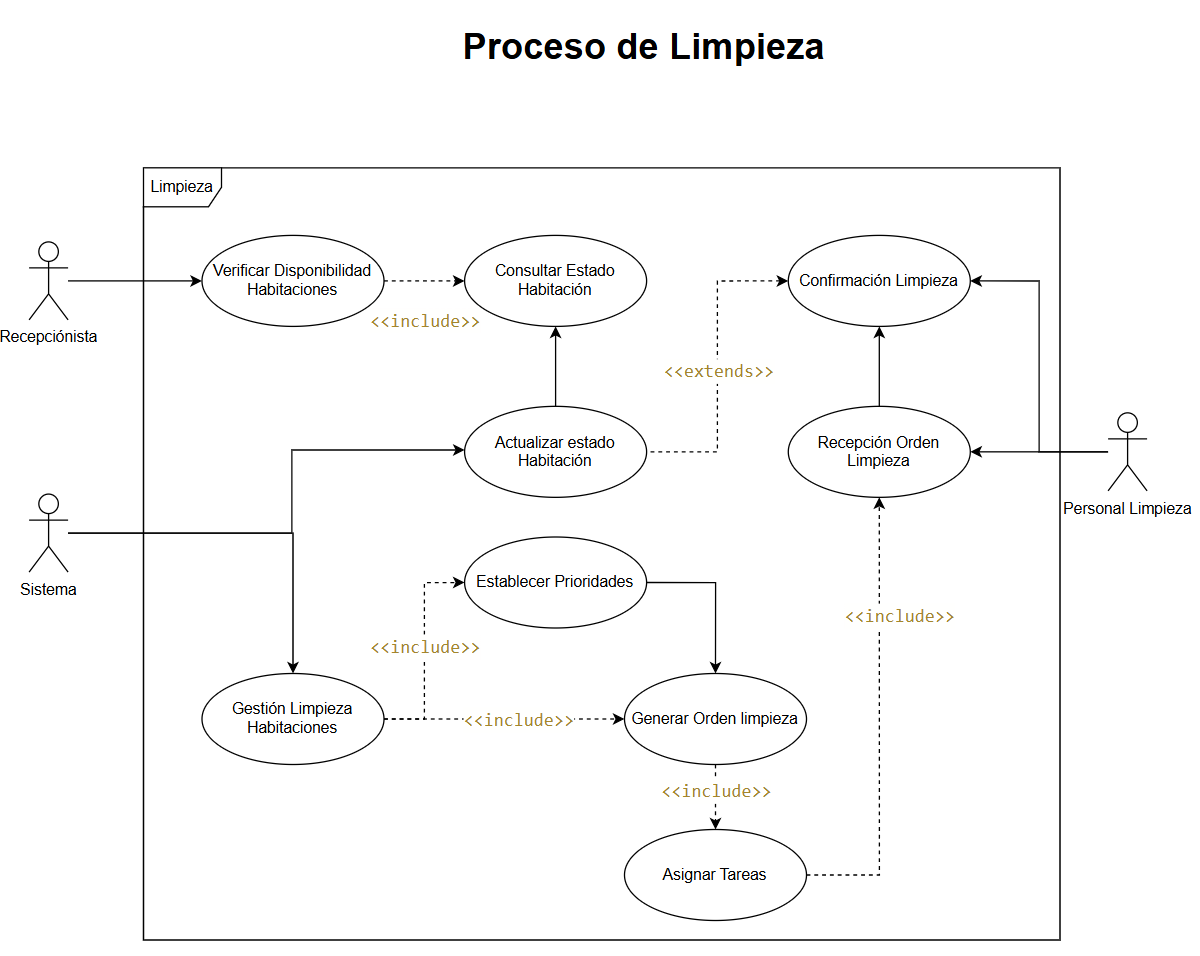
**Diagrama de Casos de uso**

**Catálogo de Elementos y relaciones**

|  |  |  |
| --- | --- | --- |
| **Actor** | **Elemento UML (Caso de Uso)** | **Relación** |
| Huésped | Explorar habitaciones disponibles | Asociación directa |
| Huésped | Seleccionar por tipo de habitación | <<extend>> desde "Explorar habitaciones disponibles" |
| Huésped | Seleccionar habitación específica | <<extend>> desde "Explorar habitaciones disponibles" |
| Huésped | Realizar reserva | Asociación directa |
| Recepcionista | Gestiona reserva telefónica | Asociación directa |
| Recepcionista | Gestiona reserva en recepción | Asociación directa |
| Recepcionista | Realizar reserva | Asociación directa |
| Sistema de Reservas Existente | Buscar habitaciones disponibles | Asociación directa |
| Sistema de Reservas Existente | Registrar información del huésped | Asociación directa |
| Sistema de Reservas Existente | Procesar pago | Asociación directa |

**2 Escenario: Proceso de limpieza**

Este escenario representa el proceso de limpieza en donde se coordinan y supervisan las tareas de aseo en las habitaciones. El sistema permite organizar todo: desde verificar la disponibilidad y el estado de las habitaciones, hasta definir prioridades, generar órdenes y asignarlas al personal de limpieza. Finalmente, el personal confirma que la tarea fue realizada. Con esto se asegura que cada habitación esté lista a tiempo y cumpla con los estándares del hotel antes de recibir a un nuevo huésped



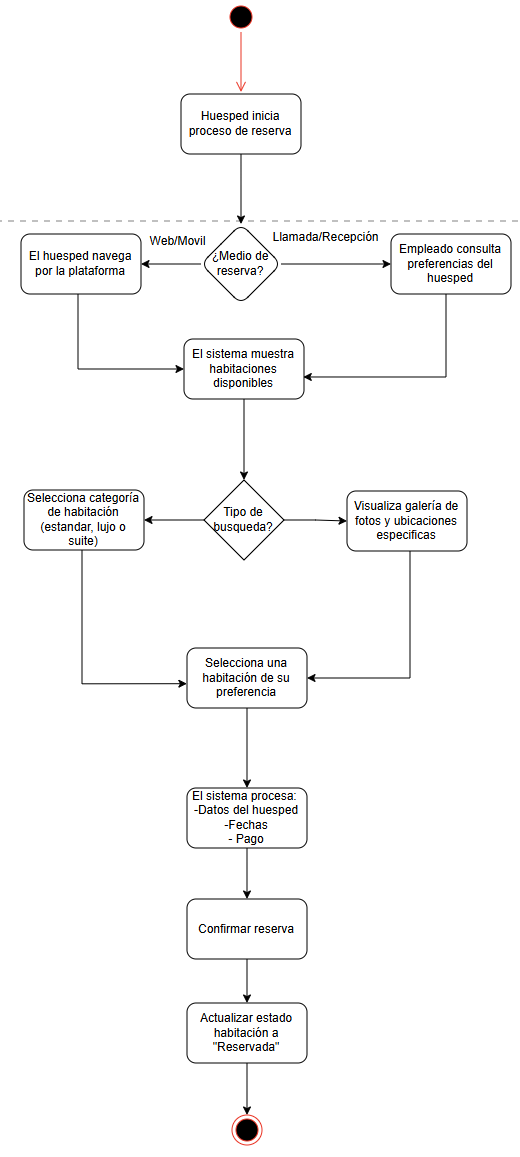
|  |  |  |
| --- | --- | --- |
| **Actor** | **Elemento UML (Caso de Uso)** | **Relación** |
| Recepcionista | Verificar Disponibilidad Habitaciones | Interacción directa del recepcionista con el sistema para obtener habitaciones disponibles. |
| Recepcionista | Consultar Estado Habitación | Incluido en “Verificar Disponibilidad Habitaciones”, permite ver detalles del estado de limpieza. |
| Sistema | Gestión Limpieza Habitaciones | Inicia el proceso general de gestión de limpieza basado en la información del sistema. |
| Sistema | Establecer Prioridades | Incluido dentro de “Gestión Limpieza Habitaciones”,define criterios para decidir qué habitaciones limpiar primero. |
| Sistema | Generar Orden Limpieza | Incluido después de establecer prioridades; crea órdenes específicas para el personal de limpieza. |
| Sistema | Asignar Tareas | Incluido dentro de “Generar Orden Limpieza”,detalla las habitaciones y tareas específicas para cada miembro del personal. |
| Personal de Limpieza | Recepción Orden Limpieza | Caso de uso donde el personal recibe digitalmente las tareas desde el sistema. |
| Personal de Limpieza | Confirmación Limpieza | Se extiende desde “Recepción Orden Limpieza”, notifica al sistema que las tareas asignadas han sido completadas. |
| Sistema | Actualizar estado Habitación | Acción que se activa una vez se confirma la limpieza, cambiando el estado de la habitación (ej. de “sucia” a “limpia disponible”). |

## Vista Lógica

La vista lógica representa la estructura funcional del sistema, enfocándose en los procesos, decisiones y comportamientos que permiten cumplir con los requerimientos del negocio.

**Diagrama de actividades Escenario 1**

El diagrama ilustra el flujo lógico del sistema de reservas desde el inicio de la interacción del huésped hasta la confirmación de la reserva, mostrando cómo se conecta cada decisión. Este diagrama se acopla a dicha vista al mostrar el flujo detallado del proceso de reserva, desde la interacción inicial del huésped hasta la actualización del estado de la habitación, incluyendo las distintas rutas según el medio de reserva y el tipo de búsqueda

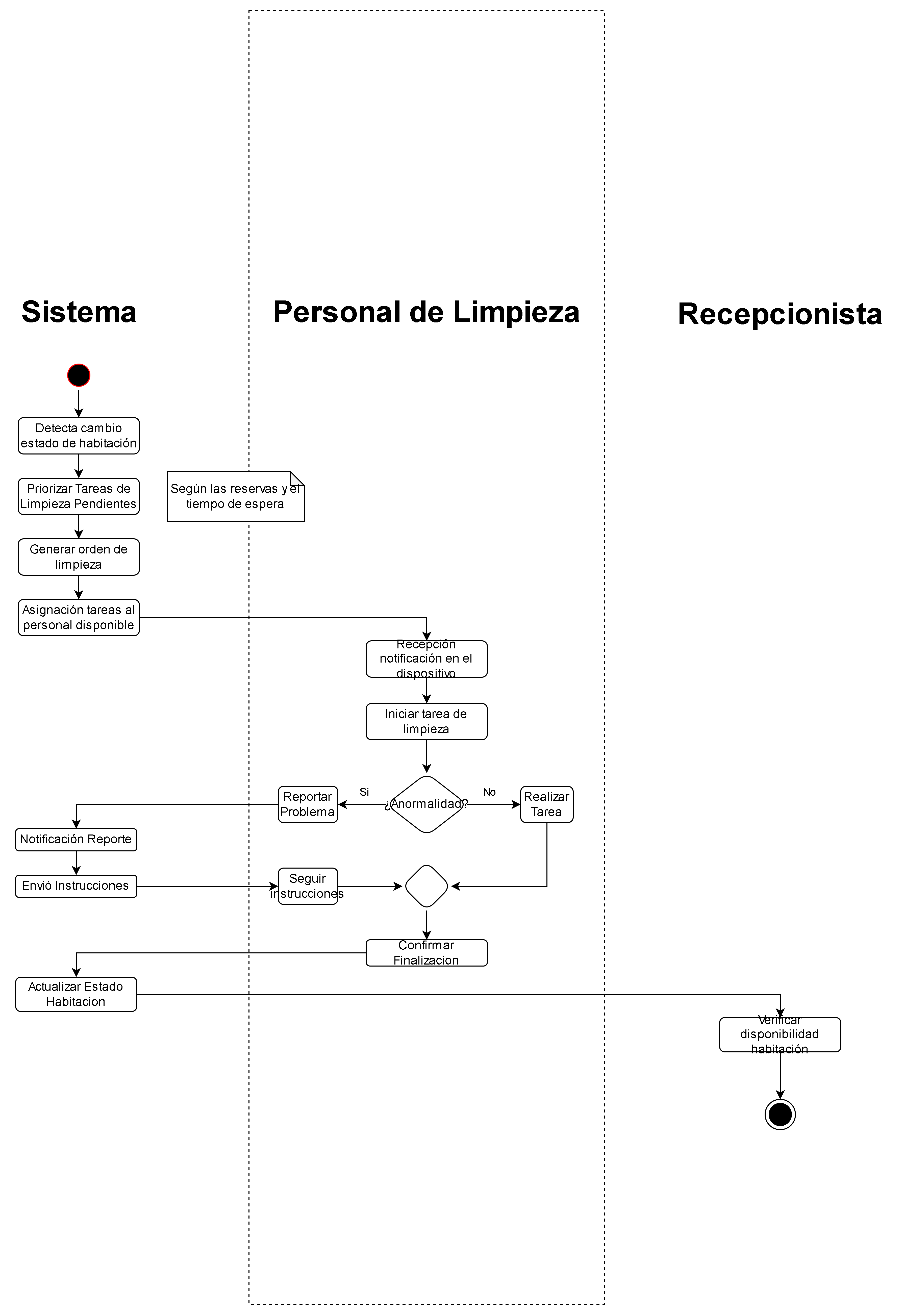


**Catálogo de Elementos y relaciones**

|  |  |  |
| --- | --- | --- |
| **Elemento UML** | **Relaciones** | **Descripción** |
| Actividad: Iniciar reserva | Flujo hacia la decisión del medio de reserva | Punto de entrada del proceso donde se inicia la intención de reservar una habitación |
| Decisión: Medio de reserva | Dirige el flujo hacia plataforma digital o atención personalizada | Determina si la reserva se hará por web/móvil o de forma asistida (empleado) |
| Actividad: Navegar sistema | Flujo desde "web o móvil" hacia selección de habitación | Acción en la que se exploran opciones y se visualiza la información de las habitaciones. |
| Actividad: Consultar preferencia | Flujo desde atención personalizada hacia la decisión de tipo de búsqueda | Paso donde se identifican las preferencias del huésped para continuar el proceso. |
| Decisión: Tipo de búsqueda | Dirige el flujo hacia categoría o habitación específica | Define si se busca por tipo de habitación (estándar, suite, etc.) o por una habitación concreta con fotos y ubicación |
| Actividad: Seleccionar habitación | Continúa el flujo hacia el registro de datos | Acción en la que se elige la habitación deseada, de acuerdo con la búsqueda realizada. |
| Actividad: Procesar datos y pago | Conecta con el sistema de reservas existente | Paso donde se ingresan y gestionan los datos del huésped, fechas y forma de pago. |
| Actividad: Confirmar reserva | Flujo final, seguido por la actualización de estado | Se genera la confirmación de la reserva y se comunica al sistema. |
| Actividad: Actualizar estado de habitación | Finaliza el flujo lógico del proceso | Actualiza el estado interno de la habitación a “reservada”, permitiendo su correcta gestión futura por el personal del hotel. |

**Diagrama de actividades Escenario 2**

El diagrama de actividades de limpieza en "Cuarto con Vista" describe el proceso detallado para garantizar que cada habitación esté lista para su próximo huésped. Tres actores principales participan en este flujo de trabajo: el Sistema, el Personal de Limpieza y el Recepcionista. El proceso comienza cuando el Sistema detecta un cambio en el estado de una habitación, generalmente después de un check-out, y asigna tareas de limpieza según las próximas reservas y el tiempo disponible. El Personal de Limpieza recibe la notificación, evalúa la situación y procede con la tarea, reportando cualquier anomalía si es necesario y confirmando la finalización una vez concluida. Luego, el Sistema actualiza el estado de la habitación, permitiendo al Recepcionista verificar la disponibilidad para nuevas reservas sin demoras. Este flujo asegura que el trabajo se realice de manera eficiente, manteniendo al personal informado y mejorando la experiencia del cliente con habitaciones listas en el momento adecuado.



**Catálogo de Elementos y relaciones**

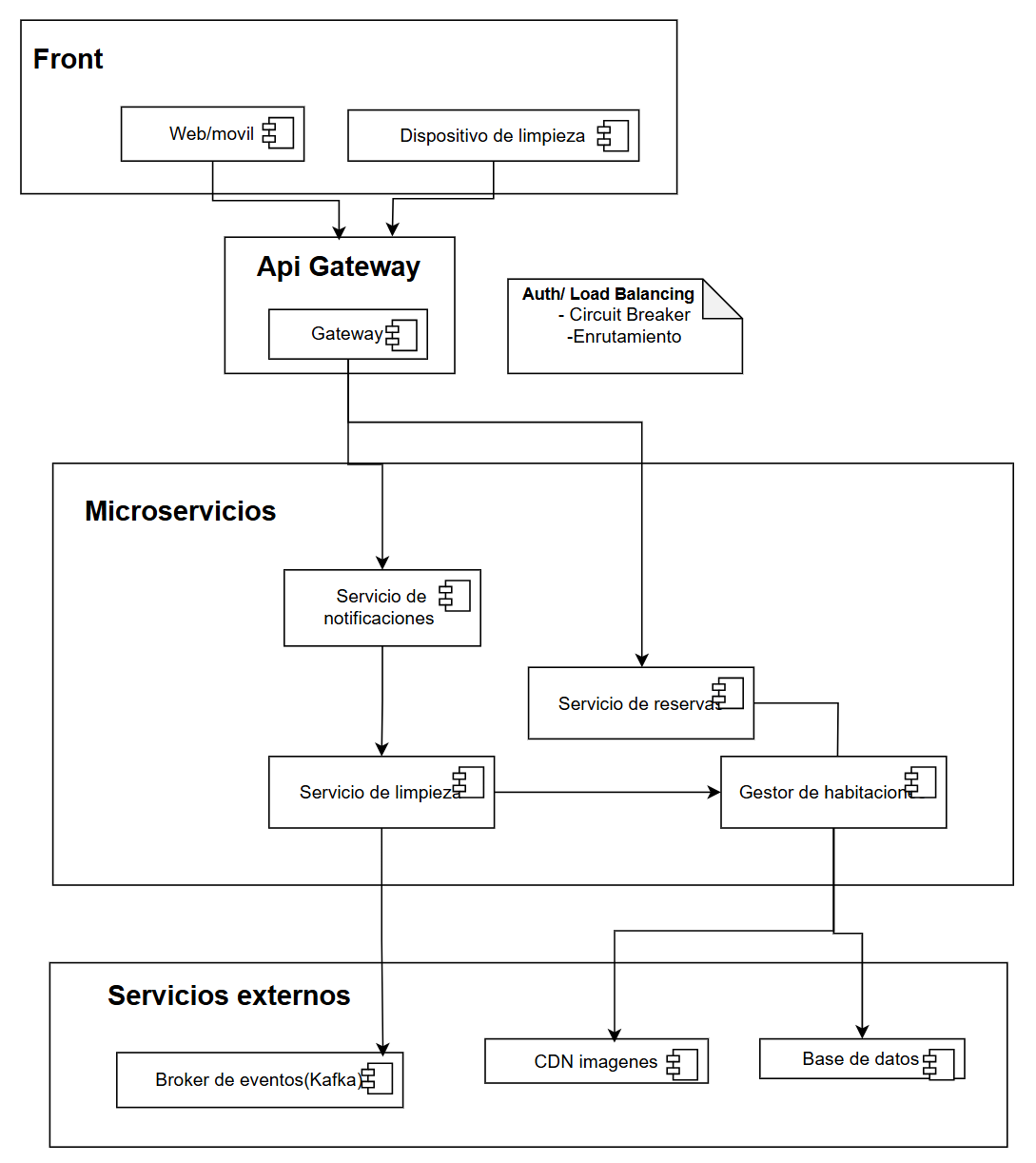
|  |  |  |
| --- | --- | --- |
| **Elemento UML** | **Relaciones** | **Descripción** |
| **Nodo Inicial** | Se conecta con "Detectar y Priorizar Limpieza" | Punto de inicio del flujo de actividades del proceso de limpieza. |
| **Carriles (Swimlanes)** | Organizan actividades por actor (Sistema, Personal de Limpieza, Recepción) | División visual que muestra qué actor es responsable de cada actividad. |
| **Actividad: Detectar y Priorizar Limpieza** | Precede a "Asignar y Notificar Tarea" | El sistema identifica cambios de estado de habitación y prioriza tareas según reservas y tiempo de espera. |
| **Actividad: Asignar y Notificar Tarea** | Conecta con "Ejecutar Limpieza" | El sistema asigna tareas al personal disponible y envía notificación al dispositivo móvil. |
| **Actividad: Ejecutar Limpieza** | Precede a decisión "¿Problema Detectado?" | El personal inicia y realiza la limpieza de la habitación asignada. |
| **Decisión: ¿Problema Detectado?** | Sí: conecta con "Gestionar Incidencia"; No: conecta con "Finalizar y Actualizar" | Evalúa si se presenta alguna situación inesperada durante la limpieza. |
| **Actividad: Gestionar Incidencia** | Conecta con "Finalizar y Actualizar" | El personal reporta el problema, recibe instrucciones del sistema y resuelve la incidencia. |
| **Actividad: Finalizar y Actualizar** | Conecta con "Verificar Disponibilidad" | El personal confirma finalización y el sistema actualiza el estado de la habitación a disponible. |
| **Actividad: Verificar Disponibilidad** | Precede al Nodo Final | La recepción verifica que la habitación esté lista para nuevas reservas. |
| **Nodo Final** | Finaliza el flujo | Indica el término exitoso del proceso de limpieza de la habitación. |

## Vista de Implementación

**Diagrama de Componentes**

Esta vista refleja la organización física del software, mostrando como los distintos componentes como microservicios se interconectan.

Este diagrama representa la arquitectura de una solución basada en microservicios, en donde los distintos servicios se comunican entre sí y con servicios externos mediante un API Gateway. El front (web/móvil o dispositivo de limpieza) se comunica con el backend a través del gateway



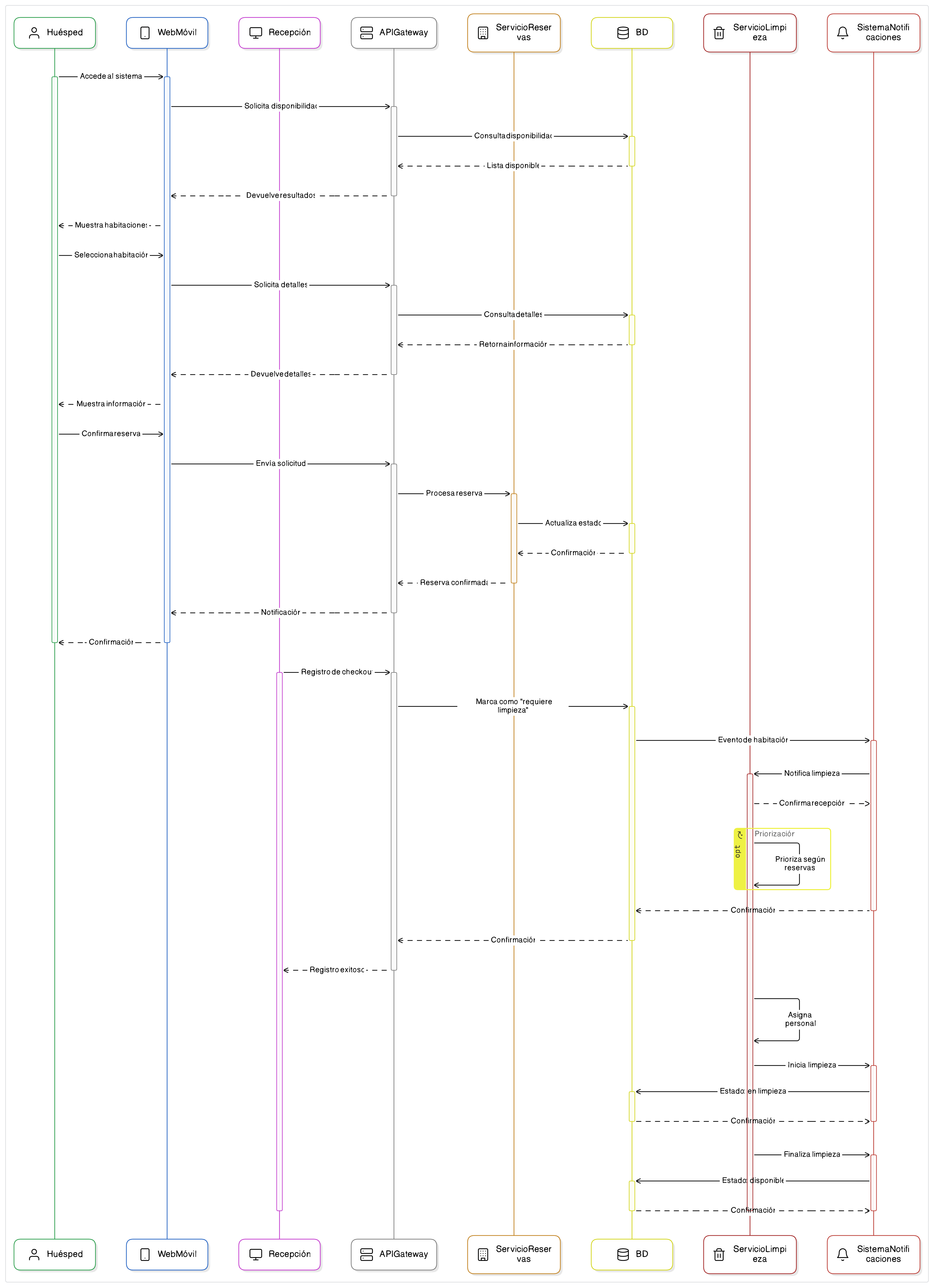
**Catálogo de Elementos y relaciones**

|  |  |  |
| --- | --- | --- |
| **Subsistema** | **Elementos UML** | **Relaciones principales** |
| Front | Web/móvil, Dispositivo de limpieza | Se comunican con el API Gateway. |
| API Gateway | Gateway, Componente de Auth/Load Balancing | Gestiona el acceso a los microservicios. Relación directa con todos los servicios del backend |
| Microservicios | Servicio de notificaciones, limpieza, reservas, gestor de habitación | - El servicio de limpieza depende del de notificaciones y se comunica con el gestor de habitación.El servicio de reservas también interactúa con el gestor. |
| Servicios externos | Broker de eventos (Kafka), CDN imágenes, Base de datos | - El servicio de limpieza pública eventos en Kafka.El gestor de habitación accede a CDN de imágenes y a la base de datos para su funcionamiento. |

## Vista de Procesos

**Diagrama de Secuencia**

Esta vista representa los principales flujos de interacción entre los actores del sistema de reservas del hotel "Cuarto con vista". A través del diagrama de secuencia se detallan procesos clave como la consulta, selección y reserva de habitaciones por parte de los huéspedes, así como la gestión interna de limpieza tras el checkout. El sistema está orientado a resorts de lujo y spas, optimizado para atender a cientos de huéspedes con una operación eficiente por parte de menos de 20 empleados.



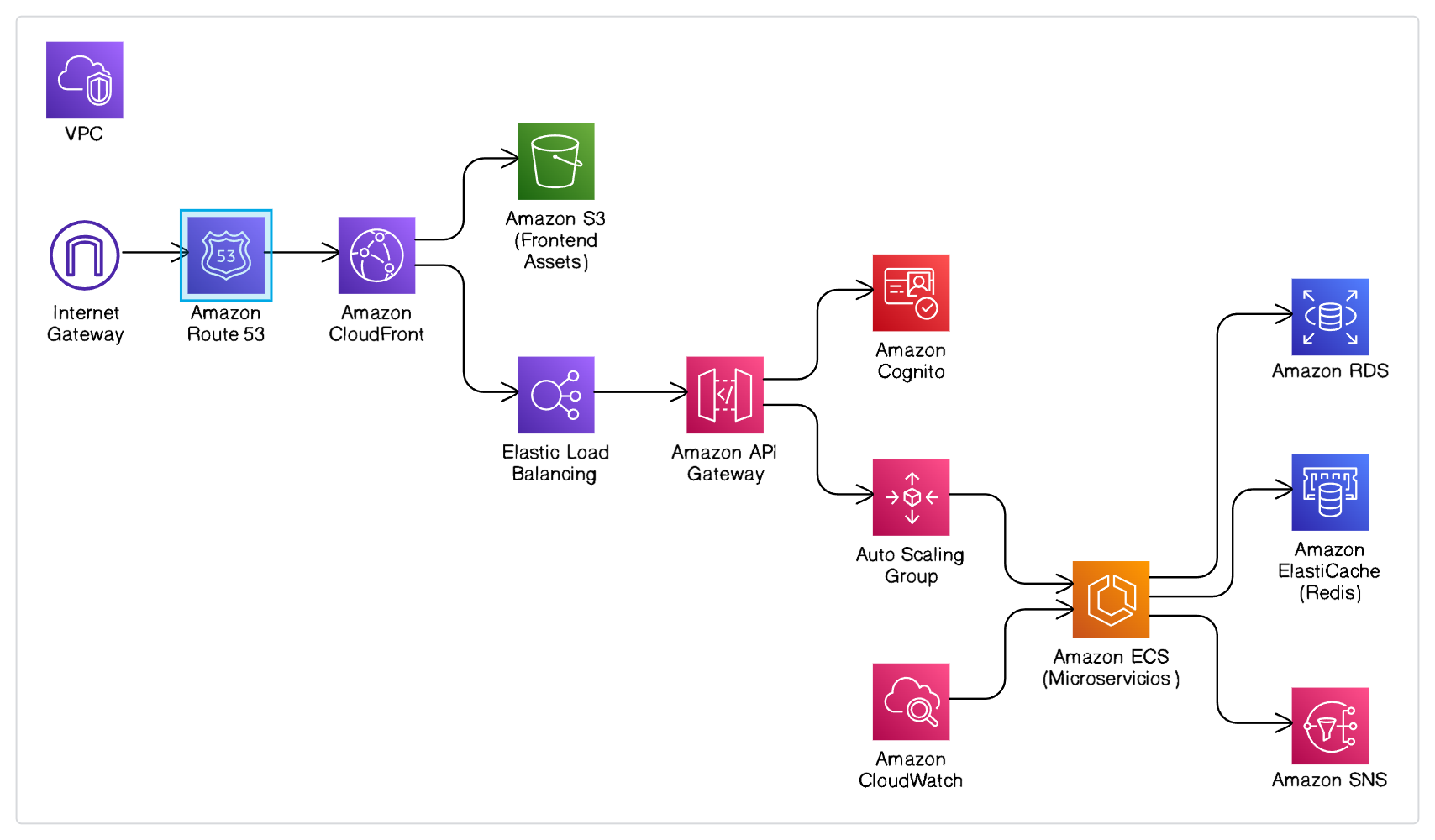
**Catálogo de Elementos y relaciones**

|  |  |  |
| --- | --- | --- |
| **Actor** | **Activaciones / Respuestas** | **Descripción** |
| **Huésped** | - Accede al sistema - Selecciona habitación - Confirma reserva | Usuario final que utiliza la app para consultar y reservar habitaciones. |
| **WebMóvil** | - Solicita disponibilidad - Solicita detalles - Envía solicitud de reserva - Muestra resultados | Aplicación cliente (web/móvil) para la interacción del huésped con el sistema. |
| **Recepción** | - Registra checkout | Personal administrativo que marca la salida del huésped y activa el proceso de limpieza. |
| **APIGateway** | - Gestiona solicitudes de disponibilidad, detalles y reservas - Interactúa con BD y servicios - Notifica | Punto de entrada centralizado para peticiones al sistema; enruta solicitudes a servicios específicos. |
| **ServicioReservas** | - Procesa reserva - Actualiza estado en BD | Servicio que gestiona y almacena las reservas de habitaciones. |
| **BD** (Base de Datos) | - Consulta disponibilidad - Devuelve detalles - Actualiza estados - Envía eventos | Almacena la información de habitaciones, reservas y su estado. |
| **SistemaNotificaciones** | - Recibe eventos - Notifica al servicio de limpieza - Actualiza estados | Sistema que intermedia notificaciones entre servicios y coordina cambios de estado. |
| **ServicioLimpieza** | - Recibe notificación - Prioriza limpieza - Asigna personal - Notifica limpieza iniciada/finalizada | Servicio operativo que organiza y ejecuta la limpieza de habitaciones. |
| **Opcional** | (Interno en Servicio Limpieza) | Opcionalmente prioriza tareas según reglas configuradas. |

## Vista Física

**Diagrama de AWS**

Este diagrama representa la vista física utilizando los servicios web de Amazon. Al principio del lado izquierdo se encuentra lo necesario para la conexión por internet al sistema alojado por AWS, pasando por balanceador de carga para mejor rendimiento, se hace uso de Amazon cognito para manejar las sesiones y un módulo de escalado para manejar el incremento de flujo de usuarios, por último, se tiene los módulos seleccionados para la construcción del sistema en sí. Todo se encuentra interconectado por el servicio de Amazon Virtual Private Cloud (VPC).



**Catálogo de Elementos y relaciones**

|  |  |  |
| --- | --- | --- |
| **Componente AWS** | **Cómo se conecta** | **Qué función cumple** |
| **Internet Gateway** | Entrada principal al sistema | Conecta la red privada (VPC) del sistema con Internet para permitir el acceso externo. |
| **Amazon Route 53** | Desde Internet Gateway | Se encarga de traducir el nombre del dominio del hotel a una dirección IP. También permite enrutamiento inteligente. |
| **Amazon CloudFront** | Desde Route 53 | Distribuye contenido del sitio (como imágenes y scripts) globalmente con baja latencia. Ayuda a mejorar el rendimiento y la seguridad. |
| **Amazon S3** | Conectado a CloudFront | Almacena los archivos del frontend del sitio, como imágenes, el SPA y otros recursos estáticos. |
| **Elastic Load Balancer** | Después de CloudFront | Reparte el tráfico de los usuarios entre múltiples servidores para garantizar disponibilidad y rendimiento. |
| **Amazon API Gateway** | A través del Load Balancer | Recibe y maneja todas las peticiones API del sistema. Se encarga de enrutar, validar y proteger las llamadas. |
| **Amazon Cognito** | Integrado con API Gateway | Maneja la autenticación de usuarios (huéspedes y empleados), manteniendo roles separados y seguros. |
| **Auto Scaling Group** | Desde API Gateway | Ajusta automáticamente el número de servidores que ejecutan los servicios según la demanda. |
| **Amazon ECS** | Ejecutado por el grupo de autoescalado | Plataforma que corre los microservicios del sistema, como reservas, limpieza y notificaciones. |
| **Amazon RDS** | Accedido desde ECS | Base de datos relacional donde se almacenan las reservas, habitaciones y usuarios. |
| **Amazon ElastiCache** | Conectado a ECS | Proporciona un sistema de caché en memoria (Redis) para agilizar las consultas frecuentes y datos en tiempo real. |
| **Amazon SNS** | Usado por ECS | Envía notificaciones automáticas, como alertas al personal o confirmaciones a los clientes. |
| **Amazon CloudWatch** | Monitorea a ECS y demás servicios | Recolecta métricas, registros y envía alertas si ocurre algún problema. |
| **VPC (Virtual Private Cloud)** | Contenedor general de todos los servicios | Crea una red segura y controlada donde viven todos los componentes del sistema. |