|  |
| --- |
| [**Hotel Pacific Reef**] |
| Especificación de requisitos de software |
| *Proyecto: [***Sistema de Gestión de Reserva Hotelera***]* |
|  |
| **Revisión*: [2.00]*** |
| **[Seleccionar 08/05/2025]** |

|  |
| --- |
| Especificación de Requisitos según estándar de IEEE 830. |

**Contenido**

Contenido

[Descripción de la actividad 8](#_Toc135731789)

[Instrucciones específicas 8](#_Toc135731790)

[Contenido 11](#_Toc135731791)

[Ficha del documento 12](#_Toc135731792)

[1. Introducción 13](#_Toc135731793)

[1.1. Propósito 13](#_Toc135731794)

[1.2. Ámbito del Sistema 13](#_Toc135731795)

[1.3. Definiciones, Acrónimos y Abreviaturas 13](#_Toc135731796)

[1.4. Referencias 13](#_Toc135731797)

[1.5. Visión General del Documento 14](#_Toc135731798)

[2. Descripción General 15](#_Toc135731799)

[2.1. Perspectiva del Producto 15](#_Toc135731800)

[2.2. Funciones del Producto 15](#_Toc135731801)

[2.3. Características de los Usuarios 15](#_Toc135731802)

[2.4. Restricciones 16](#_Toc135731803)

[2.5. Suposiciones y Dependencias 16](#_Toc135731804)

[2.6. Requisitos Futuros 17](#_Toc135731805)

[3. Requisitos Específicos 18](#_Toc135731806)

[3.1 Requisitos comunes de los interfaces 20](#_Toc135731807)

[3.1.1 Interfaces de usuario 20](#_Toc135731808)

[3.1.2 Interfaces de hardware 20](#_Toc135731809)

[3.1.3 Interfaces de software 20](#_Toc135731810)

[3.1.4 Interfaces de comunicación 20](#_Toc135731811)

[3.2 Requisitos funcionales 21](#_Toc135731812)

[3.3 Requisitos no funcionales 22](#_Toc135731813)

[3.3.1 Requisitos de rendimiento 22](#_Toc135731814)

[3.3.2 Seguridad 22](#_Toc135731815)

[3.3.3 Fiabilidad 22](#_Toc135731816)

[3.3.4 Disponibilidad 22](#_Toc135731817)

[3.3.5 Mantenibilidad 23](#_Toc135731818)

[3.3.6 Portabilidad 23](#_Toc135731819)

[3.4 Planilla de especificación de Requisitos 23](#_Toc135731820)

[**Dato** 24](#_Toc135731821)

[**Links de Interés** 24](#_Toc135731822)

# Ficha del documento

|  |  |  |  |
| --- | --- | --- | --- |
| **Fecha** | **Revisión** | **Autor** | **Modificación** |
|  |  |  |  |
|  |  |  |  |

Documento validado por las partes en fecha:

|  |  |  |
| --- | --- | --- |
| Por el cliente |  | Por la empresa suministradora |
| [Firma] |  | [Firma] |
| Sr./Sra. |  | Sr./Sra. |

# 1. Introducción

Este documento tiene como objetivo establecer los requisitos funcionales y no funcionales para el desarrollo del **Sistema de Gestión de Reserva Hotelera** para el **Hotel Pacific Reef**, basándose en las iteraciones previas de desarrollo (como el MVP, diagramas UML y las vistas front-end diseñadas).

## 1.1. Propósito

El propósito de este documento es proporcionar una descripción detallada de los requisitos para el sistema de **Gestión de Reserva Hotelera** del Hotel Pacific Reef. El documento está dirigido a todas las partes interesadas en el proyecto, incluyendo al **dueño del hotel**, **los administradores**, **los asesores tecnológicos**, y los **desarrolladores del sistema**.

## 1.2. Ámbito del Sistema

**Nombre del Sistema:**

* **Sistema de Gestión de Reserva Hotelera - Hotel Pacific Reef**

**Usuarios:**

* Consultar disponibilidad de habitaciones y reservar habitaciones disponibles
* Pagar con tarjeta Transbank (API Transbank)
* Editar su perfil (número telefónico, correo electrónico, etc.)
* Cancelar reservas
* Consultar su historial de reservas
* Restablecer contraseñas
* Ver notificaciones
* Enviar mensajes a administradores

**Administradores:**

* Ver historial de reservas de todos los clientes
* Gestionar reservas de clientes
* Ver mensajes de clientes
* Gestionar habitaciones (crear, modificar, eliminar)
* Gestionar usuarios (crear, modificar, eliminar)
* Enviar mensaje a clientes

**Lo que no hará el sistema:**

* No estará diseñado para gestionar otro tipo de servicios adicionales que no sean habitaciones (por ejemplo, servicios de restaurante, spa, etc.).

**Beneficios, objetivos y metas:**

* **Beneficios**: El sistema mejorará la eficiencia en la gestión de reservas, reducirá errores humanos y permitirá a los clientes gestionar sus reservas de manera autónoma.
* **Objetivos**:
  + Automatizar el proceso de reservas y pagos.
  + Proporcionar una plataforma visual y fácil de usar para los clientes.
  + Mejorar la experiencia del cliente con la disponibilidad de información en tiempo real.
* **Metas**:
  + A futuro se debe desarrollar un backend que guarde los datos en una nube, y no de manera local.

## 1.3. Definiciones, Acrónimos y Abreviaturas

* **Reserva**: Proceso de selección y pago parcial de una habitación en el hotel para una estancia futura.
* **Administrador de Reservas**: Persona encargada de gestionar las reservas y los usuarios en el sistema.
* **Código QR**: Un tipo de código de barras bidimensional que será utilizado para verificar la reserva del cliente durante el check-in.
* **Catálogo de Habitaciones**: Una lista visual interactiva de las habitaciones disponibles, con información detallada sobre cada una.
* **Sistema de Gestión de Reserva Hotelera**: El sistema informático que permitirá la consulta, gestión y pago de reservas para el **Hotel Pacific Reef**.
* **Stakeholder**: Personas o entidades interesadas en el proyecto, como el dueño del hotel, el asesor tecnológico y los usuarios finales.
* **Front-end**: La parte del sistema con la que interactúa directamente el usuario, es decir, la interfaz visual. Incluye los elementos gráficos, botones, menús y todo lo que el usuario ve en su dispositivo (móvil o de escritorio).
* **Back-end**: La parte del sistema que no es visible para el usuario, pero que gestiona la lógica interna, el procesamiento de datos, la base de datos y la comunicación con el front-end. Es donde se manejan los datos de las reservas, la autenticación de usuarios, etc.
* **API (Interfaz de Programación de Aplicaciones)**: Un conjunto de reglas que permite que diferentes sistemas o aplicaciones se comuniquen entre sí. En este contexto, las APIs podrían ser utilizadas para integrar el sistema con servicios de pago, bases de datos externas o plataformas de alojamiento en la nube.
* **Nube (Cloud)**: Infraestructura de servidores remotos a los cuales se accede a través de internet para almacenar y procesar datos. El sistema de reservas se alojará en la nube para facilitar su acceso desde diferentes dispositivos y garantizar la disponibilidad y seguridad de los datos.
* **Base de Datos**: Un sistema organizado para almacenar, gestionar y recuperar datos. En este caso, almacenará información de las reservas, los usuarios, las habitaciones y los pagos realizados.
* **Full-stack**: Un término utilizado para describir a un desarrollador que trabaja tanto en el front-end como en el back-end del sistema.
* **Diseño responsivo**: Un enfoque de diseño de interfaces que asegura que el sistema se vea y funcione correctamente en diferentes dispositivos y tamaños de pantalla, como móviles, tabletas y computadoras de escritorio.
* **UI (Interfaz de Usuario)**: Conjunto de elementos gráficos y visuales con los que interactúa el usuario. Esto incluye botones, menús, formularios y otros componentes visibles de la aplicación.
* **UX (Experiencia de Usuario)**: El conjunto de sensaciones, emociones y percepciones que experimenta un usuario al interactuar con el sistema. Un buen diseño de UX asegura que la aplicación sea fácil de usar y agradable para los usuarios.

## 1.4. Referencias

* **Brief de Proyecto "Gestión de Reserva Hotelera"** – Documento base para la definición del sistema y sus funcionalidades.
* **Normas ISO/IEC 9126** – Referencias para los estándares de calidad del software

## 1.5. Visión General del Documento

Este documento presenta de forma estructurada los requisitos necesarios para el desarrollo del sistema de reservas hoteleras. En él se incluye información introductoria, una descripción general del sistema desde distintas perspectivas, y el detalle completo de los requisitos funcionales y no funcionales. Además, se incorporan especificaciones sobre interfaces, características esperadas del producto y condiciones bajo las cuales debe operar. Su organización facilita la comprensión progresiva del sistema, desde un nivel general hasta el detalle técnico requerido para su implementación.

# 2. Descripción General

En esta sección se describe el contexto en el que se desarrollará el sistema de **Gestión de Reserva Hotelera** para el **Hotel Pacific Reef**, detallando cómo interactuará con otros sistemas o productos y cuáles son los factores relevantes que influirán en el diseño y desarrollo del producto.

## 2.1. Perspectiva del Producto

El sistema de **Gestión de Reserva Hotelera** es una aplicación independiente que servirá como herramienta para los usuarios (turistas) que deseen hacer reservas en línea en el **Hotel Pacific Reef**, y para los administradores y empleados del hotel que gestionarán el proceso de reservas y usuarios. El sistema será accesible tanto en dispositivos móviles como de escritorio, proporcionando una experiencia de usuario fluida en ambas plataformas.

**Relación con otros productos:**

* **Plataforma de pago**: El sistema se integrará con una API de pagos Transbank y notificará si hubo un correcto pago o no de la reserva.
* **Base de datos local**: De momento el MVP solo considera datos locales, sin embargo para la próxima actualización se espera que la base de datos sea guardada en la nube (Cloud)
* **Sistema de notificaciones por correo electrónico**: Si bien el sistema actualmente envia mensajes de confirmación a los usuarios a través de las notificaciones, a futuro se espera que el sistema enviará notificaciones automáticas al correo electrónico de cada usuario.

**Pantallas clave y funcionalidad:**

1. **Pantalla de inicio de sesión**:
   * Los usuarios deberán iniciar sesión para interactuar con el sistema. Los turistas deberán ingresar su correo electrónico y contraseña (si ya están registrados), o registrarse como nuevos usuarios. Los administradores y empleados también ingresarán con sus credenciales específicas para acceder a funciones adicionales.
2. **Pantalla de perfil del usuario**:
   * Una vez que el usuario inicie sesión, accederá a su perfil donde podrá ver su información personal (nombre, correo electrónico, historial de reservas, etc.), editar sus datos, y gestionar sus preferencias de idioma (español o inglés).
3. **Pantalla de consulta de disponibilidad de habitaciones**:
   * El sistema mostrará un calendario interactivo donde el usuario podrá seleccionar las fechas de su estadía. Al seleccionar las fechas, el sistema mostrará un listado de las habitaciones disponibles para ese período específico.
4. **Pantalla de detalles de la habitación**:
   * Cuando el usuario haga clic en una habitación de la lista de disponibilidad, verá una página con información detallada sobre la habitación: fotos (al menos tres imágenes), características, equipamiento, número de la habitación, precio por noche, y una opción para seleccionar la cantidad de noches que desea reservar.
5. **Pantalla de reserva**:
   * Después de seleccionar una habitación y las fechas, el sistema calculará el costo total de la estadía y mostrará el precio. El usuario podrá confirmar la reserva, y el sistema generará un ticket de reserva con un código QR. El ticket será enviado por correo electrónico al cliente.
6. **Pantalla de pago**:
   * El sistema permitirá al usuario realizar el pago de la reserva (30% del total) a través de una plataforma de pago externa (API de pago). El pago será procesado y se generará una confirmación.
7. **Pantalla de administración para los administradores**:
   * Los administradores tendrán acceso a un panel de administración donde podrán ver todas las reservas, gestionar cuentas de usuarios (clientes y empleados del hotel), y actualizar el catálogo de habitaciones con nuevos precios, características y disponibilidad.
8. **Pantalla de reportes para administradores**:
   * Los administradores podrán generar reportes de las reservas realizadas, consultar estadísticas de ocupación por fechas, y realizar otras tareas administrativas necesarias para el hotel.

**Diagramas de bloques (pendiente por hacer):**

1. **Frontend (Interfaz de Usuario)**:
   * **Pantalla de inicio de sesión**
   * **Pantalla de consulta de disponibilidad**
   * **Pantalla de detalles de habitación**
   * **Pantalla de reserva y pago**
2. **Backend (Lógica del sistema)**:
   * **Procesamiento de reserva**
   * **Generación de tickets**
   * **Gestión de usuarios (clientes y administradores)**
3. **Base de datos**:
   * **Datos de reservas, clientes y habitaciones**
   * **Historial de pagos y transacciones**

## 2.2. Funciones del Producto

**Funciones principales (requerimientos funcionales):**

1. **Consulta de disponibilidad de habitaciones**:
   * El sistema permite a los usuarios consultar la disponibilidad de habitaciones seleccionando una fecha en el calendario. El sistema muestra las habitaciones disponibles para esas fechas.
2. **Realización de reservas**:
   * Los usuarios pueden seleccionar la habitación deseada y realizar una reserva indicando la cantidad de días para la estancia.
3. **Generación de ticket de reserva**:
   * Una vez realizada la reserva, el sistema genera un ticket que es enviado al cliente.
4. **Gestión de pagos**:
   * El sistema calculará el valor total de la reserva y a través de una API Transbank solicita los datos de tarjeta para el pago.
5. **Administración de cuentas de usuarios**:
   * El administrador del hotel puede crear y gestionar cuentas de clientes y empleados del hotel. Además, puede modificar o cancelar reservas existentes.
6. **Administración del catálogo de habitaciones**:
   * El sistema permite al administrador agregar, editar o eliminar habitaciones del catálogo, actualizar precios y características (ubicación, equipamiento, etc.).

**Funciones no funcionales (requerimientos no funcionales):**

1. **Accesibilidad en múltiples dispositivos**:
   * La aplicación debe ser accesible tanto en dispositivos móviles como en computadoras de escritorio.
2. **Seguridad**:
   * El sistema debe garantizar que todos los datos personales y de pago estén protegidos mediante métodos de encriptación adecuados.
3. **Rendimiento**:
   * El sistema debe ser capaz de manejar múltiples usuarios realizando reservas simultáneamente sin afectar su rendimiento.
4. **Usabilidad**:
   * La interfaz debe ser clara, fácil de usar y accesible, incluso para usuarios con conocimientos técnicos limitados.

## 2.3. Características de los Usuarios

El sistema esta diseñado para ser utilizado por diferentes tipos de usuarios, cada uno con diferentes características y necesidades. A continuación se describen las características generales de estos usuarios:

1. **Usuarios finales (turistas y clientes potenciales)**:
   * **Nivel educativo**: Los usuarios pueden tener distintos niveles educativos, pero no se requiere un conocimiento técnico específico para realizar reservas.
   * **Experiencia técnica**: Los turistas pueden ser usuarios con poca experiencia técnica, por lo que el sistema debe ser intuitivo y fácil de usar.
   * **Acceso al sistema**: Los usuarios interactuarán principalmente con la interfaz de usuario para realizar reservas, ver disponibilidad y hacer pagos.
2. **Administradores del hotel**:
   * **Nivel educativo**: Los administradores pueden tener formación técnica, pero no es estrictamente necesario un título especializado en informática.
   * **Experiencia técnica**: Se espera que los administradores tengan un conocimiento básico en el manejo de sistemas informáticos y una comprensión general de la gestión hotelera.
   * **Acceso al sistema**: Los administradores tendrán acceso a las funciones de gestión de reservas, cuentas de usuarios, precios y reportes.
3. **Empleados del hotel (por ejemplo, personal de recepción)**:
   * **Nivel educativo**: Los empleados pueden tener formación en áreas como administración hotelera o servicios turísticos, pero no es necesario que tengan una formación técnica avanzada.
   * **Experiencia técnica**: No se requiere experiencia avanzada en tecnología, aunque deberán ser capaces de operar el sistema para verificar reservas y consultar la disponibilidad.
   * **Acceso al sistema**: Los empleados tendrán un acceso limitado a ciertas funciones, principalmente para consultar reservas y gestionar el check-in

## 2.4. Restricciones

Esta sección describe las limitaciones y restricciones que deberán tenerse en cuenta durante el desarrollo del sistema de **Gestión de Reserva Hotelera**. Estas restricciones afectan tanto al diseño como a la implementación del producto.

1. **Políticas de la empresa**:
   * El desarrollo debe ajustarse a las políticas de seguridad de la empresa, especialmente en lo relacionado con la gestión de datos sensibles de los clientes (información personal y de pago). Además, se debe seguir la normativa interna para la creación y administración de cuentas de usuarios.
2. **Limitaciones del hardware**:
   * El sistema debe ser accesible desde dispositivos móviles (iOS, Android) y computadoras de escritorio (Windows). Si bien el sistema debe adaptarse a las distintas resoluciones de pantalla, debe ser ligero y eficiente para dispositivos con características más modestas, ya que algunos usuarios pueden tener dispositivos de gama media o baja.
3. **Interfaces con otras aplicaciones**:
   * El sistema debe integrarse con una **API de pago externa** para procesar las transacciones de reserva.
   * Debe comunicarse con la **base de datos** para almacenar y recuperar los datos del sistema (reservas, clientes, habitaciones).
4. **Operaciones paralelas**:
   * El sistema debe permitir la realización de múltiples reservas simultáneas sin afectar su rendimiento. Los usuarios deben poder realizar reservas de manera independiente, y los administradores deben poder gestionar las reservas y actualizar el catálogo de habitaciones de manera eficiente.
5. **Funciones de auditoría**:
   * El sistema debe mantener un registro detallado de las acciones realizadas por los usuarios y administradores (por ejemplo, reservas realizadas, pagos efectuados, cambios en el catálogo de habitaciones), para garantizar la trazabilidad de las operaciones y cumplir con las políticas de auditoría.
6. **Funciones de control**:
   * El sistema debe permitir el control de acceso a funciones específicas según el rol del usuario. Por ejemplo, solo los administradores podrán modificar el catálogo de habitaciones o generar reportes.
7. **Lenguaje(s) de programación**:
   * Se utilizarán lenguajes de programación como **CSS** para el desarrollo del front-end, y **Python (Django)** para el back-end. El uso de estos lenguajes se debe ajustar a las mejores prácticas de desarrollo ágil y de código limpio.
8. **Protocolos de comunicación**:
   * El sistema se comunicará utilizando el protocolo **HTTPS** para garantizar la seguridad de la transmisión de datos. Las interacciones con la API de pagos y con la base de datos en la nube también deben garantizar una comunicación segura.
9. **Requisitos de habilidad**:
   * El equipo de desarrollo debe tener conocimientos en desarrollo web (front-end y back-end), integración de APIs, bases de datos, y en protocolos de seguridad para manejar pagos.
10. **Criticalidad de la aplicación**:
    * La aplicación será de **alta criticidad**, ya que el sistema manejará datos sensibles de los usuarios y procesará pagos. El tiempo de inactividad o errores en el sistema pueden afectar directamente la experiencia del cliente y la reputación del hotel.
11. **Consideraciones acerca de la seguridad**:
    * El sistema debe seguir las mejores prácticas en términos de seguridad de la información. Los datos personales y de pago de los clientes deben ser cifrados en tránsito y en reposo. Además, el sistema debe implementar autenticación robusta (contraseñas fuertes, autenticación de dos factores si es posible) y protección contra ataques comunes como inyecciones SQL, XSS, CSRF, entre otros.

## 2.5. Suposiciones y Dependencias

En esta sección se describen los factores clave que se han asumido durante la creación de los requisitos y que, si cambian, pueden afectar la implementación del sistema.

1. **Suposiciones:**
   * Se asume que el **Hotel Pacific Reef** tendrá acceso constante a una conexión a internet confiable, ya que el sistema depende de la conectividad para gestionar reservas y pagos en línea.
   * Se asume que los **clientes del hotel** tendrán acceso a dispositivos conectados a internet, ya sea mediante computadoras de escritorio, portátiles o teléfonos móviles, para poder realizar sus reservas de forma remota.
   * Se asume que los **empleados del hotel** estarán capacitados para manejar el sistema, ya que se les proporcionará formación sobre cómo utilizar las funcionalidades de administración y gestión de reservas.
2. **Dependencias:**
   * El funcionamiento del sistema depende de la integración exitosa con la **API de pago externa**. Si esta plataforma de pago no está disponible o presenta problemas, el proceso de pago de reservas no podrá completarse correctamente.
   * El sistema depende de la **base de datos** para almacenar los datos de los usuarios, reservas y habitaciones. Una interrupción en los servicios de la base de datos podría afectar el funcionamiento del sistema.
   * La correcta configuración y mantenimiento de las **interfaces de red** es crucial para garantizar la disponibilidad y accesibilidad del sistema a través de internet.

## 2.6. Requisitos Futuros

A continuación se esbozan algunas posibles mejoras al sistema que podrían implementarse en futuras versiones o fases del proyecto:

1. **Expansión de las funciones de pago**:
   * Integración con más métodos de pago, como **PayPal**, y **criptomonedas**, para ofrecer una mayor variedad de opciones a los usuarios.
2. **Aplicación móvil nativa**:
   * Desarrollar una versión de la aplicación como **app nativa** para dispositivos iOS y Android, que podría ofrecer características adicionales, como notificaciones push, y permitir una experiencia más fluida para los usuarios móviles.
3. **Sistema de recomendaciones**:
   * Implementar un sistema de recomendaciones basado en **machine learning** para sugerir habitaciones o paquetes especiales a los clientes, basado en su historial de reservas o preferencias.
4. **Gestión avanzada de clientes frecuentes**:
   * Introducción de un sistema de fidelización donde los usuarios frecuentes puedan acumular puntos o beneficios, mejorando la experiencia del cliente y alentando las reservas repetidas.
5. **Integración con sistemas de gestión de inventario**:
   * El sistema podría integrarse con un sistema de gestión de inventarios del hotel para coordinar las reservas con la disponibilidad de otros recursos como servicios adicionales (transporte, tours, etc.).
6. **Automatización del check-in/check-out**:
   * Incorporar funcionalidades para que los clientes puedan realizar un **check-in/check-out completamente automatizado**, utilizando su código QR recibido en el ticket de reserva.

# 3. Requisitos Específicos

Esta sección contiene los requisitos a un nivel de detalle suficiente como para permitir a los diseñadores diseñar un sistema que satisfaga estos requisitos, y que permita al equipo de pruebas planificar y realizar las pruebas que demuestren si el sistema satisface, o no, los requisitos. Todo requisito aquí especificado describirá comportamientos externos del sistema, perceptibles por parte de los usuarios, operadores y otros sistemas. Esta es la sección más larga e importante de la ERS. Deberán aplicarse los siguientes principios:

* El documento debería ser perfectamente legible por personas de muy distintas formaciones e intereses.
* Deberán referenciarse aquellos documentos relevantes que poseen alguna influencia sobre los requisitos.
* Todo requisito deberá ser unívocamente identificable mediante algún código o sistema de numeración adecuado.
* Lo ideal, aunque en la práctica no siempre realizable, es que los requisitos posean las siguientes características:
* **Corrección**: La ERS es correcta si y sólo si todo requisito que figura aquí (y que será implementado en el sistema) refleja alguna necesidad real. La corrección de la ERS implica que el sistema implementado será el sistema deseado.
* **No ambiguos**: Cada requisito tiene una sola interpretación. Para eliminar la ambigüedad inherente a los requisitos expresados en lenguaje natural, se deberán utilizar gráficos o notaciones formales. En el caso de utilizar términos que, habitualmente, poseen más de una interpretación, se definirán con precisión en el glosario.
* **Completos**: Todos los requisitos relevantes han sido incluidos en la ERS. Conviene incluir todas las posibles respuestas del sistema a los datos de entrada, tanto válidos como no válidos.
* **Consistentes**: Los requisitos no pueden ser contradictorios. Un conjunto de requisitos contradictorio no es implementable.
* **Clasificados**: Normalmente, no todos los requisitos son igual de importantes. Los requisitos pueden clasificarse por importancia (esenciales, condicionales u opcionales) o por estabilidad (cambios que se espera que afecten al requisito). Esto sirve, ante todo, para no emplear excesivos recursos en implementar requisitos no esenciales.
* **Verificables**: La ERS es verificable si y sólo si todos sus requisitos son verificables. Un requisito es verificable (testeable) si existe un proceso finito y no costoso para demostrar que el sistema cumple con el requisito. Un requisito ambiguo no es, en general, verificable. Expresiones como a veces, bien, adecuado, etc. Introducen ambigüedad en los requisitos. Requisitos como “en caso de accidente la nube tóxica no se extenderá más allá de 25Km" no es verificable por el alto costo que conlleva.
* **Modificables**: La ERS es modificable si y sólo si se encuentra estructurada de forma que los cambios a los requisitos pueden realizarse de forma fácil, completa y consistente.
  + La utilización de herramientas automáticas de gestión de requisitos facilita enormemente esta tarea.
* **Trazables**: La ERS es trazable si se conoce el origen de cada requisito y se facilita la referencia de cada requisito a los componentes del diseño y de la implementación. La trazabilidad hacia atrás indica el origen (documento, persona, etc.) de cada requisito. La trazabilidad hacia delante de un requisito R indica que componentes del sistema son los que realizan el requisito R.

## 3.1 Requisitos comunes de los interfaces

Descripción detallada de todas las entradas y salidas del sistema de software.

### 3.1.1 Interfaces de usuario

Los interfaces de usuario son fundamentales para garantizar una experiencia de usuario óptima. A continuación se describen los requisitos del interfaz de usuario para el sistema **Gestión de Reserva Hotelera**.

1. **Pantalla de Inicio de Sesión**:
   * **Descripción**: La pantalla de inicio de sesión permitirá a los usuarios autenticarse en el sistema utilizando su nombre de usuario y contraseña.
   * **Requisitos**:
     + El sistema debe permitir que el usuario seleccione entre dos tipos de acceso: **Cliente** o **Administrador**.
     + La interfaz debe ser limpia, con un diseño intuitivo, usando los colores institucionales del hotel (Colores recomendados: Azul y blanco).
     + El sistema debe permitir la opción de cambiar el idioma entre **Español** e **Inglés**.
     + Los usuarios deben tener la opción de **recuperar la contraseña** mediante correo electrónico en caso de olvido.
2. **Pantalla de Reserva**:
   * **Descripción**: El usuario podrá consultar la disponibilidad de habitaciones y realizar una reserva.
   * **Requisitos**:
     + El sistema debe mostrar un calendario visual donde el usuario podrá seleccionar las fechas de su estadía.
     + Al seleccionar la fecha de entrada y salida, el sistema debe mostrar las habitaciones disponibles con sus respectivas fotos, precios y detalles (equipamiento, características, etc.).
     + El sistema debe permitir seleccionar el número de habitaciones y los días de estadía.
     + Debe haber un botón de **"Confirmar Reserva"** para continuar al proceso de pago.
3. **Pantalla de Pago**:
   * **Descripción**: El usuario realizará el pago de la reserva y obtendrá un ticket de confirmación.
   * **Requisitos**:
     + El sistema debe solicitar al usuario que ingrese los datos de pago (por ejemplo, tarjeta de crédito).
     + El pago debe ser procesado a través de una API de pagos externa.
     + Una vez realizado el pago, el sistema debe generar un ticket de reserva.
4. **Pantalla de Administración de Reservas**:
   * **Descripción**: Los administradores del hotel podrán ver, modificar o cancelar reservas.
   * **Requisitos**:
     + Los administradores deben poder ver una lista de todas las reservas por fecha y habitación.
     + Los administradores deben poder cancelar o modificar la información de las reservas, como fechas, habitaciones, o datos del cliente.
5. **Pantalla de Catálogo de Habitaciones**:
   * **Descripción**: Los administradores podrán actualizar la información de las habitaciones.
   * **Requisitos**:
     + El administrador debe poder añadir nuevas habitaciones o actualizar las existentes.
     + Cada habitación debe contar con una descripción de sus características y equipamiento.

### 3.1.2 Interfaces de hardware

El sistema **Gestión de Reserva Hotelera** interactuará con diferentes dispositivos hardware, tanto para los usuarios finales como para el personal administrativo del hotel.

1. **Dispositivos de usuario final**:
   * **Descripción**: Los usuarios podrán acceder al sistema desde dispositivos móviles y computadoras de escritorio.
   * **Requisitos**:
     + **Móviles**: El sistema debe ser completamente funcional en smartphones y tabletas (iOS y Android).
     + **Escritorios**: El sistema debe ser accesible en dispositivos de escritorio con los sistemas operativos más comunes (Windows, MacOS).
2. **Dispositivos de administración**:
   * **Descripción**: Los administradores del hotel utilizarán dispositivos para gestionar las reservas y las habitaciones.
   * **Requisitos**:
     + El sistema debe ser accesible desde computadoras de escritorio o portátiles, con una interfaz optimizada para estos dispositivos.
     + **Impresora de tickets**: El sistema debe ser capaz de generar un ticket de reserva.

### 3.1.3 Interfaces de software

El sistema deberá integrar varias aplicaciones de software para cumplir con sus funcionalidades.

1. **API de pago**:
   * **Descripción**: Se deberá integrar una API de pago para procesar las transacciones de los usuarios.
   * **Propósito**: Facilitar el proceso de pago de las reservas.
   * **Definición del interfaz**:
     + El sistema interactuará con la API mediante solicitudes HTTPS seguras.
     + El sistema deberá manejar errores de transacción y mostrar un mensaje adecuado en caso de fallo.
2. **Base de Datos**:
   * **Descripción**: El sistema debe integrarse con una base de datos para almacenar la información de reservas, usuarios, habitaciones y pagos.
   * **Propósito**: Almacenar y recuperar la información necesaria para el funcionamiento del sistema.
   * **Definición del interfaz**:
     + La base de datos será de tipo relacional y estará alojada en el entorno de desarrollo correspondiente (SQLite).
     + El sistema interactuará con la base de datos a través de consultas SQL estándar.

### 3.1.4 Interfaces de comunicación

El sistema debe ser capaz de comunicarse con otros sistemas y servicios mediante protocolos específicos.

1. **Protocolos de comunicación**:
   * **Descripción**: El sistema debe utilizar protocolos estándar para la comunicación de datos.
   * **Requisitos**:
     + **HTTPS** para la transmisión segura de datos entre el cliente y el servidor.
     + **API RESTful** para la comunicación con la API de pagos y otros servicios externos.

## 3.2 Requisitos funcionales

**1. Consulta de Disponibilidad de Habitaciones**

* **Descripción**: El sistema debe permitir a los usuarios consultar la disponibilidad de habitaciones para un rango de fechas específico.
* **Entrada**: Fechas seleccionadas por el usuario.
* **Proceso**: El sistema consultará la base de datos para verificar la disponibilidad de las habitaciones para las fechas solicitadas.
* **Salida**: Lista de habitaciones disponibles, con detalles como el tipo de habitación, el precio y las características.

**2. Registro de Reserva**

* **Descripción**: El sistema debe permitir a los usuarios registrar una reserva para las habitaciones disponibles.
* **Entrada**: Datos de reserva (fecha de entrada, fecha de salida, tipo de habitación, cantidad de personas).
* **Proceso**: El sistema verificará la disponibilidad, calculará el precio total de la reserva y generará un ticket de reserva.
* **Salida**: Ticket de reserva con código QR, enviado al correo electrónico del cliente.

**3. Administración de Reservas (por Administradores)**

* **Descripción**: Los administradores deben poder visualizar, modificar o cancelar las reservas realizadas por los usuarios.
* **Entrada**: Solicitudes de modificación o cancelación de reservas.
* **Proceso**: El sistema actualizará o eliminará la reserva de acuerdo con las indicaciones del administrador.
* **Salida**: Confirmación de la acción realizada (modificación o cancelación de la reserva).

**4. Gestión del Catálogo de Habitaciones**

* **Descripción**: Los administradores deben poder añadir, modificar o eliminar habitaciones disponibles para la reserva.
* **Entrada**: Detalles de la habitación (precio, descripción, características, imágenes, capacidad).
* **Proceso**: El sistema actualizará el catálogo de habitaciones en la base de datos.
* **Salida**: Actualización de las habitaciones en el catálogo visible en el sistema.

**5. Cálculo del Pago de la Reserva**

* **Descripción**: El sistema debe calcular automáticamente el monto a pagar por el usuario en función de la duración de su estancia y el tipo de habitación seleccionada.
* **Entrada**: Fechas de la reserva y tipo de habitación.
* **Proceso**: El sistema calculará el precio total, aplicando el pago inicial (por ejemplo, el 30% de la tarifa total).
* **Salida**: Monto a pagar y detalles de la reserva.

**6. Procesamiento de Pagos**

* **Descripción**: El sistema debe integrar un sistema de pagos en línea para procesar pagos de reservas realizadas.
* **Entrada**: Datos de pago (tarjeta de crédito o débito transbank)
* **Proceso**: El sistema enviará la información de pago a través de una API de pago y procesará la transacción.
* **Salida**: Confirmación de pago, actualizada en la base de datos y en el estado de la reserva.

**7. Generación de Confirmaciones y Tickets**

* **Descripción**: El sistema debe generar una confirmación de reserva y enviarla por correo electrónico al cliente.
* **Entrada**: Confirmación de reserva generada.
* **Proceso**: El sistema generará un ticket, que incluirá detalles de la reserva.
* **Salida**: Mensaje notificando la confirmación de la reserva.

**8. Gestión de Perfiles de Usuario**

* **Descripción**: El sistema debe permitir a los usuarios crear y gestionar sus perfiles personales.
* **Entrada**: Datos del perfil (nombre, correo electrónico, número de teléfono, preferencias).
* **Proceso**: El sistema debe permitir la creación, edición y actualización del perfil de usuario.
* **Salida**: Perfil de usuario actualizado.

**9. Registro de Clientes Nuevos**

* **Descripción**: El sistema debe permitir que nuevos usuarios se registren en el sistema para realizar una reserva.
* **Entrada**: Datos de registro (nombre, correo electrónico, número de teléfono, contraseña).
* **Proceso**: El sistema validará los datos de registro y creará una cuenta de usuario en el sistema.
* **Salida**: Confirmación de registro, acceso a la cuenta de usuario.

**10. Recuperación de Contraseña**

* **Descripción**: El sistema debe permitir a los usuarios recuperar su contraseña en caso de olvido.
* **Entrada**: Correo electrónico del usuario.
* **Proceso**: El sistema enviará un enlace de recuperación al correo electrónico del usuario.
* **Salida**: Enlace para recuperar la contraseña.

**11. Gestión de Estado de Reservas (Confirmadas, Pendientes, Canceladas)**

* **Descripción**: Los administradores deben poder cambiar el estado de las reservas (pendiente, confirmada, cancelada).
* **Entrada**: Cambio de estado solicitado por el administrador.
* **Proceso**: El sistema actualizará el estado de la reserva en la base de datos.
* **Salida**: Confirmación del cambio de estado de la reserva.

**12. Notificación de Confirmación de Reserva**

* **Descripción**: El sistema debe enviar notificaciones automáticas por correo electrónico para confirmar las reservas.
* **Entrada**: Confirmación de la reserva.
* **Proceso**: El sistema enviará un mensaje con los detalles de la reserva.
* **Salida**: Mensaje de confirmación enviado al cliente.

**13. Control de Disponibilidad de Habitaciones**

* **Descripción**: El sistema debe asegurarse de que las habitaciones no se reserven múltiples veces para las mismas fechas.
* **Entrada**: Solicitudes de reserva de habitaciones.
* **Proceso**: El sistema verificará la disponibilidad de la habitación antes de permitir la reserva.
* **Salida**: Estado actualizado de las habitaciones (disponible/no disponible)

|  |  |
| --- | --- |
| **Actor** | **Requisitos Funcionales Asociados** |
| **Usuario (Cliente)** | RF 1 - Consulta de Disponibilidad de Habitaciones |
| RF 2 - Registro de Reserva |
| RF 5 - Cálculo del Pago de la Reserva |
| RF 6 - Procesamiento de Pagos |
| RF 7 - Generación de Confirmaciones y Tickets |
| RF 8 - Gestión de Perfiles de Usuario |
| RF 9 - Registro de Clientes Nuevos |
| RF 10 - Recuperación de Contraseña |
| RF 13 - Control de Disponibilidad de Habitaciones |
| **Administrador** | RF 3 - Administración de Reservas (por Administradores) |
| RF 4 - Gestión del Catálogo de Habitaciones |
| RF 11 - Gestión de Estado de Reservas (Confirmadas, Pendientes, Canceladas) |
| RF 12 - Notificación de Confirmación de Reserva |
| **Sistema de Pagos (API)** | RF 6 - Procesamiento de Pagos |
| **Base de Datos** | RF 1 - Consulta de Disponibilidad de Habitaciones |
| RF 2 - Registro de Reserva |
| RF 3 - Administración de Reservas |
| RF 4 - Gestión del Catálogo de Habitaciones |
| RF 5 - Cálculo del Pago de la Reserva |
| RF 6 - Procesamiento de Pagos |
| RF 7 - Generación de Confirmaciones y Tickets |
| RF 8 - Gestión de Perfiles de Usuario |
| RF 11 - Gestión de Estado de Reservas (Confirmadas, Pendientes, Canceladas) |

## 3.3 Requisitos no funcionales

## **3.3.1 Requisitos de Rendimiento**

## El sistema debe cumplir con los siguientes requisitos de rendimiento:

## El sistema debe ser capaz de manejar hasta 100 usuarios simultáneos sin experimentar caídas significativas en el rendimiento.

## El tiempo de respuesta para la consulta de disponibilidad de habitaciones debe ser inferior a 3 segundos.

## El sistema debe soportar un mínimo de 10 transacciones de reserva por segundo sin que el rendimiento se vea afectado.

## Se utilizarán bases de datos optimizadas como MySQL o SQLite para asegurar la rápida consulta y actualización de los datos.

## **3.3.2 Seguridad**

## La seguridad del sistema es un aspecto crucial para garantizar la protección de los datos y las transacciones. Para ello, se implementarán los siguientes mecanismos de seguridad:

## Autenticación y autorización: Se utilizarán métodos de autenticación seguros, como el uso de JSON Web Tokens (JWT), para garantizar que solo los usuarios autorizados puedan acceder a los recursos del sistema.

## Almacenamiento seguro de contraseñas: Las contraseñas de los usuarios serán almacenadas de forma segura utilizando técnicas de hashing con algoritmos como bcrypt en el backend.

## Protección contra vulnerabilidades: El sistema se protegerá contra ataques comunes como inyecciones SQL o Cross-Site Scripting (XSS) mediante el uso de buenas prácticas de codificación y herramientas de validación adecuadas.

## **3.3.3 Fiabilidad**

## El sistema debe ser fiable y capaz de operar sin interrupciones durante largos periodos. Para ello, se implementarán las siguientes medidas:

## Tiempo entre fallos: El sistema deberá tener un tiempo promedio entre fallos (MTBF) superior a 500 horas de uso continuo, lo que garantiza una operación estable y sin interrupciones graves.

## Recuperación ante fallos: En caso de que ocurra un fallo, el sistema debe ser capaz de recuperarse automáticamente en menos de 5 minutos, minimizando el impacto para los usuarios.

## **3.3.4 Disponibilidad**

## El sistema debe estar disponible para los usuarios en casi todo momento, lo que se garantiza con los siguientes requisitos de disponibilidad:

## Disponibilidad: El sistema debe estar disponible al menos el 99% del tiempo, lo que significa que la cantidad máxima de tiempo de inactividad será de aproximadamente 4 horas al mes.

## Mantenimiento sin interrumpir el servicio: Las tareas de mantenimiento del sistema, como actualizaciones de software o cambios en la base de datos, se deben realizar sin interrumpir el servicio de los usuarios.

## **3.3.5 Mantenibilidad**

## El sistema debe ser fácil de mantener y actualizar. Para lograrlo, se seguirán las siguientes pautas:

## Comentarios y documentación del código: Todo el código del sistema será comentado adecuadamente para facilitar su comprensión y mantenimiento. Esto permitirá a los desarrolladores realizar cambios o resolver problemas de manera eficiente.

## Pruebas unitarias: Se implementarán pruebas unitarias para cada componente del sistema.

## Registro de errores (logs): Se configurará un sistema de logs para registrar todas las actividades importantes del sistema, como errores o transacciones críticas, lo que facilitará la detección de problemas y su resolución.

## **3.3.6 Portabilidad**

## El sistema debe ser fácilmente portable a diferentes entornos y plataformas. Los requisitos de portabilidad son los siguientes:

## Uso de tecnologías multiplataforma: El sistema será desarrollado utilizando tecnologías que son ampliamente compatibles con diferentes sistemas operativos. En el backend, se usará Python, que es compatible con la mayoría de los entornos de desarrollo. En el frontend, se utilizará JavaScript, que es compatible con todos los navegadores modernos.

## 3.4 Planilla de especificación de Requisitos

**Revisa en la introducción de la semana este documento que está en formato Excel y continúa con la actividad.**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| [R-N°] | [Nombre del Requerimiento] | Tipo Requerimiento [Funcional, No Funcional] | Actores Relacionados | [Declración corta del requerimiento] | Estado |
| R. 1 | Consulta de Disponibilidad de Habitaciones | Funcional | Usuario (Cliente), Base de Datos | El sistema debe permitir a los usuarios consultar la disponibilidad de habitaciones para un rango de fechas específico. | Hecho |
| R. 2 | Registro de Reserva | Funcional | Usuario (Cliente), Base de Datos | El sistema debe permitir a los usuarios registrar una reserva para las habitaciones disponibles. | Hecho |
| R. 3 | Administración de Reservas (por Administradores) | Funcional | Administrador, Base de Datos | Los administradores deben poder visualizar, modificar o cancelar las reservas realizadas por los usuarios. | Hecho |
| R. 4 | Gestión del Catálogo de Habitaciones | Funcional | Administrador, Base de Datos | Los administradores deben poder añadir, modificar o eliminar habitaciones disponibles para la reserva. | Hecho |
| R. 5 | Cálculo del Pago de la Reserva | Funcional | Usuario (Cliente), Sistema de Pagos, Base de Datos | El sistema debe calcular automáticamente el monto a pagar por el usuario en función de la duración de su estancia y el tipo de habitación seleccionada. | Hecho |
| R. 6 | Procesamiento de Pagos | Funcional | Sistema de Pagos (API) | El sistema debe integrar un API de pago para procesar los pagos de las reservas. | Hecho |
| R. 7 | Generación de Confirmaciones y Tickets | Funcional | Usuario (Cliente), Base de Datos | El sistema debe generar una confirmación de reserva y enviarla por correo electrónico al cliente. | Hecho |
| R. 8 | Gestión de Perfiles de Usuario | Funcional | Usuario (Cliente), Base de Datos | El sistema debe permitir a los usuarios crear y gestionar sus perfiles personales. | Hecho |
| R. 9 | Registro de Clientes Nuevos | Funcional | Usuario (Cliente), Base de Datos | El sistema debe permitir que nuevos usuarios se registren en el sistema para realizar una reserva. | Hecho |
| R. 10 | Recuperación de Contraseña | Funcional | Usuario (Cliente) | El sistema debe permitir a los usuarios recuperar su contraseña en caso de olvido. | Hecho |
| R. 11 | Gestión de Estado de Reservas (Confirmadas, Pendientes, Canceladas) | Funcional | Administrador, Base de Datos | Los administradores deben poder cambiar el estado de las reservas (pendiente, confirmada, cancelada). | Hecho |
| R. 12 | Notificación de Confirmación de Reserva | Funcional | Usuario (Cliente), Base de Datos | El sistema debe enviar notificaciones automáticas por correo electrónico para confirmar las reservas. | Hecho |
| R. 13 | Control de Disponibilidad de Habitaciones | Funcional | Usuario (Cliente), Administrador, Base de Datos | El sistema debe asegurarse de que las habitaciones no se reserven múltiples veces para las mismas fechas. | Hecho |
| R. 14 | Requisitos de Rendimiento | No Funcional | Sistema, Usuarios | El sistema debe ser capaz de manejar hasta 100 usuarios simultáneos sin caídas de rendimiento significativas y garantizar un tiempo de respuesta para la consulta de disponibilidad de menos de 3 segundos. | Hecho |
| R. 15 | Seguridad | No Funcional | Sistema, Usuario (Cliente), Administrador | El sistema debe implementar seguridad como autenticación con JWT y protección contra vulnerabilidades como inyecciones SQL y XSS. | Hecho |
| R. 16 | Fiabilidad | No Funcional | Sistema | El sistema debe ser fiable, con un MTBF superior a 500 horas de uso continuo y capacidad de recuperación ante fallos en menos de 5 minutos. | Hecho |
| R. 17 | Disponibilidad | No Funcional | Sistema | El sistema debe estar disponible al menos el 99% del tiempo, con mantenimiento sin interrumpir el servicio. | Hecho |
| R. 18 | Mantenibilidad | No Funcional | Sistema | El sistema debe ser fácil de mantener y actualizar, con documentación adecuada y pruebas unitarias. | Hecho |
| R. 19 | Portabilidad | No Funcional | Sistema | El sistema debe ser fácilmente portable a diferentes entornos y plataformas utilizando tecnologías multiplataforma. | Hecho |





**Duoc UC**