**UNIVERSIDAD NACIONAL JOSÉ MARÍA ARGUEDAS**

**FACULTAD DE INGENIERÍA**

**ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMAS**



**PROYECTO**

DESARROLLO DE UN SISTEMA DE GESTIÓN ADMINISTRATIVA

PARA EL HOSTAL “EL ENCUENTRO”

**ASIGNATURA:**

INGENIERÍA DE SOFTWARE

**DOCENTE:**

MINAYA GUTIÉRREZ DANIEL

**PRESENTADO POR:**

1. Lara Leon Moises Augusto
2. Vargas Abollaneda Leydi Marilin
3. Mañuico Saqui Diego Armando

**Andahuaylas – Apurímac – Perú**

**2024**

**ÍNDICE**

[1 INTRODUCCIÓN 1](#_Toc184345157)

[1.1 OBJETIVOS DEL PROYECTO 2](#_Toc184345158)

[1.2 ALCANCE DEL PROYECTO 2](#_Toc184345159)

[1.3 JUSTIFICACIÓN DEL USO DE LA METODOLOGÍA XP 3](#_Toc184345160)

[1.4 CONTEXTO DEL PROYECTO 3](#_Toc184345161)

[2 VISIÓN DEL PROYECTO 3](#_Toc184345162)

[2.1 DESCRIPCIÓN GENERAL DEL SISTEMA DE GESTIÓN DEL HOSTAL 4](#_Toc184345163)

[2.2 SITUACIÓN PROBLEMÁTICA 4](#_Toc184345164)

[2.3 METAS DEL SISTEMA 5](#_Toc184345165)

[3 REQUISITOS DEL PROYECTO 5](#_Toc184345166)

[3.1 REQUISITOS FUNCIONALES 5](#_Toc184345167)

[3.1.1 Listar Estancia (Recepcionista) 5](#_Toc184345168)

[3.1.2 Gestionar habitaciones (Recepcionista) 6](#_Toc184345169)

[3.1.3 Asignar la habitación al huésped (Recepcionista) 6](#_Toc184345170)

[3.1.4 Liberar la habitación una vez que el huésped se retira (Recepcionista) 6](#_Toc184345171)

[3.1.5 Registrar al huésped si en caso es nuevo (Recepcionista) 6](#_Toc184345172)

[3.1.6 Agregar nuevas habitaciones (Administrador) 7](#_Toc184345173)

[3.1.7 Agregar nuevos productos (Administrador) 7](#_Toc184345174)

[3.1.8 Editar productos (Administrador) 7](#_Toc184345175)

[3.1.9 Mostrar lista de productos (Administrador) 7](#_Toc184345176)

[3.1.10 Mostrar lista de habitaciones (Administrador) 8](#_Toc184345177)

[3.1.11 Agregar nuevas habitaciones (Administrador) 8](#_Toc184345178)

[3.1.12 Editar habitaciones (Administrador) 8](#_Toc184345179)

[3.1.13 Generar Reportes detallados sobre las habitaciones más utilizadas mensualmente (Administrador) 9](#_Toc184345180)

[3.1.14 Generar reportes de los ingresos totales mensuales (Administrador). 9](#_Toc184345181)

[3.1.15 Mostrar lista de empleados (Administrador) 9](#_Toc184345182)

[3.1.16 Agregar nuevos empleados (Administrador) 9](#_Toc184345183)

[3.1.17 Editar datos de los empleados (Administrador) 10](#_Toc184345184)

[3.1.18 Gestionar roles de los empleados (Administrador) 10](#_Toc184345185)

[3.2 REQUISITOS NO FUNCIONALES 10](#_Toc184345186)

[3.3 REQUISITOS DE DOMINIO 11](#_Toc184345187)

[4 VALORES Y PRINCIPIOS DE XP EN EL PROYECTO 11](#_Toc184345188)

[4.1 COMUNICACIÓN 11](#_Toc184345189)

[4.2 SIMPLICIDAD 11](#_Toc184345190)

[4.3 RETROALIMENTACIÓN 12](#_Toc184345191)

[4.4 CORAJE 12](#_Toc184345192)

[4.5 RESPETO 12](#_Toc184345193)

[5 PRÁCTICAS DE XP APLICADAS 13](#_Toc184345194)

[5.1 DESARROLLO GUIADO POR PRUEBAS (TDD) 13](#_Toc184345195)

[5.2 PROGRAMACIÓN EN PAREJAS 13](#_Toc184345196)

[5.3 REFACTORIZACIÓN CONTINUA 14](#_Toc184345197)

[5.4 INTEGRACIÓN CONTINUA 14](#_Toc184345198)

[5.5 METÁFORA 14](#_Toc184345199)

[5.6 DISEÑO SIMPLE 14](#_Toc184345200)

[5.7 PROPIEDAD COLECTIVA DEL CÓDIGO 15](#_Toc184345201)

[5.8 RITMO SOSTENIBLE 15](#_Toc184345202)

[5.9 REUNIONES DIARIAS 15](#_Toc184345203)

[6 DESCRIPCIÓN DEL CLIENTE 15](#_Toc184345204)

[7 PLANIFICACIÓN 16](#_Toc184345205)

[8 HISTORIA DE USUARIO 16](#_Toc184345206)

[8.1 Módulo Recepción 18](#_Toc184345207)

[8.2 Módulo Reportes 21](#_Toc184345208)

[8.3 Modulo Empleados 21](#_Toc184345209)

[8.4 Módulo Productos 23](#_Toc184345210)

[8.5 Modulo Habitaciones 24](#_Toc184345211)

[9 METÁFORA DEL SISTEMA DE GESTIÓN ADMINISTRATIVA DEL HOSTAL 25](#_Toc184345212)

[9.1 PLAN DE ENTREGA DEL PROYECTO 26](#_Toc184345213)

[9.1.1 Priorización 26](#_Toc184345214)

[9.1.2 Plan de entregas 27](#_Toc184345215)

[9.1.3 Entrega Final y Revisión General 28](#_Toc184345216)

[10 DISEÑO 29](#_Toc184345217)

[10.1 DISEÑO DEL SISTEMA WEB DE GESTIÓN DEL HOSTAL 29](#_Toc184345218)

[10.2 ARQUITECTURA DEL SISTEMA 31](#_Toc184345219)

[10.2.1 Modelo 31](#_Toc184345220)

[10.2.2 Controlador 31](#_Toc184345221)

[10.2.3 Vistas 31](#_Toc184345222)

[10.3 DIAGRAMAS DE CASOS DE USO 31](#_Toc184345223)

[10.4 DIAGRAMAS DE CLASES 32](#_Toc184345224)

[11 CODIFICACIÓN 34](#_Toc184345225)

[11.1 CLIENTE SIEMPRE PRESENTE 34](#_Toc184345226)

[11.2 EL CÓDIGO SE ESCRIBE SIGUIENDO ESTÁNDARES 34](#_Toc184345227)

[11.3 PROPIEDAD COLECTIVA DEL CÓDIGO 34](#_Toc184345228)

[11.4 APLICACIÓN DE PRUEBAS 34](#_Toc184345229)

[12 PRUEBAS Y CONTROL DE CALIDAD 35](#_Toc184345230)

[12.1 PRUEBAS DE ACEPTACIÓN 35](#_Toc184345231)

[12.2 PRUEBAS UNITARIAS 35](#_Toc184345232)

[12.3 PRUEBAS DE INTEGRACIÓN 36](#_Toc184345233)

[12.4 PRUEBAS DE RENDIMIENTO 36](#_Toc184345234)

[12.4.1 ¿Qué probaremos? 36](#_Toc184345235)

[12.4.2 ¿Cómo lo haremos? 37](#_Toc184345236)

[12.4.3 Ejemplo de prueba de rendimiento: 37](#_Toc184345237)

[12.4.4 Criterio de éxito: 37](#_Toc184345238)

[13 LANZAMIENTO Y DESPLIEGUE DEL SISTEMA 37](#_Toc184345239)

[13.1 ESTRATEGIA DE DESPLIEGUE 37](#_Toc184345240)

[13.2 PREPARACIÓN DE ENTORNO DE PRODUCCIÓN 38](#_Toc184345241)

[13.3 CAPACITACIÓN Y SOPORTE A USUARIOS 38](#_Toc184345242)

[14 CONCLUSIONES Y RECOMENDACIONES 39](#_Toc184345243)

[14.1 LOGROS DEL PROYECTO 39](#_Toc184345244)

[14.2 LECCIONES APRENDIDAS 40](#_Toc184345245)

[14.3 RECOMENDACIONES PARA FUTUROS PROYECTOS 40](#_Toc184345246)

# INTRODUCCIÓN

El sistema de gestión administrativa para el hostal “El encuentro” es un proyecto de software diseñado para facilitar la administración eficiente de las operaciones diarias del hostal. Este sistema permitirá gestionar habitaciones, manejo de empleados, productos, y la generación de reportes. La implementación de un sistema automatizado no solo mejorará la eficiencia operativa, sino también la satisfacción del cliente, al proporcionar un servicio más rápido y preciso.

El propósito de este documento es definir de manera clara y detallada el objetivo del proyecto, la situación problemática, los requisitos funcionales y no funcionales, como también la planificación según la metodología XP.

El sistema proporciona una interfaz interactiva para dos actores principales: recepcionistas y administradores, cada uno con funcionalidades específicas que mejoran la eficiencia y precisión en las operaciones diarias del hostal.

## OBJETIVOS DEL PROYECTO

Desarrollar un sistema de gestión que permita a los hostales u hoteles optimizar la administración de habitaciones, empleados y productos, mejorando la experiencia de usuario tanto para los huéspedes como para el personal del hostal. Por ello se espera que el sistema permita:

❖ Automatizar el proceso de gestión de habitaciones, desde la asignación hasta la liberación tras el check-out (proceso de salida del huésped, donde se liquidan los gastos y se entrega la habitación).

❖ Proporcionar al administrador herramientas para gestionar empleados, productos y habitaciones, así como generar reportes detallados sobre el tipo habitaciones más rentadas

❖ Mejorar la precisión en el registro de huéspedes y la gestión de datos personales, permitiendo una experiencia más personalizada.

❖ Reducir errores humanos en la gestión manual de las operaciones y centralizar la información en un único sistema.

## ALCANCE DEL PROYECTO

El Sistema de Gestión del Hostal “El Encuentro” incluye las siguientes funcionalidades:

* **Registro de huéspedes**: El personal de recepción podrá registrar a los huéspedes nuevos.
* **Gestión de habitaciones**: El personal de recepción podrá asignar y liberar habitaciones a los huéspedes.
* **Gestión de productos**: El administrador podrá agregar, editar y mostrar la lista de productos del hostal, como servicios adicionales.
* **Administración de habitaciones**: Se podrán agregar y editar habitaciones, y se mostrará un listado actualizado de las mismas.
* **Gestión de empleados**: El administrador podrá agregar nuevos empleados, editar sus datos y gestionar sus roles dentro del sistema.
* **Generación de reportes**: El sistema permitirá generar reportes sobre las habitaciones más rentadas por fechas específicas, mostrando tendencias y estadísticas útiles para la administración del hotel.

El sistema está destinado para su uso en hostales de tamaño mediano a grande. Quedan fuera del alcance integraciones con sistemas de terceros para reservas y pagos en línea.

## JUSTIFICACIÓN DEL USO DE LA METODOLOGÍA XP

La metodología XP (Extreme Programming) ha sido seleccionada para el desarrollo de este proyecto debido a sus ventajas en proyectos de software donde los requisitos pueden cambiar con frecuencia y es necesario un enfoque ágil. En el contexto del sistema de gestión administrativa del Hostal “El Encuentro”, XP proporciona un marco adecuado para garantizar entregas rápidas, alta calidad del software y una constante adaptación a las necesidades del cliente.

Las prácticas fundamentales de XP, como el desarrollo iterativo, las pruebas continuas y la retroalimentación constante, permiten al equipo de desarrollo ajustar rápidamente el sistema según los comentarios de los usuarios finales. Además, XP fomenta la colaboración cercana con el cliente, lo que es esencial en un entorno hotelero dinámico, donde las operaciones diarias pueden requerir cambios rápidos.

## CONTEXTO DEL PROYECTO

El proyecto se desarrolla en un contexto hotelero donde la digitalización de procesos es esencial para mejorar la competitividad y la eficiencia operativa. En la actualidad, muchos hostales aún gestionan diferentes procesos mediante métodos manuales o sistemas desactualizados, lo que aumenta el margen de error y reduce la productividad. Este sistema de gestión administrativa busca proporcionar una solución que responda a las necesidades actuales del sector, modernizando la forma en que los hostales manejan sus operaciones cotidianas.

El sistema será diseñado teniendo en cuenta hostales de tamaño mediano a grande, con múltiples habitaciones y un flujo constante de huéspedes. Se espera que la implementación de este sistema no solo optimiza el funcionamiento interno del hostal, sino que también permita una mejor experiencia para los clientes, garantizando una gestión eficiente y precisa en cada aspecto.

# VISIÓN DEL PROYECTO

El proyecto tiene como visión proporcionar una solución tecnológica que permita al hostal optimizar sus operaciones diarias. El sistema pretende facilitar la administración y o gestionar habitaciones, empleados y productos, así como generar reportes precisos para la toma de decisiones. Al utilizar este sistema, se busca mejorar la eficiencia, reducir errores humanos, aumentar la satisfacción del cliente y fortalecer la competitividad del hostal en el mercado.

## DESCRIPCIÓN GENERAL DEL SISTEMA DE GESTIÓN DEL HOSTAL

Mediante el sistema de gestión administrativa del Hostal “El Encuentro” se podrá realizar los procesos fundamentales, tales como la gestión habitaciones (asignación y liberación), el registro de huéspedes, la administración del inventario de productos y la gestión de empleados. La interfaz del sistema es accesible para dos tipos de usuarios:

● **Recepcionista:** Responsable de la gestión de habitaciones, registro de huéspedes y quienes pueden consultar la disponibilidad de habitaciones,

● **Administrador:** Encargado de gestionar empleados, habitaciones, productos y generar reportes estadísticos para optimizar la operación del hotel.

El sistema se desarrollará de manera modular, permitiendo a cada actor interactuar con las funcionalidades específicas según su rol, garantizando la seguridad y el control de acceso.

## SITUACIÓN PROBLEMÁTICA

Actualmente, muchos hoteles aún dependen de procesos manuales o sistemas obsoletos para gestionar sus operaciones diarias. Esto resulta en una serie de problemas, tales como:

● **Errores en la asignación de habitaciones:** Al llevar un control manual o desactualizado, es común que se produzcan errores en las reservas, como el overbooking o la asignación incorrecta de habitaciones.

● **Ineficiencia en el manejo de información de huéspedes:** Los datos de los huéspedes pueden ser mal registrados o duplicados, afectando la experiencia del cliente y dificultando el seguimiento de historial.

● **Dificultad para generar reportes:** La falta de un sistema automatizado complica la recopilación y análisis de datos, lo que impide que los administradores tomen decisiones basadas en información precisa.

● **Mala gestión del personal:** Sin una plataforma centralizada, la gestión de empleados es poco eficiente y genera retrasos en la operación diaria.

## METAS DEL SISTEMA

Las metas del sistema de gestión administrativa del hostal son:

● **Optimizar el proceso de asignación y liberación de habitaciones**, reduciendo errores y mejorando la experiencia del huésped.

● **Facilitar la administración de los empleados** mediante un control eficiente de roles, permisos y datos.

● **Generar reportes detallados** sobre las operaciones del hotel, lo que permitirá a los administradores tomar decisiones informadas.

● **Mejorar la satisfacción del cliente**, ofreciendo una experiencia rápida, eficiente y personalizada.

● **Automatizar el control de servicios adicionales** que el hostal ofrece, mejorando su disponibilidad y seguimiento.

# REQUISITOS DEL PROYECTO

Los requisitos del proyecto están divididos en tres categorías: funcionales, no funcionales y de dominio. Estos definen las características que el sistema debe cumplir para satisfacer las necesidades del hotel.

## REQUISITOS FUNCIONALES

El Sistema de Gestión Administrativa para el Hostal El Encuentro tiene como objetivo principal optimizar y automatizar las operaciones diarias del hostal. En esta sección, se describen los requisitos funcionales que especifican las características esenciales que el sistema debe incluir para cumplir con las necesidades del hostal y proporcionar una herramienta eficiente para la gestión administrativa.

### Listar Estancia (Recepcionista)

* **Descripción:** El recepcionista puede ver un listado de todas las estancias activas y pasadas en el hotel.
* **Requisitos:**
  + El sistema debe permitir listar todas las estancias con información relevante, como nombre del huésped, habitación asignada, fechas de estadía.
  + El listado debe incluir filtros por fecha de estancia, estado de la estancia.

### Gestionar habitaciones (Recepcionista)

* **Descripción:** El recepcionista debe poder administrar el estado de las habitaciones (ocupada, libre, mantenimiento).
* **Requisitos:**
  + El sistema debe permitir al recepcionista cambiar el estado de una habitación.
  + Los estados de la habitación incluyen: libre, ocupada, en limpieza, en mantenimiento.
  + El sistema debe mostrar el estado actual de cada habitación de manera clara.

### Asignar la habitación al huésped (Recepcionista)

* **Descripción:** Asignación formal de la habitación al huésped cuando llega al hostal.
* **Requisitos:**
  + El recepcionista debe poder asignar una habitación a un huésped
  + El sistema debe permitir verificar si la habitación está lista antes de la asignación (por ejemplo, confirmando su limpieza).
  + Una vez asignada, la habitación debe marcar como ocupada en el sistema.

### Liberar la habitación una vez que el huésped se retira (Recepcionista)

* **Descripción:** Cuando el huésped se retira, el recepcionista debe marcar la habitación como libre.
* **Requisitos:**
  + El recepcionista debe poder liberar una habitación en el sistema, lo que la marca como disponible para futuras asignaciones.
  + El sistema debe registrar la fecha y hora de salida del huésped.
  + Después de liberar, la habitación debe marcarse como "limpieza pendiente" antes de ser ocupada nuevamente.

### Registrar al huésped si en caso es nuevo (Recepcionista)

* **Descripción:** Registrar al huésped si no existe previamente en la base de datos.
* **Requisitos:**
  + El recepcionista debe poder registrar los datos del huésped si este no ha sido previamente registrado.
  + Los datos del huésped deben ser almacenados de forma segura en la base de datos.
  + El sistema debe realizar una verificación para asegurarse de que no exista un registro duplicado.

### Agregar nuevas habitaciones (Administrador)

* **Descripción:** Permitir al administrador agregar nuevas habitaciones al inventario del hotel.
* **Requisitos:**
  + El sistema debe permitir registrar nuevas habitaciones con detalles como número, tipo, tarifa, y disponibilidad inicial.

### Agregar nuevos productos (Administrador)

* **Descripción:** El administrador puede agregar productos que el hotel ofrece, como productos de primera necesidad, servicios adicionales, etc.
* **Requisitos:**
  + El sistema debe permitir al administrador agregar nuevos productos con detalles como nombre, descripción, y precio.
  + Los productos deben estar disponibles para el huesped

### Editar productos (Administrador)

* **Descripción:** El administrador puede modificar los productos existentes.
* **Requisitos:**
  + El sistema debe permitir al administrador editar el nombre, descripción o precio de los productos existentes.
  + Los cambios deben ser registrados en tiempo real en el sistema para que estén disponibles a los huéspedes.

### Mostrar lista de productos (Administrador)

* **Descripción:** Visualizar el listado completo de productos que ofrece el hostal.
* **Requisitos:**
  + El sistema debe mostrar una lista de todos los productos registrados.
  + Cada producto debe tener opciones para ser editado o eliminado por el administrador.

### Mostrar lista de habitaciones (Administrador)

* **Descripción:** Ver un listado de todas las habitaciones registradas en el sistema.
* **Requisitos:**
  + El sistema debe mostrar una lista de todas las habitaciones, incluyendo su estado actual (ocupada, libre, en mantenimiento).
  + El administrador debe tener opciones para editar o eliminar habitaciones desde esta vista.

### Agregar nuevas habitaciones (Administrador)

* **Descripción:** Permitir al administrador agregar nuevas habitaciones al inventario del hostal en caso de que crezca.
* **Requisitos:**
  + El sistema debe permitir registrar nuevas habitaciones con detalles como número, tipo, tarifa, y disponibilidad inicial.
  + Los cambios deben reflejarse inmediatamente en el sistema

### Editar habitaciones (Administrador)

* **Descripción:** El administrador puede modificar los detalles de las habitaciones existentes.
* **Requisitos:**
  + El sistema debe permitir al administrador modificar detalles como el número de habitación, tipo, tarifas, y estado de disponibilidad.
  + Los cambios deben reflejarse inmediatamente en el sistema.

### Generar Reportes detallados sobre las habitaciones más utilizadas mensualmente (Administrador)

* **Descripción:** Permitir al administrador generar reportes sobre las habitaciones más utilizadas mensualmente
* **Requisitos:**
  + El sistema debe permitir seleccionar visualizar al administrador estadísticas mediante grafico de barras las habitaciones más usadas mensualmente

### Generar reportes de los ingresos totales mensuales (Administrador).

* **Descripción:** Permitir al administrador generar reportes detalladamente sobre los ingresos totales mensualmente.
* **Requisitos:**
  + El sistema debe permitir seleccionar el mes y año y generar el reporte del mes y año seleccionado

### Mostrar lista de empleados (Administrador)

* **Descripción:** El administrador puede visualizar una lista de todos los empleados registrados en el sistema.
* **Requisitos:**
  + El sistema debe mostrar un listado con los nombres, apellidos, cargos y otros detalles de los empleados registrados.
  + El administrador debe tener opciones para editar o eliminar empleados desde esta lista.

### Agregar nuevos empleados (Administrador)

* **Descripción:** Permitir al administrador registrar nuevos empleados en el sistema.
* **Requisitos:**
  + El sistema debe solicitar datos personales y laborales del empleado (nombre, apellido, cargo, etc.).
  + Los empleados deben ser agregados a la base de datos y tener un perfil de usuario creado.

### Editar datos de los empleados (Administrador)

* **Descripción:** El administrador puede modificar la información de los empleados registrados.
* **Requisitos:**
  + El sistema debe permitir al administrador editar los datos del empleado, como su nombre, apellido, y cargo.
  + Cualquier modificación debe ser actualizada en la base de datos de forma segura.

### Gestionar roles de los empleados (Administrador)

* **Descripción:** El administrador debe poder asignar roles y permisos a los empleados.
* **Requisitos:**
  + El sistema debe permitir asignar distintos roles a los empleados, como recepcionista, administrador, etc.
  + Los permisos deben determinar las acciones que cada empleado puede realizar dentro del sistema.

## REQUISITOS NO FUNCIONALES

Los **requisitos no funcionales** especifican las características de calidad que debe tener el sistema:

● **Seguridad:** El sistema debe garantizar la protección de los datos personales de los huéspedes y empleados mediante autenticación de usuarios y encriptación de datos sensibles.

● **Escalabilidad:** El sistema debe ser escalable para permitir su uso en hoteles de diferentes tamaños y poder manejar un mayor volumen de datos y usuarios.

● **Rendimiento:** El sistema debe ser capaz de procesar grandes volúmenes de datos en tiempo real sin afectar el rendimiento general.

● **Usabilidad:** La interfaz del sistema debe ser intuitiva y fácil de usar para los distintos tipos de usuarios, sin requerir un entrenamiento extenso.

● **Disponibilidad:** El sistema debe estar disponible 24/7, con tiempos de inactividad mínimos para no interrumpir las operaciones del hotel.

## REQUISITOS DE DOMINIO

Los requisitos de dominio son aquellos relacionados con las reglas de negocio específicas del entorno del hostal:

● El sistema debe cumplir con las regulaciones locales e internacionales sobre la protección de datos de los huéspedes, cumplir con políticas de gestión de la seguridad de la información (ISO/IEC 27001), asegurando que los datos de los clientes estén encriptados y seguros.

● El sistema debe permitir la gestión de varios tipos de habitaciones (estándar, matrimonial,) con precios variables según la temporada.

● El sistema debe manejar el control de inventario de productos adicionales ofrecidos por el hotel.

● El sistema debe generar reportes financieros que ayuden en la toma de decisiones estratégicas para la administración del hotel.

# VALORES Y PRINCIPIOS DE XP EN EL PROYECTO

La metodología de programación extrema enfatiza la importancia de la adaptabilidad y la comunicación en el proceso de desarrollo, prefiriendo invertir en controlar los cambios de requerimientos del proyecto a definirlos al comienzo del proyecto. Es por ello que se apoya de cinco valores:

## COMUNICACIÓN

La comunicación constante es esencial en el desarrollo del sistema para el Hostal El Encuentro. Hemos asegurado un flujo de información claro entre el equipo de desarrollo y los stakeholders, como el administrador y el recepcionista, para capturar los requerimientos específicos del hostal.

Se llevan a cabo reuniones frecuentes para discutir los avances y aclarar dudas sobre funcionalidades como la gestión de habitaciones o generación de reportes.

Además, se utiliza documentación ágil y código limpio para que el equipo pueda colaborar sin malentendidos.

## SIMPLICIDAD

La simplicidad en el proyecto se refleja en la construcción de un sistema que satisface únicamente las necesidades esenciales del hostal, evitando características innecesarias.

Por ejemplo, decidimos no incluir la funcionalidad de reservaciones en línea para enfocarnos en optimizar las operaciones internas del hostal.

Además, el diseño del sistema se mantiene modular para permitir ajustes futuros sin complejidad adicional.

## RETROALIMENTACIÓN

La retroalimentación es un pilar fundamental en XP y se aplica continuamente en el desarrollo del sistema del **Hostal El Encuentro**.

* **Interacción con stakeholders**: El administrador del hostal participa activamente en las revisiones regulares del sistema, proporcionando comentarios sobre funcionalidades como la gestión de habitaciones y reportes. Esto asegura que las entregas parciales del proyecto cumplan con las expectativas reales del negocio.
* **Pruebas constantes**: Se realizan pruebas frecuentes, tanto unitarias como de aceptación, para recibir retroalimentación técnica inmediata sobre el funcionamiento del sistema y resolver posibles errores antes de avanzar a nuevas etapas.
* **Iteraciones rápidas**: Cada iteración incluye revisiones y ajustes basados en los comentarios recibidos, fomentando un proceso de mejora continua.

## CORAJE

El valor del coraje se refleja en la capacidad del equipo para tomar decisiones desafiantes, priorizando siempre las necesidades del proyecto por encima de las comodidades.

* **Toma de decisiones informadas**: Se decidió limitar las funcionalidades del sistema al entorno interno del hostal (sin incluir reservaciones en línea), asumiendo que este enfoque es más práctico y manejable para el equipo y los usuarios finales.
* **Enfrentar cambios en los requerimientos**: El equipo está preparado para adaptarse a modificaciones en las necesidades del administrador, entendiendo que las prioridades pueden evolucionar durante el desarrollo.

## RESPETO

El respeto es esencial en XP y se fomenta tanto en la interacción del equipo de desarrollo como en la relación con los usuarios del sistema.

* **Respeto por los roles**: Se reconoce la importancia de las opiniones del administrador y del recepcionista como usuarios finales del sistema, asegurando que sus necesidades y preocupaciones sean escuchadas y atendidas.
* **Cultura de colaboración**: Dentro del equipo, se promueve un ambiente en el que todas las ideas son valoradas y cualquier duda o sugerencia puede ser expresada sin temor a juicio.
* **Calidad del trabajo**: El equipo muestra respeto hacia los usuarios finales entregando un sistema funcional, confiable y fácil de usar, acorde con los estándares del hostal.

# PRÁCTICAS DE XP APLICADAS

En el desarrollo del DESARROLLO DE UN SISTEMA DE GESTIÓN ADMINISTRATIVA

PARA EL HOSTAL “EL ENCUENTRO”, aplicaremos las prácticas de **Extreme Programming (XP)** para asegurar que el proyecto se mantenga ágil, adaptable y siempre enfocado en satisfacer las necesidades del cliente. A continuación, explicaremos cómo cada práctica se aplicará en el contexto del proyecto.

## DESARROLLO GUIADO POR PRUEBAS (TDD)

El Desarrollo Guiado por Pruebas asegura que cada funcionalidad del sistema sea validada antes de ser implementada. En el caso de nuestro sistema hotelero, por ejemplo, cada vez que se desarrolla una nueva funcionalidad, como listar las estancias o la gestión de empleados, primero se escribirán pruebas unitarias que definan el comportamiento esperado. Estas pruebas garantizarán que, antes de liberar cualquier nueva funcionalidad, esta se comporte de manera correcta. Por ejemplo, al implementar la funcionalidad de asignación de habitaciones, primero definimos cómo debería comportarse el sistema en distintos escenarios, como cuando una habitación ya está reservada. Estas pruebas garantizan que el sistema responde correctamente a cada situación.

## PROGRAMACIÓN EN PAREJAS

La programación en parejas nos permite mejorar la calidad del código desde el principio, ya que dos desarrolladores trabajan en el mismo código al mismo tiempo. En el desarrollo del sistema de gestión administrativa del hostal, esta práctica será útil para funciones críticas como la gestión de datos de huéspedes o el cálculo de costos de las estancias. Nuestro equipo podrá revisar constantemente el código, reducir la probabilidad de errores y generar soluciones más creativas y eficientes. Al trabajar en la interfaz de usuario para los recepcionistas, por ejemplo, la programación en parejas nos permitirá que dos de los desarrolladores se enfoquen en la funcionalidad técnica (validar la disponibilidad de habitaciones) mientras que los otros se aseguren de que la interfaz sea amigable y eficiente.

## REFACTORIZACIÓN CONTINUA

La refactorización continua implica que, durante el desarrollo del sistema, se mejorará constantemente el código existente sin cambiar su funcionalidad. En el caso del sistema administrativo del hostal, esto nos permitirá simplificar y optimizar el código a medida que surjan nuevas necesidades. A lo largo del desarrollo, podríamos identificar maneras de hacer que la funcionalidad del sistema sea más eficiente, como optimizar la forma en que se calculan las ocupaciones de habitaciones o se generan los reportes administrativos.

## INTEGRACIÓN CONTINUA

La integración continua será clave para asegurar las distintas partes del sistema (estancias, administración de empleados, etc.) funcionen juntas sin problemas. Cada vez que se complete una nueva funcionalidad, se integrará de inmediato al sistema completo, realizando pruebas automáticas para verificar que todas las partes del sistema sigan funcionando correctamente.

## METÁFORA

La metáfora en XP es una forma de mantener una visión clara y compartida del proyecto. En nuestro caso, podemos utilizar la metáfora de un "hostal digital", donde cada acción que los usuarios realizan en el sistema (como asignar una habitación o gestionar un empleado) es como una interacción real dentro del hotel. Esta metáfora guía el diseño del sistema para que sea intuitivo y fácil de entender.

Por ejemplo, las habitaciones pueden verse como "objetos asignables" dentro del sistema, lo que nos ayuda a visualizar y organizar mejor las distintas funcionalidades asociadas a ellas, como la disponibilidad, asignación y liberación de la habitación.

## DISEÑO SIMPLE

El diseño simple es una de las claves de XP. El objetivo es no complicar innecesariamente el código. Para el sistema de gestión del hostal, esto significa que las funcionalidades deben ser claras y directas. Por ejemplo, en la gestión de habitaciones, no necesitamos sobrecargar la funcionalidad con detalles innecesarios; basta con que el sistema permite al recepcionista consultar la disponibilidad, registrar la asignación y generar una confirmación de manera rápida y eficiente.

## PROPIEDAD COLECTIVA DEL CÓDIGO

La **propiedad colectiva del código** asegura que todos los desarrolladores del equipo puedan trabajar en cualquier parte del sistema, sin depender exclusivamente de una sola persona para cada módulo. Esto es esencial en nuestro proyecto, ya que los distintos componentes del sistema (estancias, gestión de empleados, administración de habitaciones) están interconectados.

## RITMO SOSTENIBLE

El **ritmo sostenible** es vital para mantener la productividad del equipo sin agotamiento. En el proyecto del sistema de gestión del hostal, planificamos las entregas de manera que el equipo pueda mantener un ritmo constante. Esto implica que no sobrecargaremos al equipo con demasiadas funcionalidades en poco tiempo, sino que mantendremos un flujo constante de trabajo, lo que resultará en un sistema más estable y de mejor calidad.

## REUNIONES DIARIAS

Las **reuniones diarias** son un componente esencial en la gestión ágil de proyectos, pero su implementación efectiva puede presentar desafíos. Uno de los principales obstáculos es la coordinación de horarios, especialmente con miembros que tienen diferentes cargas de trabajo. Además, mantener la brevedad y el enfoque de estas reuniones puede ser difícil; es fácil desviarse hacia discusiones no planificadas que consumen tiempo valioso. Es por ello que, aunque las reuniones diarias no se realicen efectivamente, el compromiso con la comunicación y la colaboración no debe disminuir.

# DESCRIPCIÓN DEL CLIENTE

El cliente es el Hostal “El Encuentro”, el cual se presenta como una opción de alojamiento de categoría media. Este establecimiento busca mejorar la eficiencia administrativa de los distintos procesos que se realizan en el hostal, por lo que el cliente espera lograr esta mejora mediante la implementación de un sistema de gestión administrativa en línea robusto pero económico.

Dado que para un sitio web el cliente esperaría no hacer una inversión muy grande, es crucial enfocarse en funcionalidades esenciales que maximicen la experiencia del usuario sin incurrir en costos excesivos. La prioridad debe ser una navegación sencilla y una interfaz amigable que permita a los usuarios realizar las distintas actividades de manera eficiente y acceder a información importante sobre el hostal.

# PLANIFICACIÓN

En la fase de planificación para el desarrollo del sistema del **Hostal El Encuentro**, se estableció una comunicación constante entre el equipo de desarrollo y los usuarios clave del sistema, es decir, el administrador y el recepcionista. Esto permitió identificar las necesidades específicas del hostal y priorizar los módulos esenciales que deben ser implementados.

El alcance del proyecto se definió considerando las siguientes funcionalidades principales:

* **Recepción**: Gestión del registro de huéspedes y asignación de habitaciones.
* **Administración**: Generación de reportes financieros y control general del sistema.
* **Empleados**: Registro y manejo de la información del personal del hostal.
* **Habitaciones**: Control de disponibilidad y precios según las temporadas.
* **Productos**: Inventario de servicios adicionales ofrecidos, como alimentos o bebidas.
* **Reportes**: Análisis de datos relevantes, como habitaciones más alquiladas o ingresos por período.

Durante esta etapa, se acordaron iteraciones cortas para implementar y revisar progresivamente cada módulo, garantizando que el sistema cumpla con las expectativas del administrador del hostal y que los entregables parciales sean funcionales.

# HISTORIA DE USUARIO

Las historias de usuario se deben articular en un idioma accesible, asegurando que sean comprensibles para todos los involucrados (Clientes, Desarrolladores y Usuarios). Estas narrativas reflejan las necesidades que el sistema debe satisfacer. Aunque las historias no fueron redactadas directamente por el cliente, se establecieron a partir de su visión sobre las funcionalidades que el sistema debería ofrecer:

* Listar estancias
* Gestionar habitaciones
* Asignar habitación
* Liberar habitación
* Registrar huésped
* Listar productos
* Agregar producto
* Editar producto
* Listar habitaciones
* Agregar habitación
* Editar habitación
* Listar empleados
* Agregar empleado
* Editar empleado
* Gestión de roles
* Generar reportes

La tabla 1 muestra el diseño de las historias de usuario.

**Tabla 1: Plantilla de Historia de Usuario**

| **HISTORIA DE USUARIO** | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Número:** |  | | **Usuario:** | |  | | |
| **Nombre de historia:** | | |  | | | | |
| **Prioridad en negocio:** | | |  | | | **Riesgo en desarrollo:** |  |
| **Días estimados:** | |  | | | | **Iteración asignada:** |  |
| **Programador responsable:** | | | |  | | | |
| **Descripción:** | | | | | | | |
| **Observaciones:** | | | | | | | |

*Fuente: Elaboración propia*

A continuación, se describen los campos de la plantilla de historia de usuario:

* **Número:** Es el número que identifica a cada historia de usuario.
* **Usuario:** La persona que realizará la actividad descrita en la historia de usuario.
* **Nombre de la historia:** Denominación textual por la cual se identificará a la historia de usuario.
* **Prioridad en negocio:** La importancia que la historia tiene en el negocio, la cual podrá ser alto, medio o bajo.
* **Riesgo en desarrollo:** La complejidad que se presentará al momento de desarrollar la historia de usuario; alto, medio o bajo según sea el caso.
* **Días estimados:** Cantidad de días estimados para implementar la historia de usuario.
* **Iteración asignada:** Número de iteración en el que el cliente desea que se implemente la historia de usuario.
* **Programador responsable:** Nombre de la persona que está a cargo del desarrollo de la historia de usuario.
* **Descripción:** Detalles de las actividades que tendrá la historia de usuario.

**Observaciones:** Notas importantes sobre la historia de usuario.

A continuación, las tablas 2-17 muestran las historias de usuario usadas para llevar a cabo el desarrollo del sistema.

## Módulo Recepción

Tabla 2: Listar estancias

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **HISTORIA DE USUARIO** | | | | | | | |
| **Número:** | 01 | | **Usuario:** | | Recepcionista | | |
| **Nombre de historia:** | | | Listar estancias | | | | |
| **Prioridad en negocio:** | | | Alto | | | **Riesgo en desarrollo:** | Alto |
| **Días estimados:** | |  | | | | **Iteración asignada:** |  |
| **Programador responsable:** | | | |  | | | |
| **Descripción:** En la pantalla de inicio, se visualizan todas las estancias activas con los datos del huésped (nombres, apellidos, documento de identificación, teléfono), la fechas y la habitación (número de habitación).  Se tendrá una opción que filtre las estancias por piso. | | | | | | | |
| **Observaciones:** | | | | | | | |

*Tabla 3: Gestionar habitaciones*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **HISTORIA DE USUARIO** | | | | | | | |
| **Número:** | 02 | | **Usuario:** | | Recepcionista | | |
| **Nombre de historia:** | | | Gestionar habitaciones | | | | |
| **Prioridad en negocio:** | | | Alto | | | **Riesgo en desarrollo:** | Alto |
| **Días estimados:** | |  | | | | **Iteración asignada:** |  |
| **Programador responsable:** | | | |  | | | |
| **Descripción:** En la pantalla de inicio, se visualiza el listado de las habitaciones con los campos número de habitación, piso en el que se encuentra, categoría, su estado (ocupado/libre/en limpieza), tarifa y servicios incluidos.  Se tendrá una opción para proceder con la asignación de la habitación.  Se tendrá una opción para proceder con la liberación de la habitación.  Se tendrá una opción para designar el estado de una habitación a “Mantenimiento”, solo en el caso de las habitaciones no asignadas.  Se tendrá una opción que filtre las habitaciones por piso.  Se tendrá una opción que filtre las habitaciones por tipo.  Se tendrá una opción que filtre las habitaciones por estado.  Se tendrá una opción que ordene las habitaciones por precio. | | | | | | | |
| **Observaciones:** | | | | | | | |

*Tabla 4: Asignar habitación*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **HISTORIA DE USUARIO** | | | | | | | |
| **Número:** | 03 | | **Usuario:** | | Recepcionista | | |
| **Nombre de historia:** | | | Asignar habitación | | | | |
| **Prioridad en negocio:** | | | Alto | | | **Riesgo en desarrollo:** | Alto |
| **Días estimados:** | |  | | | | **Iteración asignada:** |  |
| **Programador responsable:** | | | |  | | | |
| **Descripción:** El sistema muestra un campo para que se ingrese el número del documento de identificación del huésped, para luego mostrar los datos personales del huésped.  Se tendrá una opción para proceder con la asignación de la habitación.  Se tendrá una opción para cancelar la asignación. | | | | | | | |
| **Observaciones:**  Es necesario que el campo de documento de identidad sea obligatorio.  Es necesario informar con un mensaje de alerta si el número del documento ingresado coincide o no con la información existente en la base de datos. | | | | | | | |

*Tabla 5: Liberar habitación*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **HISTORIA DE USUARIO** | | | | | | | |
| **Número:** | 04 | | **Usuario:** | | Recepcionista | | |
| **Nombre de historia:** | | | Liberar habitación | | | | |
| **Prioridad en negocio:** | | | Alto | | | **Riesgo en desarrollo:** | Alto |
| **Días estimados:** | |  | | | | **Iteración asignada:** |  |
| **Programador responsable:** | | | |  | | | |
| **Descripción:** El sistema muestra un pop-up con el número de habitación, los datos personales del huésped (nombres y apellidos), días de estancia y el costo calculado.  Se tendrá una opción para confirmar la salida.  Se tendrá una opción para cancelar la salida. | | | | | | | |
| **Observaciones:** | | | | | | | |

*Tabla 6: Registrar huésped*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **HISTORIA DE USUARIO** | | | | | | | |
| **Número:** | 05 | | **Usuario:** | | Recepcionista | | |
| **Nombre de historia:** | | | Registrar huésped | | | | |
| **Prioridad en negocio:** | | | Alto | | | **Riesgo en desarrollo:** | Alto |
| **Días estimados:** | |  | | | | **Iteración asignada:** |  |
| **Programador responsable:** | | | |  | | | |
| **Descripción:** En la pantalla de inicio, se visualiza el listado de todos los huéspedes registrados (nombres, apellidos, documento de identificación, correo electrónico, teléfono).  Se tendrá un apartado para registrar a un nuevo huésped con los campos nombres, apellidos, documento de identificación, correo electrónico y teléfono.  Se tendrá una opción para confirmar el nuevo registro. | | | | | | | |
| **Observaciones:** Es necesario que para realizar un nuevo registro los campos documento de identidad, nombres y apellidos sean obligatorios.  Es necesario informar con un mensaje de alerta si el número del documento ingresado coincide con la información existente en la base de datos. | | | | | | | |

## Módulo Reportes

*Tabla 7: Generar reportes*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **HISTORIA DE USUARIO** | | | | | | | |
| **Número:** | 06 | | **Usuario:** | | Administrador | | |
| **Nombre de historia:** | | | Generar reportes | | | | |
| **Prioridad en negocio:** | | | Alto | | | **Riesgo en desarrollo:** | Alto |
| **Días estimados:** | |  | | | | **Iteración asignada:** |  |
| **Programador responsable:** | | | |  | | | |
| **Descripción:** En la pantalla de inicio, se visualiza un apartado con dos campos para fecha, de inicio y fin, con la opción para generar un informe de la cantidad de estancias que se realizaron entre estas fechas. | | | | | | | |
| **Observaciones:** | | | | | | | |

## Modulo Empleados

*Tabla 8: Listar empleados*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **HISTORIA DE USUARIO** | | | | | | | |
| **Número:** | 07 | | **Usuario:** | | Administrador | | |
| **Nombre de historia:** | | | Listar empleados | | | | |
| **Prioridad en negocio:** | | | Alto | | | **Riesgo en desarrollo:** | Alto |
| **Días estimados:** | |  | | | | **Iteración asignada:** |  |
| **Programador responsable:** | | | |  | | | |
| **Descripción:** En la pantalla de inicio, se visualiza la lista de empleados (nombres, apellidos, correo electrónico, teléfono, rol y estado).  Se tendrá una opción para agregar un nuevo empleado.  Se tendrá una opción para editar un empleado. | | | | | | | |
| **Observaciones:** | | | | | | | |

*Tabla 9: Agregar empleado*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **HISTORIA DE USUARIO** | | | | | | | |
| **Número:** | 08 | | **Usuario:** | | Administrador | | |
| **Nombre de historia:** | | | Agregar empleado | | | | |
| **Prioridad en negocio:** | | | Alto | | | **Riesgo en desarrollo:** | Alto |
| **Días estimados:** | |  | | | | **Iteración asignada:** |  |
| **Programador responsable:** | | | |  | | | |
| **Descripción:** El sistema muestra los campos para registrar un nuevo empleado por sus datos personales (nombre, apellido, documento de identificación, correo electrónico, teléfono).  Se tendrá una opción para proceder con el registro.  Se tendrá una opción para cancelar el registro. | | | | | | | |
| **Observaciones:** | | | | | | | |

*Tabla 10: Editar empleado*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **HISTORIA DE USUARIO** | | | | | | | |
| **Número:** | 09 | | **Usuario:** | | Administrador | | |
| **Nombre de historia:** | | | Editar empleado | | | | |
| **Prioridad en negocio:** | | | Alto | | | **Riesgo en desarrollo:** | Alto |
| **Días estimados:** | |  | | | | **Iteración asignada:** |  |
| **Programador responsable:** | | | |  | | | |
| **Descripción:** El sistema muestra los campos para editar los datos personales de un empleado (nombre, apellido, documento de identificación, correo electrónico, teléfono), así como también su estado (habilitado/no habilitado).  Se tendrá una opción para actualizar los cambios.  Se tendrá una opción para cancelar los cambios. | | | | | | | |
| **Observaciones:** | | | | | | | |

*Tabla 11: Gestión de roles*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **HISTORIA DE USUARIO** | | | | | | | |
| **Número:** | 10 | | **Usuario:** | | Administrador | | |
| **Nombre de historia:** | | | Gestión de roles | | | | |
| **Prioridad en negocio:** | | | Alto | | | **Riesgo en desarrollo:** | Alto |
| **Días estimados:** | |  | | | | **Iteración asignada:** |  |
| **Programador responsable:** | | | |  | | | |
| **Descripción:** En la pantalla de inicio, se visualiza el listado de todos los roles asignados (nombres, apellidos, correo electrónico y rol).  Se tendrá un apartado para asignar un nuevo rol. | | | | | | | |
| **Observaciones:** | | | | | | | |

## Módulo Productos

*Tabla 12: Listar productos*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **HISTORIA DE USUARIO** | | | | | | | |
| **Número:** | 11 | | **Usuario:** | | Administrador | | |
| **Nombre de historia:** | | | Listar productos | | | | |
| **Prioridad en negocio:** | | | Alto | | | **Riesgo en desarrollo:** | Alto |
| **Días estimados:** | |  | | | | **Iteración asignada:** |  |
| **Programador responsable:** | | | |  | | | |
| **Descripción:** En la pantalla de inicio, se visualiza la lista de productos  Se tendrá una opción para agregar un nuevo producto.  Se tendrá una opción para editar un producto. | | | | | | | |
| **Observaciones:** | | | | | | | |

*Tabla 13: Agregar producto*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **HISTORIA DE USUARIO** | | | | | | | |
| **Número:** | 12 | | **Usuario:** | | Administrador | | |
| **Nombre de historia:** | | | Agregar producto | | | | |
| **Prioridad en negocio:** | | | Alto | | | **Riesgo en desarrollo:** | Alto |
| **Días estimados:** | |  | | | | **Iteración asignada:** |  |
| **Programador responsable:** | | | |  | | | |
| **Descripción:** El sistema muestra los campos para registrar un nuevo producto.  Se tendrá una opción para proceder con el registro.  Se tendrá una opción para cancelar el registro. | | | | | | | |
| **Observaciones:** | | | | | | | |

*Tabla 14: Editar producto*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **HISTORIA DE USUARIO** | | | | | | | |
| **Número:** | 13 | | **Usuario:** | | Administrador | | |
| **Nombre de historia:** | | | Editar producto | | | | |
| **Prioridad en negocio:** | | | Alto | | | **Riesgo en desarrollo:** | Alto |
| **Días estimados:** | |  | | | | **Iteración asignada:** |  |
| **Programador responsable:** | | | |  | | | |
| **Descripción:** El sistema muestra los campos para editar los datos de un producto.  Se tendrá una opción para actualizar los cambios.  Se tendrá una opción para cancelar los cambios. | | | | | | | |
| **Observaciones:** | | | | | | | |

## Modulo Habitaciones

*Tabla 15: Listar habitaciones*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **HISTORIA DE USUARIO** | | | | | | | |
| **Número:** | 14 | | **Usuario:** | | Administrador | | |
| **Nombre de historia:** | | | Listar habitaciones | | | | |
| **Prioridad en negocio:** | | | Alto | | | **Riesgo en desarrollo:** | Alto |
| **Días estimados:** | |  | | | | **Iteración asignada:** |  |
| **Programador responsable:** | | | |  | | | |
| **Descripción:** En la pantalla de inicio, se visualiza la lista de habitaciones (id, numero, planta, tipo, estado, precio y servicios)  Se tendrá una opción para agregar una nueva habitación.  Se tendrá una opción para editar una habitación. | | | | | | | |

*Tabla 16: Agregar habitación*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **HISTORIA DE USUARIO** | | | | | | | |
| **Número:** | 15 | | **Usuario:** | | Administrador | | |
| **Nombre de historia:** | | | Agregar habitación | | | | |
| **Prioridad en negocio:** | | | Alto | | | **Riesgo en desarrollo:** | Alto |
| **Días estimados:** | |  | | | | **Iteración asignada:** |  |
| **Programador responsable:** | | | |  | | | |
| **Descripción:** El sistema muestra los campos para registrar una nueva habitación (id, numero, planta, tipo, precio y servicios).  Se tendrá una opción para proceder con el registro.  Se tendrá una opción para cancelar el registro. | | | | | | | |
| **Observaciones:** | | | | | | | |

*Tabla 17: Editar habitación*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **HISTORIA DE USUARIO** | | | | | | | |
| **Número:** | 16 | | **Usuario:** | | Administrador | | |
| **Nombre de historia:** | | | Editar habitación | | | | |
| **Prioridad en negocio:** | | | Alto | | | **Riesgo en desarrollo:** | Alto |
| **Días estimados:** | |  | | | | **Iteración asignada:** |  |
| **Programador responsable:** | | | |  | | | |
| **Descripción:** El sistema muestra los campos para editar los datos de una habitación.  Se tendrá una opción para actualizar los cambios.  Se tendrá una opción para cancelar los cambios. | | | | | | | |
| **Observaciones:** | | | | | | | |

# METÁFORA DEL SISTEMA DE GESTIÓN ADMINISTRATIVA DEL HOSTAL

La administración del hostal “El Encuentro” desea mejorar la eficiencia con la que se realizan las actividades en el hostal, y basados en la experiencia de manejar su control en registros físicos, vio como una oportunidad de mejora el implementar un sistema de gestión administrativo. De tal forma que los procesos realizados en el hostal sean controlados digitalmente. La gestión del hostal debe ser dada desde dos puntos de vista. En primera instancia está el punto de vista del administrador y luego la del recepcionista. Actualmente el hostal hace uso de registros físicos para el control de las actividades realizadas en el día a día del negocio, por lo que el sistema será desarrollado en base a estas herramientas de control. En el proceso de creación del proyecto, se definieron los módulos: Recepción, Administración, Empleados, Habitaciones, Productos y Reportes, cada uno de los cuales cumple con sus funciones.

* El módulo **Recepción**, permitirá gestionar las habitaciones (Check-in/Check-out), registrar a los huéspedes y consultar las estancias.
* En el módulo **Empleados**, permitirá gestionar los roles de los empleados y a los mismos, listar, agregar y editar empleados.
* En el módulo **Habitaciones**, estarán las opciones de gestión de habitaciones como, listar, agregar y editar.
* En el módulo **Productos**, estarán las opciones de gestión de productos como, listar, agregar y editar.
* En el módulo **Reportes**, estarán las opciones para la generación de reportes de estancias entre determinadas fechas.

La gestión admirativa del hostal se dará por roles, por ello es que se tendrá usuarios definidos en el sistema para acceder a cada uno de estos roles y sus funcionalidades respectivas.

## PLAN DE ENTREGA DEL PROYECTO

Para asegurar el éxito del desarrollo y entrega del sistema, se ha estructurado un plan de entrega basado en **iteraciones**, permitiendo que el cliente pueda ir probando funcionalidades clave de forma gradual y asegurando que se cumplan los plazos establecidos. Cada iteración incluirá tanto el desarrollo como la prueba de las funcionalidades entregadas.

### Priorización

La tabla 18 muestra la priorización de las iteraciones de cada historia de acuerdo a la estimación de la complejidad para desarrollar las mismas.

*Tabla 18: Priorización*

|  |  |  |  |
| --- | --- | --- | --- |
| **PRIORIZACIÓN** | | | |
| **Módulos** | **Nº Historia** | **Nombre Historia** | **Iteración** |
| Modulo Recepción | 1 | Listar estancias | 3 |
| 2 | Gestionar habitaciones | 3 |
| 3 | Asignar habitación | 3 |
| 4 | Liberar habitación | 3 |
| 5 | Registrar huésped | 3 |
| Modulo Reportes | 6 | Generar reportes | 4 |
| Modulo Empleados | 7 | Listar empleados | 1 |
| 8 | Agregar empleado | 1 |
| 9 | Editar empleado | 1 |
| 10 | Gestión de roles | 1 |
| Modulo Productos | 11 | Listar productos | 2 |
| 12 | Agregar producto | 2 |
| 13 | Editar producto | 2 |
| Modulo Habitaciones | 14 | Listar habitaciones | 1 |
| 15 | Agregar habitación | 1 |
| 16 | Editar habitación | 1 |

### Plan de entregas

La tabla 19 muestra el plan de entregas, donde se definen las fechas de inicio y fin de cada historia de usuario basadas en las iteraciones y los días. Dado que se desconoce cuánto podría llegar a tardar en desarrollarse cada historia se omitieron las fechas y estimaciones.

*Tabla 19: Plan de entregas*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **PLAN DE ENTREGAS** | | | | | |
| **Nº Historia** | **Nombre Historia** | **Iteración** | **Días Estimados** | **Fecha Inicio** | **Fecha Fin** |
| 7 | Listar empleados | 1 | - | - | - |
| 8 | Agregar empleado | 1 | - | - | - |
| 9 | Editar empleado | 1 | - | - | - |
| 10 | Gestión de roles | 1 | - | - | - |
| 14 | Listar habitaciones | 1 | - | - | - |
| 15 | Agregar habitación | 1 | - | - | - |
| 16 | Editar habitación | 1 | - | - | - |
| 11 | Listar productos | 2 | - | - | - |
| 12 | Agregar producto | 2 | - | - | - |
| 13 | Editar producto | 2 | - | - | - |
| 1 | Listar estancias | 3 | - | - | - |
| 2 | Gestionar habitaciones | 3 | - | - | - |
| 3 | Asignar habitación | 3 | - | - | - |
| 4 | Liberar habitación | 3 | - | - | - |
| 5 | Registrar huésped | 3 | - | - | - |
| 6 | Generar reportes | 4 | - | - | - |

### Entrega Final y Revisión General

**Fecha:** [10/12/2024]  
**Descripción:**Una vez completadas todas las funcionalidades del producto y habiendo pasado todas las fases de pruebas, se realizará una **revisión general del sistema** para garantizar que todas las funcionalidades trabajen en conjunto sin conflictos. Durante esta fase, también se implementarán los ajustes finales según las observaciones del cliente.

* ***Capacitación y Soporte Inicial***

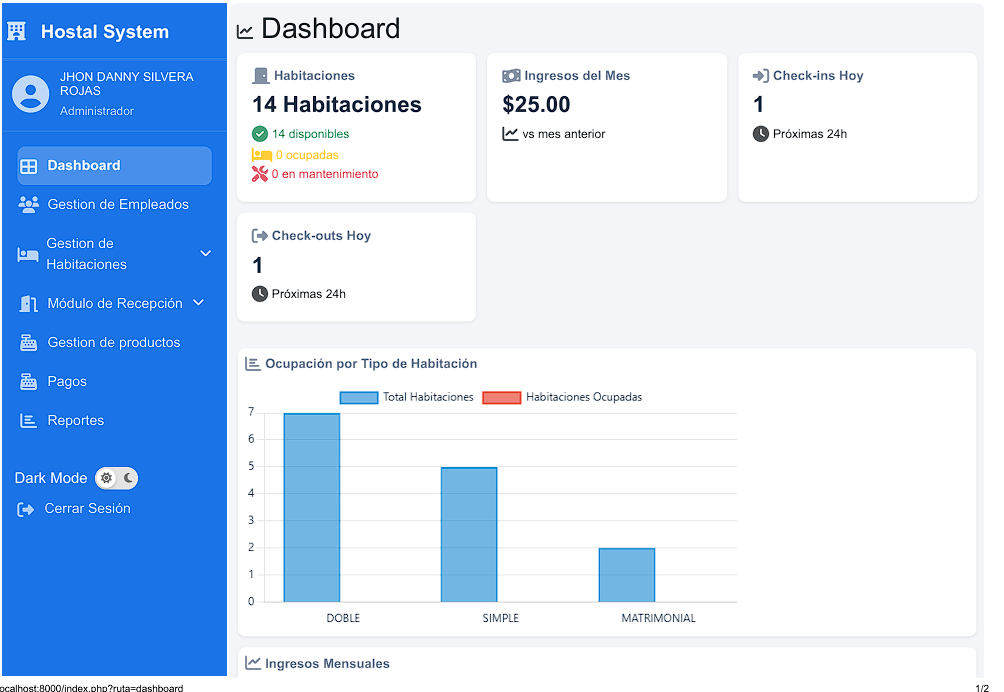
Tras la entrega final del sistema, se realizará una **capacitación al personal del hotel**, orientada al uso de cada módulo. Adicionalmente, se brindará un **período de soporte** para corregir errores no detectados durante las pruebas y asegurar una transición sin problemas al nuevo sistema.

# DISEÑO

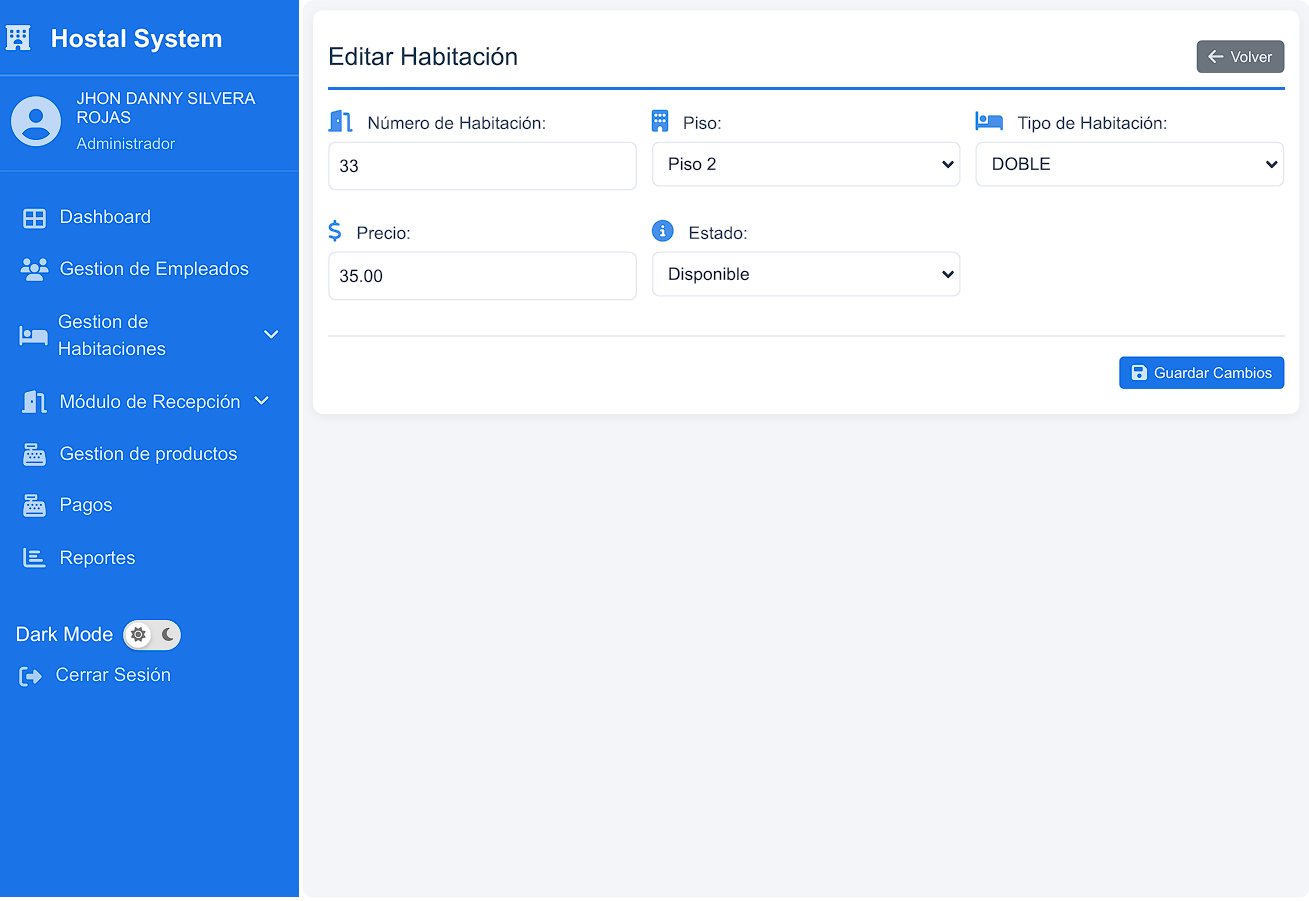
## DISEÑO DEL SISTEMA WEB DE GESTIÓN DEL HOSTAL

En el diseño del sistema web creamos interfaces tanto para el administrador y Recepcionista y con sus respectivas botones de navegación y acciones.

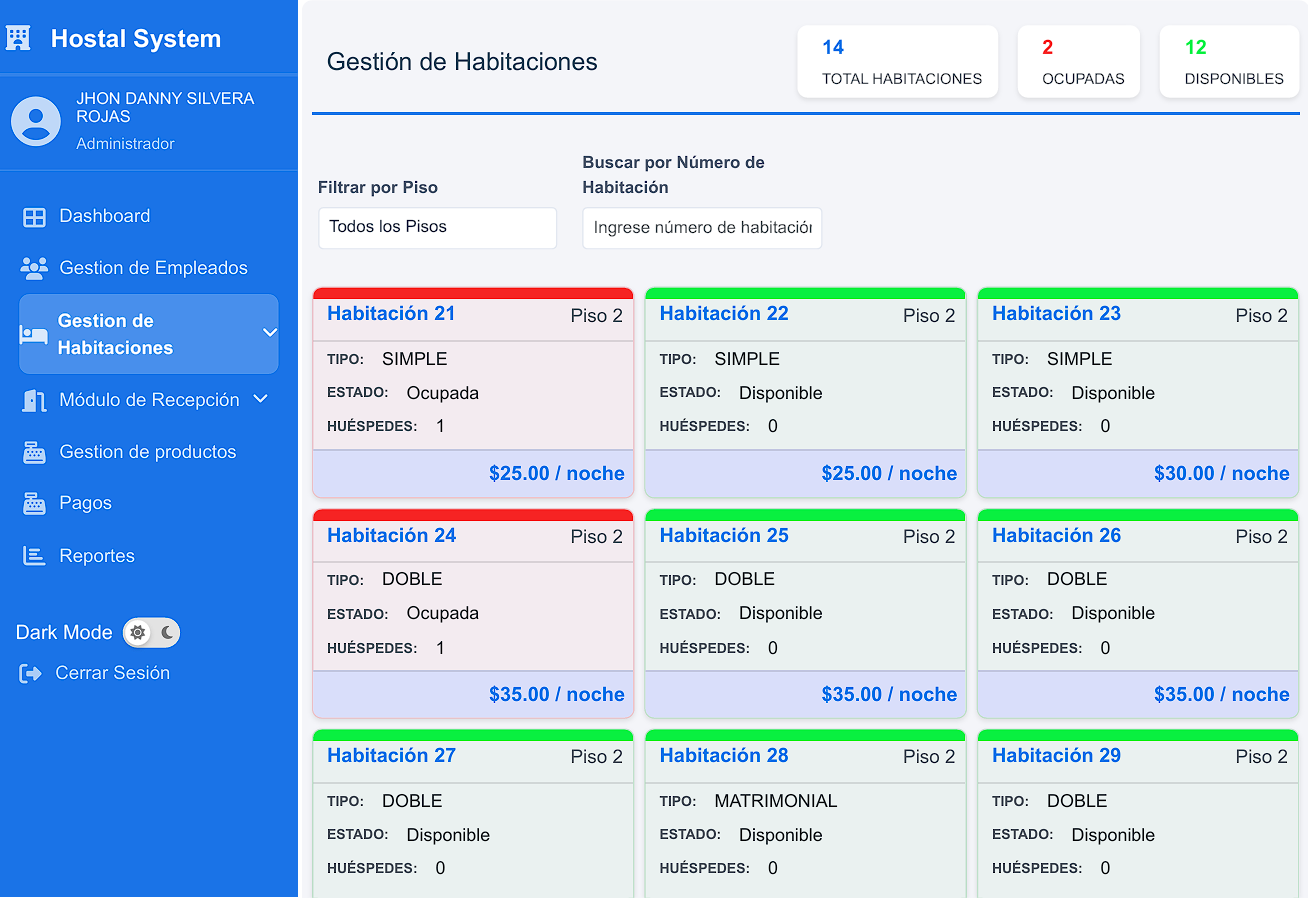
* Este es panel principal cuando inicia sesión el usuario automáticamente le redirige a dashboard.



* Gestión de habitaciones donde pueda agregar y editar según corresponda.



* Gestión de recepción y salida el usuario podrá ver habitaciones según su estado



* Gestión de productos que ofrece el hostal



## ARQUITECTURA DEL SISTEMA

En el desarrollo del sistema de gestión de hostal, he decidido utilizar el patrón arquitectónico Modelo-Vista-Controlador (MVC) debido a sus numerosas ventajas en términos de organización y mantenimiento del código. Para ello a continuación voy a detallar cada capa de este patrón de diseño.

### Modelo

El modelo es la parte del sistema que se encarga de la lógica de negocio y la gestión de datos. En el sistema de gestión Hostal, esto incluye las entidades clave como **Huésped**, **Estancia**, **Habitación** entre otros.

* Por ejemplo, en el modelo se va tener clases como Habitación, Huésped, Estancia y entre otros, y cada uno de estas va estar con sus respectivas propiedades o atributos.

### Controlador

El controlador actúa como un intermediario entre el modelo y la vista. Recibe las solicitudes de los usuarios desde la vista, procesa los datos y se comunica con el modelo para realizar las operaciones necesarias.

* Se han implementado controladores como **EstanciaController**, que maneja todas las operaciones relacionadas con las estancias, y **HuespedController**, que gestiona las operaciones de registro y actualización de datos de los huéspedes. Esto permite que las interacciones del usuario se gestionen de manera eficiente y organizada.

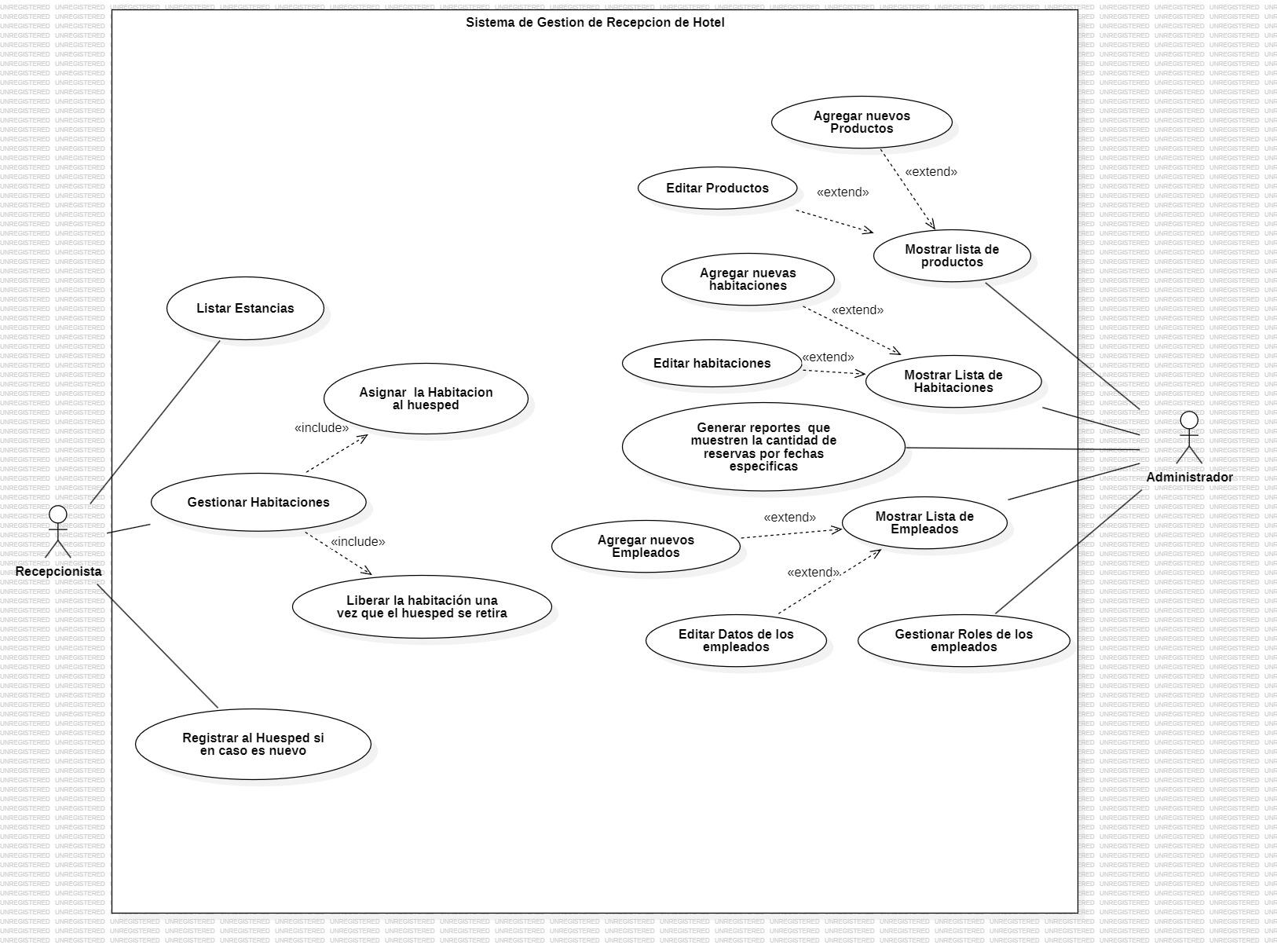
### Vistas

La vista se ocupa de la presentación de la información y de la interacción con el usuario. He diseñado interfaces intuitivas y atractivas que permiten a los usuarios realizar acciones como hacer reservas y gestionar habitaciones.

* Entre las vistas que se va crear, se encuentra la página de inicio del hostal, donde se presenta información general, un formulario de registro de huéspedes donde el recepcionista puede ingresar los datos de huéspedes y seleccionar habitaciones.

## DIAGRAMAS DE CASOS DE USO

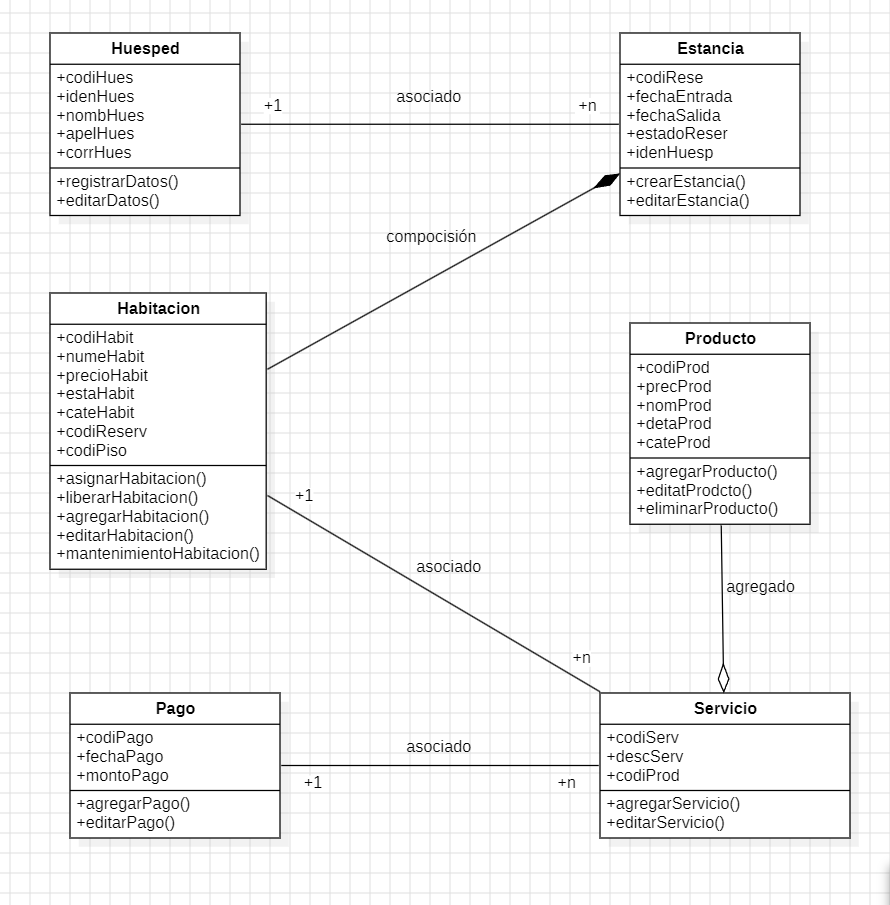
En el desarrollo de sistema de gestión de hostal, se van a crear diagramas de casos de uso para representar las interacciones entre los usuarios y el sistema. Estos diagramas me han ayudado a identificar las funcionalidades clave que debo implementar y a entender mejor las necesidades de los usuarios.





## DIAGRAMAS DE CLASES

En el desarrollo del sistema de gestión de hostal, se ha creado un diagrama de clases que representa las entidades principales del sistema y sus relaciones. Este diagrama ayuda a visualizar la estructura de la aplicación y a definir cómo interactúan las distintas partes del modelo.



# CODIFICACIÓN

## CLIENTE SIEMPRE PRESENTE

Desde el inicio del proyecto, mantenemos una comunicación constante con el cliente para asegurarnos de que todas las funcionalidades y ajustes cumplen con sus expectativas. Esto implica involucrarnos en cada fase del desarrollo, desde la definición de requisitos hasta la entrega final. Al estar siempre disponibles, el cliente puede dar retroalimentación rápida y permitir que hagamos ajustes sobre la marcha, lo que asegura que el sistema responda a sus necesidades reales sin perder tiempo en cambios inesperados más adelante.

## EL CÓDIGO SE ESCRIBE SIGUIENDO ESTÁNDARES

Nuestro equipo de desarrollo sigue buenas prácticas y estándares de codificación para asegurar que el software sea claro, legible y fácil de mantener. Esto significa que utilizamos nombres de variables descriptivos, seguimos guías claras para la estructura del código y evitamos complejidades innecesarias. De esta manera, cualquier desarrollador que se integre al proyecto podrá entender y trabajar con el código sin dificultad. Además, un código bien organizado reduce la posibilidad de errores y facilita futuras mejoras.

## PROPIEDAD COLECTIVA DEL CÓDIGO

Aquí no hay dueños exclusivos del código: todo el equipo es responsable del desarrollo y mantenimiento de cada parte del sistema. Esta práctica fomenta la colaboración y asegura que cualquiera pueda intervenir y hacer mejoras o correcciones en cualquier momento. Si alguien encuentra un error o detecta una manera más eficiente de resolver un problema, se siente libre de hacerlo, sin importar quién escribió esa sección del código inicialmente. Esto garantiza una mayor calidad del software y evita cuellos de botella si un desarrollador no está disponible.

## APLICACIÓN DE PRUEBAS

Para garantizar que cada funcionalidad del sistema funcione como se espera, realizaremos pruebas continuas durante todo el proceso de desarrollo. Antes de implementar cualquier nueva característica, primero escribimos pruebas que definen el comportamiento deseado. Esto nos permite detectar problemas desde el principio y corregirlos antes de que lleguen al usuario final. Además, realizaremos pruebas automatizadas y manuales para asegurar que los cambios no afecten otras partes del sistema. El objetivo es entregar un software confiable y de alta calidad que responda correctamente bajo cualquier circunstancia

# PRUEBAS Y CONTROL DE CALIDAD

Para asegurar que el sistema funcione de manera correcta, eficiente y sin problemas para todos los usuarios, realizaremos diferentes tipos de pruebas. Cada una de estas evaluaciones tiene un propósito específico y garantiza que cada parte del sistema cumpla su función, sea confiable y esté lista para el uso en el "Hostal el encuentro".

## PRUEBAS DE ACEPTACIÓN

Estas pruebas se realizan para confirmar que el sistema cumple con las expectativas del cliente y satisface las necesidades del hotel.

* **¿Qué probaremos?**Revisaremos si todas las funcionalidades solicitadas (como gestión de habitaciones y administración de empleados) funcionan tal como se espera.
* **¿Cómo lo haremos?**El cliente (administradores del hotel) utilizará el sistema en situaciones reales para validar que la interfaz es intuitiva y los procesos, como la reserva de habitaciones o la asignación de roles, funcionan sin complicaciones.
* **Ejemplo:** Un recepcionista realiza una reserva desde el sistema y verifica si se genera el comprobante correctamente, mientras que un administrador revisa si puede añadir nuevas habitaciones sin dificultades.
* **Criterio de éxito:** Si el cliente aprueba cada funcionalidad clave, daremos por superadas las pruebas de aceptación.

## PRUEBAS UNITARIAS

Estas pruebas se enfocan en validar que cada módulo o componente individual funcione de manera correcta y aislada.

* **¿Qué probaremos?**Módulos como la gestión de administración de habitaciones y registro de clientes.
* **¿Cómo lo haremos?**Verificaremos que cada función dentro de estos módulos entregue los resultados esperados para entradas válidas y que maneje adecuadamente los errores.
* **Ejemplo:** Cuando el recepcionista selecciona una habitación, la prueba unitaria verificará que la función que consulta la disponibilidad funcione sin errores.
* **Criterio de éxito:** El módulo debe devolver los resultados esperados para entradas válidas y manejar errores (como campos vacíos) de forma adecuada.

## PRUEBAS DE INTEGRACIÓN

Estas pruebas se encargarán de verificar que los distintos módulos del sistema se comuniquen correctamente entre sí.

* **¿Qué probaremos?**La interacción entre diferentes partes del sistema, como el registro del cliente y la actualización del estado de la habitación.
* **¿Cómo lo haremos?**Simularemos el proceso completo, desde que el recepcionista realiza registro al cliente hasta la asignación de la habitación. También se verificará la interacción entre la gestión de empleados y la administración de roles.
* **Ejemplo:** Al realizar el registro de un cliente, se comprueba que los datos del cliente se registran correctamente, la habitación cambia de estado a ocupada, y se genera correctamente el comprobante.
* **Criterio de éxito:** El sistema no debe romperse ni generar errores durante la interacción entre módulos.

## PRUEBAS DE RENDIMIENTO

Aunque el sistema estará dedicado exclusivamente a la gestión interna del hostal y no contará con una página web para realizar reservas en línea, es fundamental asegurarnos de que el rendimiento sea óptimo para los usuarios del día a día. Queremos que las operaciones internas, como el registro de huéspedes o la gestión de habitaciones, se realicen sin demoras que puedan afectar la eficiencia del personal.

### ¿Qué probaremos?

* El tiempo de respuesta de las funciones críticas, como la consulta de disponibilidad de habitaciones y el registro de huéspedes.
* La estabilidad del sistema cuando varios usuarios acceden simultáneamente, como el administrador y los recepcionistas realizando operaciones al mismo tiempo.
* El manejo de cargas pesadas en eventos particulares, como el registro de varios huéspedes en poco tiempo.

### ¿Cómo lo haremos?

* Realizaremos pruebas simulando que múltiples usuarios acceden al sistema de manera simultánea. Por ejemplo, un recepcionista puede estar ingresando un cliente mientras otro revisa el estado de las habitaciones.
* Verificaremos que las consultas y operaciones, como la generación de reportes o la actualización de datos, se realicen rápidamente sin causar bloqueos.

### Ejemplo de prueba de rendimiento:

Imaginemos que, durante un fin de semana de alta demanda, los recepcionistas registran a varios huéspedes de forma consecutiva mientras se actualizan las habitaciones a "ocupadas". En esta prueba, verificamos que el sistema responda rápidamente, sin ralentizaciones ni errores, y que el personal pueda operar de forma fluida incluso en momentos de mayor carga.

### Criterio de éxito:

* El sistema debe responder en menos de un segundo para cada operación esencial (como buscar una habitación o registrar un huésped).
* El rendimiento no debe verse afectado con varios usuarios trabajando simultáneamente.
* No deben ocurrir bloqueos, congelamientos ni demoras que impiden al personal realizar su trabajo con normalidad.

# LANZAMIENTO Y DESPLIEGUE DEL SISTEMA

## ESTRATEGIA DE DESPLIEGUE

El despliegue del sistema se realizará de forma **gradual** para minimizar riesgos y asegurar que todo funcione correctamente desde el primer momento.

* **Pruebas en un entorno controlado:** Primero, el sistema será ejecutado en un entorno de pruebas que imita al entorno real. Esto permitirá corregir errores antes del lanzamiento oficial.
* **Despliegue por fases:** Se activarán los módulos más críticos (como recepción) en la primera fase. Posteriormente, se irán activando otros módulos, como administración y reportes.
* **Supervisión constante:** Durante los primeros días del lanzamiento, se realizará un monitoreo continuo del sistema para detectar y solucionar cualquier problema rápidamente.

## PREPARACIÓN DE ENTORNO DE PRODUCCIÓN

El entorno de producción es el espacio donde la web funcionará oficialmente. A continuación, se detalla cómo nos aseguraremos de que todo esté listo:

* **Configuración del servidor web:** Se instalará el sistema en un servidor optimizado para soportar la carga esperada y garantizar accesos rápidos. Se cuidará la seguridad con certificados SSL para navegación segura (HTTPS).
* **Carga de datos iniciales:** Ingresamos los datos esenciales del hostal, como habitaciones, empleados, para que el sistema esté listo para su uso desde el primer acceso.
* **Implementación de copias de seguridad:** Se programarán respaldos automáticos diarios para evitar la pérdida de información importante.
* **Pruebas de conexión:** Verificaremos que los empleados puedan acceder al sistema desde dispositivos autorizados y que las conexiones sean rápidas y estables en todo momento.

## CAPACITACIÓN Y SOPORTE A USUARIOS

Para que los empleados del hotel aprovechen al máximo la plataforma web, se realizará lo siguiente:

* **Capacitación por roles:** Cada grupo de usuarios recibirá formación específica según sus funciones (por ejemplo, recepcionistas y administradores tendrán módulos distintos).
* **Manuales y videos:** Se entregarán guías sencillas, en formato escrito y audiovisual, que explican cómo utilizar cada módulo del sistema.
* **Soporte durante el lanzamiento:** El equipo de desarrollo estará disponible durante los primeros días para resolver dudas o inconvenientes.
* **Atención continua:** Se habilitará un canal de soporte (como correo o chat) para que los usuarios puedan reportar problemas o consultar cualquier duda en el futuro.

# CONCLUSIONES Y RECOMENDACIONES

## LOGROS DEL PROYECTO

A lo largo del proyecto, se lograron avances significativos gracias a la implementación de la metodología **Extreme Programming (XP)**, que permitió al nuestro equipo alcanzar los objetivos establecidos de forma eficiente y con alta calidad. Algunos de los logros más destacados fueron:

* **Cumplimiento de los Requisitos del Cliente:** El proyecto logró entregar el software solicitado por el cliente dentro de los plazos establecidos, con la calidad esperada y funcionalidad acorde a las necesidades identificadas en la fase inicial. Gracias a las entregas frecuentes y a la retroalimentación continua, el software evolucionó conforme a las expectativas del cliente.
* **Mejora Continua y Refactoring:** A lo largo del desarrollo, se implementó de manera constante el **refactoring** del código, lo que permitió mejorar la estructura y legibilidad del mismo sin comprometer la funcionalidad. Este enfoque garantizó que el código mantuviera una alta calidad a lo largo del ciclo de vida del proyecto.
* **Alta Cobertura de Pruebas:** Uno de los mayores logros fue la integración efectiva de **pruebas automatizadas** en todo el proceso. Se logró una cobertura de pruebas del 90%, lo que resultó en una reducción significativa de errores y una mayor confiabilidad del software al finalizar el proyecto.
* **Trabajo Colaborativo y Comunicación:** Gracias a la práctica de **programación en pareja**, el equipo logró no solo mejorar la calidad del código, sino también compartir conocimientos entre los miembros, lo que resultó en un desarrollo más eficiente y en la creación de un ambiente de trabajo colaborativo y bien integrado.

## LECCIONES APRENDIDAS

Durante la ejecución del proyecto, hubo varios desafíos y momentos de aprendizaje que nos ayudaron a mejorar tanto el proceso de desarrollo como la gestión del equipo. Entre las lecciones más importantes se encuentran:

* **Adaptación a Cambios en los Requerimientos:** Aunque el proceso de desarrollo iterativo fue efectivo para adaptarnos a cambios en los requerimientos, en algunos momentos la cantidad de cambios inesperados generó ciertos retrasos en la planificación. En futuras iteraciones, se debería trabajar más estrechamente con el cliente para anticipar estos cambios y evitar que afecten el ritmo del proyecto.
* **La Programación en Pareja Requiere Adaptación:** La programación en pareja resultó ser muy beneficiosa para la calidad del código, pero no todos los miembros del equipo se sintieron cómodos con esta práctica al principio. A medida que avanzó el proyecto, los miembros se adaptaron mejor a esta metodología, pero al inicio se requirió más tiempo para integrar completamente esta técnica de trabajo colaborativo.
* **Priorización del Refactoring:** Aunque se implementaron cambios de refactoring durante todo el proyecto, a veces no se le dio la prioridad suficiente debido a la presión por cumplir con las funcionalidades. Aprendimos que el refactoring debe considerarse una parte integral del proceso de desarrollo, no solo como una mejora posterior.
* **Tiempo de Pruebas Automáticas:** Si bien las pruebas automáticas fueron clave para asegurar la calidad del software, al principio no se consideró adecuadamente el tiempo necesario para desarrollarlas. Este retraso inicial provocó un esfuerzo adicional en etapas posteriores para alcanzar la cobertura de pruebas deseada.

## RECOMENDACIONES PARA FUTUROS PROYECTOS

A partir de la experiencia obtenida en este proyecto, hemos identificado varias áreas de mejora y recomendaciones para proyectos futuros basados en **Extreme Programming**:

* **Mejorar la Gestión de Cambios:** Es fundamental establecer un proceso más claro para gestionar los cambios en los requerimientos del cliente. Si bien XP permite cambios frecuentes, es importante tener un control más estricto sobre cuándo y cómo se implementan estos cambios para no interrumpir el flujo del proyecto.
* **Optimización de la Programación en Pareja:** Aunque la programación en pareja resultó ser exitosa, recomiendo que se proporcione más tiempo de adaptación y formación para los nuevos miembros del equipo, para que se integren más rápidamente a esta metodología.
* **Ampliar la Capacitación en Pruebas Automáticas:** En futuros proyectos, es recomendable invertir más tiempo al principio en la capacitación sobre **pruebas automatizadas** para asegurar que todo el equipo pueda desarrollar y ejecutar pruebas de manera eficiente, sin causar retrasos en las fases posteriores.
* **Planificación para Refactoring Continuo:** La práctica de refactorizar el código debe ser una prioridad constante durante todo el ciclo de vida del proyecto, no solo cuando se presentan problemas. Asegurarse de que haya tiempo dedicado exclusivamente a la mejora continua del código es esencial para mantener la calidad del software.
* **Revisión de la Comunicación con el Cliente:** Establecer reuniones más frecuentes de retroalimentación con el cliente podría ayudar a alinear mejor las expectativas y evitar malentendidos que puedan retrasar la entrega de funcionalidades clave. La comunicación continua es esencial para el éxito de cualquier proyecto á

