CORPORACION UNIREGMINTON

Arturo Ch. Taller evaluativo 02/04/2025

Introducción:

Se realizará la debida evaluación de los requerimientos definidos en el archivo "Entrevista.pdf", entregado por el docente a cargo del área y se realizará contraste con el diseño de clases presentado en el archivo de la compañera lorena nombrado: "Requerimientos Software.pdf". Se analizará la calidad del diseño propuesto y su cobertura en los requisitos solicitados previamente, con el fin de identificar posibles mejoras y encontrar una solución de software eficiente mediante la programación orientada a objetos (POO) en Python.

Informe de implementación y evaluación de requerimientos

Se identificaron los siguientes requerimientos clave del sistema:

1. Registro de hoteles y habitaciones: se debe registrar información detallada de hoteles y habitaciones, incluyendo servicios, descripciones y estados.
2. Gestión de habitaciones: registro de habitaciones por tipo, capacidad, precio, servicios y por último fotos.
3. Estados de hoteles y habitaciones: control y manejo del estado de las habitaciones y los hoteles: (mantenimiento, remodelación, disponibilidad).
4. Políticas de pago y cancelación: gestión de políticas de pago y cancelación asociadas a reservas de este.
5. Gestión de reservas y pagos: se denota la capacidad de realizar reservas de estos, calcular los costos y gestionar los debidos pagos.
6. Reseñas y calificaciones: Permitir a los clientes dejar reseñas y calificaciones.

El archivo de "Requerimientos Software.pdf" cumple en gran medida con los puntos mencionados; sin embargo, se identificaron mejoras posibles en el modelo UML.

Detalla miento de los requerimientos:

1. Registro de hoteles y habitaciones:

- El requerimiento menciona el registro de hoteles con información básica y detalles adicionales como, por ejemplo: (servicios, fotos, descripción), lo cual coincide con lo mencionado por la dama Luciana en el pdf de la entrevista sobre la necesidad de incluir información detallada del hotel.

- En el diagrama, el hotel tiene todos los atributos necesarios excepto "fotos", lo cual sería una mejora para agregar.

2. Gestión de habitaciones:

- El sistema debe registrar habitaciones de siguiente modo: por tipo, capacidad, precio, servicios y por último, pero no menos importante fotos, algo que también se menciona en la entrevista. En el diagrama, se ve reflejado la mayoría por no decir su totalidad, pero falta el atributo "fotos", que sería importante incluir para cumplir completamente el requerimiento.

3. Estado de hoteles y habitaciones:

- La entrevista menciona claramente el estado de los hoteles y las habitaciones: (mantenimiento, remodelación). Esto se ve reflejado en el diagrama con el atributo "estado" en Hotel y Habitación, cumpliendo así correctamente el requerimiento.

4. Políticas de pago y cancelación:

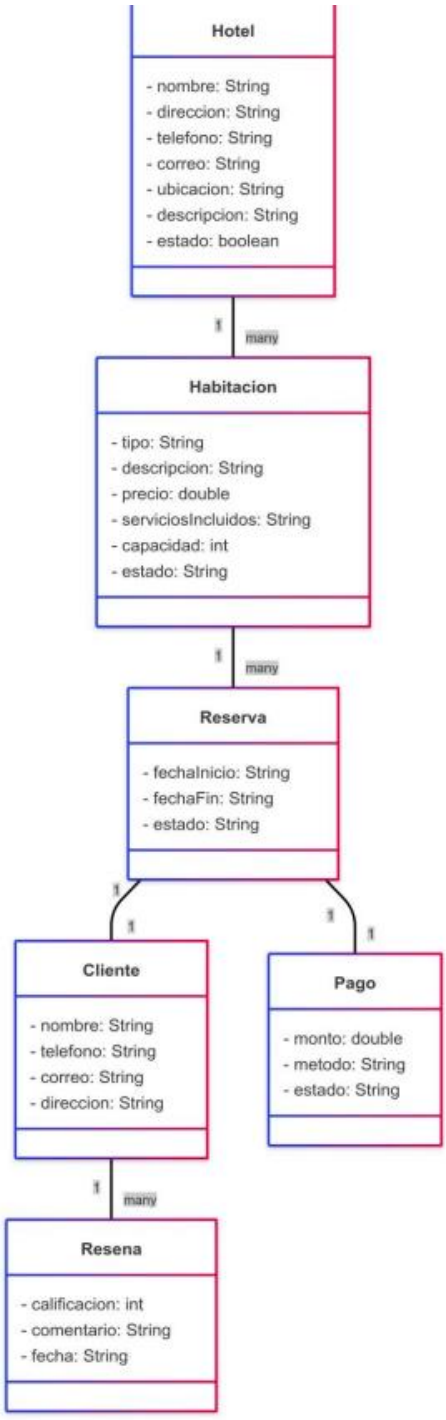
- En la entrevista se menciona que los hoteles manejan políticas de pago y cancelación por ende en el diagrama se logra apreciar que esto está incluido a través de la entidad "Pago" con sus atributos respectivos, pero sería útil agregar la política de cancelación para reflejarlo a la perfección.

5. Gestión de reservas y pagos:

- En la entrevista se habla de la reserva, los calendarios y la búsqueda de habitaciones por criterios específicos, esta se cumple bien y se ve reflejada en el diagrama con la entidad "Reserva".

6. Reseñas y calificaciones:

- La entrevista destaca la importancia de las calificaciones y comentarios, algo reflejado en la entidad "Reseña". Sin embargo, se podría mejorar añadiendo un campo para el promedio de calificación.


Hotel
- nombre: String
- direccion: String
- telefono: String
- correo: String
- ubicacion: String
- descripcion: String
- estado: boolean
1
many
Habitacion
- tipo: String
- descripcion: String
- precio: double
- serviciosIncluidos: String
- capacidad: int
- estado: String
1
many
Reserva
- fechaInicio: String
- fechaFin: String
- estado: String
1
1
1
1
Cliente
- nombre: String
- telefono: String
- correo: String
- direccion: String
Pago
- monto: double
- metodo: String
- estado: String
1
many
Resena
- calificacion: int
- comentario: String
- fecha: String

Análisis del diagrama:

Este diagrama UML refleja un correcto análisis de las mayorías de relaciones entre las entidades, pero falla en los siguientes aspectos:

- Falta del atributo de fotos tanto en hotel como en Habitación
- Las políticas de cancelación no están muy bien definidas
- No existe una entidad o atributo que gestione la disponibilidad de habitaciones por calendario
- La relación entre reserva y pago carece de lo siguientes estados: (pendiente, completado, cancelado)

Aspectos para mejorar:

- Agregar campos faltantes:
  - "Fotos" en hotel y habitación.
  - Políticas de cancelación y condiciones de pago más detalladas.
  - Promedio de calificación en reseña o en hotel.
- Incluir relaciones más complejas:
  - Un calendario de disponibilidad para cada habitación (como atributo o entidad separada).
- Profundizar en la relación Reserva - Pago:
  - Agregar detalle sobre el estado de los pagos (pendiente, completado, cancelado).
- Políticas de temporada:
  - Crear una entidad que administre precios según temporada, con un calendario detallado.

Código fuente con trazabilidad a los requerimientos:

```
class Hotel:
    """Representa un hotel con información detallada."""
    def __init__(self, nombre, direccion, servicios, fotos=[]):  # Req. 1
        self.nombre = nombre
        self.direccion = direccion
        self.servicios = servicios
        self.fotos = fotos
        self.habitaciones = []

    def agregar_habitacion(self, habitacion):
        self.habitaciones.append(habitacion)

class Habitacion:
    """Representa una habitación de hotel."""
    def __init__(self, tipo, capacidad, precio, servicios, estado, fotos=[]):  # Req. 2, 3
        self.tipo = tipo
        self.capacidad = capacidad
        self.precio = precio
        self.servicios = servicios
        self.estado = estado
        self.fotos = fotos

class Reserva:
    """Gestión de reservas de hotel."""
    def __init__(self, cliente, habitacion, fecha_entrada, fecha_salida):  # Req. 5
        self.cliente = cliente
        self.habitacion = habitacion
        self.fecha_entrada = fecha_entrada
        self.fecha_salida = fecha_salida
        self.estado_pago = "Pendiente"  # Req. 4

    def realizar_pago(self, monto):
```

```
        if monto >= self.habitacion.precio:
            self.estado_pago = "Completado"
        else:
            self.estado_pago = "Pendiente"

class Cliente:
    """Representa un cliente que realiza reservas."""
    def __init__(self, nombre, correo):
        self.nombre = nombre
        self.correo = correo

class Reseña:
    """Permite a los clientes dejar reseñas y
calificaciones."""  # Req. 6
    def __init__(self, cliente, habitacion, calificacion,
comentario):
        self.cliente = cliente
        self.habitacion = habitacion
        self.calificacion = calificacion
        self.comentario = comentario
```

Cambio correcto de diagrama UML: